SCIENCE COMPUTER YEAR 12

COMPLETE NOTES

System Development Methodologies

Projects

In business what is a project?

A project is a planned set of interrelated tasks to be executed over a fixed period and within certain cost and other limitations. Unlike most business systems one of its key characteristics is that it has a specific beginning and end. Most business systems are permanent. Or at least have some level of permanency to them. A project, on the other hand, is conceived as a temporary endeavour to achieve a defined outcome.

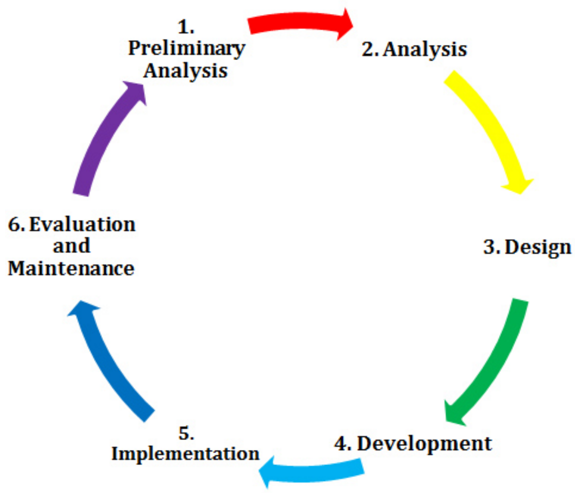
Linear - Waterfall Model - SDLC

The waterfall model is a linear sequential (non-iterative) design approach for software development, in which progress flows in one direction downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, deployment and maintenance. This is also often referred to as the system-development life cycle.

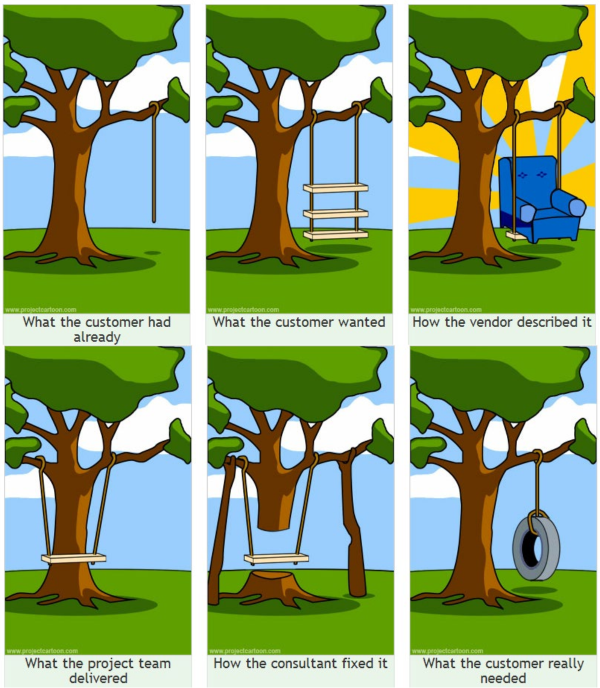
The System Development Life Cycle, "SDLC" for short, is a multistep, iterative process, structured in a methodical way. This process is used to model or provide a framework for technical and non-technical activities to deliver a quality system which meets or exceeds a business's expectations or manage decision-making progression.

The waterfall development model originated in the manufacturing and construction industries: highly structured physical environments in which after-the-fact changes are impossible or at least prohibitively expensive. At the time it was adopted for software development, there were no recognised alternatives for knowledge-based creative work.

The SDLC highlights different stages (phrases or steps) of the development process. The life cycle approach is used so users can see and understand what activities are involved within a given step. It is also used to let them know that at any time, steps can be repeated or a previous step can be reworked when needing to modify or improve the system.



Pros of Waterfall



- Clear deadlines: Waterfall's static nature and predictable workflow make it easy to estimate costs, create timelines, and stick to deadlines.
- Disciplined by design: Since each phase has a clear start point and a requirement review gate at the end of it, the team is forced to complete all tasks before the project as a whole can proceed.
- Well-documented: Waterfall requires documentation and a clear paper trail for each phase of development. This makes it easier to follow the logic of past projects and lay the groundwork for future projects.
- Clear communication: Predictable timelines and well-documented projects make it easy to give status updates to upper management, stakeholders, or clients with strict requirements.
- Easy learning curve: As the traditional approach to project management across industries, teams usually don't require any prior knowledge or training in order to start working on a project with the Waterfall method.

Cons of Waterfall

- Change can be costly: The major downside to Waterfall's rigidity is the hampered ability to handle change. Testing occurs late in the project life cycle, and if you find out that your end users don't like the product you're building, it can be too late to pivot.
- Slow delivery times: As many as four phases of development need to be completed before any coding begins—which means stakeholders and customers won't see a working product until late in the life cycle.
- Gathering requirements too early is risky: Customers and stakeholders often don't know what they really want until they've had a chance to play with a working prototype. Since Waterfall handles all the requirement gathering upfront, there's a real risk of missing the mark and causing further headaches down the project line.
- Tendency to neglect testing: Saving all the testing for the end of a project can be risky, because of the temptation to rush through it when there's a looming deadline. A poorly tested product can lead to a disastrous launch. You also lose out on valuable data you could have gained earlier in the project.

1 Preliminary analysis

The objective of phase 1 is to conduct a preliminary analysis, propose alternative solutions, describe costs and benefits and submit a preliminary plan with recommendations.

1. **Problem Definition.** Conduct the preliminary analysis: in this step, you need to find out the organization's objectives and the nature and scope of the problem under study. Even if a problem refers only to a small segment of the organization itself, you need to find out what the objectives of the organization itself are. Then you need to see how the problem being studied fits in with them. The **System Boundary** is developed defining the scope and limits or boundaries of the system.
2. **Feasibility Study.** Feasibility is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software. Information such as resource availability, cost estimation for software development, benefits of the software to the organization after it is developed and cost to be incurred on its maintenance are considered during the feasibility study. The objective of the feasibility study is to establish the reasons for developing the software that is acceptable to users, adaptable to change and conformable to established standards. Various other objectives of feasibility study are listed below.

2 Analysis

Model of current system, scrutiny of the existing system: Identify pros and cons of the current system in-place, so as to carry forward the pros and avoid the cons in the new system.

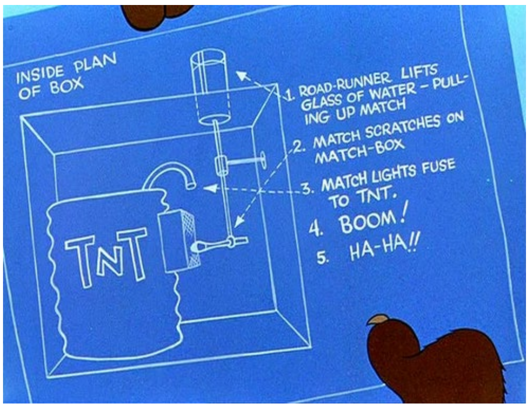
Requirements of New System. Systems analysis, requirements definition: Defines project goals into defined functions and operation of the intended application. It is the process of gathering and interpreting facts, diagnosing problems and recommending improvements to the system. Analyses end-user information needs and also removes any inconsistencies and incompleteness in these requirements.

A series of steps followed by the developer are:

1. Collection of Facts: End user requirements are obtained through documentation, client interviews, observation and questionnaires,
2. Scrutiny of the existing system: Identify pros and cons of the current system in-place, so as to carry forward the pros and avoid the cons in the new system.
3. Analysing the proposed system: Solutions to the shortcomings in step two are found and any specific user proposals are used to prepare the specifications.

3 Design

The third phase describes, in detail, the necessary specifications, features and operations that will satisfy the functional requirements of the proposed system which will be in place. This is the step for end users to discuss and determine their specific business information needs for the proposed system. It's during this phase that they will consider the essential components (hardware and/or software) structure (networking capabilities), processing and procedures for the system to accomplish its objectives.



Mockups, wireframes, and storyboards—in this phase, the designers put a face to the project. Requirements are reviewed and evaluated, team goals are set, and a plan of action is developed.

The **logical design** of a system pertains to an abstract representation of the data flows, inputs and outputs of the system.

The **physical design** relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified/authenticated, how it is processed, and how it is displayed.

4 Development

The developers start building the actual app based on flowcharts, mockups, and designs created in the previous phase.

The fourth phase is when the real work begins—in particular, when a programmer, network engineer and/or database developer are brought on to do the major work on the project. This work includes using a flow chart to ensure that the process of the system is properly organized. The development phase marks the end of the initial section of the process. Additionally, this phase signifies the start of production. The development stage is also characterized by instillation and change. Focusing on training can be a huge benefit during this phase.

During this stage you have:

- Hardware and software acquisition
- Construction and testing

5 Implementation

The fifth phase is when the majority of the code for the program is written. Additionally, this phase involves the actual installation of the newly-developed system. There are a number of **change-over methods** for this stage including:

1. **Direct Cut.** Direct cut, also referred to as immediate replacement, tends to be the least favourite of the changeover techniques. In a direct changeover, the entire system is replaced in an instant. Basically, as soon as the new system is powered up, the old system is shut down. This type of changeover carries the most risk because, if something goes wrong, reverting back to the old system usually is impossible. Using the direct changeover technique tends to work best in situations where a system failure isn't critical enough to result in a disaster for the company.

2. **Phased Changeover.** The phased changeover technique is considered a compromise between parallel and direct changeovers. In a phased changeover, the new system is implemented one stage at a time. As an example, consider a company working toward installing a new financial system. Implementing the new system one department at a time, the company converts accounts receivable, accounts payable, payroll, and so on. Advantages to phased changeovers are their low cost and isolated errors. The main disadvantage is the process takes a long time to complete because phases need to be implemented separately.
3. **Pilot Changeover.** With a pilot changeover, the new system is tried out at a test site before launching it company-wide. For example, a bank may first test the system at one of its branches. This branch is referred to as the pilot, or beta, site for the program. Since parallel changeovers tend to be expensive, using the pilot changeover technique allows companies to run the new system next to their old but on a much smaller scale. This makes the pilot changeover method much more cost-effective. After the kinks are worked out of the system at the test site, companies usually opt to use the direct changeover technique to launch the system company-wide.
4. **Parallel Changeover.** In a parallel changeover, the new system runs simultaneously with the old for a given period of time. Of all the techniques, this tends to be the most popular, mainly because it carries the lowest risk. If something goes wrong at any point, the entire system can be reverted back to its original state. A primary disadvantage in running two systems at the same time is higher costs. The parallel changeover process also can be quite time-consuming.

6 Evaluation and Maintenance

Evaluation and Maintenance involves;

- Performance evaluation
- Fault finding and correction

The Sixth and final phase involves maintenance and regular required updates. This step is when end users can fine-tune the system, if they wish, to boost performance, add new capabilities or meet additional user requirements. During the maintenance stage of the SDLC, the system is assessed to ensure it does not become obsolete. This is also where changes are made to initial software. It involves continuous evaluation of the system in terms of its performance.

This stage is also where the system that was developed, as well as the entire process, is evaluated. Some of the questions that need to be answered include: does the newly implemented system meet the initial business requirements and objectives? Is the system reliable and fault-tolerant? Does the system function according to the approved functional requirements? In addition to evaluating the software that was released, it is important to assess the effectiveness of the development process. If there are any aspects of the entire process, or certain stages, that management is not satisfied with, this is the time to improve. Evaluation and assessment is a difficult issue. However, the company must reflect on the process and address weaknesses.

Iterative - Rapid Application Development

Rapid-application development (RAD) is both a general term used to refer to alternatives to the conventional waterfall model of software development as well as the name for James Martin's approach to rapid development. In general, RAD approaches to software development put less emphasis on planning and more emphasis on process. In contrast to the waterfall model, which calls for rigorously defined specification to be established prior to entering the development phase, RAD approaches emphasize adaptability and the necessity of adjusting requirements in response to knowledge gained as the project progresses. Prototypes are often used in addition to or sometimes even in place of design specifications.

RAD is especially well suited for (although not limited to) developing software that is driven by user interface requirements. In the slow, methodical software development methods of olde, receiving useful and concrete user feedback has been inherently difficult, costly, and time consuming. Long meetings and phone calls, and even longer design docs, were a necessary evil to lay out even the most basic concrete plans of proper software design. With typical waterfall methods, rudimentary user feedback was often many months if not years in the future, after all planning and most development had taken place.

In stark contrast, one of the biggest benefits to rapid application development is the ability to both easily and frequently receive feedback from users who are directly interfacing with the application during development and prototyping. While this advantage is most readily visible within the UI/UX components of the system, iterative design intrinsically means user feedback can be at the forefront of the process.

The James Martin approach to RAD divides the process into four distinct phases:

1. **Requirements planning phase** – combines elements of the system planning and systems analysis phases of the Systems Development Life Cycle (SDLC). Users, managers, and IT staff members discuss and agree on business needs, project scope, constraints, and system requirements. It ends when the team agrees on the key issues and obtains management authorization to continue.
2. **User design phase** – during this phase, users interact with systems analysts and develop models and prototypes that represent all system processes, inputs, and outputs. The RAD groups or subgroups typically use a combination of Joint Application Development (JAD) techniques and CASE tools to translate user needs into working models. User Design is a continuous interactive process that allows users to understand, modify, and eventually approve a working model of the system that meets their needs.
3. **Construction phase** – focuses on program and application development task similar to the SDLC. In RAD, however, users continue to participate and can still suggest changes or improvements as actual screens or reports are developed. Its tasks are programming and application development, coding, unit-integration and system testing.
4. **Cutover phase** – resembles the final tasks in the SDLC implementation phase, including data conversion, testing, changeover to the new system, and user training. Compared with traditional methods, the entire process is compressed. As a result, the new system is built, delivered, and placed in operation much sooner.

The advantages of RAD include:

- **Better quality.** By having users interact with evolving prototypes the business functionality from a RAD project can often be much higher than that achieved via a waterfall model. The software can be more usable and has a better chance to focus on business problems that are critical to end users rather than technical problems of interest to developers.
- **Risk control.** Although much of the literature on RAD focuses on speed and user involvement a critical feature of RAD done correctly is risk mitigation. It's worth remembering that Boehm initially characterized the spiral model as a risk based approach. A RAD approach can focus in early on the key risk factors and adjust to them based on empirical evidence collected in the early part of the process. E.g., the complexity of prototyping some of the most complex parts of the system.
- **More projects completed on time and within budget.** By focusing on the development of incremental units the chances for catastrophic failures that have dogged large waterfall projects is reduced. In the Waterfall model it was common to come to a realization after six months or more of analysis and development that required a radical rethinking of the entire system. With RAD this kind of information can be discovered and acted upon earlier in the process.

The disadvantages of RAD include:

- **The risk of a new approach.** For most IT shops RAD was a new approach that required experienced professionals to rethink the way they worked. Humans are virtually always averse to change and any project undertaken with new tools or methods will be more likely to fail the first time simply due to the requirement for the team to learn.
- **Requires time of scarce resources.** One thing virtually all approaches to RAD have in common is that there is much more interaction throughout the entire life-cycle between users and developers. In the waterfall model, users would define requirements and then mostly go away as developers created the system. In RAD users are involved from the beginning and through virtually the entire project. This requires that the business is willing to invest the time of application domain experts. The paradox is that the better the expert, the more they are familiar with their domain, the more they are required to actually run the business and it may be difficult to convince their supervisors to invest their time. Without such commitments RAD projects will not succeed.
- **Less control.** One of the advantages of RAD is that it provides a flexible adaptable process. The ideal is to be able to adapt quickly to both problems and opportunities. There is an inevitable trade-off between flexibility and control, more of one means less of the other. If a project (e.g. life-critical software) values control more than agility RAD is not appropriate.
- **Poor design.** The focus on prototypes can be taken too far in some cases resulting in a "hack and test" methodology where developers are constantly making minor changes to individual components and ignoring system architecture issues that could result in a better overall design. This can especially be an issue for methodologies such as Martin's that focus so heavily on the user interface of the system.
- **Lack of scalability.** RAD typically focuses on small to medium-sized project teams. The other issues cited above (less design and control) present special challenges when using a RAD approach for very large scale systems.

Prototyping

Margaret Rouse

The prototyping model is a development method in which a prototype (an early approximation of a final system or product) is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved.

The Prototyping Model is a development method in which a prototype (an early approximation of a final system or product) is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved from which the complete system or product can now be developed. This model works best in scenarios where not all of the project requirements are known in detail ahead of time. It is an iterative, trial-and-error process that takes place between the developers and the users.

There are several steps in the Prototyping Model:

1. The new design requirements are defined in as much detail as possible.
2. A preliminary design is created for the new system.
3. A first prototype of the new design is constructed from the preliminary design. This is usually scaled-down, and represents an approximation of the characteristics of the final product.
4. The users thoroughly evaluate the first prototype, noting its strengths and weaknesses, what needs to be added, and what should to be removed. The developer collects and analyses the feedback from the users.
5. The first prototype is modified, based on the comments supplied by the users, and a second prototype of the new design is constructed.
6. The second prototype is evaluated in the same manner as was the first prototype.
7. The preceding steps are iterated as many times as necessary, until the users are satisfied that the prototype represents the final product desired.
8. The final design is constructed, based on the final prototype.
9. The final system is thoroughly evaluated and tested. Routine maintenance is carried out on a continuing basis to prevent large-scale failures and to minimize downtime.

Documentation

User Documentation

User Documentation helps the user USE the system that has been built. So explains how to do data entry, operate the Software, get output and what the output means.

System Documentation

System Documentation explains how the system was built for other programmers so they can modify/ update/ fix it later as well as the Update Strategy. It includes the data dictionary.

Activities

1. Explain the difference between a linear and an iterative development method. What are some advantages and disadvantages of each method?
2. What type of project would benefit from a linear approach to development?
3. What type of project would benefit from a RAD approach to development?
4. Which approach would you tend to use? Why?
5. What is the Software Development Cycle (not in the notes)? How is this different from the linear development method described in the notes.

Images

1. http://www.larevueautomobile.com/photo-voiture/photo-voiture.php?src=/images/Renault/Twizy-ZE-Concept/Exterieur/Renault_Twizy_ZE_Concept_051.jpg
2. http://www.projectcartoon.com/images/project.jpg
3. https://s.blogcdn.com/www.dailyfinance.com/media/2013/04/cost-benefit-analysis-604cs041813.jpg
4. http://farm6.static.flickr.com/5226/5672767256_e735bc255d.jpg
5. http://cms.bbcomcdn.com/fun/images/2015/arnold-blueprint-mass-training-4-graphics.jpg
6. http://www.tecco.com.au/images/Implementation0.jpg
7. http://smallbusiness.chron.com/changeover-techniques-34890.html
8. https://media.giphy.com/media/xThuWg7lusylvpAVu8/giphy.gif
9. https://s3.amazonaws.com/production-wordpress-assets/blog/wp-content/uploads/2016/11/23155646/RAD.png
10. https://www.innovativearchitects.com/KnowledgeCenter/basic-IT-systems/system-development-life-cycle.aspx

Data Gathering

Observation

Seeing is believing, they say. Making direct observations, when the situation allows for it, is a very quick and effective way of collecting data with minimal intrusion. Establishing the right mechanism for making the observation is all you need.

Advantages:

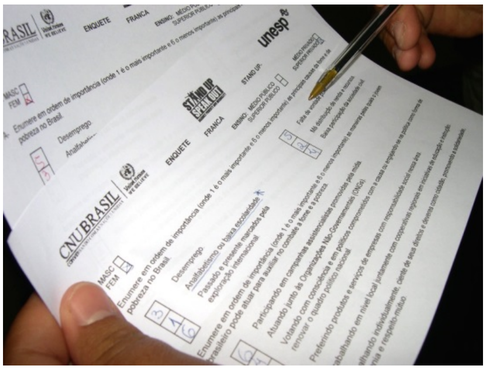
- Non-responsive sample subjects are a non-issue when you're simply making direct observation.
- This mode doesn't require a very extensive and well-tailored training regime for the survey workforce.
- It is not as time-consuming as the other modes that we will be discussing below.
- Infrastructure requirement and preparation time are minimal.

Disadvantages:

- Heavy reliance on experts who must know what to observe and how to interpret the observations once the data collection is done.
- There is the possibility of missing out on the complete picture due to the lack of direct interaction with sample subjects.

Making direct observations can be a good way of collecting information about mechanical, orderly tasks, like checking the number of manual interventions required in a day to keep an assembly line functioning smoothly.

Questionnaires



Questionnaires, as we consider them here, are stand-alone instruments of data collection that will be administered to the sample subjects either through mail, phone or online. They have long been one of the most popular data collection techniques.

Advantages:

- Questionnaires give the researchers an opportunity to carefully structure and formulate the data collection plan with precision.
- Respondents can take these questionnaires at a convenient time and think about the answers at their own pace.
- The reach is theoretically limitless. The questionnaire can reach every corner of the globe if the medium allows for it.

Disadvantages:

- Questionnaires without human intervention (as we have taken them here) can be quite passive and miss out on some of the finer nuances, leaving the responses open to interpretation. Interviews and Focus Group Sessions, as we shall see later, are instrumental in overcoming this shortfall of questionnaires.
- Response rates can be quite low. Questionnaires can be designed well by choosing the right question types to optimize response rates, but very little can be done to encourage the respondents without directly conversing with them.

The survey can be carried out through directly-administered questionnaires when the sample subjects are relatively well-versed with the ideas being discussed and comfortable at making the right responses without assistance. A survey about newspaper reading habits, for example, would be perfect for this mode.

Interviews

Conducting interviews can help you overcome most of the shortfalls of the previous two data collection techniques that we have discussed here by allowing you to build a deeper understanding of the thinking behind the respondents' answers.

Advantages:

- Interviews help the researchers uncover rich, deep insight and learn information that they may have missed otherwise.
- The presence of an interviewer can give the respondents additional comfort while answering the questionnaire and ensure correct interpretation of the questions.
- The physical presence of a persistent, well-trained interviewer can significantly improve the response rate.

Disadvantages:

- Reaching out to all respondents to conduct interviews is a massive, time-consuming exercise that leads to a major increase in the cost of conducting a survey.
- To ensure the effectiveness of the whole exercise, the interviewers must be well-trained in the necessary soft skills and the relevant subject matter.

Interviews are the most suitable technique for sample forms that touch upon complex issues like healthcare and family welfare. The presence of an interviewer to help respondents interpret and understand the questions can be critical to the success of the survey.

Sample Forms

Sample forms are a good way of gathering a large amount of data, providing a broad perspective. Sample forms can be administered electronically, by telephone, by mail or face to face. Mail and electronically administered sample forms have a wide reach, are relatively cheap to administer, information is standardised and privacy can be maintained. They do, however, have a low response rate, are unable to investigate issues to any great depth, require that the target group is literate and do not allow for any observation.

As sample forms are self-reported by participants, there is a possibility that responses may be biased particularly if the issues involved are sensitive or require some measure of disclosure on trust by the participant. It is therefore vital that sample forms used are designed and tested for validity and reliability with the target groups who will be completing the sample forms.

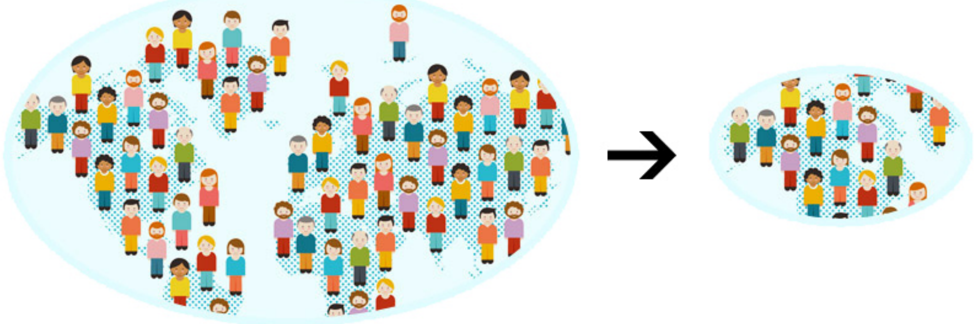
Careful attention must be given to the design of the sample form. If possible the use of an already designed and validated sample form instrument will ensure that the data being collected is accurate. If you design your own sample form it is necessary to pilot test the sample form on a sample of your target group to ensure that the sample form instrument is measuring what it intends to measure and is appropriate for the target group.

Questions within the sample form can be asked in several ways and include: closed questions, open-ended and scaled questions, and multiple choice questions. Closed questions are usually in the format of yes/no or true/false options. Open-ended questions on the other hand leave the answer entirely up to the respondent and therefore provide a greater range of responses. Additionally, the use of scales is useful when assessing participants' attitudes. A multiple choice question may ask respondents to indicate their favourite topic covered in the program, or most preferred activity. Other considerations when developing a sample form instrument include: question sequence, layout and appearance, length, language, and an introduction and cover letter. Sensitive questions should be placed near the end of a sample form rather than at the beginning.

Offering young people an incentive for completing the sample form or embedding the sample form as a compulsory item within the program schedule or curriculum may be useful to maximise the response rate.

Sample Volume (Size)

Sample size determination is the act of choosing the number of observations or replicates to include in a statistical sample. The sample size is an important feature of any empirical study in which the goal is to make inferences about a population from a sample. In practice, the sample size used in a study is determined based on the expense of data collection, and the need to have sufficient statistical power.



Quiz

Volkswagen wants to gather information on customer satisfaction with their car service centre. Outline two methods this information could be gathered. Outline the methods and the advantages and disadvantages of each method for this case.

References

https://blog.socialcops.com/academy/resources/4-data-collection-techniques-ones-right/

https://media.licdn.com/mpr/mpr/AAEAAQAAAAAAAAVVAAAAJDQwZDkyMGU0LWQyYmYtNGE2MC1hYzM0LWQwYjExZTEwNzFjOA.jpg

http://mypeer.org.au/monitoring-evaluation/data-collection-methods/

https://en.wikipedia.org/wiki/Sample_size_determination

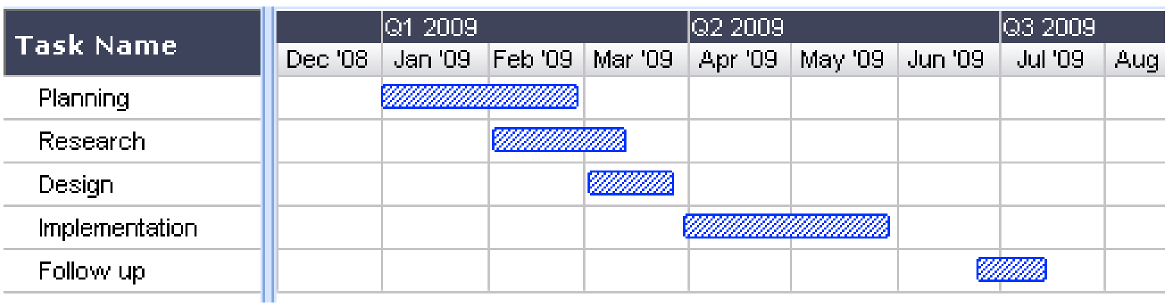
Gantt Charts

http://www.gantt.com/ (Yes there is a webpage just about Gantt charts)

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale. Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. This allows you to see at a glance:

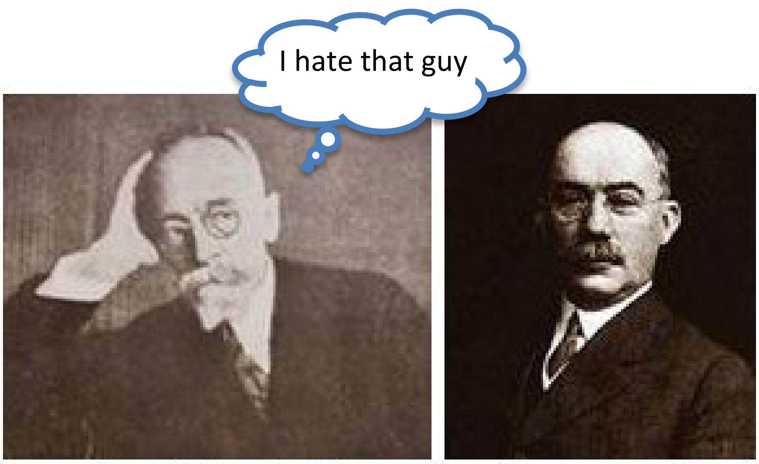
- What the various activities are
- When each activity begins and ends
- How long each activity is scheduled to last
- Where activities overlap with other activities, and by how much
- The start and end date of the whole project

To summarize, a Gantt chart shows you what has to be done (the activities) and when (the schedule).



Gantt Chart History

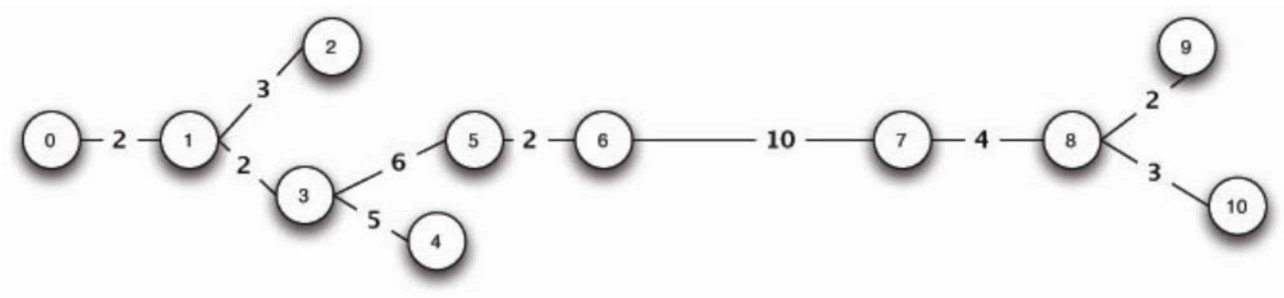
The first Gantt chart was devised in the mid 1890s by Karol Adamiecki, a Polish engineer who ran a steelworks in southern Poland and had become interested in management ideas and techniques. Some 15 years after Adamiecki, Henry Gantt, an American engineer and management consultant, devised his own version of the chart and it was this that became widely known and popular in western countries. Consequently it was Henry Gantt whose name was to become associated with charts of this type due mainly to the difficulty in spelling Adamiecki. I may have made that last bit up.



Karol Adamiecki Henry Gantt

Program Evaluation Review Technique (PERT)

A Program, Evaluation, and Review Technique (PERT) analysis is used to determine a realistic duration for tasks, by taking into account optimistic, expected, and pessimistic duration estimates.



A PERT chart is a project management tool used to schedule, organize, and coordinate tasks within a project. PERT stands for Program Evaluation Review Technique, a methodology developed by the U.S. Navy in the 1950s to manage the Polaris submarine missile program. A similar methodology, the Critical Path Method (CPM) was developed for project management in the private sector at about the same time.

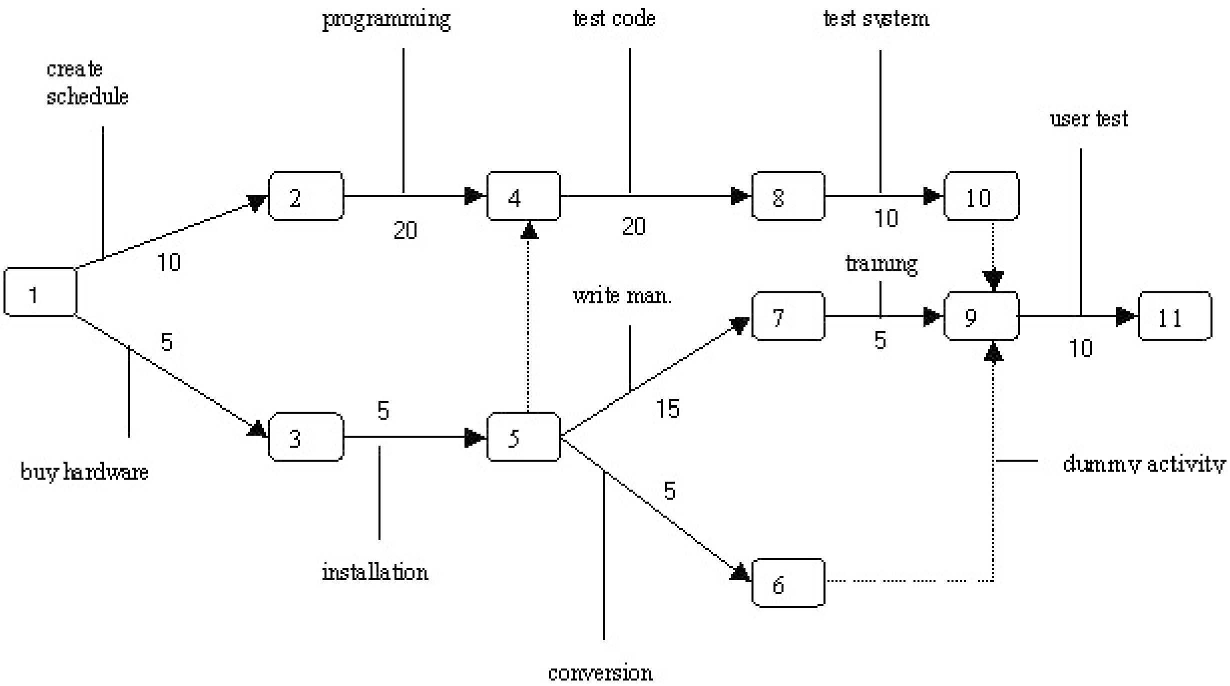


Fig. 1:
PERT Chart

* Numbered rectangles are nodes and represent events or milestones.
* Directional arrows represent dependent tasks that must be completed sequentially.
* Diverging arrow directions (e.g. 1-2 & 1-3) indicate possibly concurrent tasks
* Dotted lines indicate dependent tasks that do not require resources.

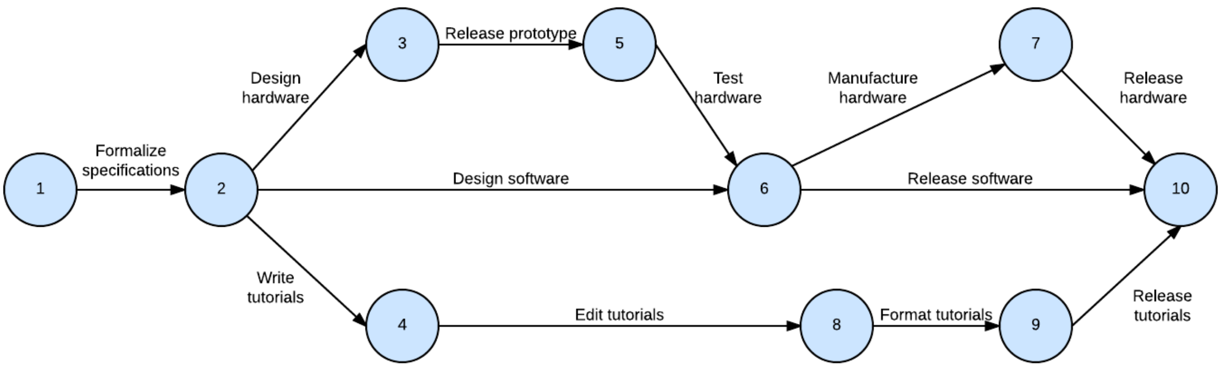
A PERT chart presents a graphic illustration of a project as a network diagram consisting of numbered nodes (either circles or rectangles) representing events, or milestones in the project linked by labelled vectors (directional lines) representing tasks in the project. The direction of the arrows on the lines indicates the sequence of tasks. In the diagram, for example, the tasks between nodes 1, 2, 4, 8, and 10 must be completed in sequence. These are called dependent or serial tasks. The tasks between nodes 1 and 2, and nodes 1 and 3 are not dependent on the completion of one to start the other and can be undertaken simultaneously. These tasks are called parallel or concurrent tasks. Tasks that must be completed in sequence but that don't require resources or completion time are considered to have event dependency. These are represented by dotted lines with arrows and are called dummy activities. For example, the dashed arrow linking nodes 6 and 9 indicates that the system files must be converted before the user test can take place, but that the resources and time required to prepare for the user test (writing the user manual and user training) are on another path. Numbers on the opposite sides of the vectors indicate the time allotted for the task.

The PERT chart is sometimes preferred over the Gantt chart, another popular project management charting method, because it clearly illustrates task dependencies. On the other hand, the PERT chart can be much more difficult to interpret, especially on complex projects. Frequently, project managers use both techniques.

How to draw a PERT chart

You can design your PERT chart in a few different ways. The traditional PERT chart (also known as an activity-on-arrow diagram) contains two different elements:

- **Nodes** represent events or milestones in your project. You can use either numbered circles or numbered boxes.
- **Arrows** represent tasks. The direction of the arrows shows the sequence of tasks. Diverging arrows indicate that you can complete those tasks concurrently. In the example below, tasks 1, 2, 4, and 6 have to be completed in order.



Now the activity-on-node diagram, a variation of the original PERT chart, has become more common. The nodes represent activities, and the arrows simply show dependencies and relationships between tasks. In this version, the nodes are usually small charts (the ERD shapes in Lucidchart work well for this purpose). These charts show the task, the date you anticipate starting the task, the time it’ll take to complete the task, and finally the date you anticipate finishing the task.

Regardless of which structure you follow, go through these steps to gather information and build your PERT chart:

1. List the activities involved in the project. Break your project down into more manageable tasks. Limit your list to only high-level activities; otherwise, you may have trouble managing the resulting PERT chart.

   Note: You may want to start with a work breakdown structure to determine all the tasks required to complete a project and then create a PERT chart to arrange those tasks and predict milestone and completion dates.

2. Consider dependencies. For every activity, ask yourself whether the task depends on the completion of another task. This step will determine the order of nodes on your PERT chart.

3. Place nodes and arrows based on the information you have gathered. For activity-on-arrow diagrams, number a node for each milestone in the project. These milestones are generally based on the tasks you wrote down, such as "Project start," "Hardware designed," "Tutorials drafted," etc. If you'd like to keep track of these milestones, you can create a numbered list to accompany the diagram. For activity-on-node diagrams, create a node for each task and write the task directly on the node.

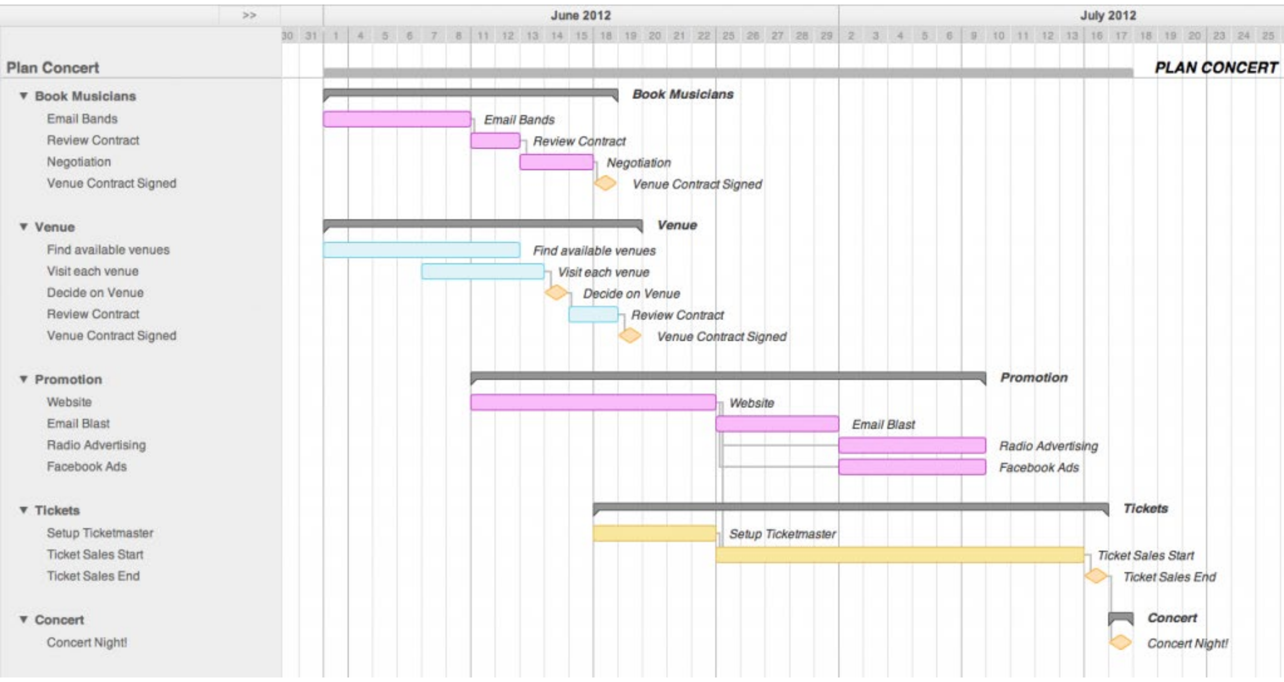
   Organize these nodes based on the dependencies you found. If activities need to be completed in sequence, place them in the right order and connect with arrows. You can use a diverging arrow to show tasks that do not depend on other tasks. For activity-on-arrow diagrams, once you connect your nodes, you'll write the activity name on the arrow.

4. Add the completion time for each activity. Estimate the amount of time you and your team need to complete each task. Write that amount under the task name. If you're not confident in your estimation, experts recommend a three-point method. For each task, write down:
   - An optimistic time estimate (o), or the shortest possible time the task will take
   - A most-likely time estimate (m),
   - A pessimistic time estimate (p), or the longest possible time if the task takes longer than expected

Depending on the style of your PERT chart, you can add expected start and finish dates for each tasks. And that's it!

Quiz

1. Make a PERT chart from the Gantt chart below.



2. Make a PERT chart from the Gantt chart below.

Number	Task	Start	End	Duration	6 / 8	13 / 8	20 / 8	27 / 8	3 / 9	10 / 9	17 / 9	24 / 9
1	Project Initiated	7 / 8 / 2012	9 / 8 / 2012	1								
2	Research / Brainstorming	10 / 8 / 2012	16 / 8 / 2012	5								
3	Initial Designs	10 / 8 / 2012	21 / 8 / 2012	10								
4	User Surveys / QA	17 / 8 / 2012	23 / 8 / 2012	5								
5	Analysis	22 / 8 / 2012	31 / 8 / 2012	8								
6	First Design Release to Dept	1 / 9 / 2012	7 / 9 / 2012	5								
7	Final Designs	8 / 9 / 2012	13 / 9 / 2012	4								
8	Final Design Release to Dept	14 / 9 / 2012	20 / 9 / 2012	5								
9	Implementation	21 / 9 / 2012	26 / 9 / 2012	4								
10	User Training	21 / 9 / 2012	25 / 9 / 2012	3								

References

http://www.gantt.com/ (Yes there is a webpage just about Gantt charts)
http://searchsoftwarequality.techtarget.com/definition/PERT-chart
https://www.lucidchart.com/blog/advantages-of-pert-charts-vs-gantt-charts

Gantt Chart

A Gantt chart is a horizontal bar chart developed as a production control tool in 1917 by Henry L. Gantt, an American engineer and social scientist. Frequently used in project management, a Gantt chart provides a graphical illustration of a schedule that helps to plan, coordinate, and track specific tasks in a project

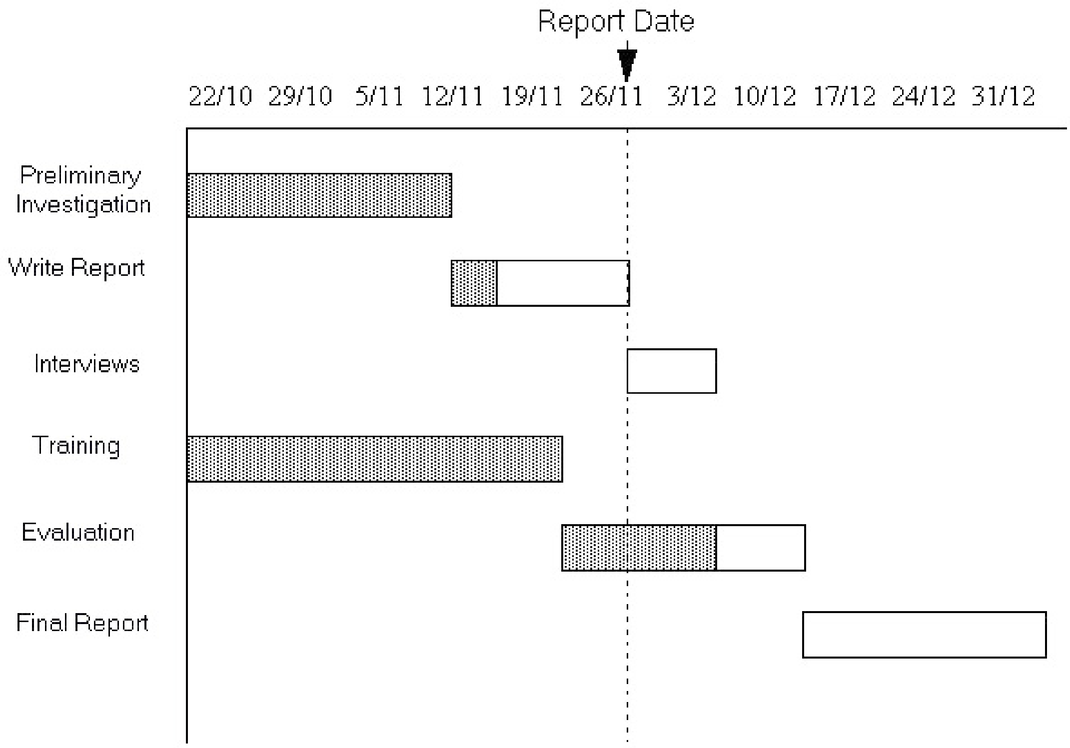


Figure 1: Gantt Chart

Gantt charts may be simple versions created on graph paper or more complex automated versions created using project management applications such as Microsoft Project or Excel.

A Gantt chart is constructed with a horizontal axis representing the total time span of the project, broken down into increments (for example, days, weeks, or months) and a vertical axis representing the tasks that make up the project (for example, if the project is outfitting your computer with new software, the major tasks involved might be: conduct research, choose software, install software). Horizontal bars of varying lengths represent the sequences, timing, and time span for each task. Using the same example, you would put "conduct research" at the top of the vertical axis and draw a bar on the graph that represents the amount of time you expect to spend on the research, and then enter the other tasks below the first one and representative bars at the points in time when you expect to undertake them. The bar spans may overlap, as, for example, you may conduct research and choose software during the same time span. As the project progresses, secondary bars, arrowheads, or darkened bars may be added to indicate completed tasks, or the portions of tasks that have been completed. A vertical line is used to represent the report date.

Gantt charts give a clear illustration of project status, but one problem with them is that they don't indicate task dependencies - you cannot tell how one task falling behind schedule affects other tasks. The PERT chart, another popular project management charting method, is designed to do this. Automated Gantt charts store more information about tasks, such as the individuals assigned to specific tasks, and notes about the procedures. They also offer the benefit of being easy to change, which is helpful. Charts may be adjusted frequently to reflect the actual status of project tasks as, almost inevitably, they diverge from the original plan.

Gantt Chart Benefits

Gantt charts are useful for planning and scheduling projects. They help you assess how long a project should take, determine the resources needed, and plan the order in which you'll complete tasks. They're also helpful for managing the dependencies between tasks.

- Clarity. One of the biggest benefits of a Gantt chart is the tool's ability to boil down multiple tasks and timelines into a single document.
- Communication.
- Motivation
- Coordination
- Creativity
- Time Management. ...
- Flexibility. ...
- Manageability

PERT chart (Program Evaluation Review Technique)

A PERT chart is a project management tool used to schedule, organize, and coordinate tasks within a project. PERT stands for *Program Evaluation Review Technique*, a methodology developed by the U.S. Navy in the 1950s to manage the Polaris submarine missile program. A similar methodology, the *Critical Path Method* (CPM) was developed for project management in the private sector at about the same time.

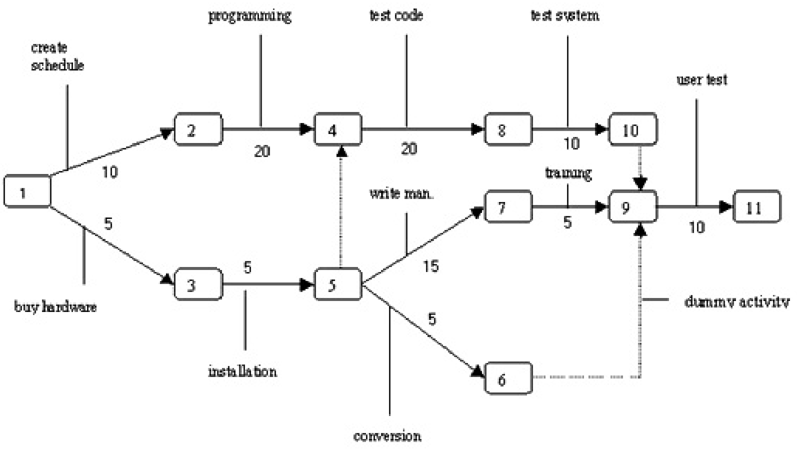


Fig. 1:
PERT Chart

* Numbered rectangles are nodes and represent events or milestones.
* Directional arrows represent dependent tasks that must be completed sequentially.
* Diverging arrow directions (e.g. 1-2 & 1-3) indicate possibly concurrent tasks
* Dotted lines indicate dependent tasks that do not require resources.

A PERT chart presents a graphic illustration of a project as a network diagram consisting of numbered *nodes* (either circles or rectangles) representing events, or milestones in the project linked by labelled *vectors* (directional lines) representing tasks in the project. The direction of the arrows on the lines indicates the sequence of tasks. In the diagram, for example, the tasks between nodes 1, 2, 4, 8, and 10 must be completed in sequence. These are called *dependent* or *serial* tasks. The tasks between nodes 1 and 2, and nodes 1 and 3 are not dependent on the completion of one to start the other and can be undertaken simultaneously. These tasks are called *parallel* or *concurrent* tasks. Tasks that must be completed in sequence but that don't require resources or completion time are considered to have *event dependency*. These are represented by dotted lines with arrows and are called *dummy activities*. For example, the dashed arrow linking nodes 6 and 9 indicates that the system files must be converted before the user test can take place, but that the resources and time required to prepare for the user test (writing the user manual and user training) are on another path. Numbers on the opposite sides of the vectors indicate the time allotted for the task.

The PERT chart is sometimes preferred over the Gantt chart, another popular project management charting method, because it clearly illustrates task dependencies. On the other hand, the PERT chart can be much more difficult to interpret, especially on complex projects. Frequently, project managers use both techniques.

PERT charts are generally used before a project begins to plan and determine the duration of each task—so they don't have to show the actual dates of your project. They also do a better job of showing whether certain tasks need to be completed in order or whether they can be completed simultaneously. Use a PERT chart if you need to:

- Show the interdependency of certain tasks.
- Anticipate the amount of time it'll take to complete a project.
- Determine the critical path to meet your deadlines.
- Plan for large or more complex projects.

The Wizzo project has two phases. Each phase consists of three main tasks:analysis, coding and test, which must be performed sequentially. Analysis forphase two can begin immediately on the completion of analysis for phase one,but coding for phase two must await satisfactory testing of the phase one code. Analysis is expected to take 3 weeks for phase one, and 4 weeks for phase two,while coding is expected to take 2 weeks for phase one and 3 weeks for phase two. Testing is expected to take 1 week for both phases.

a) Draw the PERT and GANTT charts for the Wizzo project, and define the critical path. [5 marks]
b) During testing of phase one, a serious bug is found resulting in 2 weeks of extra work being required in that phase. How will this affect the overall timescale and critical path of the project? [5 marks]
c) An extra programmer is assigned to the project in week 6, potentially increasing code productivity by 50%, but first requiring 1 week's training by the project analyst. Will this allow the project to finish early, with or without the event described in (b) above? [5 marks]
d) What other tasks would you expect to be under the control of the Wizzo project manager and be included in a typical software project? [5 marks]

Introduction to Data Flow Diagrams

There are various levels of detail a Data Flow Diagram (DFD) can take. In this course you have discriminate between Context DFDs, Level 0 and Level 1 DFDs and be able to draw DFDs at each of these levels.

Context DFD

A **Context DFD** sets the scene for the system and shows it data inputs and out puts, by showing the Sources and Sinks of data it also sets the environmental boundaries and scope for the system. Context DFD's:

- One process called process ZERO (0) which represents the entire system.
- Do not show stores as these are within process zero.
- Show data flows into and out of the system.

Level Zero DFD

The analyst now thinks about what processes are contained within process Zero and this provides more detail on the system. Level Zero shows the data stores, data flows and the sinks and sources. It must be consistent with the Context DFD, so no new data inputs and outputs should appear and no new Sinks/Sources.

You need to keep in mind the **System Boundary.** The System Boundary defines the scope and limits or boundaries of the system. You wouldn't include data stores for example the exist outside the system. So for example if your organisation works with a bank that bank would have data stores with information relevant to but outside of the system so they would not be included.

The main processes you should look for usually fall into the following categories:

1. Data capture from customers.
2. Maintaining (updating) data stores.
3. Producing data to sinks in the form of reports or data transfers.
4. Hi Level data transformations – ie from a burger order to kitchen instruction.

Each process in a level zero DFD ends with .0 ie 1.0 2.0…x.0

These are the highest level processes in the system and can (inevitably will be) be broken down to lowers levels that show increasing levels of detail for each process. This continues until one process cannot be decomposed any further. A decomposition to Level One of a DFD process labelled 1.0 would result in processes labelled 1.1, 1.2 1.3, 1.4 1.5 1.6 etc. As a rule of thumb about 7 processes is the limit, anymore and it's too complex.

Tips

Completeness

Are all components present to model the system? Are all your stores, sinks, processes connected? Do your data flows go somewhere? Do your sinks get their data?

Consistency

Are the lower levels consistent with higher ones – have you created a Level One process that is not owned by a level zero process? Have you created a new data flow from thin air?

Timing

Draw DFDs as if the system has never started and will never stop. Timing is not something DFDs show…

Iterative

Do them over and over again, improving as you go.

Primitive DFDs – or When Do I stop????

Well, somewhat cynically, you stop when you get to level 1- the course does not require you to go further. However the following are guidelines of when to stop:

- When a process represents a single calculation/ decision/ Db operation CRUD.
- When a data store represents data for a single entity.
- When data flows no longer need to be split to show how data is handled.
- When each transaction is shown as a single flow.

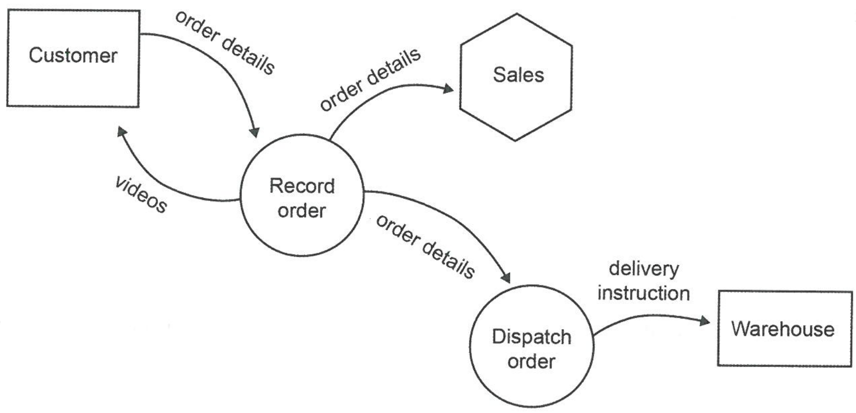
Errors in DFDs – see the attached exercise on black holes and immaculate conceptions.

There are a stack of rules for drawing DFDs and the attached exercise will walk you the basic ones.

Sample Exam Question

Question 7 **(4 marks)**

Consider the Level 0 Data Flow Diagram (DFD) below, modelling an online ordering system.



(a) Name a process in this diagram. (1 mark)

(b) Name an entity in this diagram. (1 mark)

(c) An analyst has looked at this DFD and believes that it has **not** been drawn correctly. Identify **two** errors in the diagram. (2 marks)

Context Diagram

http://www.bawiki.com/wiki/techniques/context-diagram/

A Context Diagram is a common tool that Business Analysts use to understand the context of an entity being examined. Most descriptions of a Context Diagram limit this entity to a system that is being created or modified as part of a project, but the Context Diagram can also be applied to other entities.

There are at least three different specifications for creating Context Diagrams (that I can find at this time). They are: Gane-Sarson, Yourdon-DeMarco, and TOGAF. The one referred to in the syllabus is Yourdon-DeMarco so this in the one we will cover.

Yourdon-DeMarco

Context Diagrams in Yourdon-DeMarco styles generally consist of just four (4) standard elements. These are:

The Process

The Process (or system, or business entity, etc) being investigated. There should only be one process per Context diagram and it is generally displayed in the centre of the diagram. The Process contains the name of the process or entity being investigated.

- In the Yourdon-DeMarco style the process is represented as a circle.

Data Stores

Data Stores are databases that are either created by the Process under review and used by outside parties, or created by outside parties and used by the Process.

- In the Yourdon-DeMarco style, a Data Store is represented by two parallel lines (essentially an open-sided long, flat rectangle)

Note that data stores/databases are not included in context diagrams. If the database is part of an external system, you would show the overall system, not the database.

Entities (or Actors)

Entities, are the parties that communicate either directly with the Process, or indirectly with the Process through an intermediary Data Store. According to Yourdon, in a Context Diagram the Entities should not be shown as communicating directly with each other.

- In the Yourdon-DeMarco style, an Entity is displayed as a rectangle.

Flows

Flows (or Relationships) represent data or events flowing between the three other components above. Flows are labelled and can be displayed as unidirectional or bi-directional.

- In Yourdon-DeMarco styles, a Flow is displayed as a solid-line with an directional arrow indicator at either or both ends.

Why do it?

In general, a Context Diagram can be used to examine:

- An existing system (to help determine what users and entities it interacts with)
- Stakeholder Identification
- A conceptual solution system (to help determine what users and entities it will interact with)
- A business unit (to help in understanding what entities a business unit interacts with)
- A problem space (to help identify the entities impacted by the problem or who help cause the problem)
- A project (to help identify the organizations, functions, services, processes, applications, data, and technology that will be added, removed, or impacted by the project)

Some more specific uses include:

System Context

As indicated above, most Context Diagrams are focused on identifying the context of a system.

- This can be a very good early step to help identify scope and potential stakeholders for a project involving that system.
  - A limitation for Stakeholder identification is that Context Diagram will generally only identify the stakeholders who directly interact with the system, and may lead you to miss stakeholders further away down the process change who might be significantly impacted by changes.
  - The Context Diagram can help with scope in identifying the specific events or data flows that are in scope. By creating a Context Diagram identifying all entities and data flows that are in scope, you can limit future scope creep or protect yourself from "how did you miss the XXX report requirements?!" if that report is not captured in the Context Diagram.
- The system context diagram can be improved over time with more detail as additional information is learned. For example, if part of the interaction of the system with an external entity takes place via an API, or email, this can be captured in the diagram. This identification of specific interfaces and methods then supports interface analysis efforts and similar system detailed analysis.

Business Knowledge

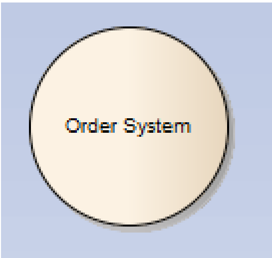
A great use for Context Diagram's comes up when BA's are first starting to work with a particular business unit or client. By starting with a Context Diagram that shows the standard interactions and relationships for that business unit, the BA can learn a great deal about the business and identify specific processes or use cases that merit further exploration.

How do I do it?

In the following steps I will demonstrate the creation of a Context Diagram for a simplified web-based order system using the Yourdon-DeMarco format, created with Enterprise Architect.

Step 1

The first step is the Process (or business entity, or in this case a system) that you are focusing on to the diagram. There should only be one Process per context diagram and this should be your first action. Note however, that the subsequent steps below can be done in whatever order you like.

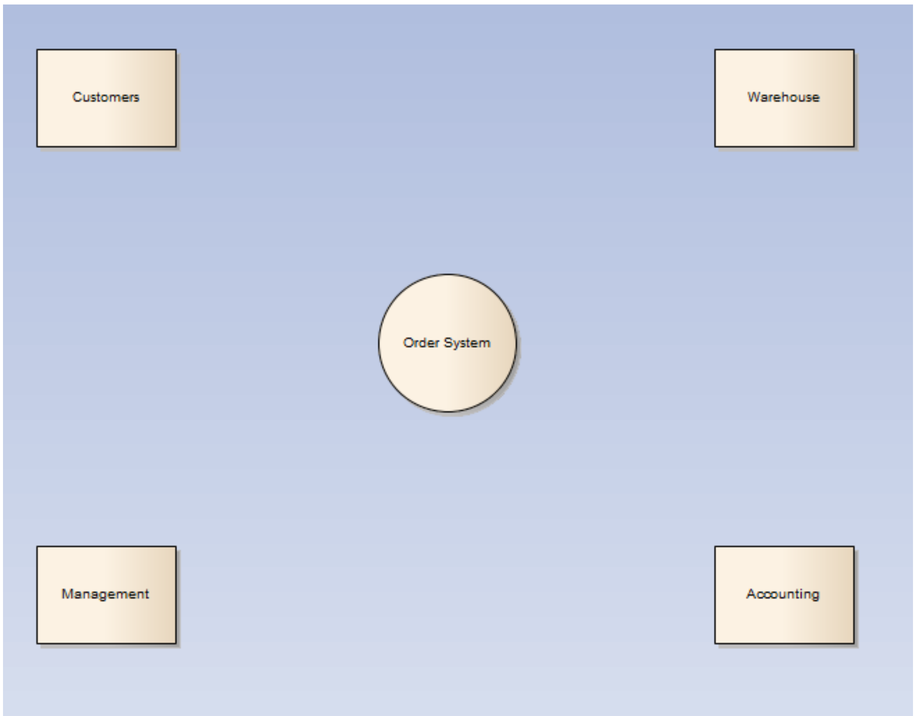


Step 2

For the second step, I am going to add the Entities. To start off, I can think of four entities who would interact with the Order system. They are:

- Customers
- Management
- Accounting
- Warehouse Operations

The updated diagram now looks like this:

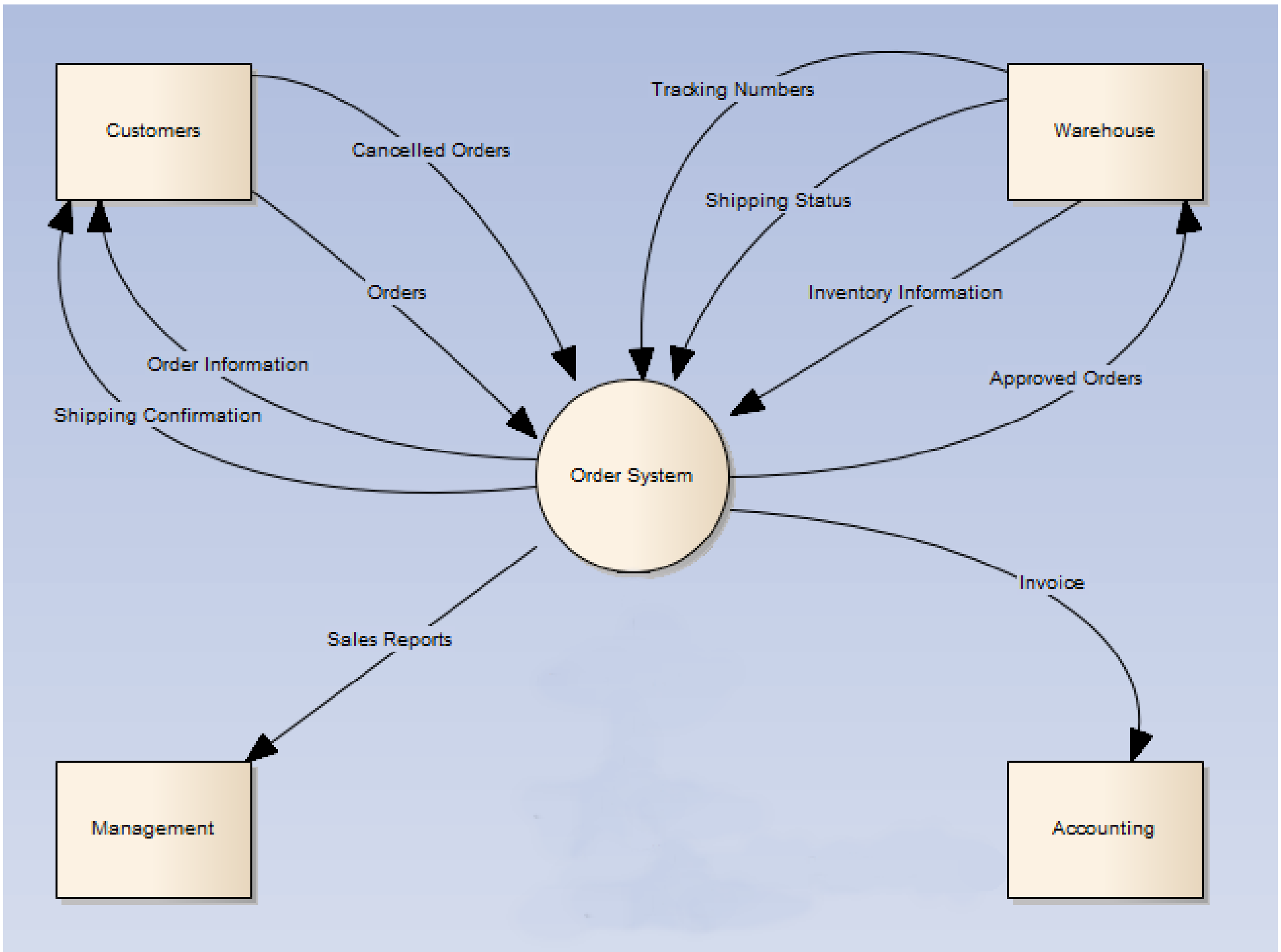


Step 3

Now that the entities and data stores have been defines, the next step is show the interactions between them. If I was working with stakeholders to determine the interactions of an Order system, I might find the following common interactions and data flows between the entities diagrammed in Step 3.

- Customers:
  - Place orders
  - Cancel orders
  - Receive order information
  - Receive a shipping confirmation with a tracking number when their order ships
- Accounting:
  - Receives an invoice when a customer places an order
  - Sends a transaction request to the transaction processor for the payment method and amount of the customers order
- The Transaction Processor:
  - Sends a status message to the order system indicating if the customers order has been successfully paid for
- The Warehouse:
  - Provides inventory information to the order system, so that customers know if what they want to order is in stock
  - Receives approved orders
  - Provides a shipping status that is part of the customers order information
  - Provides a tracking # when the customers order actually ships
- Management:
  - Receives sales reports so that they can manage the business

With all of those interactions captures, the updated diagram now looks like this:



What Should the Results be?

From a Business Analysis perspective, the context diagram is mostly used to help determine the scope of the effort that is being undertaken. If for example you were beginning an initiative to improve the Order System itself or the order processing process in the business case above, you could work with your stakeholders to agree that only the items in the diagram are in scope.

For example, if the Warehouse later wanted to add a capability to auto-order certain products when inventory reached a certain point, you could point out that interactions with suppliers are not in the diagram above and thus not in scope.

The entities, systems, and flows captured in a context diagram also provide a starting point for Use Case Diagrams, process flow diagrams, class diagrams, and similar efforts to capture more detailed knowledge.

Risks

As always with diagrams, there are risks in abstracting too much from reality (leaving important things out), or focusing so much on trying to include everything that the diagram loses part of its value by being too complex.

Tip

Yourdon indicates that Entities should never communicate directly with each other in a Context Diagram, even if they do so in reality.

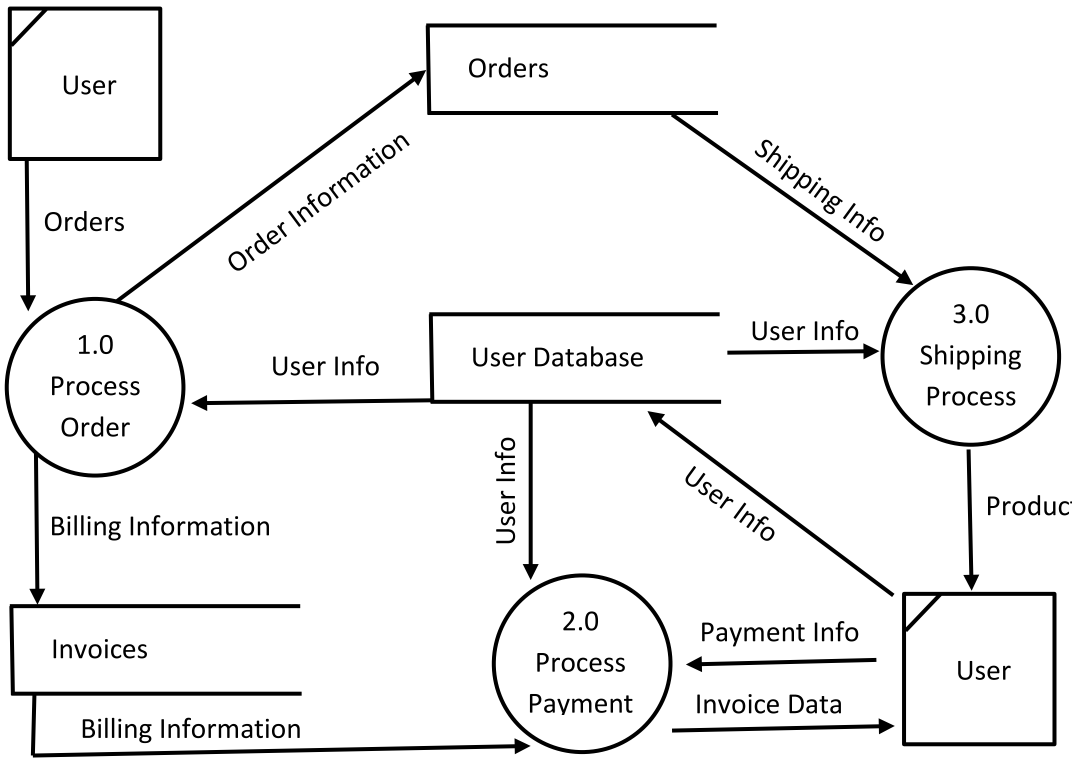
Data Flow Diagrams

https://www.lucidchart.com/pages/data-flow-diagram

This guide provides everything you need to know about data flow diagrams, including definitions, and symbols and notations. You'll learn the different levels of a DFD, the difference between a logical and a physical DFD and tips for making a DFD.

What is a data flow diagram?

A data flow diagram (DFD) maps out the flow of information for any process or system. A data flow diagram shows how data is processed within a system based on inputs and outputs. Visual symbols are used to represent the flow of information, data sources and destinations, and where data is stored. Data flow diagrams are often used as a first step toward redesigning a system. They provide a graphical representation of a system at any level of detail, creating an easy-to-understand picture of what the system does. A general overview of a system is represented with a context diagram, also known as a level 0 DFD, which shows a system as a single process. A level 1 diagram provides greater detail, focusing on a system's main functions. Diagrams that are level 2 or higher illustrate a system's functioning with increasing detail. It's rare for a DFD to go beyond level 2 because of the increasing complexity, which makes it less effective as a communication tool.

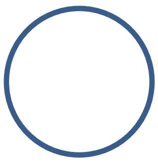


Symbols and Notations Used in DFDs

Using any convention's DFD rules or guidelines, the symbols depict the four components of data flow diagrams.

- **External entity:** an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram. External entities canbe are any organisation or person that provides data to the system or receives data from the system.
  - They exist outside of the system.
  - An external entity can be both a source and a sink.
  - They should be named in the singular as a person, place or thing.

- **Process:** any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules. A short label is used to describe the process, such as "Submit payment." These are actions taking place that transform inputs into outputs.
  - They must always have at least one inflow and one outflow.
  - They should be named with an active verb associated with a noun or very short phrases of that type, reflecting what transformation the process is making to the data passing through it.
  - The numbering of a process does not indicate timing or sequence.
  - The data flowing out of a process should differ from that going in. (e.g. payment_cheque_details goes in and cancelled_cheque_details comes out of an enter cheque transaction process.)



- **Data store:** files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as "Orders." These store data used within a system.
  - They cannot transform data, and must usually contain at least one inflow and one outflow.
  - A data store's identifier should be a noun reflecting the data it contains and not its physical nature.

- **Data flow:** the route that data takes between the external entities, processes and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name, like "Billing details." These vectors indicate the data being transferred (not physical objects) e.g. invoice_details not invoice.
    - They should connect at each end directly to their source and destination with only one arrowhead.

Sometimes in order to simplify a diagram, an entity or data store requires duplication. Each of the duplicated objects should contain a diagonal line(s) in the bottom corner as shown below:

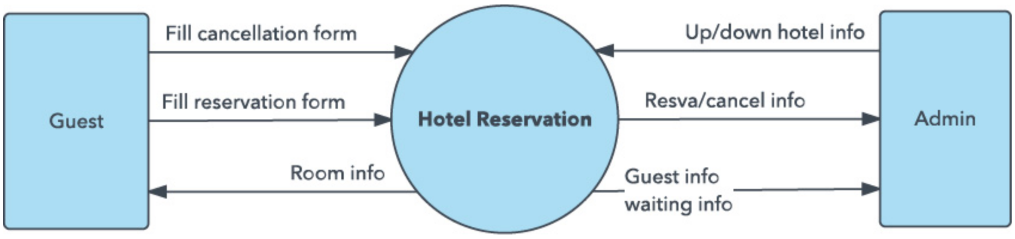
DFD Rules

- Each process should have at least one input and an output.
- Each data store should have at least one data flow in and one data flow out.
- Data flows must have a process at one end.
- Add details, info, or data to your data flow names.
- A DFD must be balanced - The data that flow into or out of a bubble must match the data flow at the next level of DFD.

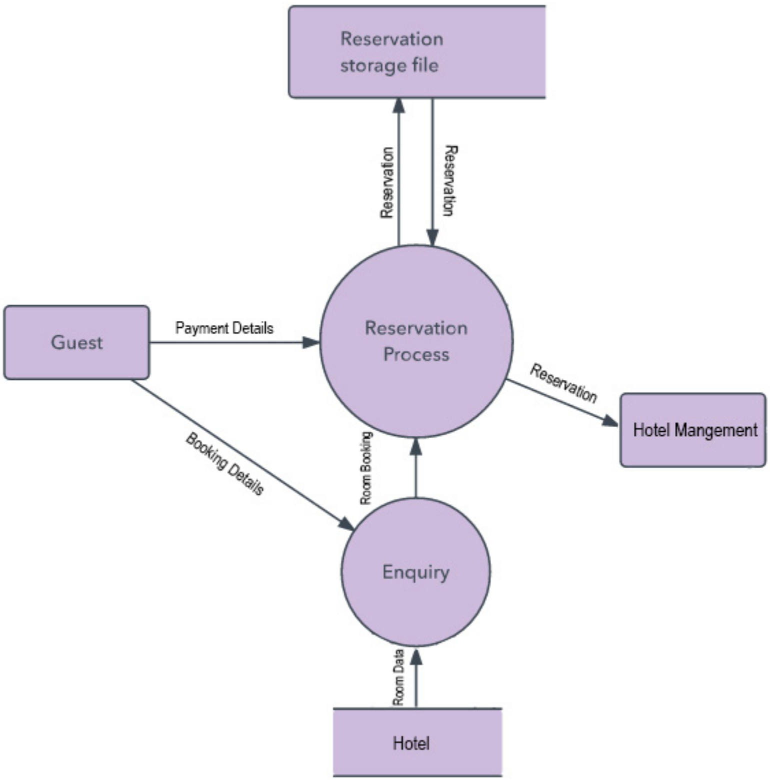
DFD levels and layers: From context diagrams to pseudocode

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

- Context Diagram. It's a basic overview of the whole system or process being analysed or modelled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.



- DFD Level 0 provides a more detailed breakout of pieces of the Context Level Diagram. A level 0 data flow diagram should show all external entities. The processes are numbered, but do not indicate sequence.
- In the level 0 data flow diagram, the same total number of inflows and outflows (and external entities) must exist as in the context diagram.



Documentation

User Documentation

User Documentation - helps the user USE the system that has been built. So explains how to do data entry, operate the Software, get output and what the output means.

System Documentation

SYS Documentation explains how the system was built for other programmers so they can modify/ update/ fix it later as well as the Update Strategy. It includes the data dictionary.

Activities

1. Explain the function and purpose of a Data Flow Diagram.
2. Why are DFD’s known as Process Models? Draw a diagram showing the 4 components of any DFD.
3. List the naming conventions for all of the components of a DFD.
4. What activity is represented by a Process Bubble in a DFD.
5. List some examples of the types of things a Process Bubble will achieve.
6. How would a database be represented in a DFD?
7. For each of the other components of a DFD explain their function and purpose.
8. Some objects in a DFD can NEVER be linked, what are they and why can they not be linked.
9. Draw at least 2 diagrams of a DFD that would not make any sense. Use only 1 process bubble and 1 vector.
10. DFD’s are said to be timeless, what does this mean? Correct the attached document.
11. What is an entity in a DFD?
12. Give five rules for DFDs.

CSC 12 ATAR Unit 3 Process Modeling – Data Flow Diagrams

Types of DFD
The point of modeling the current system is to understand what is going on and if it will change or how it needs to change, detailed diagrams of current system have limited value. Focus should be on the new logical system. This is an important part of Requirements Determination – which is the main goal of the Analysis Stage of the SDLC. At the end of Analysis, we should have a New Logical Model (DFD), Some alternatives and a set of Structured Requirements.

Levels of Data Flow Diagrams
Context – shows the whole system as a single process
Level 0 - Just to make life difficult some define Level 0 as being a context diagram but for this course - a level 1 DFD notates each of the main sub-processes that together form the complete system.
Level 1 – Is just a further breakdown of the processes in Level 0 diagram.

Level-0 diagrams
A level-0 diagram represents a system's major processes, data flows and data stores at a high level of detail.

- each process will be labeled as 1.0, 2.0 etc

Context diagram
Then:

- Level 0 DFD (top level; overview DFD)
- Level -n diagram (Middle level DFD(s))
- Primitive DFDs
- Each level shows an 'explosion' of the process at the level above. Thus the context DFD explodes into processes 1, 2, 3...; process 2 explodes into 2.1, 2.2… Process 2.2 explodes into 2.2.1, 2.2.2...number of decimal points tells you the level you're at
- you will probably need all the levels to describe a complex system, in simpler systems you may be able to omit middle level and maybe overview DFDs

How to draw a DFD - 1

1. Read and re-read the narrative that describes the process you wish to diagram.
2. Identify the **nouns** which describe objects that interact with the system.
3. Identify **verbs** that describe the activities performed by the objects.
4. **Match** the **nouns** and **verbs**. Identify nouns that do not perform any information processing activities for the system.
5. Generally, non-information processing activities relate to sending and receiving data to things or people – Banks, Customers. These are the external entities.
   In contrast, information processing activities relate to entering, filing, preparing, classifying, sorting, verifying, etc.....
6. Draw the **context** diagram using 1 circle and rectangles for the external entities.
   Draw the data flow lines.
   Label the process 0 and data flows.
7. Draw the level 0 diagram.
   a. Group like activities related to information processing.
   b. Determine suitable name describing the process.
   c. Draw bubbles for each level 0 process. Include the external entities and their data flows. Connect level 0 processes with data flow lines.
   d. Label new data flow lines.
   e. Include data files if needed to accurately describe processing.
   f. Verify that the level 0 diagram balances with the context diagram.

Balancing DFDs

The inputs and outputs to processes must be conserved when decomposing to lower levels.
Balancing means ensuring no details get lost when a process on one DFD is exploded to a more detailed DFD - so levels are consistent
All flows to and from a 'parent' process must be carried down to the next level
Check when you have exploded a process that the input and output flows are the same as to the higher level process. If you find you have 'lost' a data flow, add it in.
If you discover a new data flow at a lower level, amend the higher level DFDs to include it.
Exploding a DFD should provide more detail - not completely new information.

When to stop decomposing DFDs

- when each process has been reduced to a single operation, decision or calculation
- when each data store represents one type of entity e.g. customer, product …
- when each business form, report, transaction or display is shown as one flow
- when you have enough detail to begin development or when everyone has had enough!

Final points about DFDs

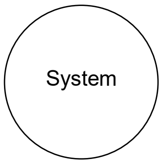
- they do not represent time well - use other techniques (e.g. state transition diagrams)
- be prepared to revise DFDs a few times
- need more accuracy for design than analysis
- CASE tools help maintain consistency
- an important part of structured analysis
- help postpone technology commitments
- Using DFDs in analysis,
    - structuring requirements and process modeling,
    - finding redundancy/ waste in processes
    - finding inefficiency - capturing and not using data

Year 11 Data Flow Diagrams

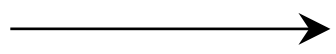
These conventions are based on the De Marko/Yourdan symbols.

Context diagram

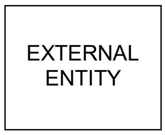
The context diagram is the top level of a set of hierarchically related diagrams that form a set that decomposes a system into successively finer detail with each move down the diagram set. This diagram represents the system being modelled as a single circle interacting with external entities. The emphasis of this diagram is to identify the boundary of the system. The name inside the single circle representing the system should describe the system being modelled. The symbols used are:



the system is represented as a circle

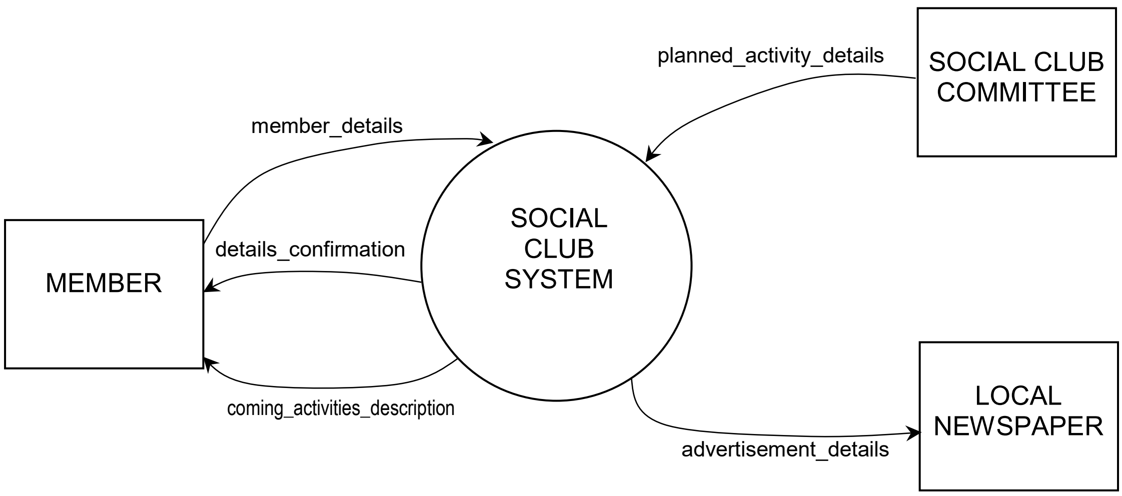


represents the flow of data between the system and the external entities



an organisation or person that provides data to the system or receives data from the system

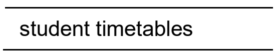
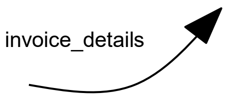
Context diagram for social club system



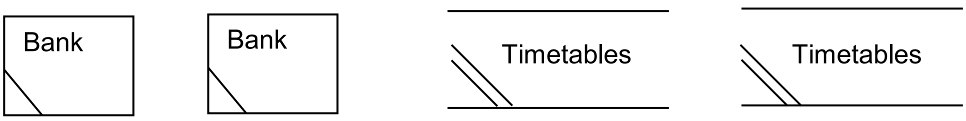
The circle is a representation of the system boundary. The system boundary defines what is inside and outside the system.

Deciding on which side objects lie is an important consideration. Is a particular object part of the system being considered, and hence invisible inside the circle, or is it really outside the system's considerations and therefore an external item supplying data, or taking information from the system?

Notice that data stores or files must never appear in a context diagram. They are part of the system and are therefore inside the circle.

	**External entities:** (sources or sinks): These are any organisation or person that provides data to the system or receives data from the system. ▪ They exist *outside* of the system. ▪ An external entity can be both a source and a sink. ▪ They should be named in the singular as a *person*, *place* or *thing*.
	**Processes:** These are actions taking place that *transform* inputs into outputs. ▪ They must always have at least one inflow and one outflow. ▪ They should be named with an *active verb* associated with a *noun* or very short phrases of that type, reflecting what transformation the process is making to the data passing through it. ▪ The numbering of a process does not indicate timing or sequence. ▪ The data flowing out of a process should differ from that going in. (e.g. **payment_cheque_details** goes in and **cancelled_cheque_details** comes out of an enter cheque transaction process.)
	**Data stores:** (files, repositories of data or temporary data stores) These store *data* used within a system. ▪ They cannot transform data, and must usually contain at least one inflow and one outflow. ▪ A data store's identifier should be a noun reflecting the data it contains and not its physical nature e.g. customer details NOT sorted magnetic tape file.
	**Data flows:** These vectors indicate the *data being transferred* (not physical objects) e.g. **invoice_details** not **invoice**. ▪ They should connect at each end directly to their source and destination with only one arrowhead.

Sometimes in order to simplify a diagram, an entity or data store requires duplication. Each of the duplicated objects should contain a diagonal line(s) in the bottom corner as shown below:

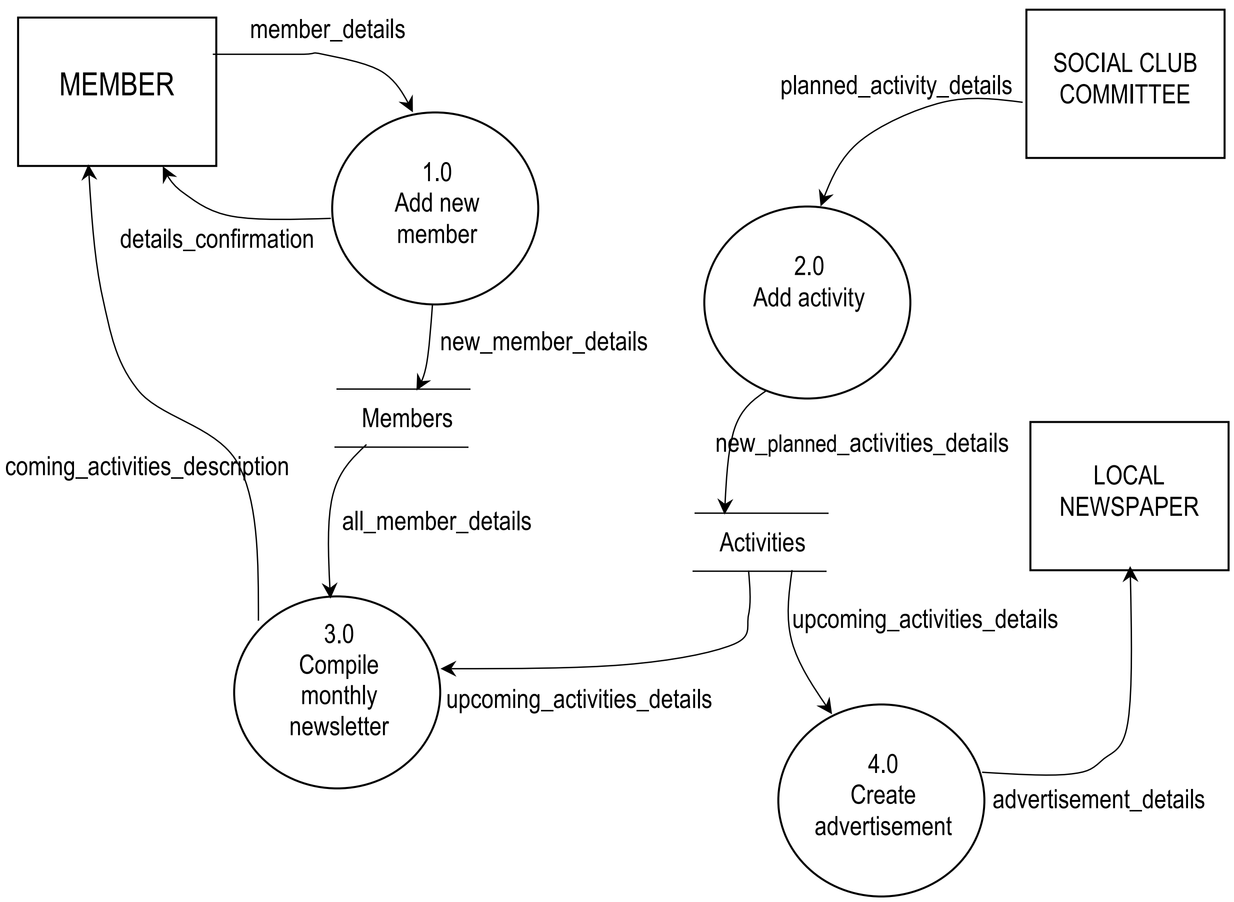


Levelled data flow diagrams

A level 0 data flow diagram should show all external entities. The processes are numbered, but do not indicate sequence.

In the level 0 data flow diagram, the same total number of inflows and outflows (and external entities) must exist as in the context diagram.

Level 0 DFD for the social club system



Any similar information that data flows carry are resolved in the data dictionary. The number of processes that are in the level 0 data flow diagram depend on the number of major processes described.

Year 12 levelled data flow diagrams

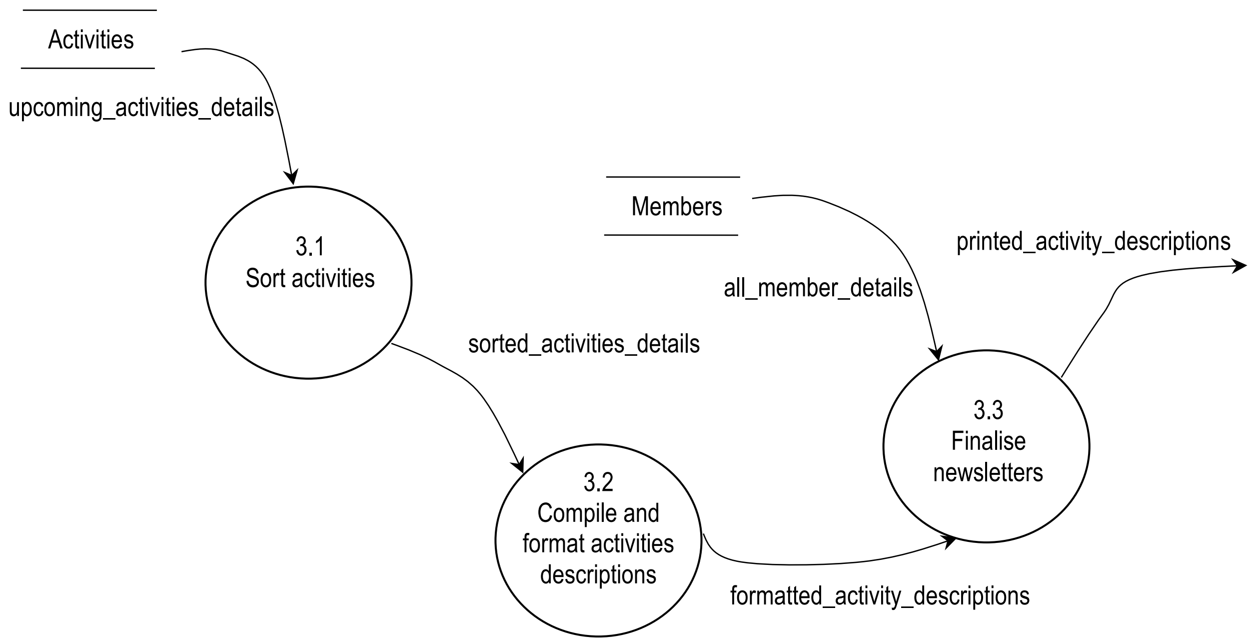
(Refer to syllabus content on p. 18)

Unit 3A build on from Unit 2A data flow diagrams.

Level 1 DFD for Process 3.0 Compile monthly newsletter

In the level 1 data flow diagram, the same total number of inflows and outflows must exist as in the level 0 diagram. External entities are not shown.

Process 3 can be expanded to show more detail.



Computer-Aided Software Engineering

What is CASE?

Computer-Aided Software Engineering (CASE) is the use of software tools to assist in the System Development Life Cycle. Tools used to assist in this way are known as CASE Tools.

What is CASE Tool?

A CASE tool is a computer-based product aimed at supporting one or more software engineering activities within a software development process.

Computer-Aided Software Engineering tools are those software which are used in any and all phases of developing an information system, including analysis, design and programming. For example, data dictionaries and diagramming tools aid in the analysis and design phases, while application generators speed up the programming phase.

CASE tools provide automated methods for designing and documenting traditional structured programming techniques. The ultimate goal of CASE is to provide a language for describing the overall system that is sufficient to generate all the necessary programs needed.

Components of CASE Tools

CASE Tool are used to support a wide variety of SDLC. CASE Tools are used to help in the project identification and selection, project initiation and planning, and design phase, and in the implementation and maintenance phases. The components of CASE Tools are categorized into 3 mainly;

UpperCASE Tool

UpperCASE Tool is a Computer-Aided Software Engineering (CASE) software tool that supports the software development activities upstream from implementation. Uppercase tool focus on the analysis phase (but sometimes also the design phase) of the software development lifecycle (diagramming tools, report and form generators, and analysis tools)

LowerCASE Tool

LowerCASE Tool Computer-Aided Software Engineering (CASE) software tool that directly supports the implementation (programming) and integration tasks. LowerCASE tools support database schema generation, program generation, implementation, testing, and configuration management.

I CASE

Tools that integrate both upper and lower CASE, for example making it possible to design a form and build the database to support it at the same time. An automated system development environment that provides numerous tools to create diagrams, forms and reports. It also offers analysis, reporting, and code generation facilities and seamlessly shares and integrates data across and between tools.

Types of CASE Tools

The general types of CASE tools are listed below:

1. **Diagramming tools:** enable system process, data and control structures to be represented graphically.
2. **Computer display and report generators:** help prototype how systems look and feel. It makes it easier for the systems analyst to identify data requirements and relationship.
3. **Analysis tools:** automatically check for importance, inconsistent, or incorrect specifications in diagrams, forms, and reports.
4. **Central repository:** enables the integrated storage of specifications, diagrams, reports and project management information.
5. **Documentation Generators:** produce technical and user documentation in standard formats.
6. **Code generators:** enable the automatic generation of program and data base definition code directly from the design documents, diagrams, forms, and reports.

Benefits of CASE Tools?

- Improve the quality of the system developed.
- Increase the speed with which systems are designed and developed.
- Ease and improve the testing process through the use of automated checking.
- Improve the integration of development activities via common methodologies.
- Improve the quality and completeness to documentation.
- Help standardize the development process.
- Improve the management of the project.
- Simplify system maintenance.
- Promote reusability of modules and documentation.

Functions of a CASE Tool

1. **Analysis**
   CASE analysis tools automatically check for incomplete, inconsistent, or in correct specifications in diagrams, forms and reports.
2. **Design**
   This is where the technical blueprint of the system is created by designing the technical architecture – choosing amongst the architectural designs of telecommunications, hardware and software that will best suit the organization's system and future needs. Also designing the systems model – graphically creating a model from graphical user interface, screen design, and databases, to placement of objects on screen
3. **Code generation**
   CASE Tool has code generators which enable the automatic generation of program and data base definition code directly from the documents, diagrams, forms, and reports.
4. **Documentation**
   CASE Tool has documentation generators to produce technical and user documentation in standard forms. Each phase of the SDLC produces documentation. The types of documentation that flow from one face to the next vary depending upon the organization, methodologies employed and type of system being built.

Quiz

1. What would the benefits be to the use of CASE tools for developing a system by a team of people?
2. How can CASE tools help with documentation?

References

http://blog.salamtura.com/post/computer-aided-software-engineering/

computer aided software engineering

http://en.wikipedia.org/wiki/Computer-aided_software_engineering

http://www.answers.com/case%20tool

http://www.sei.cmu.edu/legacy/case/case_whatis.html

http://www.opfro.org/index.html?Components/Producers/Tools/UpperCASETools.html~Contents

http://www.opfro.org/index.html?Components/Producers/Tools/LowerCASETools.html~Contents.

Jefferey A. Hoffer, Joey F. George, and Joseph S. Valacich. Modern Systems Analysis and Design. Fourth edition .2005. New Jersey. Pg 602-617.

Hardware Components for a Purpose

Server Vs Consumer Grade Hardware

Many hardware component are available in server or consumer grade. Server grade hardware is designed for businesses with critical computers that need to be as reliable as possible. So for example if you have a computer acting as a server to a large business if it crashes it would result in a lot of down time for staff. A desktop computer used by a single member of staff on the other hand might crash with much less impact to the whole business.

Server components are designed to be more scalable. The majority of most core server hardware (motherboards, CPU, RAM) anticipate upgrades to one degree or another. Most desktops do not anticipate a larger quantity of resources. This is usually a function of the motherboard chipset more than anything, but server chipsets are significantly different from desktops.

Processors

As you know the CPU is the main processor of a computer. The faster the CPU the faster the computer runs. Modern desktop CPUs are very powerful so for most applications there would be little difference in usability between the top processor and lower end of the processor market.

Generally speaking, Core i7s are better than Core i5s, which are in turn better than Core i3s. Each have desktop and laptop variants. Intel also make Atom processes made for smaller portable devices.

Mobile processors, whether they be for laptops, smartphones, or IOT devices, are generally designed for efficiency first and performance second. Desktop processors being less limited by thermals and not at all by potential battery life, are by far the performance champs and will on average have more cache, higher stock and Turbo speeds, and higher TDP (Thermal Design Power).

AMD has a similar range AMD Ryzen, AMD FX, AMD Athlon and so on.
Cycles are what most processors are measured in "GHz," means one billion hertz, or cycles per second. The higher the number, the more often a processor can process. These numbers are not comparable between CPUs, however, because modern CPU architecture has different instruction sets, multiple cores, cache, hyper threading, sub cycle processing, and so on. All these variations in the processor architecture effects how quickly the processor performs it's tasks.

If you only run one or two programs at once, go on the internet, type up letters and do the odd spread sheet you will not notice much difference between processors on desktop system.

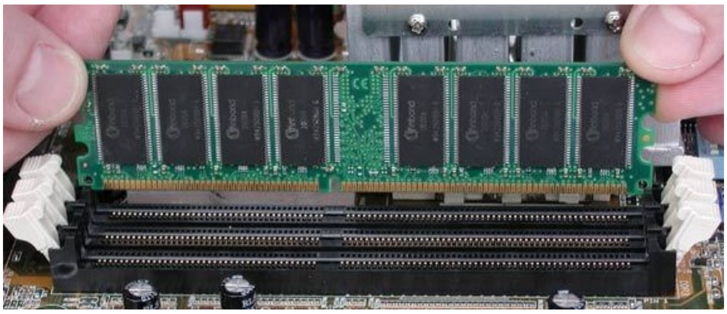
If you run multiple high end software Adobe Creative Suit for example at the same time, then a multicore CPU will do the job better.

If you are into Video Editing then a faster processor with hyper threading with do the job better.
If you play video games a faster multicore processor will do the job better.
More on board cache on the processor will also improve all the above hi-end tasks.

Primary Memory

Primary memory is computer memory that is accessed directly by the CPU. This includes several types of memory, such as the processor cache and system ROM. However, in most cases, primary memory refers to system RAM.

As mentioned in the last section most CPUs include a cache built into the chip. These come in three basic forms: a level 1 (L1) cache right next to the cores, and a L2 cache on the same chip but further away. Most of the resources of a processor are spent trying to keep these caches filled with any needed data. They use complex prediction algorithms to make sure that the CPU never has to wait a hundred cycles for RAM to send information over. And yes there is L3 cache as well but basically at this level all you need to know is cache on a CPU is very fast RAM - L1 is the fastest L2 second fastest and L3 is the third fastest.

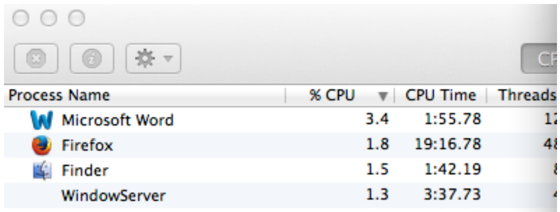


The bulk of the system ram is in the form of memory sticks on the motherboard. The more RAM you have the more programs you can run simultaneously without having the access the much slower secondary memory (your hard drive).

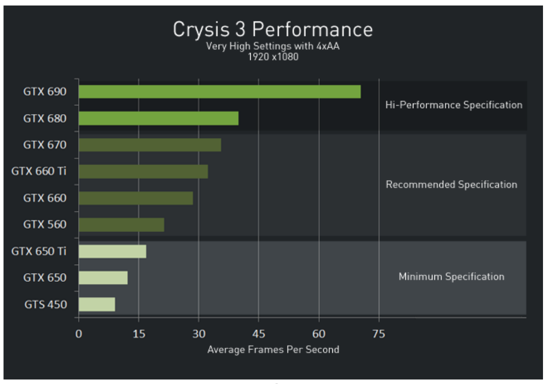
Like CPUs RAM is a technology that is evolving and so newer RAM will allow for faster transfer of information than older RAM providing that the motherboards supports it.

Like processor speed - generally modern desktops and laptops come with plenty of memory for your casual user. Again it is your high end users that will might want more.

Test The System



On Windows press CTRL, SHIFT, and ESC at the same time. This will bring up the Task Manager. Click on the performance tab. On Macs open Activity Monitor (it's in Applications/Utilities). Then fire up all the software you are likely to run at the same time and start doing what you would normally do. Writing this in Word is taking up a whopping 3.4% of the processing power of my 2007 iMac.



These applications will show how much physical memory the computer is using, as a number, and a percentage. This is the minimum memory needed, and, preferably, at least double this amount. If the memory is at 100%, then that means the system is already caching memory on the hard drive. When that happens, users need a lot more memory in the system. They will also show the heaviness of the CPU load. If the numbers are well under 100% usage, then the system is plenty fast.

Obviously there are other similar applications that will allow you to test your system. Some games have inbuilt bench marking so that you can figure out how well a system will run that particular game.

Graphics Cards

Graphic cards perform the 2D and 3D rendering of a system. The 2D rendering is handled well by pretty much any card. So if you do not play games you are unlikely to see much difference between the quality of various graphics cards.

If you do play games there will be a huge difference both in frame rate and the quality of the image that can be produced. Other 3D applications that use 3D rendering like 3D Ray Traced Animation or Video Production software like After Effects can also take advantage of a 3D card and will show major performance differences. 2D video rendering also utilised mathematics that GPUs are good at so the GPU will have a large effect on these applications too. With all these applications though the software has to be written to make use of the graphics cards GPU to see the benefit.

Modern operating systems such as Windows 10 and OSX have some fancy 3D effect to them as well so if you are running with them switched on they you might see a small deference between cards. Other graphical programs such as Photoshop also can take advantage of the processing power of the graphics card. But again the difference between modern cards won’t be huge.

As a random consequence of the types of mathematics that GPUs are good at is they are also very good and encryption and decryption. Which means they a very effective code breakers - much more effective than a conventional CPU. Passwords and encryptions that were previously thought of as (for practical purposes) uncrackable are not so much so anymore thanks to GPUs.

Secondary Memory

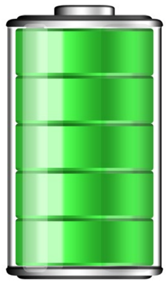
Secondary memory is not accessed directly by the CPU. Instead, data accessed from secondary memory is first loaded into RAM and is then sent to the processor.

Secondary Memory will affect the loading times of software. Generally solid state drives are much faster in practical use but traditional hard drives offer much more storage space and are much cheaper. Because traditional hard drives have moving parts they are more susceptible to damage through movement and vibration. They are a very mature technology though so, although they suffer wear and tear because of moving parts, they are very reliable and last a long time.

Solid state drives are much less susceptible to movement and vibration but do have a limited amount of writes. That is to say you can only save onto them so many times before they stop working. There is varying quality of these devices so your cheap thumb drive with survive fewer writes than the solid state hard drive in your laptop. But they will fail and generally without warning. The internal SSDs are designed to last the useful life of the system.

Always keep your data on more than one drive regardless of the technology you are using.

Laptops



Laptops use much the same hardware as desktops but have limitations on space which limits their expandability and their ability to eliminate heat because there is not the room for air to flow and multiple large extraction fans to be installed.

This means that CPUs that use less power are much more desirable. CPUs that can change the power they are using to suit the task are also important to mobile devices. The same limitations apply to their GPU.

You don't *need* the most powerful option, of course, and having a more powerful processor can negatively affect your battery life, so choosing a laptop with the right processor is more important than you might think. Both Intel and competitor AMD create processors specifically designed for laptops.

Plenty of other factors like screen resolution and storage capacity and type contribute to a laptop's power consumption, too. Windows, OSX or any other operating system will do its best to moderate power, performance and battery life and hit a good compromise, but the fact remains that some laptops use more power than others. Reading around, doing your research and choosing carefully can save you from the constant stress and headache of running out of battery when you're on the go.

The size of your laptop plays a *huge* role in how long its battery will last when you're using it on the go. A super-thin laptop simply can't hide away a massive battery pack, no matter how efficient its processor and display and storage are. Some manufacturers, especially those that integrate batteries into the shells of their laptops to use as much space as possible, can get amazing longevity from thin and light devices, but as a general rule the larger your laptop is the longer it will last. Settings such as brightness and wifi and also be adjusted to help the battery last longer.

The usability would also vary depending on the input devices supplied with the laptop. Most will have a touch pad but the features of these vary and now many laptops provide touch screens as well.

iMac Computer System Specifications

Display

27-inch (diagonal) Retina display with IPS technology; 5120-by-2880 resolution with support for millions of colours.

- Retina refers to the Dots Per Inch (DPI) of the monitor. A retina displays dots are in so small that you shouldn't be able to discern one pixel from the one next to it.

Processor

3.5GHz quad-core Intel Core i5 processor (Turbo Boost up to 4.1GHz)
Configurable to 4.2GHz quad-core Intel Core i7 (Turbo Boost up to 4.5GHz)

- 3.5 is how fast the processor is – processes 3.5 Billion times a second.
- i5 is a class of CPU architecture.
- quad-core means it has four processes on the single CPU chip.
- Turbo Boost means the processor can vary it's speed up to this speed for a burst. For how long? Depends on the cooling of the system.

Memory

8GB (two 4GB) of 2400MHz DDR3 memory; four user-accessible SO-DIMM slots
Configurable to 16GB or 32GB

- Primary Memory (RAM)
- 8GB is the amount
- 2400MHz is the speed of the RAM
- DDR4 is the type of RAM
- DIMM slots are where you can plug them into the motherboard

Storage

1TB Fusion Drive
Configurable to 3TB Fusion Drive, or 256GB, 512GB or 1TB of flash storage (SSD)

- Secondary Memory
- 1TB is the amount
- Fusion Drive means it is a traditional hard drive with flash storage as well (for speed).

Graphics
Radeon Pro 575 with 4GB of VRAM
Configurable to Radeon Pro 580 with 8GB of VRAM

- Graphics cards, the second one is newer and faster

Connections and Expansion
Headphone port
- For headphones :P

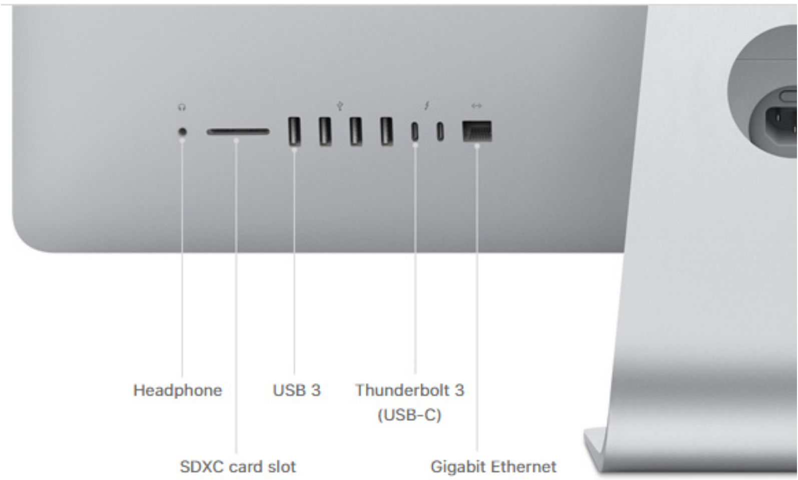
SDXC card slot
- For memory cards like out of a camera

Four USB 3 ports
- External Bus

Two Thunderbolt 3 ports
- An external bus standard that handles video in the form of Mini Display and other high demand devices.

10/100/1000BASE-T Gigabit Ethernet (RJ-45 connector)
- Wired networking

Kensington lock slot
- A physical security thing



Wireless

Wi-Fi
802.11ac Wi-Fi wireless networking; IEEE 802.11a/b/g/n compatible

Bluetooth
Bluetooth 4.2 wireless technology
- Networking Standards

Exercise

1. Find a computer system that would be suitable for a video producer. Explain how the specifications of your system would be suitable for this type of user.

References

https://www.williams-sonoma.com/products/cuisinart-elemental-13-cup-food-processor/
http://www.techretard.com/ram-types-ram-uses/
https://www.slashgear.com/nvidia-geforce-314-07-drivers-bring-crysis-3-optimizations-19269925/
https://www.digitaltrends.com/game-reviews/doom-2016-review/

CPUs in Detail

The central processing unit (CPU), sometimes referred to simply as the central processor, but more commonly called processor, the CPU is the brains of the computer where most calculations take place.

Control Unit

The control unit (CU) is a component of a computer's central processing unit (CPU) that directs the operation of the processor. It tells the computer's memory, arithmetic/logic unit and input and output devices on how to respond to a program's instructions.

A control unit or CU is circuitry that directs operations within a computer's processor. It lets the computer's logic unit, memory, as well as both input and output devices know how to respond to instructions received from a program. Examples of devices that utilize control units include CPUs and GPUs.

A control unit works by receiving input information that it converts into control signals, which are then sent to the central processor. The computer's processor then tells the attached hardware what operations to carry out. The functions that a control unit performs are dependent on the type of CPU, due to the variance of architecture between different manufacturers. The following diagram illustrates how instructions from a program are processed.

Arithmetic Logic Unit

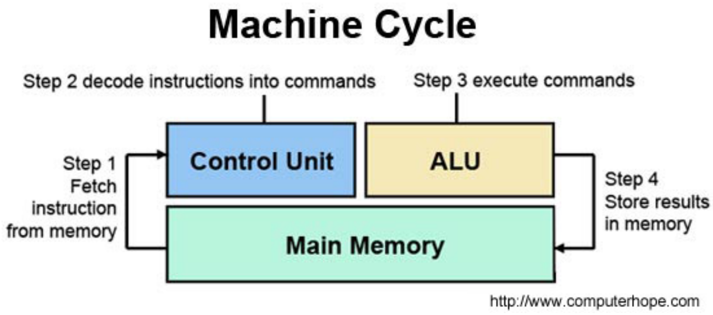
Short for Arithmetic Logic Unit, the ALU is a complex digital circuit; one of many components within a computer's central processing unit. It performs both bitwise and mathematical operations on binary numbers and is the last component to perform calculations in the processor. The ALU uses to operands and code that tells it which operations to perform for input data. After the information has been processed by the ALU, it is sent to the computer's memory.

Multiple Arithmetic Logic Units can be found in CPUs, GPUs and FPUs. In some computer processors, the ALU is divided into an AU and LU. The AU performs the arithmetic operations, and the LU performs the logical operations.

Fetch-Execute Cycle *Always in Exams*

Before an instruction can be executed, program instructions and data must be placed into memory from an input device or a secondary storage device. As the figure below shows once the necessary data and instruction are in memory, the central processing unit performs the following four steps for each instruction:

1. The control unit *fetches* the instruction from memory. The next instruction is fetched from the memory address that is currently stored in the program counter and stored into the instruction register.
2. The control unit *decodes* the instruction and directs that the necessary data be moved from memory to the ALU. These first two steps together are called instruction time or **I-time**.
3. The arithmetic/logic unit *executes* the arithmetic or logical instructions. It performs the actual operation on the data.
4. The arithmetic/logic unit *stores* the result of this operation in memory. Steps 3 and 4 together are called execution time, or **E-time**.



Register

A processor register (CPU register) is one of a small set of data holding places that are part of the computer processor.

A register may hold an instruction, a storage address, or any kind of data (such as a bit sequence or individual characters). Some instructions specify registers as part of the instruction. For example, an instruction may specify that the contents of two defined registers be added together and then placed in a specified register.

A register must be large enough to hold an instruction - for example, in a 64-bit computer, a register must be 64 bits in length. In some computer designs, there are smaller registers - for example, half-registers - for shorter instructions. Depending on the processor design and language rules, registers may be numbered or have arbitrary names.

A processor typically contains multiple index registers, also known as address registers or registers of modification. The effective address of any entity in a computer includes the base, index, and relative addresses, all of which are stored in the index register. A shift register is another type. Bits enter the shift register at one end and emerge from the other end. Flip flops, also known as bistable gates, store and process the data.

Program Counter

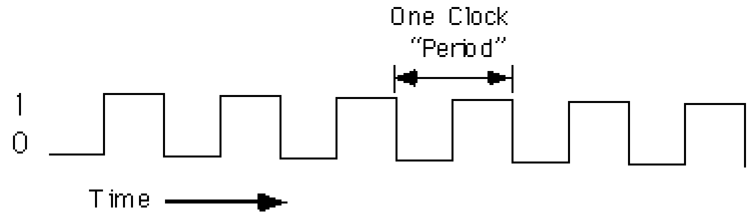
A program counter is a register in a computer processor that contains the address (location) of the instruction being executed at the current time. As each instruction gets fetched, the program counter increases its stored value by 1. After each instruction is fetched, the program counter points to the next instruction in the sequence. When the computer restarts or is reset, the program counter normally reverts to 0.

In computing, a program is a specific set of ordered operations for a computer to perform. An instruction is an order given to a computer processor by a program. Within a computer, an address is a specific location in memory or storage. A register is one of a small set of data holding places that the processor uses.

Some engineers refer to a program counter as an instruction address register or an address pointer.

System Clock

At the most basic level, the system clock handles all synchronization within a computer system. The system clock is an electrical signal on the control bus which alternates between zero and one at a periodic rate:



CPUs are a good example of a complex synchronous logic system. The system clock gates many of the logic gates that make up the CPU allowing them to operate in a synchronized fashion.

The frequency with which the system clock alternates between zero and one is the system clock frequency. The time it takes for the system clock to switch from zero to one and back to zero is the clock period. One full period is also called a clock cycle. On most modern systems, the system clock switches between zero and one at rates far exceeding million times per second. The clock frequency is simply the number of clock cycles which occur each second.

To ensure synchronization, most CPUs start an operation on either the falling edge (when the clock goes from one to zero) or the rising edge (when the clock goes from zero to one). The system clock spends most of its time at either zero or one and very little time switching between the two. Therefore clock edge is the perfect synchronization point.

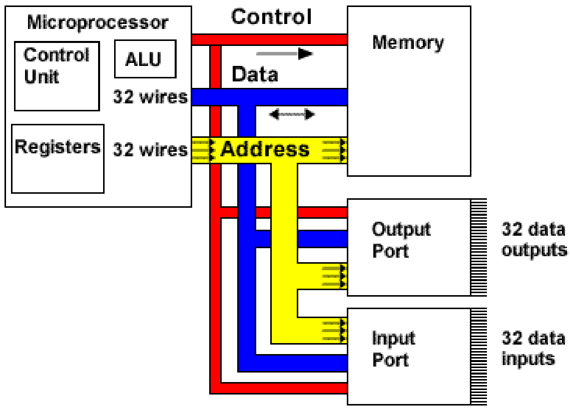
Since all CPU operations are synchronized around the clock, the CPU cannot perform tasks any faster than the clock. However, just because a CPU is running at some clock frequency doesn't mean that it is executing that many operations each second. Many operations take multiple clock cycles to complete so the CPU often performs operations at a significantly lower rate.

Data, Address and Control Bus

In isolation, the microprocessor, the memory and the input/output ports are interesting components, but they cannot do anything useful. In combination, they can form a complete system if they can communicate with each other. This communication is accomplished over bundles of signal wires (known as buses) that connect the parts of the system together.

There are normally three types of bus in any processor system:

- An address bus: this determines the location in memory that the processor will read data from or write data to.
- A data bus is a bus that operates only within the internal circuitry of the CPU, communicating among the internal caches of memory that are part of the CPU chip's design. This bus is typically rather quick and is independent of the rest of the computer's operations.
- A control bus: this manages the information flow between components indicating whether the operation is a read or a write and ensuring that the operation happens at the right time.



Types of Processing

Different processors and different systems process data and execute programs differently. Depending upon the task and purpose of the task and how a possessor is optimised to run. Processor architecture can be different depending upon the type of system so a mobile phone processor will manage tasks differently than a super computer.

Distributed processing

Distributed processing is a phrase used to refer to a variety of computer systems that use more than one computer (or processor) to run an application. In distributed computing/processing, each processor has its own private memory. The processing can be on one or many computers. Information is exchanged by passing messages between the processors.

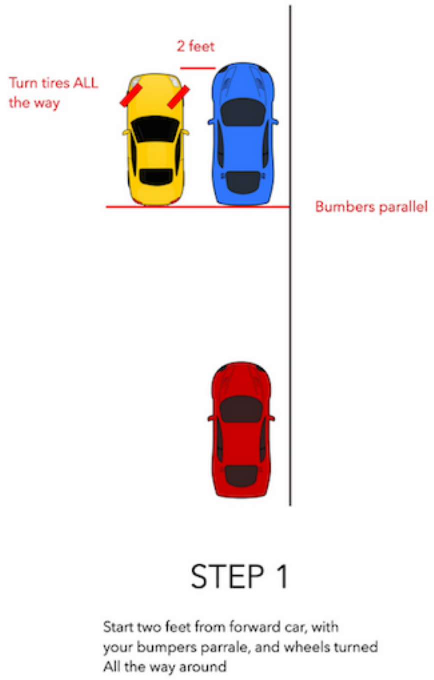
More often, however, distributed processing refers to local-area networks (LANs) designed so that a single program can run simultaneously at various sites. Most distributed processing systems contain sophisticated software that detects idle CPUs on the network and parcels out programs to utilize them.

Another form of distributed processing involves distributed databases. This is databases in which the data is stored across two or more computer systems. The database system keeps track of where the data is so that the distributed nature of the database is not apparent to users.

Sequential processing

Sequential processing is a term used to describe the processing that occurs in the order that it is received. Sequential processing is in contrast to parallel processing or multitasking.

Parallel Processing



The simultaneous use of more than one CPU to execute a program. Parallel Processing usually occurs on a single computer and with multiple processors – that may have access to a shared memory to exchange information between processors for the completion of the processing task. Ideally, parallel processing makes a program run faster because there are more engines (CPUs) running it. In practice, it is often difficult to divide a program in such a way that separate CPUs can execute different portions without interfering with each other.

Most computers have just one CPU, but some models have several. There are even computers with thousands of CPUs. Note that parallel processing differs from multitasking, in which a single CPU executes several programs at once.

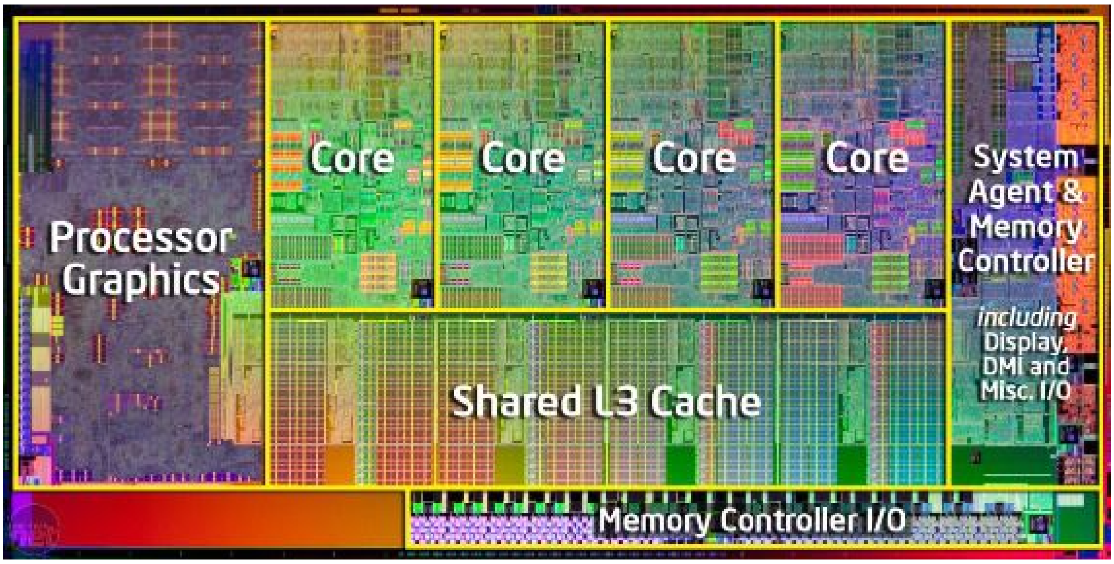
Multi-Core Processing

In multicore processing – processing occurs on a single processor with multiple cores.

A multi-core processor is an integrated circuit (IC) to which two or more processors have been attached for enhanced performance, reduced power consumption, and more efficient simultaneous processing of multiple tasks (see parallel processing). A dual core set-up is somewhat comparable to having multiple, separate processors installed in the same computer, but because the two processors are actually plugged into the same socket, the connection between them is faster.

Ideally, a dual core processor is nearly twice as powerful as a single core processor. In practice, performance gains are said to be about fifty percent: a dual core processor is likely to be about one-and-a-half times as powerful as a single core processor.

Multi-core processing is a growing industry trend as single-core processors rapidly reach the physical limits of possible complexity and speed. Most current systems are multi-core. Systems with a large number of processor core - tens or hundreds - are sometimes referred to as many-core or massively multi-core systems.


Processor Graphics
Core
Core
Core
Core
System Agent & Memory Controller
including Display, DMI and Misc. I/O
Shared L3 Cache
Memory Controller I/O

Quiz

1. How are the components of a computer system synchronized?
2. What happens in I time and E time?
3. What are the current generation of processes from Intel and AMD? What is one processor Intel makes for mobile devices?
4. Create an illustration for distributed processing, sequential processing and parallel processing.

References

https://www.webopedia.com/TERM/D/distributed_processing.html
https://searchdatacenter.techtarget.com/definition/multi-core-processor
http://bgr.com/2015/06/29/parallel-parking-how-to-tips/
https://www.youtube.com/watch?v=YS-QvfCZWvc
http://mayfield.mgfl.net/2016/08/24/lining-up/
https://i.ytimg.com/vi/H7gxsGAsugA/maxresdefault.jpg
http://www.computerhope.com/
https://s-media-cache-k0.pinimg.com/originals/dc/7a/c4/dc7ac40ccbf37c5d9ac14fe110785f45.jpg
http://i1.ebayimg.com/thumbs/images/g/WV4AAOSwzgBY4rvg/s-l225.jpg
https://fthmb.tqn.com/Us1gUpRgejVigveq1t8Hh13ZYr4=/1458x832/filters:fill(auto,1)/about/western-digital-my-passport-2tb-56a6fa9b3df78cf772913ec1.jpg
http://www.allwhitebackground.com/cash-register-white-background.html
https://courses.engr.illinois.edu/ece390/books/artofasm/CH03/CH03-2.html
http://www-mdp.eng.cam.ac.uk/web/library/enginfo/mdp_micro/lecture1/lecture1-3-1.html

File Systems

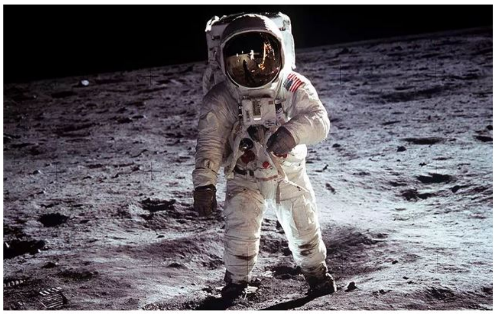
In computing, a file system or filesystem is used to control how data is stored and retrieved. Without a file system, information placed in a storage medium would be one large body of data with no way to tell where one piece of information stops and the next begins. By separating the data into pieces and giving each piece a name, the information is easily isolated and identified. Taking its name from the way paper-based information systems are named, each group of data is called a "file". The structure and logic rules used to manage the groups of information and their names is called a "file system".

There are many different kinds of file systems. Each one has different structure and logic, properties of speed, flexibility, security, size and more. Some file systems have been designed to be used for specific applications. For example, the ISO 9660 file system is designed specifically for optical discs.

File systems can be used on numerous different types of storage devices that use different kinds of media. The most common storage device in use today is a hard disk drive. Other kinds of media that are used include flash memory, magnetic tapes, and optical discs. In some cases, such as with tmpfs, the computer's main memory (random-access memory, RAM) is used to create a temporary file system for short-term use.

The file system manages access to both the content of files and the metadata about those files. It is responsible for arranging storage space; reliability, efficiency, and tuning with regard to the physical storage medium are important design considerations.

Space management



With any working file system files are added and removed. Deleting files create spaces, files systems need to manage these spaces so that they are used later on.

When you delete a file the file isn't actually removed from the media the file system just changes the metadata to allow the operating system to write over that data if more files are added. So for example if you accidently delete your photos from a digital camera so long as you do not take more photos you would most likely be able to retrieve those photos with file retrieving software.

Because files are different sizes the operating system will break files apart (fragment) and save those parts on different areas of the media to make the most of the free space available. This is causes fragmentation which with disc based systems will slow down the loading of that file because the drive has access different parts of the disc to retrieve it. This is why it is good to occasionally defragment the disc to go through the hard drive and rearrange these files so the complete files are in one position and the spaces are all consolidated. Solid state drives aren't slowed down by fragmentation and have a limited (although very large) amount of time they can be written to so defragmentation isn't recommend for them.

Filenames

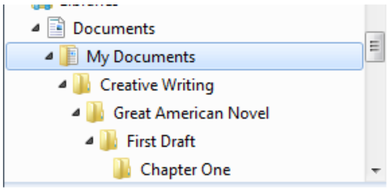
Adopt consistent methods for file and folder naming

When learning how to manage files and folders, it is important that you develop a naming scheme for the kinds of files you create most often and then stick to it. To change an existing file or folder name, right-click the name in the folder structure. Click **Rename**, and then type the new name.

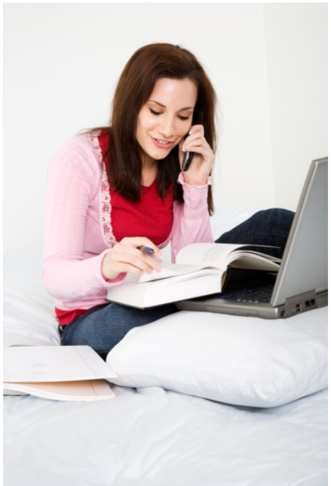
Keep names short

Even though you can use long file names in Windows and MacOS, you should not necessarily do so. Long file names can be harder to read.

Let your folder structure do some of the naming. For example, rather than creating a file called Great American Novel Chapter One First Effort, you can build a structure like this:



Separate ongoing and completed work



To keep the Documents folder from becoming too unwieldy, use it only for files you're actively working on. As a result, you can reduce the number of files you need to search through and the amount of data you need to back up. Every month or so, move the files you're no longer working on to a different folder or location, such as a special archive folder.

Store like with like.

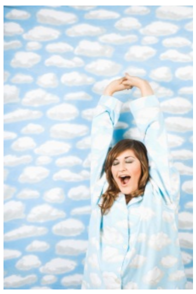
Restricting folders to a single document type (or predominantly one type) makes it easier for you to find files. For example, with all of your graphics in a single folder it's easier to find the right picture for your newsletter.

Use shortcuts and shortcut links instead of multiple copies.

If you need to get to the same file from multiple locations, don't create copies of the file. Create shortcuts to it instead. Shortcuts are links to files or programs and are represented by icons with an arrow in the lower-left corner. To create a shortcut, **right-click** the file and then **click Create Shortcut**. You can drag the shortcut to other locations. Microsoft Office includes some built-in shortcuts with the new Backstage view. To see Backstage view, open an Office file and then click the File tab. In Backstage view, click the Recent tab for a list of links to your recent documents. The Recent tab even includes a Recover Unsaved Documents option. In Backstage view, you can create, save, and send documents, inspect documents for hidden metadata or personal information, set options, and more.

Consider storing documents online.

You can also keep documents in ‘the cloud’ so that you can easily access them from outside the office, share them, and edit them anywhere.



Version Control

It is often handy to keep older versions of documents you are working on. It allows you to go back if mistakes are made and retrace your footprints if you are documenting the design process.

You can achieve this by applying version numbers to the document. So for example –

Newsletter1.0, Newsletter1.1, Newsletter1.2, Newsletter2.0…. and so on

Another method would be to include the date in the name of the document so –

Newsletter_12_1_13, Newsletter_12_2_13, Newsletter_14_3_13… and so on.

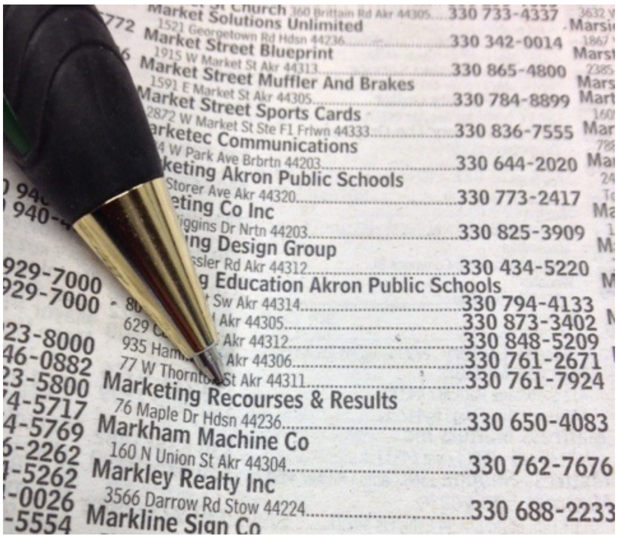
Backup Your Work!!!

You can use the built in features of your operating system to keep on going backups or you can manually periodically backup your work to another drive. You need to have you files on at least to drives because disks drives will fail. It's not a matter of if but when.

Online backups have the added benefit of keeping the backup at a different location.

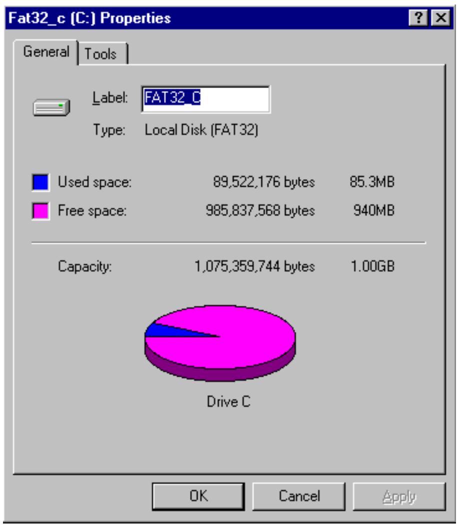
Directories

File systems typically have directories (also called folders) which allow the user to group files into separate collections. Directory structures may be flat (i.e. linear), or allow hierarchies where directories may contain subdirectories. The native file systems of Unix-like systems also support arbitrary directory hierarchies, as do, for example, Apple's HFS, the FAT file system and in Microsoft Windows, the NTFS file.



Disk File Systems

File Allocation Table



File Allocation Table (FAT) is a computer file system architecture and a family of industry-standard file systems utilizing it. The FAT file system is a continuing standard which borrows source code from the original, legacy file system and proves to be simple and robust. It offers useful performance even in lightweight implementations, but cannot deliver the same performance, reliability and scalability as some modern file systems. It is, however, supported for compatibility reasons by nearly all currently developed operating systems for personal computers and many mobile devices and embedded systems, and thus is a well-suited format for data exchange between computers and devices of almost any type and age from 1981 up to the present.

HFS Plus

HFS Plus or HFS+ is a file system developed by Apple Inc. It replaced the Hierarchical File System (HFS) as the primary file system of Apple computers with the 1998 release of Mac OS 8.1. HFS+ continued as the primary Mac OS X file system until it was itself replaced with the release of the Apple File System (APFS) with macOS High Sierra in 2017. HFS+ is also one of the formats used by the iPod digital music player. It is also referred to as Mac OS Extended or HFS Extended, where its predecessor, HFS, is also referred to as Mac OS Standard or HFS Standard. During development, Apple referred to this file system with the codename Sequoia.

HFS Plus is an improved version of HFS, supporting much larger files (block addresses are 32-bit length instead of 16-bit) and using Unicode (instead of Mac OS Roman or any of several other character sets) for naming items.

New Technology File System

NTFS ("New Technology File System") is a proprietary file system developed by Microsoft. Starting with Windows NT 3.1, it is the default file system of the Windows NT family.

NTFS has several technical improvements over the file systems that it superseded – File Allocation Table (FAT) and High Performance File System (HPFS) – such as improved support for metadata and advanced data structures to improve performance, reliability, and disk space use. Additional extensions are a more elaborate security system based on access control lists (ACLs) and file system journaling.

NTFS is supported in other desktop and server operating systems as well. Linux and BSD have a free and open-source NTFS driver, called NTFS-3G, with both read and write functionality. macOS comes with read-only support for NTFS; its disabled-by-default write support for NTFS is unstable.

Quiz

1. Explain the difference between a file and a directory.
2. What happens when you delete a file?
3. FAT files system has a few different variants, FAT 16, Fat 32 and exFAT. Explain the benefits of one over the other.

References

http://www.winzip.com/en/features/file-manager.html

https://www.wickes.co.uk/Bahco-Handyman-Wood-File---8in/p/226407

https://www.telegraph.co.uk/science/2016/03/12/faked-moon-landing-would-have-been-exposed-within-four-years-sci/

https://en.wikipedia.org/wiki/File_system

Operating Systems

Standard Operating Environment

A Standard Operating Environment (SOE) is a standard implementation of an operating system and its associated software.

Administrators typically implement SOE as a standard disk image for mass deployment to multiple computers in an organisation. SOEs can include the base operating system, a custom configuration, standard applications used within an organisation, software updates and service packs. An SOE can apply to servers, desktops, laptops, thin clients, and mobile devices.

Enterprises, universities and other organizations may have separate SOEs for workstations, notebook computers and mobile devices such as smartphones and tablets.

Components of an SOE typically include the OS, its standard configuration, associated service packs, common applications and their associated updates. Disk images are typically used to speed deployment of an SOE across networks to multiple clients.

The standardized base of an SOE can help maintain consistent reliability and performance throughout an organization. The SOE reduces variables that can create individual software issues that complicate diagnosis, troubleshooting and issue resolution. The standardization can also provide a compatible environment for extra software that may be required by specialized departments. The SOE's base of approved, secure applications should also be resistant to exploits.

Sometimes SOE are simply a starting point, and freedoms granted to individual users diverge from the initial install. Organizations that have more stringent SOE constraints may also have exemption requests for use of applications outside the ones included in the environment. In either case, IT is encouraged to confer with staff about their needs and to find the most effective, yet still secure application that will fill the bill.

The major advantage of an SOE in a business environment is the reduction in the cost and time taken to deploy, configure, maintain, support and manage computers. By standardising the hardware and software platforms used within an organization, an IT department or service provider can deploy new computers and correct problems with existing computers quickly. A standardized, repeatable and automated solution creates a known, expected and supportable environment. A standardised solution ensures maintaining known outcomes, with automation fostering speed, repeatability and standardization.

The introduction of BYOD (Bring Your Own Device) and the significant increase in employee-supplied devices has led many organisations to reconsider the use of an SOE. A number have implemented an Unmanaged Operating Environment (UOE) where users manage and maintain their own devices, subject to policies enforcing minimum standards.

Roles of an Operating System

Scheduling

The act of determining which process is in the ready state, and should be moved to the running state is known as Process Scheduling.

The prime aim of the process scheduling system is to keep the CPU busy all the time and to deliver minimum response time for all programs. For achieving this, the scheduler must apply appropriate rules for swapping processes IN and OUT of CPU.

Scheduling fell into one of the two general categories:

- Non Pre-emptive Scheduling: When the currently executing process gives up the CPU voluntarily.
- Pre-emptive Scheduling: When the operating system decides to favour another process, pre-empting the currently executing process.

What are Scheduling Queues?

- All processes, upon entering into the system, are stored in the Job Queue.
- Processes in the Ready state are placed in the Ready Queue.
- Processes waiting for a device to become available are placed in Device Queues. There are unique device queues available for each I/O device.

A new process is initially put in the Ready queue. It waits in the ready queue until it is selected for execution(or dispatched). Once the process is assigned to the CPU and is executing, one of the following several events can occur:

- The process could issue an I/O request, and then be placed in the I/O queue.
- The process could create a new subprocess and wait for its termination.
- The process could be removed forcibly from the CPU, as a result of an interrupt, and be put back in the ready queue.

Concurrency

All modern operating systems allow multiple programs to run at the same time. This is called concurrency. A running program is known as a process. In fact modern operating systems even allow a single program to execute different routines at the same time or to put it another way, they allow a process to execute several threads of control at the same time.

If multiple threads are running and accessing shared state then it is possible that these threads will conflict with each other on the usage of said state and it becomes necessary to regulate each threads access to that state. This regulation is called concurrency control.

In computing, multitasking is the concurrent execution of multiple tasks (also known as processes) over a certain period of time. New tasks can interrupt already started ones before they finish, instead of waiting for them to end. As a result, a computer executes segments of multiple tasks in an interleaved manner, while the tasks share common processing resources such as central processing units (CPUs) and main memory. Multitasking automatically interrupts the running program, saving its state (partial results, memory contents and computer register contents) and loading the saved state of another program and transferring control to it. This "context switch" may be initiated at fixed time intervals (pre-emptive multitasking), or the running program may be coded to signal to the supervisory software when it can be interrupted (cooperative multitasking).

Multitasking does not require parallel execution of multiple tasks at exactly the same time; instead, it allows more than one task to advance over a given period of time. Even on multiprocessor computers, multitasking allows many more tasks to be run than there are CPUs.

Multitasking is a common feature of computer operating systems. It allows more efficient use of the computer hardware; where a program is waiting for some external event such as a user input or an input/output transfer with a peripheral to complete, the central processor can still be used with another program. In a time sharing system, multiple human operators use the same processor as if it was dedicated to their use, while behind the scenes the computer is serving many users by multitasking their individual programs. In multiprogramming systems, a task runs until it must wait for an external event or until the operating system's scheduler forcibly swaps the running task out of the CPU. Real-time systems such as those designed to control industrial robots, require timely processing; a single processor might be shared between calculations of machine movement, communications, and user interface.

Often multitasking operating systems include measures to change the priority of individual tasks, so that important jobs receive more processor time than those considered less significant. Depending on the operating system, a task might be as large as an entire application program, or might be made up of smaller threads that carry out portions of the overall program.

A processor intended for use with multitasking operating systems may include special hardware to securely support multiple tasks, such as memory protection, and protection rings that ensure the supervisory software cannot be damaged or subverted by user-mode program errors.

The term "multitasking" has become an international term, as the same word is used in many other languages such as German, Italian, Dutch, Danish and Norwegian.

Memory management (Operating systems)

When an operating system manages the computer's memory, there are two broad tasks to be accomplished:

1. Each process must have enough memory in which to execute, and it can neither run into the memory space of another process nor be run into by another process.

2. The different types of memory in the system must be used properly so that each process can run most effectively.

The first task requires the operating system to set up memory boundaries for types of software and for individual applications.

Disk storage is only one of the memory types that must be managed by the operating system, and it's also the slowest. Ranked in order of speed, the types of memory in a computer system are:

- High-speed cache - This is fast, relatively small amounts of memory that are available to the CPU through the fastest connections. Cache controllers predict which pieces of data the CPU will need next and pull it from main memory into high-speed cache to speed up system performance.

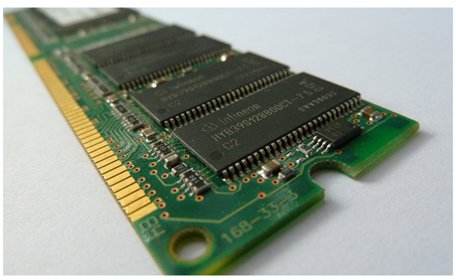
- Primary memory - This is the RAM that you see measured in Gigabytes when you buy a computer.

- Secondary memory - This is your hard drives, thumb drives and optical drives – basically any storage that isn't RAM or ROM. Measured in Gigabytes or Terabytes.

In the OS, memory management involves the allocation (and constant reallocation) of specific memory blocks to individual programs as user demands change. At the application level, memory management ensures the availability of adequate memory for the objects and data structures of each running program at all times. Application memory management combines two related tasks, known as allocation and recycling.

When the program requests a block of memory, a part of the memory manager called the allocator assigns that block to the program.

When a program no longer needs the data in previously allocated memory blocks, those blocks become available for reassignment. This task can be done manually (by the programmer) or automatically (by the memory manager).



Device Management

Device management generally performs the following:

- Installing device and component-level drivers and related software
- Configuring a device so it performs as expected using the bundled operating system, business/workflow software and/or with other hardware devices.
- Implementing security measures and processes.

Devices usually refer to physical/hardware devices such as computers, laptops, servers, mobile phones and more. They could also be virtual, however, such as virtual machines or virtual switches. In Windows, device management is also an administrative module that is used for managing or configuring the physical devices, ports and interfaces of a computer or server.

The path between the operating system and virtually all hardware not on the computer's motherboard goes through a special program called a driver. Much of a driver's function is to be the translator between the electrical signals of the hardware subsystems and the high-level programming languages of the operating system and application programs. Drivers take data that the operating system has defined as a file and translate them into streams of bits placed in specific locations on storage devices, or a series of laser pulses in a printer.

One reason that drivers are separate from the operating system is so that new functions can be added to the driver -- and thus to the hardware subsystems -- without requiring the operating system itself to be modified, recompiled and redistributed.

Managing input and output is largely a matter of managing queues and buffers, special storage facilities that take a stream of bits from a device, perhaps a keyboard or a serial port, hold those bits, and release them to the CPU at a rate with which the CPU can cope. This function is especially important when a number of processes are running and taking up processor time. The operating system will instruct a buffer to continue taking input from the device, but to stop sending data to the CPU while the process using the input is suspended. Then, when the process requiring input is made active once again, the operating system will command the buffer to send data. This process allows a keyboard or a modem to deal with external users or computers at a high speed even though there are times when the CPU can't use input from those sources.

Managing all the resources of the computer system is a large part of the operating system's function.

Drivers

In computing, a device driver is a computer program that operates or controls a particular type of device that is attached to a computer. A driver provides a software interface to hardware devices, enabling operating systems and other computer programs to access hardware functions without needing to know precise details about the hardware being used.

The main purpose of device drivers is to provide abstraction by acting as a translator between a hardware device and the applications or operating systems that use it. Programmers can write the higher-level application code independently of whatever specific hardware the end-user is using.

Types of Operating Systems

Embedded

An embedded operating system is a type of operating system that is embedded and specifically configured for a certain hardware configuration and purpose. Hardware that uses embedded operating systems is designed to be lightweight and compact, forsaking many other functions found in non-embedded computer systems in exchange for efficiency at resource usage. This means that they are made to do specific tasks and do them efficiently.

Embedded operating systems are usually used for hardware that have very little computing power, little RAM/ROM and a slow CPU, so they tend to be very specific in their applications and scope. They are usually made using assembly language in order to really take advantage of the limited computing resources, since it is the closest to machine language and is able to squeeze every drop of computing power available. This means that the OS is optimized for whatever hardware it was developed for and will not be compatible with other hardware systems with different configurations.

In most embedded OSs, the applications are built in to the OS or part of the OS, so they are loaded immediately when the OS starts. The most common examples of devices with an embedded OS would be cell phones before Android and iOS popularized the mobile operating system, which may still be considered as embedded but are also arguably desktop-like in the way they handle tasks and apps and their access to vast amounts of computing power. Embedded OSs can also be found in cars, large laser printers, some home appliances, and even military systems.

Stand-Alone Operating System

A stand-alone is a complete operating system works on desktop or notebook computer. So Windows, Mac OS and Linux.

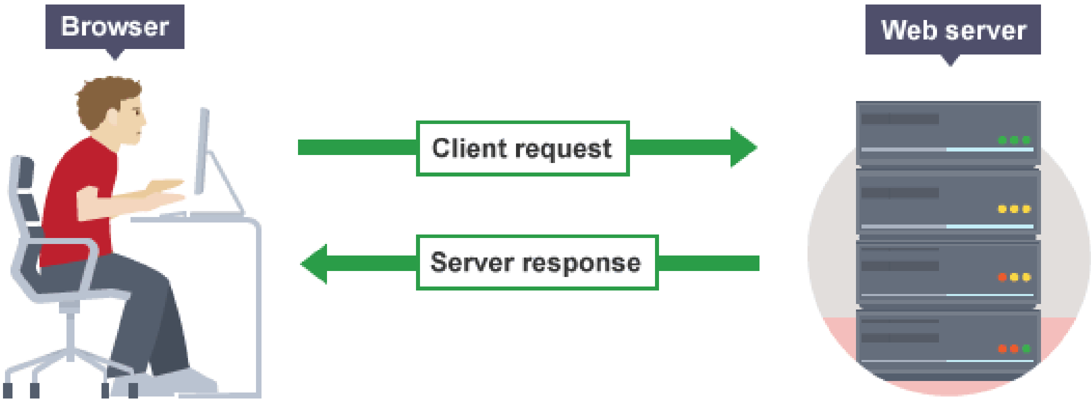
Server Operating System

A server operating system, also called a server OS, is an operating system specifically designed to run on servers, which are specialized computers that operate within a client/server architecture to serve the requests of client computers on the network.

Server OSes are designed from the ground up to provide platforms for multi-user, frequently business-critical, networked applications. As such, the focus of such operating systems tends to be security, stability and collaboration, rather than user interface.

Server operating systems help enable and facilitate typical server roles such as Web server, mail server, file server, database server, application server and print server.

Popular server operating systems include Windows Server, Mac OS X Server, variants of Linux such as Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server and UNIX.



Quiz

1. How are disc images used to create and maintain standard operating environments.
2. There are four roles of an operating system identified in this section. List the four roles. What is another role of an operating system that was not included that could have been? Explain how operating systems fulfil this role.
3. Modern operating systems are released with the option of "Safe Mode Boot". Safe Mode Boot loads only important drivers (minimum) and not all. Safe Mode Boot is used for diagnostic purposes to find the driver with bugs. What are drivers and why do operating systems use them?
4. How does an operating system manage interacting with all of the possible graphics cards that could be installed in a computer?
5. Give an example of a device that uses an embedded operating system. Why does it use an embedded operating system rather than a stand-alone operating system like Windows?
6. What does a server operating system do over a stand-alone operating system?

References

https://en.wikipedia.org/wiki/Standard_Operating_Environment
http://whatis.techtarget.com/definition/standard-operating-environment-SOE
https://www.freepik.com/free-photos-vectors/monitor
https://www.studytonight.com/operating-system/cpu-scheduling
https://www.tutorialspoint.com/operating_system/os_process_scheduling.htm
https://en.wikipedia.org/wiki/Scheduling_(computing)
https://www.quora.com/What-is-concurrency-control-in-operating-systems
https://en.wikipedia.org/wiki/Computer_multitasking
http://whatis.techtarget.com/definition/memory-management
https://www.robotshop.com/letsmakerobots/tags/fun-light-switch-box
https://www.studytonight.com/operating-system/process-scheduling
https://www.techopedia.com/definition/30014/embedded-operating-system
https://en.wikipedia.org/wiki/Device_driver

Benchmarking

Benchmarking is a mechanism used to test software/hardware/operating systems for a given purpose. In doing so indicating using a rating or score for its performance when compared with other products.

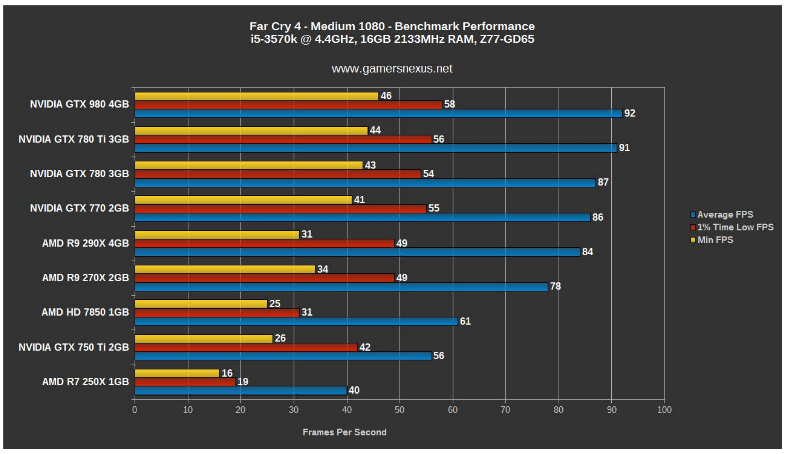
In computing, a benchmark is the act of running a computer program, a set of programs, or other operations, in order to assess the relative performance of an object, normally by running a number of standard tests and trials against it. The term 'benchmark' is also mostly utilized for the purposes of elaborately designed benchmarking programs themselves.

Benchmarking is usually associated with assessing performance characteristics of computer hardware, for example, the floating point operation performance of a CPU, but there are circumstances when the technique is also applicable to software. Software benchmarks are, for example, run against compilers or database management systems.

Benchmarks provide a method of comparing the performance of various subsystems across different chip/system architectures.

Test suites are a type of system intended to assess the correctness of software.

Bench marking is also used to test Graphical Processing Units. Their ability to render video, effects and 3D. These bench marks often use real software like photoshop or particular games to create their results. Many games come with benchmark tools built in.



Purpose

As computer architecture advanced, it became more difficult to compare the performance of various computer systems simply by looking at their specifications. Therefore, tests were developed that allowed comparison of different architectures. A slower processor, with regard to clock frequency, may perform as well as a processor operating at a higher frequency.

Benchmarks are designed to mimic a particular type of workload on a component or system. Synthetic benchmarks do this by specially created programs that impose the workload on the component. Application benchmarks run real-world programs on the system. While application benchmarks usually give a much better measure of real-world performance on a given system, synthetic benchmarks are useful for testing individual components, like a hard disk or networking device.

Benchmarks are particularly important in CPU design, giving processor architects the ability to measure and make tradeoffs in microarchitectural decisions. For example, if a benchmark extracts the key algorithms of an application, it will contain the performance-sensitive aspects of that application. Running this much smaller snippet on a cycle-accurate simulator can give clues on how to improve performance.

CPUs that have many execution units typically have slower clock rates than a sequential CPU with one or two execution units when built from transistors that are just as fast. Nevertheless, CPUs with many execution units often complete real-world and benchmark tasks in less time than the supposedly faster high-clock-rate CPU.

Given the large number of benchmarks available, a manufacturer can usually find at least one benchmark that shows its system will outperform another system; the other systems can be shown to excel with a different benchmark.

Manufacturers commonly report only those benchmarks (or aspects of benchmarks) that show their products in the best light. They also have been known to mis-represent the significance of benchmarks, again to show their products in the best possible light. Taken together, these practices are called bench-marketing.

Ideally benchmarks should only substitute for real applications if the application is unavailable, or too difficult or costly to port to a specific processor or computer system. If performance is critical, the only benchmark that matters is the target environment's application suite.

Quiz

1. Define the term benchmarking in computer science terms
2. Explain why benchmark tests are hard to do.
3. Find at least 3 examples of Software, Hardware and OS benchmarks in use today for Hard drives, CPU and software. How is the speed measured – on what units?

References

https://en.wikipedia.org/wiki/Benchmark_(computing)

https://www.t-nation.com/workouts/how-i-benched-400

https://www.newegg.com/global/uk/Product/Product.aspx?Item=N82E16814125869

Disaster Recovery Plan

A disaster recovery plan (DRP) is a documented process or set of procedures to recover and protect a business IT infrastructure in the event of a disaster. Such plan, ordinarily documented in written form, specifies procedures an organization is to follow in the event of a disaster. It is "a comprehensive statement of consistent actions to be taken before, during and after a disaster." The disaster could be natural, environmental, man-made or Godzilla related. Man-made disasters could be intentional (for example, an act of a terrorist) or unintentional (that is, accidental, such as the breakage of a man-made dam).

The hard truth is that no company is as invulnerable as the King of Monsters, nor does it have to be.

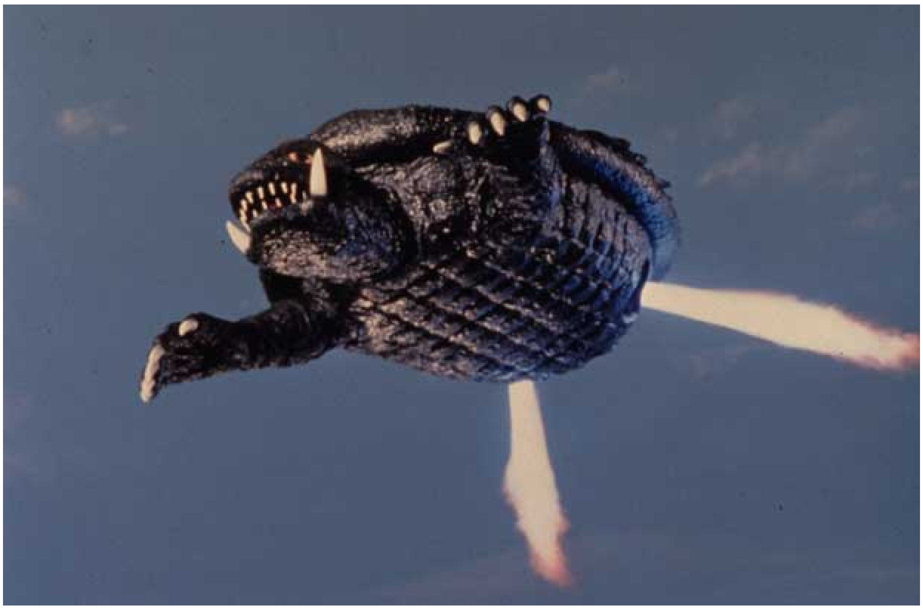
Like every insurance plan, there are benefits that can be obtained from the drafting of a disaster recovery plan. Some of these benefits are:

- Providing a sense of security
- Minimizing risk of delays
- Guaranteeing the reliability of standby systems
- Providing a standard for testing the plan
- Minimizing decision-making during a disaster
- Reducing potential legal liabilities
- Lowering unnecessarily stressful work environment

Disaster Recovery: Godzilla can’t break your data if he can’t reach it

The future of data back-up and recovery is in cloud technology. To this day, even Godzilla cannot reach the clouds, real or digital. Cloud technology is an umbrella term for data that is accessed and stored in a server that is not physically located in the vicinity of the user. In other words, if your business is physically in ground zero of the monster’s attack, your data is stored safely in a server far, far away.

I know what you're thinking – "what about Gamera?" and it is true cloud storage may still leave you vulnerable to titanic, fire-breathing, prehistoric species of tortoise who's diet of petroleum-based material gives them the ability to breathe fire and fly by "jets" ignited when the monster retracts its legs.



Backup Techniques

http://www.genie9.com

If you were to ask a person that was not familiar with computer backups, most would think that a backup was just an identical copy of all the data on the computer.

While it is possible to configure backups in this way, it is likely that you would not. To understand more about this, we must first understand the different types of backups that can be created. They are:

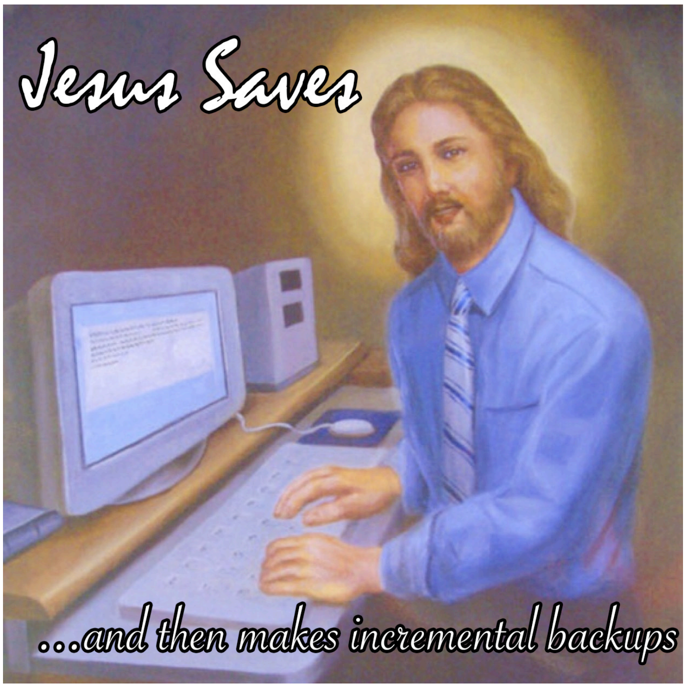
- Full Backups
- Incremental Backups

Full Backups

Full backup is the starting point for all other backups, and contains all the data in the folders and files that are selected to be backed up. Because full backup stores all files and folders, frequent full backups result in faster and simpler restore operations. Remember that when you choose other backup types, restore jobs may take longer.

This approach is good when the project includes smaller amounts of data.

Incremental Backups



An incremental backup stores all files that have changed since the last backup. The advantage of an incremental backup is that it takes the least time to complete. However, during a restore operation, each incremental backup must be processed, which could result in a lengthy restore job.

This approach is good when the project includes too many files to back up them all each time. It's fast and takes less time for incremental backups. Incremental backups take less disk space. It allows you to create backups frequently. However, to restore all the files, you have to restore the last full backup, and all the following incremental backups.

RAID

RAID is a technology that is used to increase the performance and/or reliability of data storage. The abbreviation stands for Redundant Array of Inexpensive Disks. A RAID system consists of two or more drives working in parallel. These disks can be hard discs, but there is a trend to also use the technology for SSD (solid state drives). There are different RAID levels, each optimized for a specific situation.

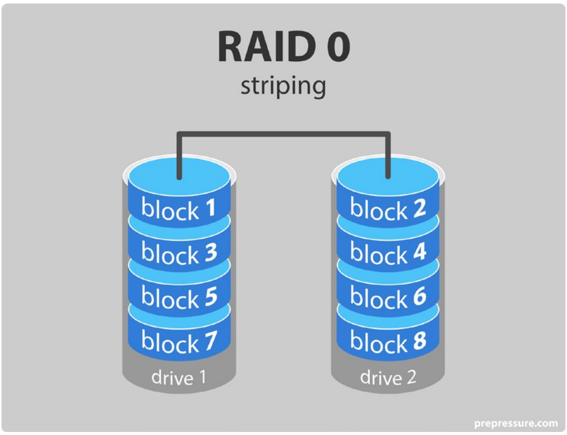
This syllabus covers the following RAID levels:

- RAID 0 – striping
- RAID 1 – mirroring
- RAID 10 – combining mirroring and striping

The software to perform the RAID-functionality and control the drives can either be located on a separate controller card (a hardware RAID controller) or it can simply be a driver. Some versions of Windows, such as Windows Server 2012 as well as Mac OS X, include software RAID functionality. Hardware RAID controllers cost more than pure software, but they also offer better performance.

RAID level 0 – Striping

In a RAID 0 system data are split up into blocks that get written across all the drives in the array. By using multiple disks (at least 2) at the same time, this offers superior I/O performance. This performance can be enhanced further by using multiple controllers, ideally one controller per disk.



Advantages

- RAID 0 offers great performance, both in read and write operations. There is no overhead caused by parity controls.
- All storage capacity is used, there is no overhead.
- The technology is easy to implement.

Disadvantages

- RAID 0 is not fault-tolerant. If one drive fails, all data in the RAID 0 array are lost. It should not be used for mission-critical systems.

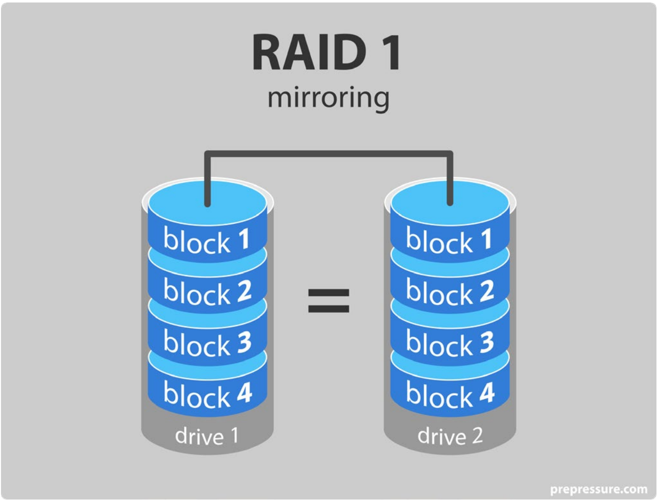
Ideal use

RAID 0 is ideal for non-critical storage of data that have to be read/written at a high speed, such as on an image retouching or video editing station.

If you want to use RAID 0 purely to combine the storage capacity of twee drives in a single volume, consider mounting one drive in the folder path of the other drive. This is supported in Linux, OS X as well as Windows and has the advantage that a single drive failure has no impact on the data of the second disk or SSD drive.

RAID level 1 – Mirroring

Data are stored twice by writing them to both the data drive (or set of data drives) and a mirror drive (or set of drives). If a drive fails, the controller uses either the data drive or the mirror drive for data recovery and continues operation. You need at least 2 drives for a RAID 1 array.



Advantages

- RAID 1 offers excellent read speed and a write-speed that is comparable to that of a single drive.
- In case a drive fails, data do not have to be rebuild, they just have to be copied to the replacement drive.
- RAID 1 is a very simple technology.

Disadvantages

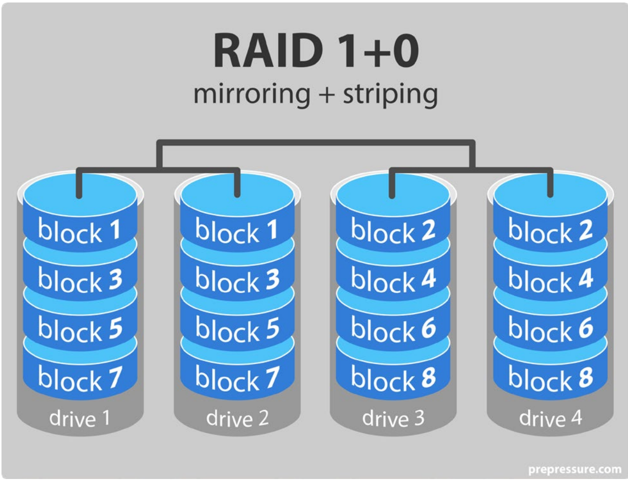
- The main disadvantage is that the effective storage capacity is only half of the total drive capacity because all data get written twice.
- Software RAID 1 solutions do not always allow a hot swap of a failed drive. That means the failed drive can only be replaced after powering down the computer it is attached to. For servers that are used simultaneously by many people, this may not be acceptable. Such systems typically use hardware controllers that do support hot swapping.

Ideal use

RAID-1 is ideal for mission critical storage, for instance for accounting systems. It is also suitable for small servers in which only two data drives will be used.

RAID level 10 – combining RAID 1 & RAID 0

It is possible to combine the advantages (and disadvantages) of RAID 0 and RAID 1 in one single system. This is a nested or hybrid RAID configuration. It provides security by mirroring all data on secondary drives while using striping across each set of drives to speed up data transfers.



Advantages

- If something goes wrong with one of the disks in a RAID 10 configuration, the rebuild time is very fast since all that is needed is copying all the data from the surviving mirror to a new drive. This can take as little as 30 minutes for drives of 1 TB.

Disadvantages

- Half of the storage capacity goes to mirroring, so compared to large RAID 5 or RAID 6 arrays, this is an expensive way to have redundancy.

RAID is no substitute for back-up!

All RAID levels except RAID 0 offer protection from a single drive failure. A RAID 6 system even survives 2 disks dying simultaneously. For complete security, you do still need to back-up the data from a RAID system.

- That back-up will come in handy if all drives fail simultaneously because of a power spike.
- It is a safeguard when the storage system gets stolen.
- Back-ups can be kept off-site at a different location. This can come in handy if a natural disaster or fire destroys your workplace.
- The most important reason to back-up multiple generations of data is user error. If someone accidentally deletes some important data and this goes unnoticed for several hours, days or weeks, a good set of back-ups ensure you can still retrieve those files.

Uninterruptible Power Supply

An uninterruptible power supply, also uninterruptible power source, is a continual power system, an electrical apparatus that provides emergency power to a load when the input power source or mains power fails. A UPS differs from an auxiliary or emergency power system or standby generator in that it will provide near-instantaneous protection from input power interruptions, by supplying energy stored in batteries or supercapacitors. The on-battery runtime of most uninterruptible power sources is relatively short (only a few minutes) but sufficient to start a standby power source or properly shut down the protected equipment.

A UPS is typically used to protect hardware such as computers, data centers, telecommunication equipment or other electrical equipment where an unexpected power disruption could cause injuries, fatalities, serious business disruption or data loss.

Common power problems

The primary role of any UPS is to provide short-term power when the input power source fails. However, most UPS units are also capable in varying degrees of correcting common utility power problems like voltage spikes or reductions in input voltage. Basically a good UPS can clean up the power making it predictable and consistent which can be good for the long term functioning of electrical equipment.

Quiz

1. How could you protect the data of your business against a disaster like a fire?
2. What gives Gamera the ability to breathe fire and fly by jets from he’s retracted legs?
3. What is the befits of full backups over incremental backups ?
4. What is the befits of incremental backups over full backups?
5. What are the benefits of RAID 0?
6. What are the benefits of RAID 1?
7. What are the benefits of RAID 10?
8. What is the name of the dog in UP?

References

https://en.wikipedia.org/wiki/Data_management
https://en.wikipedia.org/wiki/Data_security
https://en.wikipedia.org/wiki/Disaster_recovery_plan
https://huddle.eurostarsoftwaretesting.com/organizational-technical-challenges-implementing-test-data-management/
https://wall.alphacoders.com/by_sub_category.php?id=171667&name=2012+Wallpapers
https://www.npr.org/2014/05/02/308934760/movie-monsters-monster-movies-and-why-godzilla-endures
http://knowyourmeme.com/memes/subcultures/gamera
http://philosophyofscienceportal.blogspot.com.au/2012/10/the-numbers-games-of-spy-vs-spy-in.html
https://www.genie9.com/support/kb/KnowledgeArticle.aspx?KBID=104
https://www.pinterest.com.au/pin/497999671272166484/
https://www.prepressure.com/library/technology/raid
https://en.wikipedia.org/wiki/Uninterruptible_power_supply

Virtualisation

In computing, virtualisation refers to the act of creating a virtual (rather than actual) version of something, including virtual computer hardware platforms, storage devices, and computer network resources.

Desktop Virtualisation

Desktop virtualisation is a virtualisation technology that separates an individual's PC applications from his or her desktop. Virtualized desktops are generally hosted on a remote central server, rather than the hard drive of the personal computer. Because the client-server computing model is used in virtualizing desktops, desktop virtualisation is also known as client virtualisation.

Desktop virtualisation provides a way for users to maintain their individual desktops on a single, central server. The users may be connected to the central server through a LAN, WAN or over the Internet.

Desktop virtualisation has many benefits, including a lower total cost of ownership (TCO), increased security, reduced energy costs, reduced downtime and centralized management.

Personal Computer Virtualisation *cough* Hardware Virtualisation *cough*

Hardware virtualisation is the virtualisation of computers as complete hardware platforms, certain logical abstractions of their componentry, or only the functionality required to run various operating systems. Virtualisation hides the physical characteristics of a computing platform from the users, presenting instead an abstract computing platform. At its origins, the software that controlled virtualisation was called a "control program", but the terms "hypervisor" or "virtual machine monitor" became preferred over time.

A virtual machine can be more easily controlled and inspected from outside than a physical one, and its configuration is more flexible. This is very useful in kernel development and for teaching operating system courses.

- Running one or more applications that are not supported by the host OS: A virtual machine running the required guest OS could allow the desired applications to be run, without altering the host OS.
- Evaluating an alternate operating system: The new OS could be run within a VM, without altering the host OS.
- Server virtualisation: Multiple virtual servers could be run on a single physical server, in order to more fully utilize the hardware resources of the physical server.
- Duplicating specific environments: A virtual machine could, depending on the virtualisation software used, be duplicated and installed on multiple hosts, or restored to a previously backed-up system state.
- Creating a protected environment: if a guest OS running on a VM becomes damaged in a way that is difficult to repair, such as may occur when studying malware or installing badly behaved software, the VM may simply be discarded without harm to the host system, and a clean copy used next time.

Server Virtualisation

Server virtualisation is a virtualisation technique that involves partitioning a physical server into a number of small, virtual servers with the help of virtualisation software. In server virtualisation, each virtual server runs multiple operating system instances at the same time.

Typical enterprise data centers contain a huge number of servers. Many of these servers sit idle as the workload is distributed to only some of the servers on the network. This results in a waste of expensive hardware resources, power, maintenance and cooling requirements. Server virtualisation attempts to increase resource utilization by partitioning physical servers into several multiple virtual servers, each running its own operating system and applications. Server virtualisation makes each virtual server look and act like a physical server, multiplying the capacity of every single physical machine.

The concept of server virtualisation is widely applied in IT infrastructure as a way of minimizing costs by increasing the utilization of existing resources. Virtualizing servers is often a good solution for small- to medium-scale applications. This technology is widely used for providing cost-effective web hosting services.

Application Virtualisation

Application virtualisation is software technology that encapsulates computer programs from the underlying operating system on which it is executed. A fully virtualized application is not installed in the traditional sense, although it is still executed as if it were. The application behaves at runtime like it is directly interfacing with the original operating system and all the resources managed by it, but can be isolated or sandboxed to varying degrees.

In this context, the term "virtualisation" refers to the artifact being encapsulated (application), which is quite different from its meaning in hardware virtualisation, where it refers to the artifact being abstracted (physical hardware).

Platform Virtualisation

Platform Virtualisation is a portable virtual desktop. It allows you to take the platform (Apps/OS and so on) with you so it appears same whatever platform you are on. Generally, very slow in practical implementation.

Storage Virtualisation

Storage virtualisation is the process of grouping the physical storage from multiple network storage devices so that it looks like a single storage device.

The process involves abstracting and covering the internal functions of a storage device from the host application, host servers or a general network in order to facilitate the application and network-independent management of storage.

Storage virtualisation is also known as cloud storage.

The management of storage and data is becoming difficult and time consuming. Storage virtualisation helps to address this problem by facilitating easy backup, archiving and recovery tasks by consuming less time. Storage virtualisation aggregates the functions and hides the actual complexity of the storage area network (SAN).

Storage virtualisation can be implemented by using software applications or appliances. There are three important reasons to implement storage virtualisation:

1. Improved storage management in a heterogeneous IT environment
2. Better availability and estimation of down time with automated management
3. Better storage utilization

Storage virtualisation can be applied to any level of a SAN. The virtualisation techniques can also be applied to different storage functions such as physical storage, RAID groups, logical unit numbers (LUNs), LUN subdivisions, storage zones and logical volumes, etc.

The storage virtualisation model can be divided into four main layers:

1. Storage devices
2. Block aggregation layer
3. File/record layer
4. Application layer

Some of the benefits of storage virtualisation include automated management, expansion of storage capacity, reduced time in manual supervision, easy updates and reduced downtime.

Quiz

1. Why would desktop virtualisation be useful for network manager.
2. What is RetroArch?
3. How does server virtualisation allow for the efficient use of hardware.

References

1. https://en.wikipedia.org/wiki/Virtualisation
2. https://en.wikipedia.org/wiki/Desktop_virtualisation
3. https://searchvirtualdesktop.techtarget.com/definition/desktop-virtualisation
4. https://en.wikipedia.org/wiki/Hardware_virtualisation
5. https://www.techopedia.com/definition/688/server-virtualisation
6. https://www.techopedia.com/definition/4798/storage-virtualisation
7. http://www.slashfilm.com/brett-leonard-interview-the-lawnmower-man/
8. https://digitalmediaservices.wordpress.com/2011/04/13/tron-legacy-from-imax-to-blu-ray/
9. http://tankarselfstorage.com.au/storing-solutions/three-storage-solutions-everyone-needs-know/
10. https://www.pluralsight.com/blog/it-ops/what-is-hypervisor
11. https://br.pinterest.com/pin/241153755021413608/

E-Waste

http://www.abc.net.au/ Penny Travers 22 Mar 2017

PHOTO: Michael Mullaney dismantles a computer at the MRI Drop Zone recycling facility in Canberra.

Millions of televisions, computers and mobile phones are discarded in Australia each year.

So what happens to all that electronic waste?

To prevent it ending up in landfill, the Federal Government introduced the National Television and Computer Recycling Scheme (NTCRS) in 2011.

Under the industry-funded scheme, companies that import or manufacture more than 5,000 televisions, computers or printers per year are required to help recover end-of-life products and divert them from landfill.

They do this by being members of one of four authorised recyclers which organise collection and recycling of e-waste on their behalf.

Each year 40,000 tonnes of computers, tablets, televisions, printers, scanners, photocopiers, keyboards and mice are recycled through the scheme.

How does the recycling process work?

There are more than 1,800 collection points across the country where householders and small businesses can drop off their unwanted televisions, computers and accessories for free.

The items are then collected by one of the authorised recyclers and dismantled.

"So for a TV, they'll remove the plastics, they'll remove the glass," MRI Drop Zone chief executive Rose Read said.

"If it's an old CRT (cathode ray tube) television, they'll separate the leaded glass from the normal glass.

"In the case of computers, they'll take off the plastics and the casings, they'll remove the circuit boards, separate the batteries, and then they are further recycled."

The leaded glass is processed at a smelter in Port Pirie to recover the lead.

Circuit boards go to Japan where precious metals like gold, silver and copper are recovered through a heat and chemical process.

"The batteries will go to a company in Korea where they are further processed to recover, in the case of lithium ion batteries, lithium and cobalt," Ms Read said.

"In the case of nickel-cadmium batteries, they recover cadmium."

Steel, copper and aluminium are also recycled in Australia while some plastics are sent to China for processing.

"As a result of those processes we recover more than 95 per cent of the materials that are collected," Ms Read said.

What about computer data safety?

Ms Read said there were no data safety issues under the NTCRS.

Australian e-waste in African dump

A computer monitor from St George Bank is found on a toxic e-waste dump in west Africa.

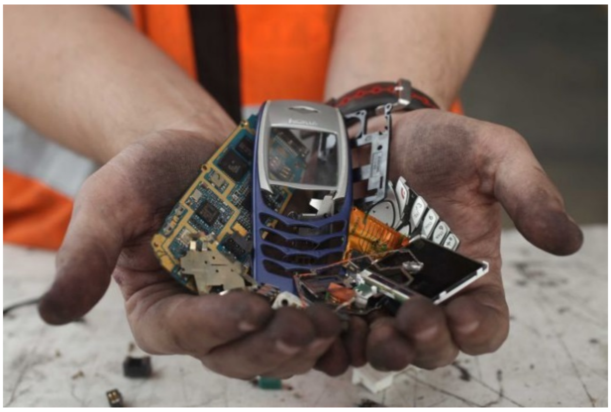
"All of the elements will be physically dismantled and destroyed, so if there is any data left on the computer it will be destroyed through the process," she said.

However if you decide to donate your computer for re-use, for example through a charity organisation, it is important to wipe the hard drive first.

"You need to be very careful about your own personal data on those computers; it's your responsibility to remove that data if you're going to pass it on down the reuse pathway," Ms Read said.

"Or you need to make sure the company you give it to does do data wiping and provides a data wiping certificate for you."

Can I recycle my old phone safely?



Mobiles phones are recycled under an industry-led voluntary scheme called Mobile Muster which began nearly 20 years ago.

The scheme is funded by major carriers and retailers through a 42-cent levy on each of the eight to 10 million handsets that are imported annually.

Each year Mobile Muster collects more than one million handsets, batteries and accessories for recycling.

"Everything collected by the program is recycled for resource recovery so it's a safe and secure way of disposing of old unwanted phones," Mobile Muster's Spyro Kalos said.

Like with televisions and computers, everything collected through the program is dismantled in Australia into its various components ahead of further processing.

Batteries are either processed in Singapore or South Korea.

Circuit boards go to Singapore for processing while plastics, steel and copper are smelted in Australia.

"Ninety-eight per cent of your mobile phone can be recycled back to its former material and put back into the supply chain for making new products," Mr Kalos said.

Clean up before dropping off

Mobile Muster does not reuse or resell any items that it collects, and all data is destroyed in the recycling process.

"We do encourage people before they drop off their phone to remove any data they want to keep like pictures, music files," Mr Kalos said.

"But if you can't turn the phone on and it doesn't work, it will get destroyed through our recycling process.

"We don't switch anything on or test anything to see if it is working as it comes through the program."

Consumers can drop off their old handsets and accessories at one of more than 3,000 Mobile Muster collection points across the country, including at all major retailer shopfronts.

"If you can't get to one of those drop-off points, you can actually pick up a free reply-paid satchel from Australia Post and post back your old unwanted phone and accessories," Mr Kalos said.

Why recycle e-waste?

Electronic waste is growing three times faster than any other type of waste in Australia.

"TVs that would stay in a house for seven, 10, 12 years are now being turned over in two years," Ms Read said.

She urged all householders to recycle their old electronics and prevent hazardous materials going to landfill.

"They are also made of finite resources which are becoming more and more scarce.

"The material in TVs and computers is very recyclable ... and can be recovered and reused as raw materials for new products including electronic products."

Earlier this month the Federal Government announced it was conducting a review of the Product Stewardship Act 2011 which oversees the NTCRS and Mobile Muster.

DVD and VCR players are not currently covered under the NTCRS and Ms Read said she was hopeful the scheme would be expanded to include television peripherals as part of the review.

To find your local e-waste recycling collection points, visit the ***Recycling Near You*** website.

https://recyclingnearyou.com.au/

Australian e-waste ending up in toxic African dump, torn apart by children

Rebecca Le Tourneau 10 Mar 2017

A computer monitor from St George Bank, destined for recycling in Australia, has been found on a toxic e-waste dump in west Africa, being pulled apart by children as young as five.

At Agbogbloshie dump, in Ghana's capital, Accra, children tear apart e-waste from Western nations with their hands, and burn circuit boards over open fires to melt out the precious metals.

Broken or redundant computers are considered hazardous waste and are illegal to ship out of Australia — so the discovery of the bank monitor raises serious questions about the integrity and regulation of Australia's growing e-waste problem.

St George Bank, wholly owned by Westpac, claims gold standard environmental stewardship.

It says it followed the "right processes to ensure the St George Bank monitor was despatched" to their recycling partner.

Ghanaian environmental reporter, Mike Anane, on assignment for RN's Background Briefing, discovered the broken monitor during a routine visit to check on the health and welfare of children working at Agbogbloshie dump, considered the worst dump in the world.

"Over 500 container loads of electronic waste are coming from these developed countries, including Australia, every single month," said Anane, speaking from Accra.

"Lately there is so much coming from Australia. I see about three container loads of electronic waste coming from Australia every single month.

"It is not just immoral, it is criminal to ship these things here."

Australia's pile of e-waste growing

Australia is one of the biggest consumers of electronics in the world, buying millions of items a year, which translates into almost 600,000 tonnes of e-waste annually.

The St George Bank monitor is part of that growing pile.

Background Briefing showed a video of a 13-year-old boy on the Agbogbloshie dump holding the bank's monitor to Don Quinn, operations manager at WorkVentures, which has the contract for Westpac's 15,000 e-waste items every year.

Mr Quinn said he did not believe the monitor passed through WorkVentures.

But when asked to confirm this by checking back through their asset register, the company advised by email a week later that the monitor had indeed been put through their system as e-waste in 2012.

"We looked up the sticker number on the monitor from the video you showed us and found we picked it up from St George Bank and decommissioned it in May 2012," the email said.

Because the monitor was deemed broken and not able to be fixed, it was sent on to another recycler.

Westpac 'concerned' by monitor's disposal

WorkVentures is a not-for-profit group that refurbishes computers and sells them to the disadvantaged and community groups at discounted prices.

Items that cannot be repaired are disposed of through other recyclers.

From what the ABC can tell, the monitor made its way from one of these other recyclers to Ghana.

Neither WorkVentures nor Westpac would agree to further interviews about the integrity of their e-waste disposal chain, or how the monitor ended up on the Agbogbloshie dump.

WorkVentures declined to identify the third-party Australian recycler they used, but said they severed that relationship in 2012 because their documentation was not up to standard.

WorkVentures also declined to say if there were other monitors and computers in the batch that found its way to Ghana.

Westpac said in an email: "We can't speculate on how the monitor ended up in Ghana five years after we dispatched it, however it is of great concern to us.

"We are determined to work with WorkVentures and our suppliers to understand how this has happened."

Recyclers contacted by rogue dealers

Westpac is not alone in losing control of its hazardous e-waste.

Anane says the lack of regulatory oversight is one of the reasons Australian e-waste is ending up on dumps in Africa.

He has been warning Western nations for years about the temptation for recyclers to avoid costly, legitimate disposal of what they collect.

Indeed, it seems very easy to find third-party recyclers who are prepared to illegally export hazardous e-waste from Australia.

Background Briefing made contact with an e-waste dealer in the Middle East who offered $500 per 1,000 kilograms of broken and smashed computers.

A major Australian national e-waste recycler, Geordie Gill, confirmed he was regularly contacted by rogue dealers hungry for his e-waste.

"On a fortnightly basis we will get emails from offshore and basically it comes down to: 'I will buy your e-waste from you'," Mr Gill said.

"The majority of the emails come from Africa and we've been offered up to $20,000 per shipping container of e-waste."

When asked if he believed there were Australian operators selling to these dealers Mr Gill said: "The opportunity is there. I would have to say yes."

Kids at dump 'suffer skin diseases, heart problems'

Background Briefing is not asserting that Westpac or their recyclers sent the broken St George Bank monitor to Ghana, but its appearance at Agbogbloshie dump reveals a lack of oversight.

Anane says the health problems suffered by the children exposed to e-waste are life-threatening.

"Each time I go to the dump ... I see the children with all these open sores, I see them with skin diseases," he said.

"They tell me, 'We cannot run, I have a problem with my heart, my heart beats faster, I cannot play football, I have headaches all the time.'

"It's obvious that these children will not live to see their 20th birthday. A lot of the kids disappear from the dumps and it's obvious what happens to them."

Quiz

1. What recycling facilities are available near where you live?
2. What features make E-Waste different that other typical house hold waste?
3. What can you do to minimise your impact in the production of waste?

Data Destruction

When getting rid of electronic media it is wise and in the case of many businesses a legal requirement that data stored on the media must be destroyed.

Data destruction is the process of destroying data stored on tapes, hard disks and other forms of electronic media so that it is completely unreadable and cannot be accessed or used for unauthorized purposes.

When data is deleted, it is no longer readily accessible by the operating system or application that created it. But the files aren't actually removed basically the operating system has just lowered the status of the file so that it won't be seen by the operating system and it can be overwritten by new files. The same is true with formatting a disk. If you accidently delete a file it can retrieved using data recovery software tools so long as you don't write new files to the disk (that over write the old files). So deleting a file is not enough.

Data can also be destroyed through degaussing, which destroys data on magnetic storage tapes and disk drives by changing the magnetic field. One caveat with this method is that the person who wishes to destroy data will need to know the exact strength of degaussing needed for each drive type.

Software can be also used to destroy data by systematically writing over the entire drive. When data is overwritten, a pattern of 1s and 0s is written over the original information. Sometimes a random pattern is used but a set pattern can also be used which allows for later verification that the drive was wiped by detecting the set pattern. Overwriting data once is enough for most situations. However, for high security applications multiple wipes may be required. This provides an extra measure of surety that the old data is destroyed and that there are no bit shadows present.

Storage media can also be destroyed by using a mechanical device called a shredder to physically mangle tape, optical media and hard disk drives. There can be major problems with physical destruction. First, it is prone to human error and manipulation. There is no reliable way to audit the physical destruction process. Typically a technician will scan or record a hard drive serial number before destroying a drive. But what's to prevent the technician from forgetting to scan a drive or, even worse, scanning the drive and then not destroying it?

Second, most methods of physical destruction leave large portions of the drive platter intact, even if the drive is inoperable. As previously discussed, data could still be recovered using forensic methods in such cases. Only pulverizing the disk to particles ensures the data is irrecoverable.

References

https://searchstorage.techtarget.com/definition/data-destruction

http://www.alfatekcorp.com/services/data-destruction-services

ICT Ethics

"It's not your work, Hannon — It's your attitude."

Code of Conduct

A code of conduct is a set of rules outlining the responsibilities of, or proper practices for, an individual, party or organization.

"Principles, values, standards, or rules of behaviour that guide the decisions, procedures and systems of an organization in a way that (a) contributes to the welfare of its key stakeholders, and (b) respects the rights of all constituents affected by its operations."

Example Code of Conduct Australian Public Service Commission

Code of Conduct

The Code of Conduct requires that an APS employee must:

- behave honestly and with integrity in connection with APS employment;
- act with care and diligence in connection with APS employment;
- when acting in connection with APS employment, treat everyone with respect and courtesy, and without harassment;
- when acting in connection with APS employment, comply with all applicable Australian laws;
- comply with any lawful and reasonable direction given by someone in the employee's Agency who has authority to give the direction;
- maintain appropriate confidentiality about dealings that the employee has with any Minister or Minister's member of staff;
- take reasonable steps to avoid any conflict of interest (real or apparent) and disclose details of any material personal interest of the employee in connection with the employee's APS employment
- use Commonwealth resources in a proper manner and for a proper purpose;
- not provide false or misleading information in response to a request for information that is made for official purposes in connection with the employee's APS employment;
- not improperly use inside information or the employee's duties, status, power or authority:
  1. to gain, or seek to gain, a benefit or an advantage for the employee or any other person; or
  2. to cause, or to seek to cause, detriment to the employee's Agency, the Commonwealth or any other person.
- at all times behave in a way that upholds the APS Values and Employment Principles, and the integrity and good reputation of the employee’s Agency and the APS;
- while on duty overseas, at all times behave in a way that upholds the good reputation of Australia; and
- comply with any other conduct requirement that is prescribed by the regulations.

Intellectual Property in Australia

http://www.ipaustralia.gov.au/

Intellectual property (IP) is the property of your mind or proprietary knowledge and can be an invention, a trade mark, a design or the practical application of your idea.

Protecting and managing your IP assets is critical when establishing your presence in the market. It is often the difference between success or failure.

Some forms of IP require a formal application and examination process before a right can be registered. Other rights come into play without the need for a registration process.

IP rights exist in many forms. In some cases they don't need to be registered in order to be of value. Each type of IP provides different competitive advantages for its owners.

Types of Protection

What’s protected	Type of IP protection	What it means	Example
Production designs	Registered design	The visual appearance of a product is protected, but not the way it works.	iPod kitchen appliances footwear fashion items
Logos, words letters, numbers, colours, a phrase, sound, scent, shape, picture, aspect of packaging or branding - or any combination of these	Trade mark	A trade mark identifies the particular goods or services of a trader as distinct from those of other traders.	Qantas Lonely Planet
Inventions and new processes	Patent	A patent protects how an invention works or functions.	Polymer bank notes Anti cervical cancer drug, Gardasil
Drawings, art, literature, music, film, broadcasts, computer programs	Copyright	The owner's original expression of ideas is protected, but not the ideas themselves.	Games of Thrones TV series
Trade secrets and confidential information	Other	These types of IP rights give creators certain rights and privileges depending on the type of IP protection.	Coca Cola has used trade secrets to keep its formula from becoming public for decades.
New plant variety	Plant Breeder's Rights	Plant Breeder's rights protect the commercial rights of new plant varieties	Cotton plants with insect resistance and the pink iceberg rose

Patents

A patent is a right that is granted for any device, substance, method or process that is new, inventive, and useful.

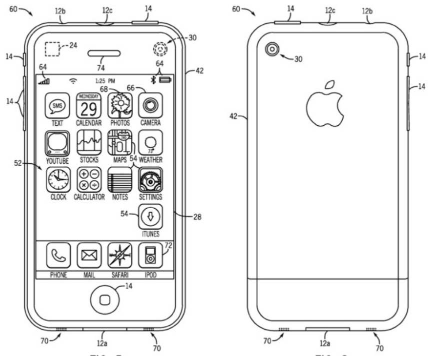


FIG. 5 FIG. 6

A patent is legally enforceable and gives you (the owner), exclusive rights to commercially exploit the invention for the life of the patent.

You can apply for patent protection for a range of inventions. These include traditional inventions such as appliances, mechanical devices and so on.
However you may also protect things such as:

- computer-related inventions
- business methods
- biological inventions
- micro-organisms and other biological materials

What cannot be patented?

You cannot patent human beings or the biological process for their generation, artistic creations, mathematical models, plans, schemes or other purely mental processes.

Trade Marks

A trade mark is used to distinguish the goods and services of one trader from those of another.

A trade mark is a right that is granted for a letter, number, word, phrase, sound, smell, shape, logo, picture and/or aspect of packaging. A registered trade mark is legally enforceable and gives you exclusive rights to commercially use, licence or sell it for the goods and services that it is registered under.

A trade mark is a way of identifying a unique product or service and it can be your most valuable marketing tool. Sometimes called a brand, your trade mark is your identity - the way you show your customers who you are. There is a difference between trade marks, business, company and domain names.

A trade mark can be a letter, number, word, phrase, sound, smell, shape, logo, picture, aspect of packaging or any combination of these.

An example of a well-known trade mark is QANTAS and the flying kangaroo.

Designs

A design refers to the features of shape, configuration, pattern or ornamentation which give a product a unique appearance, and must be new and distinctive.

Design registration is intended to protect designs which have an industrial or commercial use.

A registered design gives you, the owner, exclusive rights to commercially use it, licence or sell it.

A design is the overall appearance of a product. This includes the shape, configuration, pattern and ornamentation which, when applied to a product, give it a unique visual appearance.

A registered design that has been certified after it has been examined provides you with enforceable rights to use, licence, sell or protect your design.

A product is something that is manufactured or handmade. Importantly the mechanics of how a product works or operates is not protected by designs legislation, but may be protected using a patent.

What can be registered?

A design can be registered in Australia, provided it is both:

- 'new' - meaning it must not be identical to any design previously disclosed anywhere in the world (including on the internet), nor any design previously used in Australia; and
- 'distinctive' - meaning it must not be substantially similar in overall impression to any design previously published anywhere in the world (including on the internet), nor any design previously used in Australia.

Some designs are not registrable by law. These include designs for medals, layouts for integrated circuits, Australian currency and scandalous designs. A scandalous design is one which is shocking or offensive to the public or an individual's sense of propriety or morality.

If you have already publicly disclosed your design (e.g. exhibited, sold copies, posted your design on a website), you may not be able to register it as it may not be considered to be new and distinctive. You can use a formal publication process in the Australian Official Journal of Designs to prevent others from obtaining certification of a similar design.

Plant Breeder’s Rights

Plant Breeder's Rights (PBR) are used to protect new varieties of plants that are distinct, uniform and stable. I’m going to go out on a limb and say we don’t need to cover these in IT.

An Introduction to Copyright in Australia

This material had been adapted from Australian Copyright Council information sheets - www.copyright.org.au.

Key points

- Copyright protection is free and applies automatically when material is created.
- There is NO registration system for copyright in Australia.
- Copyright does not protect ideas, information, styles or techniques.
- Copyright does not protect names, titles or slogans.
- There are no general exemptions from copyright law for non-profit organisations.
- There are some situations where copyright law allows people to use copyright material without permission for their own personal use, but these are narrow and specific.
- Australian copyright law applies to actions that take place in Australia, even if the material used was created or first published in another country.

Copyright law

Copyright law creates incentives for people to invest their time, talent and other resources in creating new material – particularly cultural and educational material, which benefits society.

In Australia, copyright law is set out in the Copyright Act 1968. This is federal legislation, and applies throughout Australia.

What does copyright protect?

Copyright protects:

- **textual material** ("literary works") such as journal articles, novels, screenplays, poems, song lyrics and reports;
- **computer programs**
- **compilations** such as anthologies – the selection and arrangement of material may be protected separately from the individual items contained in the compilation;
- **artistic works** such as paintings, drawings, cartoons, sculpture, craft work, architectural plans, buildings, photographs, maps and plans;
- **dramatic works** such as choreography, screenplays, plays and mime pieces;
- **musical works:** that is, the music itself, separately from any lyrics or recording;
- **cinematograph films:** the visual images and sounds in a film, video or DVD;
- **sound recordings:** the particular recording itself is protected by copyright, in addition to, for example, the music or story that is recorded;
- **broadcasts:** TV and radio broadcasters have a copyright in their broadcasts, which is separate from the copyright in the films, music and other material which they broadcast; and
- **published editions:** publishers have copyright in their typographical arrangements, which is separate from the copyright in works reproduced in the edition.

What is not protected by copyright?

Ideas, concepts, styles, techniques and information

Copyright does not protect ideas, concepts, styles, techniques or information.

For example, if you write an outline of your idea for a TV show, the written text will be protected by copyright and, generally, someone wanting to reproduce it would need your permission. However, someone else could write their own script, using your ideas, without necessarily infringing your copyright.

Names, titles and slogans

Some things are too small or unoriginal to be protected by copyright. For example, single words (even invented words), names, titles, slogans and headlines are unlikely to be protected by copyright.

In some cases, however, someone using a name, title or slogan which is already being used by someone else may run into problems with other areas of law, such as trade marks.

People

People and people's images (images of their face or body) are not protected by copyright.

Copyright protection is automatic

You do not apply for copyright in Australia, and there is no system of registration here. Nor are there any forms to fill in, or fees to be paid.

You do not need to publish your work, put a copyright notice on it, or do anything else before your work is covered by copyright – the protection is free and automatic, from the time a work is first written down or recorded in some way. For example, as soon as a poem is written, or a song is recorded, it is protected.

How long does copyright last?

Until 1 January 2005, copyright generally lasted for the life of the relevant creator plus 50 years. Under the Free Trade Agreement with the United States, Australia agreed to extend the general duration of copyright so now it lasts for the life of the creator plus 70 years.

Note, however, that the duration of copyright varies from country to country.

Who owns copyright?

The general rule under the Act is that the first owner of copyright is the creator of the work. Where a work is made by an employee (rather than a freelancer) as part of that person's job, the employer usually owns copyright.

Infringement of copyright

Using copyright material without the copyright owner's permission.

Moral Rights

Individual creators have rights called “moral rights”, whether or not they own copyright. These are
the rights to:

- be attributed as the creator of their work;
- take action if their work is falsely attributed as being someone else’s work or is altered by someone else but attributed as if it were unaltered; and
- take action if their work is distorted or treated in a way that is prejudicial to their honour or reputation.

Performers’ rights

People generally need to get consent from a performer to record or broadcast a live performance. The performer’s consent may also be needed to use an unauthorised recording of a performance, or to use an authorised sound recording of a performance on a film soundtrack.

Fair Dealing

The Copyright Act 1968 allows people to use copyright material without the copyright owner’s permission in certain situations.

When are the fair dealing exceptions relevant?

If you are using copyright material in ways reserved to the copyright owner, you generally need permission to use it, unless:

- copyright has expired; or
- you are using less than a substantial part of the material; or
- one of the specific exceptions to infringement applies.

The “fair dealing” exceptions to infringement

There is no general exception for using copyright material simply because you think it is fair or because you are not making a profit. The Copyright Act allows you to use copyright material without permission if your use is a “fair dealing” for one of the following purposes:

- research or study;
- criticism or review;
- parody or satire;
- reporting news; or
- professional advice by a lawyer, patent attorney or trade marks attorney.

Fair dealing for research or study

Use of copyright material for the purpose of research or study will not infringe copyright, provided
the use is “fair”.

If you use less than a certain amount of a copyright item for research or study, the use is deemed to be fair.

Fair dealing for criticism or review

People can use copyright material for the purpose of criticism or review without infringing copyright, provided they acknowledge the author and title of the work. The criticism or review may relate to the work being used or to other material. For example, television film reviewers may show clips from other films as well as the one they are reviewing, in making their criticism or review.

The purpose of criticism or review must be genuine.

Fair dealing for parody or satire

You can use copyright material for the purposes of parody and satire.

Fair dealing for reporting news

Copyright material may be used in reporting news in a newspaper, magazine or similar periodical, or in a film, or by means of a broadcast. The author and title of the work must be acknowledged. However, music in news reports is not covered by this provision, unless the playing of the music is part of the news being reported.

Fair dealing for professional advice

It is not an infringement of copyright to use copyright material for the purpose of giving professional advice by a lawyer, patent attorney or trademarks attorney.

Piracy

United Nations Convention on the Law of the Sea 1982

Article 101
Definition of
Piracy
Piracy consists of any of the following acts:

a) any illegal acts of violence or detention, or any act of depredation, committed for private ends by the crew or the passengers of a private ship or a private aircraft, and directed:
   - on the high seas, against another ship or aircraft, or against persons or property on board such ship or aircraft;
   - against a ship, aircraft, persons or property in a place outside the jurisdiction of any State;

b) any act of voluntary participation in the operation of a ship or of an aircraft with knowledge of facts making it a pirate ship or aircraft;

…wait do you think they meant media piracy?

Piracy Copying and Downloading

Material on the internet like music, videos and photographs are protected by copyright.

- In many cases, permission to download material from a website is given on the site itself, but the fact that material is available to be viewed on a website, or is accessible using P2P software or networks over the internet does not, by itself, mean that you can use it as you wish.
- Do not assume that material you come across whilst online has been uploaded with the copyright owner's permission: if the material is an infringing copy, or the person hosting the site was not in a position to give a permission on behalf of the copyright owner, you risk infringing copyright if for example, you download or copy material.

When might you infringe copyright?

Copyright owners have a number of rights, including the right to control the "reproduction" of their material and the right to control the "communication" of that material "to the public" (which includes uploading, posting or downloading material online).
Generally, you will infringe copyright if you use copyright material in one of the ways that copyright owners control, without permission, in circumstances to which no exception applies. For example, you might infringe copyright if you do any of the following:

- print or distribute copies of material from a website without express or implied permission;
- uploading, downloading or sharing unauthorised copies of a movie, album or software;
- save material from a website onto your hard drive without express or implied permission;
- access media through circumnavigation of protection.

Can I download music, movies and software files from anywhere on the internet?

If a copyright owner has given permission, you may download a file from an authorised website or through file sharing networks.

- Many major record companies and independent bands and labels offer music downloads through their own sites, through online music stores (like iTunes or Bigpond Music) or as an option on Youtube or MySpace;
- Several TV stations make programs they broadcast available online;
- Developers may offer an option to download their software via BitTorrent.

However, the fact that you can locate a file on a website or a file-sharing network does not mean that you can copy it, even for personal use. There are a number of sites that claim copying material over networks using their software is legal. However, these statements are not true: you will generally need permission to copy someone else's material.

In Australia, there are at least three people who have ended up with criminal records as a result of illegal file sharing of music files. Another man received a criminal conviction after he recorded The Simpsons Movie in an Australian cinema on his mobile phone and placed a copy on a US-based website before the US release date. In yet another case, a provider of P2P software settled a court dispute by paying over $150 million to record companies whose copyright in music files was infringed.

How can I tell if I am accessing a website with illegitimate (infringing) material?

You'll need to adopt a common sense approach in working out whether or not content on a site is likely to be infringing. The legitimacy of material will be questionable where it is commercial material that is available for free from third party sources, is of low quality and/or is available before

Civil actions and remedies

If copyright has been infringed, the copyright owner is entitled to commence an action in court and various remedies may be awarded. Action must be taken within 6 years of the date the infringement took place.
A court may also order a person who loses a case to pay another party's legal costs. An award of costs will not, however, always cover the full amount the person who won the case has to pay their legal representatives.

Criminal offences involving copyright infringement

Not every infringement of copyright is a criminal offence. Generally, only infringements of copyright that involve commercial dealings or infringements that are on a commercial scale are criminal. For example, under the Copyright Act, it may be an offence to:

- cause infringement on a commercial scale, even if the person doing this makes no financial gain;
- make "an article" that infringes copyright for sale or hire or to obtain a commercial advantage or profit, or to sell or otherwise deal with such an article, sometimes with the intention of obtaining a commercial advantage or profit;
- import "an article" that infringes copyright for trade purposes, or to obtain a commercial advantage or profit;
- distribute "an article" that infringes copyright for trade purposes, or to obtain a commercial advantage or profit, or for any other purpose that prejudicially affects the copyright owner; or
- possess an article that infringes copyright, for specified commercial purposes, including for distribution to obtain a commercial advantage or profit or in a way that prejudicially affects the copyright owner.

Penalties

Penalties vary, and depend on whether it is an individual or a corporation that is convicted. For some indictable offences, an individual who is guilty may be fined up to $93,500 or imprisoned for up to 5 years, or both. For importation of material that infringes copyright, fines of up to $71,500 and/or imprisonment for 5 years may be imposed on an individual. Penalties can be much higher where the infringement involves the digitisation of copyright material from hardcopy (for example, from cassette to CD or from video to DVD). An individual who is found guilty of a summary offence may be fined up to $13,200 or imprisoned for up to 2 years or both.

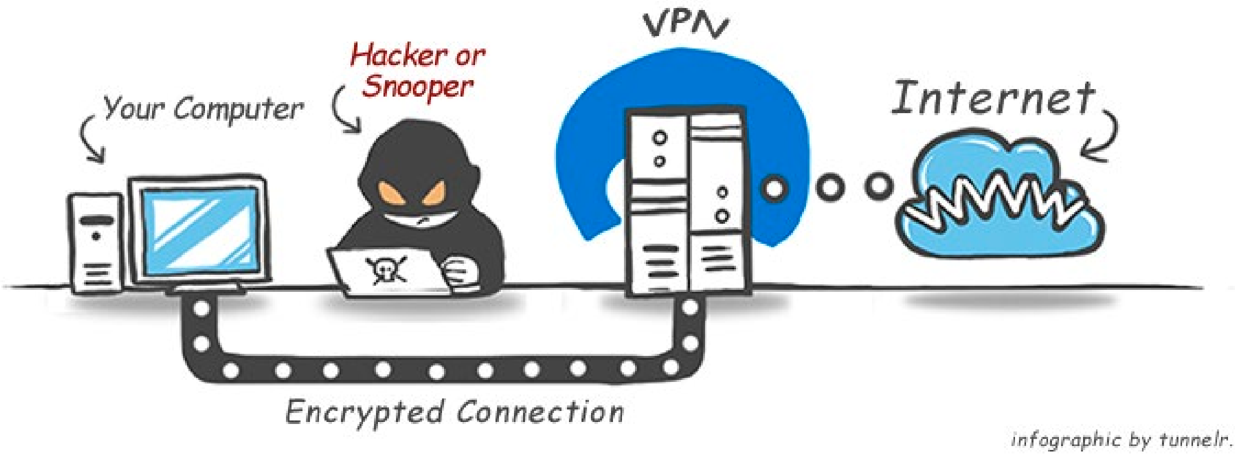
A corporation may be fined up to 5 times the amount of a maximum fine.
Where an individual is convicted of a strict liability offence (small scale piracy), the maximum penalty is $6,600. However, where police issue an infringement notice, the maximum amount of the penalty for an individual is $1,320. In some cases, the offender must have already forfeited infringing copies and illegal devices to the Commonwealth. From an offender's point of view, some of the benefits of the infringement notice scheme include that, so long as the infringement notice is not withdrawn and the offender complies with the other requirements of the scheme, the offender is not taken to have admitted guilt, nor to have been convicted of the offence, and no prosecution can be brought in relation to it.

Where a matter goes to court, courts can order that circumvention devices, infringing copies, and devices and equipment used to infringe, be destroyed, or handed over to relevant copyright owners, or otherwise dealt with.

VPNs

For people looking to hide their activities from prying eyes, a foreign-based virtual private network is the solution. It makes a secure encrypted connection from your computer, out through your ISP and across the ocean to a VPN server on the other side of the world. VPNs are designed to protect honest people against hackers but, like any tool, they can also be used by people looking to evade the authorities.

There's no practical way the government can crack down on movie pirates using a VPN, because VPNs have so many legitimate uses. It would be like shutting down the freeways during peak hour – stopping millions of people getting to work in the hope of catching a handful of wrongdoers. The Feds might be able to see that you're using a VPN and how much data you're shifting, but they can't see exactly what you're doing. This of course depends on your faith in encryption but, even if the spooks do have a backdoor into VPNs, they're certainly not going to reveal it just so they can catch you downloading Dallas Buyers Club.



Exercise

1. Why do we have laws to protect copyright?
2. Discuss with you peers whether or not they download media illegally.
3. What are their reasons for downloading/not downloading media?
4. How could someone protect themselves against being caught downloading copyright material illegally.
5. Give an example of something that would be covered under a patent, a registered trademark, a registered design and what the hell a new plant variety. And no you can't just use the ones mentioned in this task sheet.
6. Is a comedian doing a skit on something covered under copyright are the required to pay the copyright owners? Why/Why not?
7. Find an example of a code of conduct. Give a link and a brief summary of the areas it covers.

References

https://www.cartoonstock.com/directory/p/paper_air_plane.asp
https://en.wikipedia.org/wiki/Code_of_conduct
https://www.apsc.gov.au/code-conduct
http://www.ipaustralia.gov.au/
https://ideas2market.wordpress.com/2013/03/11/patents-the-road-to-riches-or-ruin/
https://techcrunch.com/2018/05/24/jury-finds-samsung-owes-apple-539m-in-patent-case-stretching-back-to-2011/
www.copyright.org.au
http://designlab.net.au/copyright-a-logo/
https://www.che-fare.com/dal-web/i-nodi-al-pettine-del-copyright/
http://tvtropes.org/pmwiki/pmwiki.php/GoodAngelBadAngel/WesternAnimation
https://www.youtube.com/channel/UCsgv2QHkT2ljEixyulzOnUQ
http://familyguy.wikia.com/wiki/Category:Star_Wars_episodes
http://vpnmachine.com/best-vpn-softwares-for-mac/
https://www.costumebox.com.au/
https://ideanav.co.za/how-to-trade-mark/
https://www.yahoo.com/entertainment/mvps-of-horror-the-little-shop-of-horrors-puppet-master-who-brought-the-blood-thirsty-plant-to-life-174643066.html
http://dublinhire.com/cheap-automatic-car-hire-from-dublin-airport
https://listverse.com/2017/02/27/10-shocking-cases-of-modern-piracy/
http://www.slashfilm.com/trailer-aardman-animation-pirates-band-misfits/

Cloud Computing

https://azure.microsoft.com/en-au/overview/what-is-cloud-computing/

Simply put, cloud computing is the delivery of computing services – servers, storage, databases, networking, software, analytics and more – over the Internet ("the cloud"). Companies offering these computing services are called cloud providers and often charge for cloud computing services based on usage, similarly to traditional services such as electricity.

Benefits of cloud computing

Cloud computing is a big shift from the traditional way businesses think about IT resources. What is it about cloud computing? Why is cloud computing so popular? Here are 6 common reasons why organisations are turning to cloud computing services:

1. Cost
Cloud computing eliminates the capital expense of buying hardware and software, and setting up and running on-site data centres – the racks of servers, the round-the-clock electricity for power and cooling, the IT experts for managing the infrastructure. It adds up quickly.

2. Speed
Most cloud computing services are provided as self service and on demand, so even vast amounts of computing resources can be provisioned in minutes, typically with just a few mouse clicks, giving businesses a lot of flexibility and taking the pressure off capacity planning.

3. Global scale
The benefits of cloud computing services include the ability to scale elastically. In cloud speak, that means delivering the right amount of IT resources – for example, more or less computing power, storage, bandwidth – exactly when it's needed, and from the right geographic location.

4. Productivity
On-site data centres typically require a lot of "racking and stacking" – hardware setup, software patching and other time-consuming IT management chores. Cloud computing removes the need for many of these tasks, so IT teams can spend time on achieving more important business goals.

5. Performance
The biggest cloud computing services run on a worldwide network of secure data centres, which are regularly upgraded to the latest generation of fast and efficient computing hardware. This offers several benefits over a single corporate data centre, including reduced network latency for applications and greater economies of scale.

6. Reliability
Cloud computing makes data backup, disaster recovery and business continuity easier and less expensive, because data can be mirrored at multiple redundant sites on the cloud provider's network.

Disadvantages of Cloud Computing

1. Downtime
Downtime is often cited as one of the biggest disadvantages of cloud computing. Since cloud computing systems are internet-based, service outages are always an unfortunate possibility and can occur for any reason.

2. Loss of Control
You are, essentially, trusting another party to take care of your data. You are trusting that they will maintain their data centres and servers with the same care as you would, if not more. You have to trust that your provider's data centres are compliant and secured both physically and online. Some find the lack of in-house control of the server unnerving.

3. Security and privacy
In cloud computing, every component is online, which exposes potential vulnerabilities. Although cloud service providers implement the best security standards and industry certifications, storing data and important files on external service providers always opens up risks. Any discussion involving data must address security and privacy, especially when it comes to managing sensitive data.

4. Limited control and flexibility
Since the cloud infrastructure is entirely owned, managed, and monitored by the service provider, it transfers minimal control over to the customer.

To varying degrees (depending on the particular service), cloud users may find they have less control over the function and execution of services within a cloud-hosted infrastructure.

5. Vendor lock-in
Vendor lock-in is another perceived disadvantage of cloud computing. Easy switching between cloud services is a service that hasn't yet completely evolved, and organizations may find it difficult to migrate their services from one vendor to another.

Types of Cloud Computing

http://en.wikipedia.org/wiki/Cloud_computing

Cloud computing involves deploying groups of remote servers and software networks that allow centralized data storage and online access to computer services or resources. Clouds can be classified as public, private. Public and private clouds aren't mentioned in the syllabus but it is useful to be aware of them.

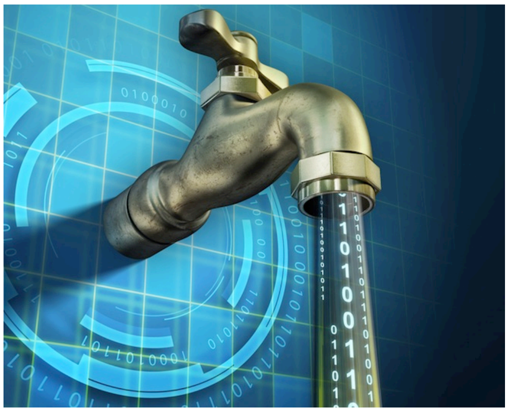
Private Cloud

Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third-party, and hosted either internally or externally. Self-run data centres are generally capital intensive. These assets have to be refreshed periodically, resulting in additional capital expenditures. They do provide the control that public clouds lack and can eliminate speed issues also.

Public cloud

A cloud is called a "public cloud" when the services are rendered over a network that is open for public use. Technically there may be little or no difference between public and private cloud architecture, however, security consideration may be substantially different for services.

Security and Privacy



Cloud computing poses privacy concerns because the service provider can access the data that is on the cloud at any time. It could accidentally or deliberately alter or even delete information. Many cloud providers can share information with third parties if necessary for purposes of law and order even without a warrant. That is permitted in their privacy policies which users have to agree to before they start using cloud services. Solutions to privacy include policy and legislation as well as end users' choices for how data is stored. Users can encrypt data that is processed or stored within the cloud to prevent unauthorized access.

There is the problem of legal ownership of the data (If a user stores some data in the cloud, can the cloud provider profit from it?). Many Terms of Service agreements are silent on the question of ownership.

Physical control of the computer equipment (private cloud) is more secure than having the equipment off site and under someone else's control (public cloud). This delivers great incentive to public cloud computing service providers to prioritize building and maintaining strong management of secure services. Some small businesses that don't have expertise in IT security could find that it's more secure for them to use a public cloud.

Fundamentally private cloud is seen as more secure with higher levels of control for the owner, however public cloud is seen to be more flexible and requires less time and money investment from the user.

Exercises

1. What is the difference between a public and private clouds? What are the benefits and disadvantages of each?
2. What are some common security concerns mentioned by the articles regarding cloud computing?
3. What are some of the possible threats to confidentiality using cloud computing?
4. How can data encryption help reduce these threats?

References

https://azure.microsoft.com/en-au/overview/what-is-cloud-computing/
http://en.wikipedia.org/wiki/Cloud_computing
http://computer.howstuffworks.com/cloud-computing/cloud-computing.htm/printable
https://blog.skrots.com/what-is-cloud-computing/
https://cloudacademy.com/blog/disadvantages-of-cloud-computing/
https://centretechnologies.com/5-cloud-computing-disadvantages/

Convergence

Digital convergence refers to the convergence of four industries;

- Information Technologies,
- Telecommunication,
- Consumer Electronics, and
- Entertainment.

Previously separate technologies such as voice (and telephony features), data (and productivity applications), and video can now share resources and interact with each other synergistically.

Telecommunications convergence or **network convergence** involves the converging of previously distinct media such as telephony and data communications into common interfaces on single devices, such as most smart phones can make phone calls and search the web.

The rise of digital communication in the late 20th century has made it possible for media organizations (or individuals) to deliver text, audio, and video material over the same wired, wireless, or fiber-optic connections. At the same time, it inspired some media organizations to explore multimedia delivery of information. Today, we are surrounded by a multi-level convergent media world where all modes of communication and information are continually reforming to adapt to the enduring demands of technologies, "changing the way we create, consume, learn and interact with each other".

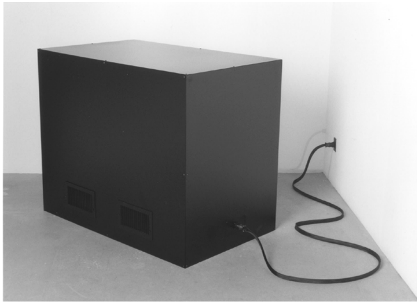
Convergence in this instance is defined as the interlinking of computing and other information technologies, media content, and communication networks that has arisen as the result of the evolution and popularization of the Internet as well as the activities, products and services that have emerged in the digital media space. Many experts view this as simply being the tip of the iceberg, as all facets of institutional activity and social life such as business, government, art, journalism, health, and education are increasingly being carried out in these digital media spaces across a growing network of information and communication technology devices.

Convergence services, such as VoIP, IPTV, Mobile TV, Smart TV, etc., will replace the old technologies and are a threat to the current service providers. IP-based convergence is inevitable and will result in new service and new demand in the market.

History

Communication networks were designed to carry different types of information independently. Radio was designed for audio, and televisions were designed for video. The older media, such as television and radio, are broadcasting networks with passive audiences. Convergence of telecommunication technology permits the manipulation of all forms of information, voice, data, and video. Telecommunication has changed from a world of scarcity to one of seemingly limitless capacity. Consequently, the possibility of audience interactivity morphs the passive audience into an engaged audience.

Some expect that we will eventually access all media content through one device, or "black box". As such, media business practice has been to identify the next "black box" to invest in and provide media for. This has caused a number of problems.



Firstly, as "black boxes" are invented and abandoned, the individual is left with numerous devices that can perform the same task, rather than one dedicated for each task. For example, one may own both a computer and a video games console, subsequently owning two DVD players. This is contrary to the streamlined goal of the "black box" theory, and instead creates clutter.

Secondly, technological convergence tends to be experimental in nature. This has led to consumers owning technologies with additional functions that are harder, if not impractical, to use rather than one specific device. For example, Intel has created a surfboard with an in-built laptop. Additionally, LG has created a microwave with a television screen. Many people would only watch the TV for the duration of the meal's cooking time, or whilst in the kitchen, but would not use the microwave as the household TV. These examples show that in many cases technological convergence is unnecessary or unneeded.

Despite the creation of "black boxes", intended to perform all of one's tasks, the trend is to use devices that can suit the consumer's physical position.

Due to the variable utility of portable technology, convergence occurs in high end mobile devices. They incorporate multimedia services, GPS, Internet access, and mobile telephony into a single device, heralding the rise of what has been termed the "smart phone," a device designed to remove the need to carry multiple devices.

Convergence of media occurs when multiple products come together to form one product with the advantages of all of them. The idea of complete convergence has become known to be a fallacy because of the inability to actually put all technical pieces into one. For example, while people can have e-mail and Internet on their phone, they still want full computers with Internet and e-mail in addition.

Convergence is till the major trend though. For example, the Playstation 4 and Xbox 1 are not only a games consoles, but media centres with a web browser, content recording and streaming and social networking tools. Mobile phones are another good example, in that they increasingly replacing dedicated digital cameras, mp3 players, camcorders, voice recorders, and other devices. This type of convergence is popular for the consumer, it means more features in less space; for media conglomerates it means remaining competitive.

However, convergence has a downside. Particularly in initial forms, converged devices are frequently less functional and reliable than their component parts (e.g., a mobile phone's web browser may not render some web pages correctly, due to not supporting certain rendering methods, such as the iPhone browser not supporting Flash content). As the number of functions in a single device escalates, the ability of that device to serve its original function decreases. For example, the iPhone (which by its name implies that its primary function is that of a mobile phone) can perform many different tasks, but does not feature a traditional numerical pad to make phone calls. Instead, the phone features a touchpad, which some users find more troublesome.

Interwebs

The role of the internet has changed from its original use as a communication tool to provide easier and faster access to information, mainly through a broadband connection. The television, radio and newspapers were the world's mediums for accessing news and entertainment; now, all three mediums have converged into one, and people all over the world can read and hear news and other information on the Internet. The convergence of the internet and conventional TV become popular in the 2010s, through smart TV, also sometimes referred to as "Connected TV" or "Hybrid TV”.

Smart TV is used to describe the current trend of integration of the internet and Web 2.0 features into modern television sets and set-top boxes, as well as the technological convergence between computers and these television sets or set-top boxes. These new devices most often also have a much higher focus on online interactive media, Internet TV, over-the-top content, as well as on-demand streaming media, and less focus on traditional broadcast media like previous generations of television sets and set-top boxes always have had.

Media Convergence

Media convergence, in reality, is more than just a shift in technology. It alters relationships between industries, technologies, audiences, genres and markets. Media convergence changes the rationality media industries operate in, and the way that media consumers process news and entertainment. Media convergence is essentially a process and not an outcome, so no single black box controls the flow of media. With proliferation of different media channels and increasing portability of new telecommunications and computing technologies, we have entered into an era where media constantly surrounds us.

Exercise

Name 20 things that a mobile phone is used for that would have previously been done by specialized technologies. Let me start you off.

Can you name 30?

References

https://en.wikipedia.org/wiki/Technological_convergence
https://pngtree.com/freepng/old-phone_2235241.html
https://www.mirror.co.uk/news/uk-news/black-white-tv-still-watched-7077902
http://content.time.com/time/specials/packages/article/0,28804,1940424_1940627_1940622,00.html
https://www.bestbuy.com/site/microsoft-xbox-one-console-pre-owned-black/4334012.p?skuId=4334012
https://thoughtcatalog.com/brianna-lerose/2014/09/3-practical-ways-to-stop-staring-at-your-goddamn-cell-phone/
https://cbutcherpiat.wordpress.com/2012/09/02/weapon-of-mass-distraction/

Databases

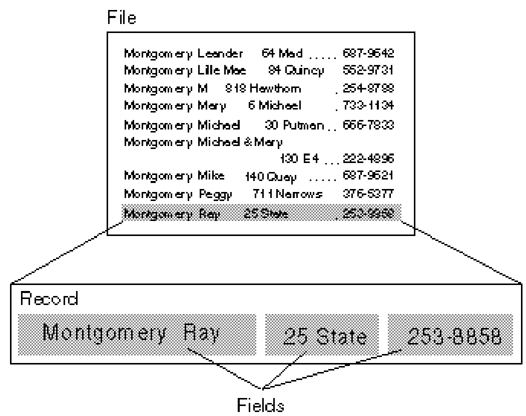
A database is a collection of information organized in such a way that a computer program can quickly select desired pieces of data.

Data

Data is distinct pieces of information, usually formatted in a special way.

Field

In computer science, data that has several parts, known as a record, can be divided into fields. Relational databases arrange data as sets of database records, also called rows. Each record consists of several fields; the fields of all records form the columns. Examples of fields: name, gender, hair colour.



Primary Key

A primary key is a special field designated to uniquely identify all table records. A primary key's main features are:

- It must contain a unique value.
- It cannot contain null values.

Foreign Key

A foreign key is a field a relational database table that provides a link between data in two tables. It acts as a cross-reference between tables because it references the primary key of another table.

In database management systems, a complete set of information. Records are composed of fields, each of which contains one item of information. A set of records constitutes a file. For example, a personnel file might contain records that have three fields: a name field, an address field, and a phone number field.

In relational database management systems, records are called tuples.

Relation

A relational database is a set of tables containing data fitted into predefined categories. Each table (which is sometimes called a relation) contains one or more data categories in columns.

Atomicity

Atomicity is a feature of databases systems dictating where a transaction must be all-or-nothing. That is, the transaction must either fully happen, or not happen at all. It must not complete partially.

The definition of what constitutes an atomic transaction is decided by its context or the environment in which it being implemented. For example, in an online airline-booking system, a booking may consist of 2 separate actions that together form a transaction - paying for the seat, and reserving the seat for the customer who's just paid. Business logic dictates that these two, though distinct and separate actions, must occur together. If one happens without the other, problems can occur. For example, the system may reserve the same seat for two separate customers.

It is essential that a database system that claims to offer atomicity be able to do so even in the face of failure in power supply or the underlying operating system or application that uses the database.

Online Vs Local Data Management

Local Storage

Speed is one of the main advantages to local storage. Storing data on external hard drives is faster than uploading to the cloud. You also have full control of your backups, which means better control of who accesses your data. Disconnecting the drives from the internet access makes your data safe from attacks.

Because you have complete control of the data it makes it easier to perform your obligations under the privacy act as well as monitor who has accessed and changed data when.

The downsides to local storage.

Creating and maintaining a local storage system is expensive. The hardware and software can cost thousands of dollars depending on how much space you need. Upgrading can also be costly. When you need more space, it also means buying more hardware.

An on-site disaster will destroy your local backups. Having off-site backups is smart, however, it requires more work to maintain them. Each time your data needs updating, your off-site backups need updating as well. This means bringing them in for updating which is a hassle and time-consuming.

Online Storage

Online data management is all about using Internet-based software systems to securely collect and manage your organisational data, giving you up-to-date alerts, insights and reports that will ensure you're always in control and are making the right decisions - no matter if you're in or out of the office.

Data management comprises data:

- Collection
- Storage
- Backup
- Secure Access
- Monitoring
- Reporting
- Analysis
- Alerting
- Integration
- Quality
- Reliability

Online Data Management can assist with these common problems:

- Poor visibility over your operations
- Difficulty collecting data from many different sources
- Not having decision-orientated data at your fingertips
- IT systems that don't "talk to each other"
- Reliance on unmaintainable Excel spreadsheets and Microsoft Access® databases
- Difficult to track manual processes
- Human errors and mistakes
- Poor data quality
- Complex data

Correct and up-to-date data is vital to the operation of all modern businesses. Without it, opportunities are missed, mistakes are made and both time and money is wasted. But often the data you need is stored in different systems and locations, is manually handled or is "dirty" and needs cleansing.

This is where a properly designed data management system can help. It can bring data together from different sources and replace outdated IT systems with a centralised solution. It can give you better access, control and visibility over the data that's important to your organisation and give you up-to-date insights and analysis to ensure you're making the right decisions.

Centralized database

A centralized database is a database that is located, stored, and maintained in a single location. This location is most often a central computer or database system, for example a desktop or server CPU, or a mainframe computer. In most cases, a centralized database would be used by an organization (e.g. a business company) or an institution (e.g. a university.) Users access a centralized database through a computer network which is able to give them access to the central CPU, which in turn maintains to the database itself.

All of the information stored on the Centralized Database is accessible from a large number of different points, which in turn creates a significant amount of both advantages and disadvantages.

Advantages

Centralized databases hold a substantial amount of advantages against other types of databases. Some of them are listed below:

- Data integrity is maximized and data redundancy is minimised,as the single storing place of all the data also implies that a given set of data only has one primary record. This aids in the maintaining of data as accurate and as consistent as possible and enhances data reliability.
- Generally better data security, as the single data storage location implies only a one possible place from which the database can be attacked and sets of data can be stolen or tampered with.
- Better data preservation than other types of databases due to often-included fault-tolerant setup.
- Easier for using by the end-user due to the simplicity of having a single database design.
- Generally easier data portability and database administration.
- More cost effective than other types of database systems as labor, power supply and maintenance costs are all minimized.
- Data kept in the same location is easier to be changed, re-organized, mirrored, or analyzed.
- Updates to any given set of data are immediately received by every end-user.

Disadvantages

Centralized databases also have a certain amount of limitations, such as those described below:

- Centralized databases are highly dependent on network connectivity. The slower the internet connection is, the longer the database access time needed will be.
- Bottlenecks can occur as a result of high traffic.
- Limited access by more than one person to the same set of data as there is only one copy of it and it is maintained in a single location.[8] This can lead to major decreases in the general efficiency of the system.
- If there is no fault-tolerant setup and hardware failure occurs, all the data within the database will be lost.

Distributed Database

A distributed database is a database in which storage devices are not all attached to a common processor. It may be stored in multiple computers, located in the same physical location; or may be dispersed over a network of interconnected computers. A distributed database system consists of loosely coupled sites that share no physical components.

System administrators can distribute collections of data (e.g. in a database) across multiple physical locations. A distributed database can reside on organized network servers or decentralized independent computers on the Internet, on corporate intranets or extranets, or on other organization networks. Because distributed databases store data across multiple computers, distributed databases may improve performance at end-user worksites by allowing transactions to be processed on many machines, instead of being limited to one.

Two processes ensure that the distributed databases remain up-to-date and current: replication and duplication.

- Replication involves using specialized software that looks for changes in the distributive database. Once the changes have been identified, the replication process makes all the databases look the same. The replication process can be complex and time-consuming depending on the size and number of the distributed databases. This process can also require a lot of time and computer resources.
- Duplication, on the other hand, has less complexity. It basically identifies one database as a master and then duplicates that database. The duplication process is normally done at a set time after hours. This is to ensure that each distributed location has the same data. In the duplication process, users may change only the master database. This ensures that local data will not be overwritten.

Both replication and duplication can keep the data current in all distributive locations.

Advantages

- Management of distributed data with different levels of transparency like network transparency, fragmentation transparency, replication transparency, etc.
- Increase reliability and availability
- Easier expansion
- Reflects organizational structure — database fragments potentially stored within the departments they relate to
- Protection of valuable data — if there were ever a catastrophic event such as a fire, all of the data would not be in one place, but distributed in multiple locations
- Improved performance — data is located near the site of greatest demand, and the database systems themselves are parallelized, allowing load on the databases to be balanced among servers. (A high load on one module of the database won't affect other modules of the database in a distributed database)
- Economics — it may cost less to create a network of smaller computers with the power of a single large computer
- Modularity — systems can be modified, added and removed from the distributed database without affecting other modules (systems)
- Continuous operation, even if some nodes go offline (depending on design)

Disadvantages

- Complexity — DBAs may have to do extra work to ensure that the distributed nature of the system is transparent. Extra work must also be done to maintain multiple disparate systems, instead of one big one. Extra database design work must also be done to account for the disconnected nature of the database — for example, joins become prohibitively expensive when performed across multiple systems.
- Economics — increased complexity and a more extensive infrastructure means extra labour costs
- Security — remote database fragments must be secured, and they are not centralized so the remote sites must be secured as well. The infrastructure must also be secured (for example, by encrypting the network links between remote sites).
- Difficult to maintain integrity — but in a distributed database, enforcing integrity over a network may require too much of the network's resources to be feasible
- Database design more complex — In addition to traditional database design challenges, the design of a distributed database has to consider fragmentation of data, allocation of fragments to specific sites and data
- Concurrency control poses a major issue. It can be solved by locking and timestamping.

Centralised databases vs. Distributed databases

The underlying idea of centralised databases is that they should be able to receive, maintain, and complete every single request that the main system must perform by themselves. There is only one database file, kept at a single location on a given network.

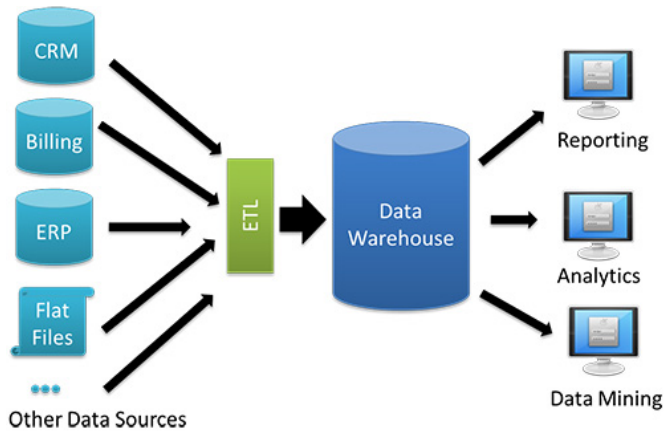
A distributed database, however, is a database in which all the information is stored on multiple physical locations. Distributed databases are divided into two groups: homogeneous and heterogeneous. It relies on replication and duplication within its multiple sub-databases in order to maintain its records up to date. It is composed of multiple database files, all controlled by a central DBMS.

The main differences between centralised and distributed databases arise due to their respective basic characteristics. Differences include but are not limited to:

- Centralized databases store data on a single CPU bound to a single certain physical/geographical location. Distributed databases, however, rely on a central DBMS which manages all its different storage devices remotely, as it is not necessary for them to be kept in the same physical and/or geographical location.
- As outlined above, centralised databases are easier to maintain up to date than distributed databases. This is so because distributed databases require additional (often manual) work to keep the data stored relevant, and to avoid data redundancy, as well as to improve the overall performance.
- If data is lost in a centralised system, retrieving it would be much harder. If, however, data is lost in a distributed system, retrieving it would be very easy, because there is always a copy of the data in a different location of the database.
- Designing a centralised database is generally much less complex than designing a distributed database, as distributed database systems are based on a hierarchical structure.

Data Warehouses

In computing, a data warehouse, is a system used for reporting and data analysis. DWs is centrally stored integrated data from one or more disparate sources. They store current and historical data and are used for creating trending reports for senior management reporting such as annual and quarterly comparisons.



The data may pass through an operational data store for additional operations before it is used in the DW for reporting

A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process.

Subject-Oriented: A data warehouse can be used to analyze a particular subject area. For example, "sales" can be a particular subject.

Integrated: A data warehouse integrates data from multiple data sources. For example, source A and source B may have different ways of identifying a product, but in a data warehouse, there will be only a single way of identifying a product.

Time-Variant: Historical data is kept in a data warehouse. For example, one can retrieve data from 3 months, 6 months, 12 months, or even older data from a data warehouse. This contrasts with a transactions system, where often only the most recent data is kept. For example, a transaction system may hold the most recent address of a customer, where a data warehouse can hold all addresses associated with a customer.

Non-volatile: Once data is in the data warehouse, it will not change. So, historical data in a data warehouse should never be altered.

Ralph Kimball provided a more concise definition of a data warehouse:

A data warehouse is a copy of transaction data (data describing an event) specifically structured for query and analysis.

Benefits

A data warehouse maintains a copy of information from the source transaction systems. This architectural complexity provides the opportunity to:

- Gather data from multiple sources into a single database so a single query engine can be used to present data.

- Reduce the problem bogging databases down when attempts to run large, long running, analysis queries in transaction processing databases.
- Maintain data history, even if the source transaction systems do not.
- Integrate data from multiple source systems, enabling a central view across the enterprise. This benefit is always valuable, but particularly so when the organization has grown by merger.
- Improve data quality, by providing consistent codes and descriptions, flagging or even fixing bad data.
- Present the organization's information consistently.
- Provide a single common data model for all data of interest regardless of the data's source.
- Restructure the data so that it makes sense to the business users.
- Add value to operational business applications, notably customer relationship management systems.
- Make decision–support queries easier to write.

Data Mining

Data Mining is an analytic process designed to explore data (usually large amounts of data - typically business or market related - also known as "big data") in search of consistent patterns and/or systematic relationships between variables, and then to validate the findings by applying the detected patterns to new subsets of data.

The data mining task is the automatic or semi-automatic analysis of large quantities of data to extract previously unknown interesting patterns such as groups of data records (cluster analysis), unusual records (anomaly detection) and dependencies (association rule mining). This usually involves using database techniques such as spatial indices. These patterns can then be seen as a kind of summary of the input data, and may be used in further analysis or, for example, in machine learning and predictive analytics. For example, the data mining step might identify multiple groups in the data, which can then be used to obtain more accurate prediction results by a decision support system.

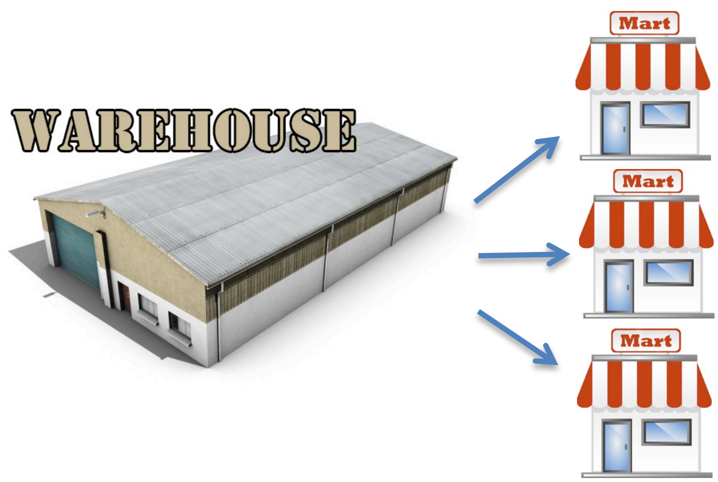
In today's world raw data is being collected by companies at an exploding rate. For example, Walmart (Kmart but bigger and they sell guns :/) processes over 20 million point-of-sale transactions every day. This information is stored in a centralized database, but would be useless without some type of data mining software to analyze it. If Walmart analyzed their point-of-sale data with data mining techniques they would be able to determine sales trends, develop marketing campaigns, and more accurately predict customer loyalty.

Every time a credit card or a store loyalty card is being used, or a warranty card is being filled, data is being collected about the users behavior. Many people find the amount of information stored about us from companies, such as Google, Facebook, and Amazon, disturbing and are concerned about privacy. Although there is the potential for our personal data to be used in harmful, or unwanted, ways it is also being used to make our lives better. For example, Ford and Audi hope to one day collect information about customer driving patterns so they can recommend safer routes and warn drivers about dangerous road conditions.

Data mining involves six common classes of tasks:

- **Anomaly detection** – The identification of unusual data records, that might be interesting or data errors that require further investigation.
- **Association rule learning** – Searches for relationships between variables. For example a supermarket might gather data on customer purchasing habits. Using association rule learning, the supermarket can determine which products are frequently bought together and use this information for marketing purposes. This is sometimes referred to as market basket analysis.
- **Clustering** – is the task of discovering groups and structures in the data that are in some way or another "similar", without using known structures in the data.
- **Classification** – is the task of generalizing known structure to apply to new data. For example, an e-mail program might attempt to classify an e-mail as "legitimate" or as "spam".
- **Regression** – attempts to find a function which models the data with the least error.
- **Summarization** – providing a more compact representation of the data set, including visualization and report generation.

Data Marts



A data mart is the access layer of the data warehouse environment that is used to get data out to the users. The data mart is a subset of the data warehouse that is usually oriented to a specific business line or team. Data marts are small slices of the data warehouse. Whereas data warehouses have an enterprise-wide depth, the information in data marts pertains to a single department. In some deployments, each department or business unit is considered the owner of its data mart including all the hardware, software and data. This enables each department to use, manipulate and develop their data any way they see fit; without altering information inside other data marts or the data warehouse. In other deployments where conformed dimensions are used, this business unit ownership will not hold true for shared dimensions like customer, product, etc.

The reasons why organizations are building data warehouses and data marts are because the information in the database is not organized in a way that makes it easy for organizations to find what they need. Also, complicated queries might take a long time to answer what people want to know since the database systems are designed to process millions of transactions per day. While transactional databases are designed to be updated, data warehouses or marts are read only. Data warehouses are designed to access large groups of related records.

Data marts improve end-user response time by allowing users to have access to the specific type of data they need to view most often by providing the data in a way that supports the collective view of a group of users.

A data mart is basically a condensed and more focused version of a data warehouse that reflects the regulations and process specifications of each business unit within an organization. Each data mart is dedicated to a specific business function or region. This subset of data may span across many or all of an enterprise's functional subject areas. It is common for multiple data marts to be used in order to serve the needs of each individual business unit (different data marts can be used to obtain specific information for various enterprise departments, such as accounting, marketing, sales, etc.).

Data warehouse:

- Holds multiple subject areas
- Holds very detailed information
- Works to integrate all data sources
- Does not necessarily use a dimensional model but feeds dimensional models.

Data mart:

- Often holds only one subject area- for example, Finance, or Sales
- May hold more summarized data (although many hold full detail)
- Concentrates on integrating information from a given subject area or set of source systems
- Is built focused on a dimensional model using a star schema.

Quiz

1. Give an example of a business that would use a local database and why it would be more appropriate than an online database.
2. Give an example of a business that would use an online database and why it would be more appropriate than a local database.
3. Give an example of a centralised that would use a local database and why it would be more appropriate than a distributed database.
4. Give an example of a business that would use a distributed database and why it would be more appropriate than a centralised database.
5. How is a data warehouse distinct from a normal database.
6. How is datamining used in the advertising industry?
7. What is a data mart used for?

References

https://www.webopedia.com/TERM/D/database.html

https://en.wikipedia.org/wiki/Data_warehouse

http://www.monitis.com/blog/top-5-data-warehouses-on-the-market-today/

https://imgflip.com/memetemplate/122739632/Indiana-Jones-Warehouse

https://www.technobuffalo.com/2014/02/26/minecraft-passes-100-million-registrations/

https://www.amazon.com/Brother-Watching-1984-Poster-36in/dp/B00EHVUAH6

http://www.cloudstoragebest.com/cloud-storage-reviews-are-important/

https://huddle.eurostarsoftwaretesting.com/organizational-technical-challenges-implementing-test-data-management/

https://en.wikipedia.org/wiki/Centralized_database

https://dilogrenme.com/dilgruplari/en_tr/8982/Central.html

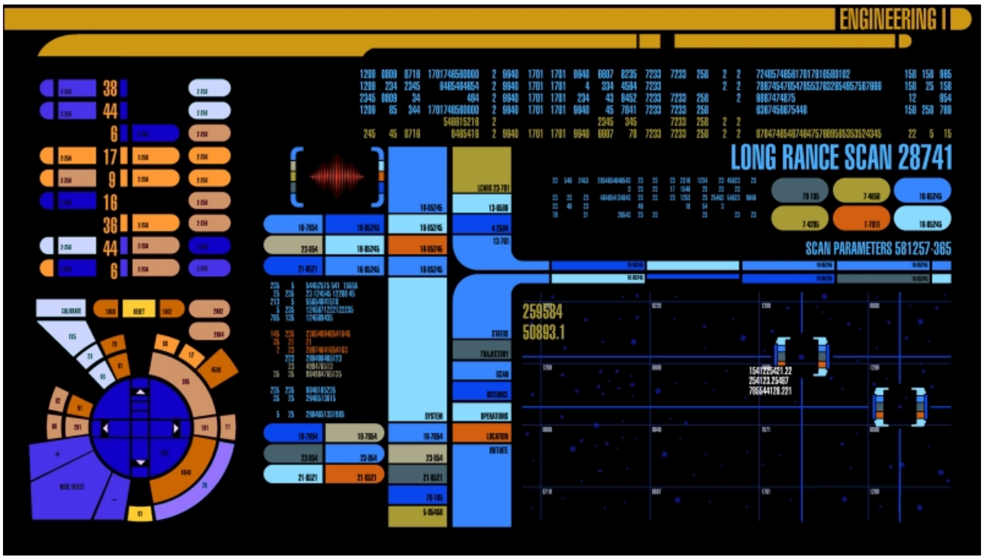
http://goinsurance.com.my/blog/what-you-need-to-know-when-buying-insurance-online/

User Interface Design… for Database Systems

Designing a good user interface (UI) is easier than it sounds - it can often be a juggling contest trying to succeed in every goal you have for the interface without there being a conflict of interest somewhere along the line. For example, by trying to make the interface too pleasing to the eye, functionality and ease of use might be impaired.

So, how does one define a UI as being "good"? A good user interface is one that allows the user to carry out their intended actions efficiently and effectively, without causing too much of a distraction.

With this in mind, it's no wonder that the best UIs aren't the most in-your-face spectacular designs, but rather the ones that work subtly in the background to allow the users to complete their tasks with ease, like a benevolent but nearly invisible presence.



Readability and Legibility and Comprehension UI Design… for databases

Legibility

Legibility is the lowest-level consideration in content usability: it's whether people are able to see, distinguish, and recognize the characters and words in your text. Legibility is thus mainly determined by visual design, specifically typography.

The main guidelines to ensure legibility are:

- Use a reasonably large default font size and allow users to change the font size. Tiny text dooms legibility — and remember that what counts as "tiny" differs across people, depending on their visual acuity, which sadly declines with age. Old users need bigger text, but even young users appreciate not having to squint to read the text. (Teens, in particular, often have ghastly posture and don't sit straight in front of their computer.)
- Have high contrast between characters and background. Preferably, employ a plain background instead of a busy or textured one, since the latter interferes with the recognition of the fine details in the letterforms.
- Use a clean typeface. With today's high-resolution monitors, serif type is fine, but strangely shaped fonts (e.g., emulating handwriting or gothic style) have reduced legibility.

Readability

Readability is the ease with which a reader can understand a written text. In natural language, the readability of text depends on its content (the complexity of its vocabulary and syntax) and its presentation (such as typographic aspects like font size, line height, and line length).

The main guidelines to ensure readability are:

- Use plainspoken words, the shorter the better. Avoid fancy words and made-up terms.
- Use short sentences. Avoid convoluted sentence structures, especially compound sentences with many subordinate clauses and conjunctions that put a strain on users' short-term memory.

Comprehension measures whether a user can understand the intended meaning of a text and can draw the correct conclusions from the text. In the case of instructional or action-oriented content, we also want users to be able to perform the intended actions after reading the text.

The main comprehension guidelines are:

- Use user-centric language; terms familiar to your audience facilitate comprehension.
- If targeting a specialized audience (e.g., for a B2B, scientific, or enthusiast/fan/hobbyist site), use the specialized terminology of that field, even if some of these words are difficult for a broad consumer audience, and thus lower your readability score.
- Use an inverted-pyramid writing style: start with the conclusion or an overview of the main point. People relate better to subsidiary points when they already know the basics.
- Minimize cognitive load by building on existing mental models and reducing the need for users to remember things from one part of text to another.
- Pictures or conceptual diagrams can sometimes explain things better than reams of words.
- Be brief. If you say less, people are more likely to make the effort to understand what you do say.

Navigation

- Navigation is logical and easy to follow.
- Naming of menus is logical.
- People expect to see particular navigation, show them normal navigation.
- Provide tabs, menus in predictable places.
- Provide focus for the most important item on the page/screen.
- Provide alternative pathways, not just one pathway in and one pathway out.
- Image navigation icons need to be able to work literally at the slide of a finger. Too small and it won't work.
- Text hyperlinks too close together because of small font size can cause navigation problems.

Logical Order

- A clear and understandable interface is the goal.
- Order items in the order in which people are going to use them.
- Put first things up front and in the expected place.
- Put yourself in the position of the person wanting to use this database, apply order to what they would expect.

Inclusivity

- Can all people use this software.
- Does it meet user friendly requirements regardless of .
    - Culture
    - Language
- Design it so it has maximum use by the people.
- If people find it difficult to use, it won't be used very often.
- Design it with the end user in mind. All end users is the focus for inclusivity.

Activities

http://ui-patterns.com/patterns/navigation/list

Check out the above website.

1. Give an example of Navigation Tabs (not from the above website).
2. What is an Accordion Menu.
3. In navigation what are breadcrumbs? Give an example.
4. Why might a database provide different interfaces for different users?

References

1. https://www.nngroup.com/articles/legibility-readability-comprehension/
2. https://marvel.com/movies/all
3. https://www.cbscorporation.com/portfolio/cbs-television-studios/
4. Dmitry Fadeyev. 8 Characteristics of Successful User Interfaces http://usabilitypost.com/2009/04/15/8-characteristics-of-successful-user-interfaces/
5. Argon Design. 5 aspects of a good user interface. http://www.argondesign.com/news/2014/feb/5/5-aspects-good-user-interface/
6. Kyle Sollenberger. 10 User Interface Design Fundamentals. http://blog.teamtreehouse.com/10-user-interface-design-fundamentals
7. https://www.thecomputingteacher.com/csc/index.php/man-data/theory-md/interface-design

Database Management

Database

A database is an organized collection of data, stored and accessed electronically. Database designers typically organize the data to model aspects of reality in a way that supports processes requiring information, such as (for example) modelling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.

Data Dictionary

A Data Dictionary, centralized repository of information about data such as meaning, relationships to other data, origin, usage, allowable values and format.

One instance of Data Dictionary is a document describing a database schema for users. In most basic case this documentation includes descriptions of tables and their columns as in the example below. Data Dictionary can include various information about data, including relationships, constraints and rules, sources and usage, to name a few. This documentation is used by database users and developers to understand the data and its structures. It can be in form of a simple document or special repository accessed by a dedicated tool.

The Key Elements of a Data Dictionary

A Data Dictionary provides information about each attribute, also referred to as fields, of a data model.

A Data Dictionary is typically organized in a spreadsheet format. Each attribute is listed as a row in the spreadsheet and each column labels an element of information that is useful to know about the attribute.

Let’s look at the most common elements included in a data dictionary.

- **Element Name** – A unique identifier, typically expressed in business language, that labels each attribute.
- **Data type** - Most programming languages support various types of data, for example: string, date/time, look-ups, real, integer or Boolean.
- **Size/format** – Size of data, default format.
- **Description** – Information on the data for users and/or programmers to better understand what the data is.
- **Constraints** – Limits on the type, size or format of data that can be entered.

While these are the core elements of a data dictionary, it’s not uncommon to document additional information about each element, which may include the source of the information, the table or concept in which the attribute is contained, the physical database field name, the field length, and any default values.

Example of a Data Dictionary

You are probably wondering how all of this comes together… well you're probably not but here we go anyway.

Here's a look at a simplified example data dictionary that contains the attribute from our Bridging the Gap article example, along with critical information about each attribute.

Attribute Name	Required	Type	Field Length	Default Values	Notes
Article Title	Yes	Text	250	n/a	Can contain HTML.
Article Author	Yes	Look-Up	n/a	n/a	
Article Category	Yes	Look-Up	n/a	Uncategorized	
Article Content	No	Text	Unlimited	n/a	Can contain HTML.

Data Definition

Data Definition Language is a computer language that is used to define data structures. In Database Management Systems (DBMS), it is used to specify a database scheme as a set of definitions

Data Duplication

Data duplication occurs when an exact copy of a piece of data is created.

- Data duplication provides benefits such as providing us with the ability to back up copies of files and create multiple versions of a file (which may be required for progress reporting or other information)
- The duplication of data is often intentional and used primarily for creating backups
- Data duplication on a database may result in data redundancy, and thus an inefficient and inconsistent database

Data Redundancy

Data redundancy is a condition created within a database or data storage technology in which the same piece of data is held in two separate places.

This can mean two different fields within a single database, or two different spots in multiple software environments or platforms. Whenever data is repeated, this basically constitutes data redundancy. This can occur by accident but is also done deliberately for backup and recovery purposes.

- Data redundancy occurs when the same data is entered in to two or more fields of a database
- For example, "Joe" is entered in to the Name field under a record called Customers
- "Joe" is also entered in to the Customer field under a record called Purchases
- Although we are referring to the same Joe in both fields, each piece of data is seen as unique
- This means that to update "Joe", we need to manually edit each reference

- This can cause problems in database systems such as data anomalies
- This differs from data duplication, as it is often not intentional and can take up potentially required storage space

Data Integrity

Relates to the accuracy and consistency of the data. Primary areas include referential, entity and domain.

Referential Integrity

Referential integrity is a relational database concept, which states that table relationships must always be consistent. In other words, any foreign key field must agree with the primary key that is referenced by the foreign key. Thus, any primary key field changes must be applied to all foreign keys, or not at all. The same restriction also applies to foreign keys in that any updates (but not necessarily deletions) must be propagated to the primary parent key.

Consider a bank database, which contains two tables:

- CUSTOMER_MASTER Table: This holds basic customer/account holder data such as name, social security number, address and date of birth.
- ACCOUNTS_MASTER Table: This stores basic bank account data such as account type, account creation date, account holder and withdrawal limits.

To uniquely identify each customer/account holder in the CUSTOMER_MASTER table, a primary key column named CUSTOMER_ID is created.

To identify a customer and bank account relationship in the ACCOUNTS_MASTER table, an existing customer in the CUSTOMER_MASTER table must be referenced. Thus, the CUSTOMER_ID column – also created in the ACCOUNTS_MASTER table – is a foreign key. This column is special because its values are not newly created. Rather, these values must reference existing and identical values in the primary key column of another table, which is the CUSTOMER_ID column of the CUSTOMER_MASTER table.

Referential integrity is a standard that means any CUSTOMER_ID value in the CUSTOMER_MASTER table may not be edited without editing the corresponding value in the ACCOUNTS_MASTER table. For example, if Andrew Smith's customer ID is changed in the CUSTOMER_MASTER table, this change also must be applied to the ACCOUNTS_MASTER table, thus allowing Andrew Smith's account information to link to his customer ID.

Domain Integrity

To understand a database domain, let's consider a few other aspects of a database:

- A database schema defines a set of attributes, also called columns or fields. For example, if you have a table "Employee Contact Information," it might include attributes for FirstName, LastName, JobTitle, StreetAddress, City, State, ZipCode, PhoneNumber, and Email.
- Each attribute has a domain that defines allowable values. This could include its data type, length, values, and other details.

For example, the domain for an attribute Post Code might specify a numeric data type, such as an integer, usually called an INT or an INTEGER, depending on the database. Or a database designer might choose to define it instead as a character, usually called a CHAR. The attribute can be further defined to require a specific length, or whether an empty or unknown value is allowed.

About Domain Integrity

The allowed values of an attribute create domain integrity, which ensures that all data in a field contains valid values.

Domain integrity is defined by:

- The data type, such as integer, character, or decimal
- The allowed length of the data
- The range, defining the upper and lower boundaries
- Any constraints, or limitations on allowable values. For example, a Australian Post code field might enforce a four-digit code.
- The type of NULL support, or whether an attribute can have an unknown, or NULL value
- The default value, if any
- The date format, if applicable (for instance, dd/mm/yy or mm/dd/yyyy)

Entity Integrity

Entity Integrity ensures that there are no duplicate records within the table and that the field that identifies each record within the table is unique and never null.

The existence of the Primary Key is the core of the entity integrity. If you define a primary key for each entity, they follow the entity integrity rule.

Entity integrity specifies that the Primary Keys on every instance of an entity must be kept, must be unique and must have values other than NULL.

Although most relational databases do not specifically dictate that a table needs to have a Primary Key, it is good practice to design a Primary Key for each table in the relational model. This mandates no NULL content, so that every row in a table must have a value that denotes the row as a unique element of the entity.

Entity Integrity is the mechanism the system provides to maintain primary keys. The primary key serves as a unique identifier for rows in the table. Entity Integrity ensures two properties for primary keys:

- The primary key for a row is unique; it does not match the primary key of any other row in the table.
- The primary key is not null, no component of the primary key may be set to null.

The uniqueness property ensures that the primary key of each row uniquely identifies it; there are no duplicates. The second property ensures that the primary key has meaning, has a value; no component of the key is missing.

The system enforces Entity Integrity by not allowing operations (INSERT, UPDATE) to produce an invalid primary key. Any operation that creates a duplicate primary key or one containing nulls is rejected.

Database Validation

Validation is an automatic check to ensure that the data entered is sensible and feasible. Validation cannot ensure data is actually accurate.

Database Management Systems allow for some handy validation methods to be implemented. These are needed because it is easier to try and prevent users from entering garbage than attempting to fix mistakes later.

Validation methods for databases. * The first two (Range and Type Check) are mentioned in the syllabus.

Validation Method	Description
Range Check	Checks the data is between an upper and lower acceptable value, within a certain range
Type Check	Checks that the data entered is of an expected type, e.g. a number or date
Length Check	Checks the number of characters meets expectations, e.g. 8 character password
Format Check	Ensures the data follows a set pattern (using an input mask), e.g. postcode.
Drop Down Box	Ensures the user can only choose a predefined option from a list, reducing the chances of spelling mistakes or unwanted responses
Presence Check	Checks the user has at least entered something into the fieldm stopping them from accidently leaving it empty

Data Anomalies

Normalization is the process of splitting relations into well structured relations that allow users to insert, delete, and update tuples without introducing database inconsistencies. Without normalization many problems can occur when trying to load an integrated conceptual model into the DBMS. These problems arise from relations that are generated directly from user views are called anomalies. There are three types of anomalies: update, deletion and insertion anomalies.

An **update anomaly** is a data inconsistency that results from data redundancy and a partial update. For example, each employee in a company has a department associated with them as well as the student group they participate in.

Employee_ID	Name	Department	Student_Group
123	J. Longfellow	Accounting	Beta Alpha Psi
234	B. Rech	Marketing	Marketing Club
234	B. Rech	Marketing	Management Club
456	A. Bruchs	CIS	Technology Org.
456	A. Bruchs	CIS	Beta Alpha Psi

If A. Bruchs' department is an error it must be updated at least 2 times or there will be inconsistent data in the database. If the user performing the update does not realize the data is stored redundantly the update will not be done properly.

A **deletion anomaly** is the unintended loss of data due to deletion of other data. For example, if the student group Beta Alpha Psi disbanded and was deleted from the table above, J. Longfellow and the Accounting department would cease to exist. This results in database inconsistencies and is an example of how combining information that does not really belong together into one table can cause problems.

An **insertion anomaly** is the inability to add data to the database due to absence of other data. For example, assume Student_Group is defined so that null values are not allowed. If a new employee is hired but not immediately assigned to a Student_Group then this employee could not be entered into the database. This results in database inconsistencies due to omission.

Update, deletion, and insertion anomalies are very undesirable in any database. Anomalies are avoided by the process of normalization.

Data Manipulation

Data Manipulation concerns the types of activities (Create, Read, Update and Delete) that may be done to data and who has the authority to do them- think CRUD - create read update delete.

Data Security

Database security concerns the use of a broad range of information security controls to protect databases (potentially including the data, the database applications or stored functions, the database systems, the database servers and the associated network links) against compromises of their confidentiality, integrity and availability. It involves various types or categories of controls, such as technical, procedural/administrative and physical. Database security is a specialist topic within the broader realms of computer security, information security and risk management.

Security risks to database systems include, for example:

- Unauthorized or unintended activity or misuse by authorized database users, database administrators, or network/systems managers, or by unauthorized users or hackers (e.g. inappropriate access to sensitive data, metadata or functions within databases, or inappropriate changes to the database programs, structures or security configurations);
- Malware infections causing incidents such as unauthorized access, interruption or denial of authorized access to the database, etc;
- Overloads, capacity issues resulting in the inability of authorized users to use databases as intended;
- Physical damage to database servers;
- Design flaws and programming bugs in databases, creating various security vulnerabilities (e.g. unauthorized privilege escalation), data loss/corruption, performance degradation etc.;
- Data corruption and/or loss caused by the entry of invalid data or commands, mistakes in database or system administration processes, sabotage/criminal damage etc.

Many layers and types of information security control are appropriate to databases, including:

- Access control
- Auditing
- Authentication
- Encryption
- Integrity controls
- Backups
- Application security
- Database Security applying Statistical Method

References

https://en.wikipedia.org/wiki/Database

https://en.wikipedia.org/wiki/Data_dictionary

http://www.bridging-the-gap.com/data-dictionary/

https://searchmicroservices.techtarget.com/definition/data-dictionary

https://www.techopedia.com/definition/18707/data-redundancy

https://www.techopedia.com/definition/1233/referential-integrity-ri

https://en.wikipedia.org/wiki/Data_integrity

https://www.lifewire.com/database-domain-definition-1019170

http://databasemanagement.wikia.com/wiki/Category:Data_Anomalies

https://en.wikipedia.org/wiki/Database_security

https://www.computerscience.gcse.guru/theory/database-validation

Normalisation

Normalisation is the process of identifying and eliminating data anomalies and redundancies, thereby improving data integrity and efficiency in a relational database. This process is designed to remove repeated data and improve database design.

Basically we are making a database where –

1. Records are unique – So two people named John Smith won't break the database
2. We aren't doubling up data – So if we update something we only need to do it in the one place
3. We can add data without creating empty fields
4. We can delete data without losing other data we would like to keep

During normalisation a relation is split into a number of smaller tables suitable for implementation in a relational database.

For the course we need to normalise up until the 3rd Normal Form 3NF.

Let's look at an example.

EB GAMES

EB Games is a store that sells a variety of consoles for example –

Microsoft Xbox 1S $399

Sony PlayStation 4 $509

$148

Here are EB games sales records.

Cust Name	Item	Shipping Address	Supplier	Supplier Phone	Price
John Smith	XBox One	35 Palm St, Melville	Microsoft	(08) BUY-XBOX	$399
Roger Banks	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509
Evan Wilson	XBox One, PS Vita	38 Rock Av, Coogee	Wholesale	Toll Free	$547
John Smith	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509

There are some problems with this database so we will Normalise it to remove these possible problems.

1st Normal Form

1. Each cell to be Single valued
2. Entries in a column are same type
3. Rows uniquely identified – Add Unique ID, or Add more columns to make unique

3. Row uniquely Identified *doesn't account for people with the same name*

Cust Name	Item	Shipping Address	Supplier	Supplier Phone	Price
John Smith	XBox One	35 Palm St, Melville	Microsoft	(08) BUY-XBOX	$399
Roger Banks	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509
Evan Wilson	XBox One, PS Vita	38 Rock Av, Coogee	Wholesale	Toll Free	$547
John Smith	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509

1. Not a Single Value

2. Not Same Type

So here is that table Normalised to the 1st Normal Form. The colour is just to highlight the changes.

Primary Key

Cust ID	Cust Name	Item	Shipping Address	Supplier	Supplier Phone	Price
1000	John Smith	XBox One	35 Palm St, Melville	Microsoft	(08) BUY-XBOX	$399
1001	Roger Banks	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509
1002	Evan Wilson	XBox One	38 Rock Av, Coogee	Microsoft	(08) BUY-XBOX	$399
1002	Evan Wilson	PS Vita	38 Rock Av, Coogee	Sony	(08) BUY-SONY	$148
1003	John Smith	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509

1. So to get to single values Evan Wilson's entry was separated to two entries one for the XBox and one for the Vita.
2. Wholesale and Toll Free were removed and replaced with proper values that reflect the information expected.
3. A Primary Key (Cust ID) was added to make sure each row is uniquely identified. So we now see clearly that the two John Smiths are two different customers.

2nd Normal Form

1. All attributes (Non-Key Columns) dependant on the key.

So which attributes are not dependant on Cust ID in previous table?

Primary Key

Cust ID	Cust Name	Item	Shipping Address	Supplier	Supplier Phone	Price
1000	John Smith	XBox One	35 Palm St, Melville	Microsoft	(08) BUY-XBOX	$399
1001	Roger Banks	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509
1002	Evan Wilson	XBox One	38 Rock Av, Coogee	Microsoft	(08) BUY-XBOX	$399
1002	Evan Wilson	PS Vita	38 Rock Av, Coogee	Sony	(08) BUY-SONY	$148
1003	John Smith	Playstation 4	47 Campus Rd, Murdoch	Sony	(08) BUY-SONY	$509

The Supplier, Supplier Phone and Price and not dependant on the key. They are independent to the person buying the console. Microsoft is the supplier of XBox regardless of who is buying it.

So we will take these items out into their own table.

I have used the name of the item as a primary key to make it a little clearer what is going on. You probably wouldn't do this in real life, there would be a stock number associated with these items.

Primary Key

Cust ID	Cust Name	Shipping Address
1000	John Smith	35 Palm St, Melville
1001	Roger Banks	47 Campus Rd, Murdoch
1002	Evan Wilson	38 Rock Av, Coogee
1003	John Smith	47 Campus Rd, Murdoch

Primary Key

Stock ID	Supplier	Supplier Phone	Price
XBox One	Microsoft	(08) BUY-XBOX	$399
PlayStation 4	Sony	(08) BUY-SONY	$509
PS Vita	Sony	(08) BUY-SONY	$148

So we have the non-dependant data out but now we can't see who bought what so we add table called a junction table to join them together.

Primary Key Cust ID	Primary Key Stock ID
1000	XBox One
1001	PlayStation 4
1002	XBox One
1002	PS Vita
1003	PlayStation 4

3rd Normal Form

1. All fields (columns) can be determined only by the key in the table and no other column.

In our example the Supplier Phone is dependent on the supplier. If we left this and the supplier changed their number we would need to change it in multiple places.

Primary Key

Cust ID	Cust Name	Shipping Address
1000	John Smith	35 Palm St, Melville
1001	Roger Banks	47 Campus Rd, Murdoch
1002	Evan Wilson	38 Rock Av, Coogee
1003	John Smith	47 Campus Rd, Murdoch

Primary Key

Stock ID	Supplier	Supplier Phone	Price
XBox One	Microsoft	(08) BUY-XBOX	$399
PlayStation 4	Sony	(08) BUY-SONY	$509
PS Vita	Sony	(08) BUY-SONY	$148

Primary Key **Primary Key**

Cust ID	Stock ID
1000	XBox One
1001	PlayStation 4
1002	XBox One
1002	PS Vita
1003	PlayStation 4

So we pull out the Supplier making it a Primary Key in its own table and a Foreign key in Stock Table. Which will means you can only add suppliers from the Supplier Table in the Stock Table.

Primary Key

Cust ID	Cust Name	Shipping Address
1000	John Smith	35 Palm St, Melville
1001	Roger Banks	47 Campus Rd, Murdoch
1002	Evan Wilson	38 Rock Av, Coogee
1003	John Smith	47 Campus Rd, Murdoch

Primary Key **Foreign Key**

Stock ID	Supplier	Price
XBox One	Microsoft	$399
PlayStation 4	Sony	$509
PS Vita	Sony	$148

Primary Key **Primary Key**

Cust ID	Stock ID
1000	XBox One
1001	PlayStation 4
1002	XBox One
1002	PS Vita
1003	PlayStation 4

Primary Key

Supplier	Supplier Phone
Microsoft	(08) BUY-XBOX
Sony	(08) BUY-SONY

And we are done! Yes there is 4th Normal form but we don't need to know about that.

3NF Terminology

Primary key

A primary key is a single attribute or multiple attributes (composite key) that uniquely identify each tuple in the relation. The primary key attribute must contain unique values.

Composite key

A composite key is a primary key that includes multiple attributes.

Foreign key

A foreign key is an attribute in a table that stores a value that must match a value in the primary key field in the related table. The foreign key attribute may have duplicate values. Therefore, a foreign key is an attribute in a table that is a primary key in another table.

Driver Licence Number	Driver Firstname	Driver Surname	Driver Email	Car Registration	Car Manufacturer	Car Manufacturer Web Site
19289385	John	Smith	jsmith@combi.net	1COB 293	Ford	www.ford.com.au
19289385	John	Smith	jsmith@combi.net	1QAZ 889	Holden	www.holden.com.au
19289385	John	Smith	jsmith@combi.net	1CCT 441	Ford	www.ford.com.au
26453790	May	Hogarth	mhogarth@combi.net	1COB 293		

Update Anomaly

An update anomaly is a problem that occurs when data that is repeated in a number of records requires updating. If all records are not updated the data could become inconsistent or inaccurate. In the relation above, if John Smith changes his email address, all 3 occurrences would need to be updated to ensure that the data remains accurate.

Insert Anomaly

There are 2 types of insert anomaly.

1. When data needs to be added to a table and not all the data is known then some fields will be null. For example, if a new driver (e.g. Sam Spade) is added to the Car Driver table and the car details are unknown then the Car Registration, Car Value, Car Colour, Car Model and Car Manufacturer Website fields would be null.
2. When data is added to a table and this results in some data being repeated. For example, because John Smith can drive 3 cars the data in the Driver Licence

Deletion Anomaly

A deletion anomaly occurs when a record is deleted and this results in the loss of other data that only occurs in that record. For example, if the second row is deleted because John Smith sells the car with registration 1QAZ 889 then the website for Holden (www.holden.com.au) is also lost as this data only occurs in that record.

Exercise 1

Student ID	Student Name	House	Year	Gender	Event No	Event	Time	Place	Points
1	John Jones	Blue	10	M	1	Boys 50 Fly	00:49:30	1	6
2	Alan Hunt	Green	10	M	1	Boys 50 Fly	00:50:19	2	4
3	Sam Lint	Blue	10	M	1	Boys 50 Fly	00:51:19	3	2
4	Mark Lo	Red	10	M	1	Boys 50 Fly	01:00:01	4	1
1	John Jones	Blue	10	M	2	Boys 50 Free	00:50:30	2	4
2	Alan Hunt	Green	10	M	2	Boys 50 Free	00:50:19	1	6
10	Mary Hill	Red	11	F	8	Boys 50 Fly	00:50:20	1	6

Other Info
1st Place – 6 points
2nd Place – 4 points
3rd Place – 2 points
4th Place – 1 point
5th Place – 1 point
6th Place – 1 point

1. What type of “database” is this?
2. What are the appropriate data types and data validations/limits for each field?
3. What is the primary key? Is there one? Can one be identified?
4. Is this table normalized in any form? Justify your answer.

Exercise 2

Consider the following report from a Home Rental Company that has recently employed you as their Information Management advisor.

Page 1			**Dream Home** Customer Rental Report			**7 Oct 1996**
	Customer Name	John Kay		**Customer Number**	CR 76	
Property Number	**Property Address**	**Rent Start**	**Rent Finish**	**Rent**	**Owner Number**	**Owner Name**
PG 4	6 Lawrie St Glasgow	1 Jul 93	31 Aug 95	350	CO40	Tina Murphy
PG 16	5 Novice Drive Glasgow	1 Sept 95	1 Sept 96	450	CO 93	Tony Shaw

Page 2			**Dream Home** Customer Rental Report			**7 Oct 1996**
	Customer Name	Aline Stewart		**Customer Number**	CR 56	
Property Number	**Property Address**	**Rent Start**	**Rent Finish**	**Rent**	**Owner Number**	**Owner Name**
PG 4	6 Lawrie St Glasgow	1 Sept 92	10 Jun 93	350	CO40	Tina Murphy
PG 36	2 Manor Road Glasgow	10 Oct 93	1 Dec 94	375	CO 93	Tony Shaw
PG 16	5 Novice Drive Glasgow	1 Jan 95	10 Aug 95	450	CO 93	Tony Shaw

Study the data from the two pages of this report and transfer the data to a table. Ignore the date of the report; this does not need to be taken into account at this time.

1. Create an un-normalised table first of all.
2. Identify a primary key (which may be composite) and put the table into 1NF.
3. Study the dependencies and create a set of tables in 2NF.
4. Look for a dependency that are not dependant on a primary keys from step 3 and move your tables to 3NF.

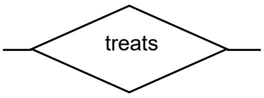
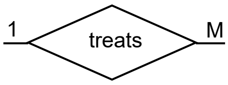
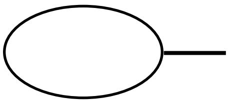
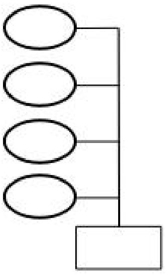
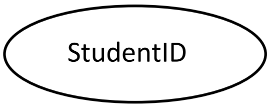
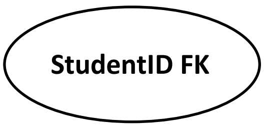
References

https://www.youtube.com/watch?v=UrYLYV7WSHM
https://ebgames.com.au/

Chen’s Notation

The Computer Science ATAR syllabus requires the use of Chen’s notation as a convention to represent Entity Relationship (ER) diagrams when modelling a data base solution. However, online and text resources are inconsistent in the representation of foreign key fields and attributes when using Chen’s notation.

To provide clarity and ensure consistency, when constructing ER diagrams using Chen's notation the following applies:

Entities		An entity is represented by a rectangle, containing the name of the entity expressed as a singular noun. An entity is connected to an attribute and/or a relationship by a straight line.
Relationships		A relationship is represented by a diamond, containing the relationship type expressed as a verb. Two single lines either side of the diamond connect the relationship to the entities.
Cardinality		Cardinality is represented by placing the type of cardinality (1:1, 1:M, M:N), at the extremities of the connectors to the entities.
Attributes	   	An attribute is represented by an oval. An oval contains a single attribute label expressed as an adjective and is connected to an entity by a single straight line. Multiple attributes can be connected to an entity by a nested connecting line. For the purpose of this course: • The primary key field/s is identified by a single underline • The foreign key field/s is identified by the use of the letters ‘FK’ next to the field

Note: The use of the following is **beyond** the requirements of the Computer Science ATAR syllabus:

- Weak entity
- Multi-valued and derived attributes
- Weak and optional relationships
- Participation constraints

Teachers may wish to provide more able students extension activities to explore these concepts however they do not reflect the examinable content of the Computer Science syllabus.

Cardinality

The **degree of relationship** (**cardinality**) is represented by characters “1”, “M” or “N” usually placed at the ends of the relationships:

- **one-to-one (1:1)**
  The employee can manage only one department, and each department can be managed by one employee only:

- **one-to-many (1:M)**
  The customer may place many orders, but each order can be placed by one customer only:

- **many-to-one (M:1)**
  Many employees may belong to one department, but one particular employee can belong to one department only:

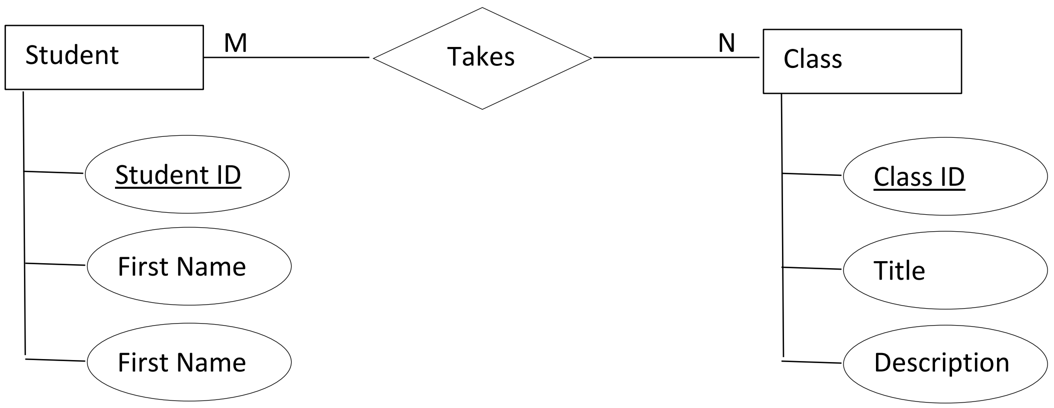
- **many-to-many (M:N)**
  One student may belong to more than one student organizations, and one organization can admit more than one student:

Cardinality Many to Many

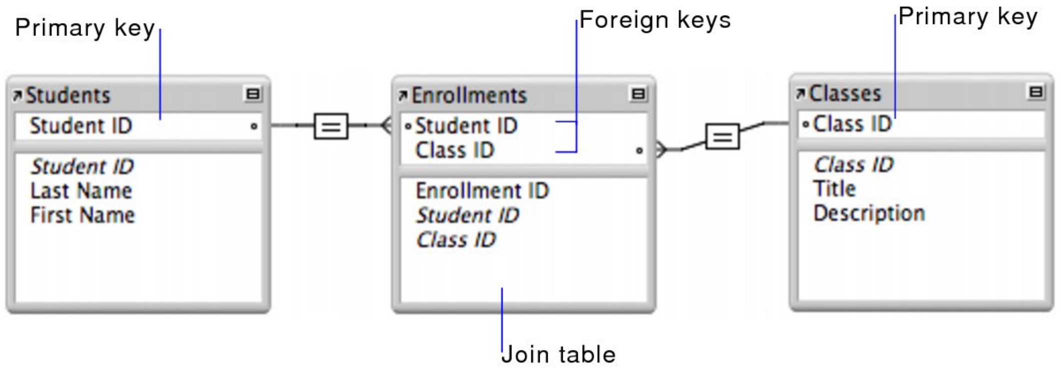
A many-to-many relationship occurs when multiple records in a table are associated with multiple records in another table. For example, a many-to-many relationship exists between customers and products: customers can purchase various products, and products can be purchased by many customers. Relational database systems usually don't allow you to implement a direct many-to-many relationship between two tables.

To avoid this problem, you can break the many-to-many relationship into two one-to-many relationships by using a third table, called a join table. Each record in a join table includes a match field that contains the value of the primary keys of the two tables it joins. (In the join table, these match fields are foreign keys.) These foreign key fields are populated with data as records in the join table are created from either table it joins. The foreign key would sit in the table with the many.

A typical example of a many-to many relationship is one between students and classes. A student can register for many classes, and a class can include many students.

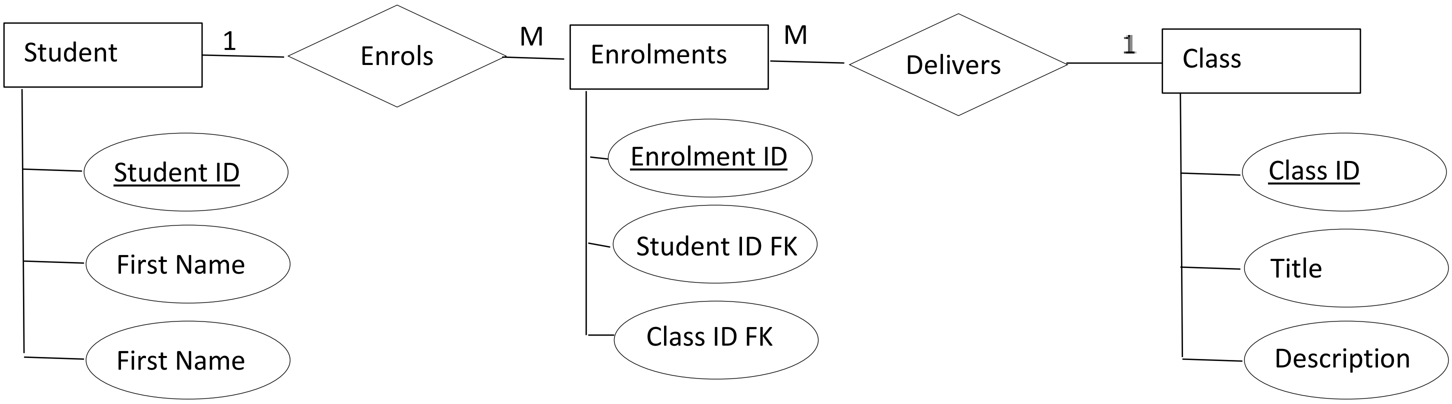


To resolve this problem we create a join table, Enrollments, creates two one-to-many relationships—one between each of the two tables.



The primary key Student ID uniquely identifies each student in the Students table. The primary key Class ID uniquely identifies each class in the Classes table. The Enrollments table contains the foreign keys Student ID and Class ID.

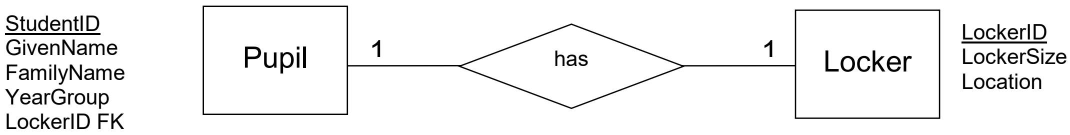
We would create a new entity to represent this table in the entity relationship diagram.



Examples

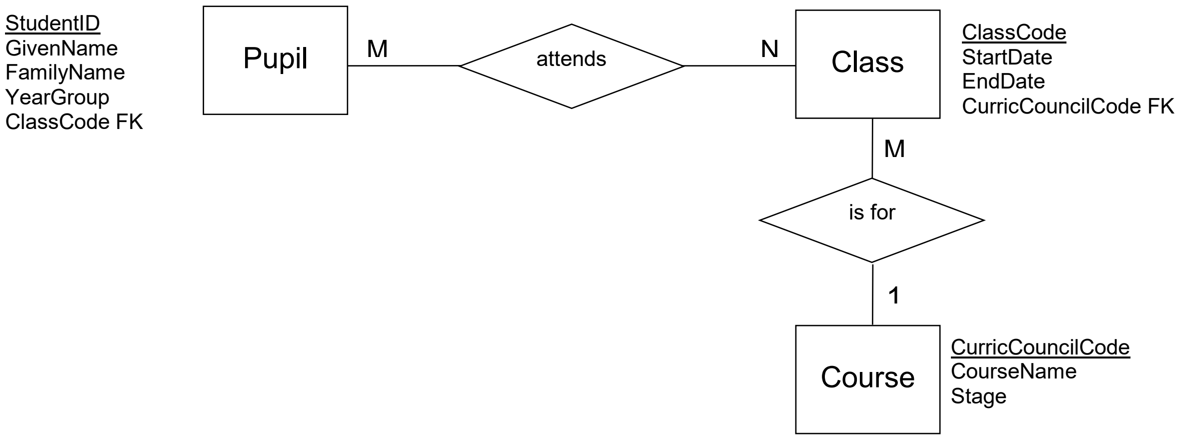
Example 1

A pupil has one locker and a locker is used by only one student at a time. The locker details are stored in a separate relation as not all students will have a locker. The following diagram describes this relationship.

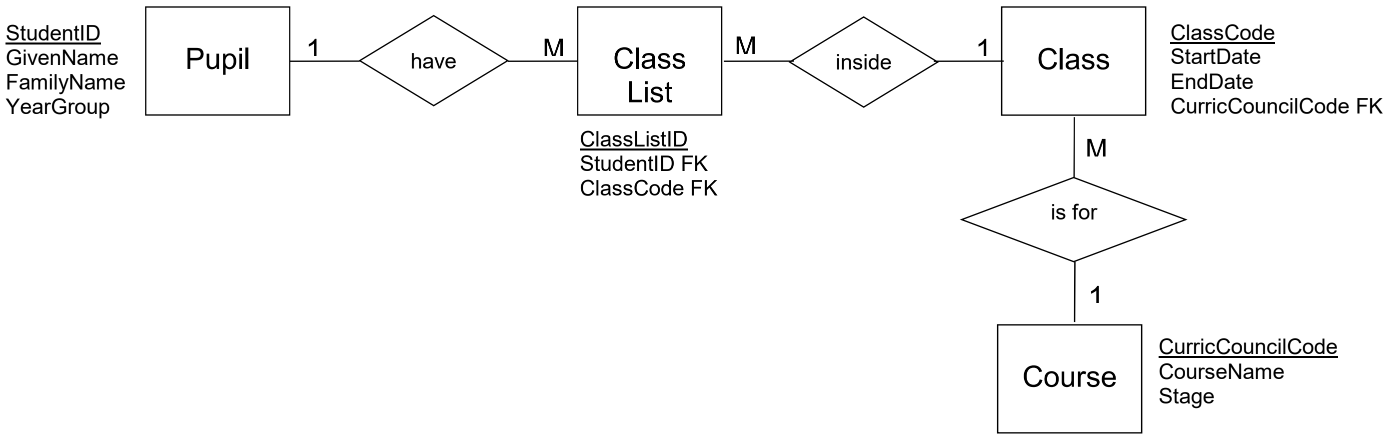


Example 2

A pupil can attend many classes and a class is attended by many students. A class is for one course but a course may have many classes. The following diagram describes this relationship.

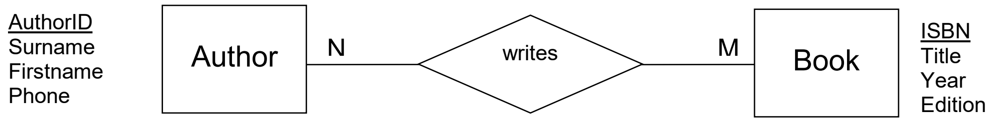


To create a model that can be implemented in a relational database the above many to many (M:N) relationship needs to be resolved by introducing an intersecting entity as shown below.
The intersecting ClassList entity could have a composite primary key, but a ClassListID field has been created to provide a single field primary key instead of having a composite primary key.

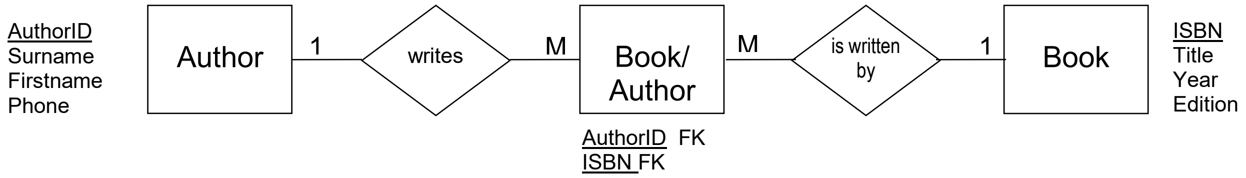


Example 3

An author can write many books and a book could have many authors. The following diagram describes this relationship.



The M:N relationship resolves to two 1:M relationships when the intersecting Book/Author entity is used.



The Book/Author entity has a composite primary key.

Activities

Krista owns a ladies clothing supply and manufacturing business that she operates from home. She sells a limited but popular range to many boutiques across Australia. Presently she stores all her sales data in the following table.

Sales

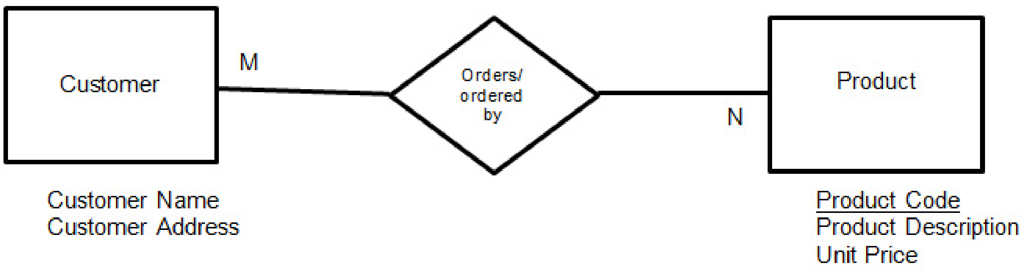
ID	Data	Customer	Address	Quantity Order	Product Code	Product Decription	Unit Price	Sub Total	Discount	Sub Total 2	GST	Total
2	15/05/2016	Fleur	18 Wavely St, I	25	G4001	Long Sleve Dress	$55.00	$1,375.00	$68.75	1306	$130.63	$1,436.88

Krista has been advised to develop an Entity Relationship Diagram (ERD) before developing the database

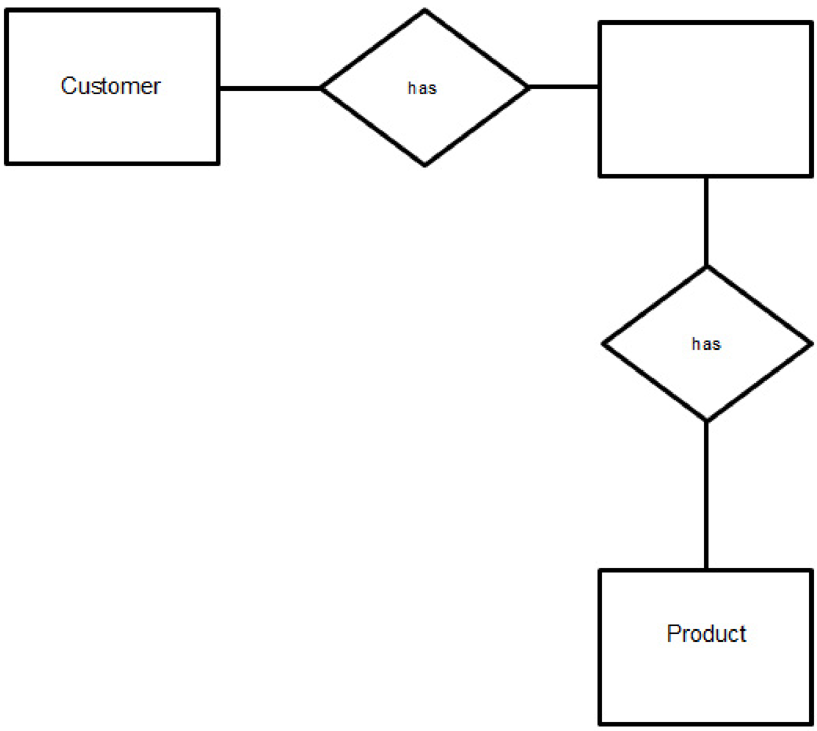
1. What is an ERD and why is it important?

Krista has sketched and ERD below.

Krista has sketched and ERD below.



2. Identify two issues with this ERD
3. Complete the ERD below. Part of it is done for you. You must identify all primary/foreign keys and cardinality.



Krista is unsure what to do with the fields 'Sub Total'; 'GST'; 'Discount' and 'Total'.

4. Why would they not be included in the ERD?

Data Driven Websites and Open Database Connectivity

Dynamic Vs. Static Websites

If you have tried to delve into data driven web design you must have come across these two terms. Here is a brief breakdown of what each entails:

- **Static website:** This website does not change every time the browser loads a page. If a user clicks a button, nothing changes on the layout and content of the page. The only changes occur when the user loads a new page or when the admin loads another page on the web browser. The content is stored on the web file system and it will always be presented in the same format.
- **Dynamic website:** As the name implies, these pages change every time they are loaded without the webmaster having to make the changes. If a user clicks on an image or text, some changes are observed on the particular page. The fact that the content is stored outside the web file system makes it easier to manage and you can manipulate data quickly.

Database Driven Websites

A database driven website is the best example of a dynamic website. The web page grabs information from your database and inserts the same on the web page every time it is loaded. In case there are any changes to the information in the database, the web page (which is connected to the database through programming) will also change automatically. These websites are more effective and they are commonly used for large industries.

Database-driven Websites removes the need for manually coding each of the possibly thousands of pages comprising a site. Instead, data are entered once into a database, which then dynamically creates appropriate pages based on the search terms entered by the user.

Websites based on static HTML pages require skilled coders not only for the initial creation of the pages but also for their revision when the content or URLs change. Processing and storing all these pages taxes even the most robust server. Websites that are database-driven usually require only a few files to serve as templates, which are then populated with content taken from the database. Knowledge of HTML is not needed to add data into the database, thereby eliminating the need for an ever-increasing number of coders.

Many data driven websites are "authenticated" sites. An authenticated site requires that the visitors identify themselves using a login ID and password. Examples of data driven sites include Amazon, Hotmail, and web banking websites. An example of a simple data driven application is Newspaper-World.com.

It is much more complex to build a data driven website than a "brochure ware" website or a web-application that doesn't require a database (e.g. an application to display amortization schedules). Fortunately, there are web authoring tools now available that can help you generate simple data driven sites using data from Access databases, spreadsheets, and XML data sources.

There are many different types of database management systems (DBMS) that can be used in a data driven website. The most widely used DBMS for data driven sites is MySQL, a relatively simple one that runs on computers that use Unix and Linux operating systems. On server computers that use Windows based operating systems, SQL Server, MS Access and XML data sources are popular.

When a request for a data driven web page is made by a web browser to a web server, the web server calls an application system (or server-side script) that in turn makes a request for data from the database. If the request is successful, the data is formatted and passed back to the web server which integrates it with other website content. The resulting webpage and related files (e.g. graphics) are then sent back to the web browser using the standard http communication protocol.

Examples of Database Driven Websites

If you are a web designer then you must have in mind the type of website you are building. This will help you determine whether a static website is best suited for your clients or not. Common database driven websites include:

- E-Commerce platforms: These businesses leverage data driven websites because of the expected changes in prices, offers and services. This guarantees the information internet users find is always fresh and up-to-date.
- Content Management Systems (CMS): If the website is going to use a CMS then it is database driven. Users can easily update content on the website even without the need for any specialized programming skills. These CMSs include WordPress and Joomla and they have an easy-to-use editor to allow publishing of content, editing and deleting.
- Blogs: Most blogs and online community forums are database driven because they involve regular updates by users. Whether people are leaving comments or liking a website there is immediate change on the page.

Dynamic websites require databases be it PostgreSQL, MySQL Community Server, Oracle Express Edition or any other. If you are creating an e-commerce website, you need a database to support shopping carts, discussion forums, registration systems and much more.

PostgreSQL

PostgreSQL, often simply Postgres, is an object-relational database management system (ORDBMS) with an emphasis on extensibility and standards compliance. It can handle workloads ranging from small single-machine applications to large Internet-facing applications (or for data warehousing) with many concurrent users; on macOS Server, PostgreSQL is the default database; and it is also available for Microsoft Windows and Linux (supplied in most distributions).

MySQL

MySQL is an open source relational database management system (RDBMS) based on Structured Query Language (SQL). MySQL runs on virtually all platforms, including Linux, UNIX and Windows. Although it can be used in a wide range of applications, MySQL is most often associated with web applications and online publishing.

Advantages of a Database-Driven Website

What are the benefits of a database driven website?

- It is easier and faster to update content. A few clicks are all the webmaster needs to update the content. Changes are made almost in real-time
- It is ideal for an ecommerce site where different products need to be added, prices changes and offers introduced.
- No need for specialized HTML knowledge or expertise to change content on the website.
- High scalability: Every business grows with time and a database driven website offers room for growth. Changing graphics, layout or interactivity can be done anytime.
- Reduced chances for error: Physical data entry by employees is bound to lead to errors, which can lead to downtime, bugs and other problems. Rectifying such problems on data-driven websites is easy and this improves user experience.

Open Database Connectivity (ODBC)

In computing, Open Database Connectivity (ODBC) is a standard application programming interface (API) for accessing database management systems (DBMS). The designers of ODBC aimed to make it independent of database systems and operating systems. An application written using ODBC can be ported to other platforms, both on the client and server side, with few changes to the data access code.

So basically using ODBC it means that Data Base developers can just write their commands with ODBC to call their database to do stuff (like insert rows of data into tables) and the ODBC engine translates these calls into whatever the correct calls are for a Postgre Database or a MySQL Database and so on, saving a lot of time and effort.

ODBC accomplishes DBMS independence by using an ODBC driver as a translation layer between the application and the DBMS. The application uses ODBC functions through an ODBC driver manager with which it is linked, and the driver passes the query to the DBMS. An ODBC driver can be thought of as analogous to a printer driver or other driver, providing a standard set of functions for the application to use, and implementing DBMS-specific functionality. An application that can use ODBC is referred to as "ODBC-compliant". Any ODBC-compliant application can access any DBMS for which a driver is installed. Drivers exist for all major DBMSs, many other data sources like address book systems and Microsoft Excel, and even for text or comma-separated values (CSV) files.

Working Example PHP - Database

If I needed to store the form values (let's say we have firstname, lastname on the form), and my database is a **PostgreSQL** server:

The **php** code would be something like:

pg_execute("INSERT INTO people (firstname, lastname) VALUES (?, ?)",$firstname, $lastname);

That **pg_execute** function is part of the **PHP Postgres** library if you wanted to do the same thing in a **MySQL** database rather than a Postgres one you'd have to change all the pg_* calls in the example.

The database abstraction layers (of which ODBC is one), would allow you to write something like:

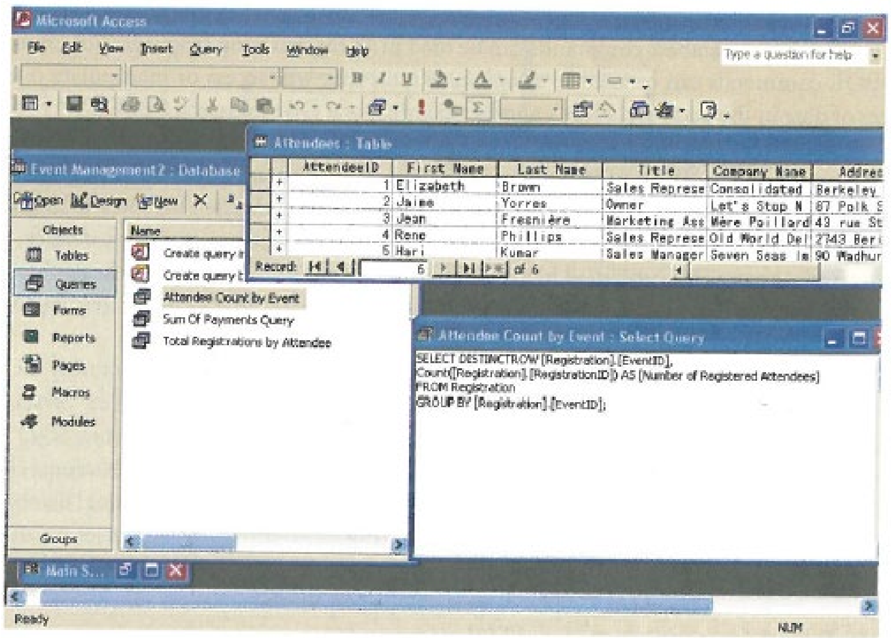
odbc_execute("INSERT INTO people (firstname, lastname) VALUES (?,?)", $firstname, $lastname);

The ODBC driver knows how to talk to multiple databases, so you don't need to rewrite commands when changing Data Bases.

Examples of SQL Commands

SQL Command	Description
SELECT ClientName, Debt FROM Client WHERE Debt>1000	This query displays all clients (ClientName) and the amount they owe the company (Debt) from a database table called Client for clients that owe the company more than $1,000 (WHERE Debt>10001.
SELECT ClientName, ClientNum, OrderNum FROM Client, Order WHERE Client.ClientNum=Order.ClientNum	This command is an example of a join command that combines data from two tables: the client table and the order table (FROM Client, Order). The command creates a new table with the client name, client number, and order number (SELECT ClientName, ClientNum, OrderNuml. Both tables include the client number, which allows them to be joined. This is indicated in the WHERE clause that states that the client number in the client table is the same (equal to) the client number in the order table (WHERE Client.ClientNum=Order.ClientNum).
GRANT INSERT ON Client to Guthrie	This command is an example of a security command. It allows Bob Guthrie to insert new values or rows into the Client table.

SQL lets programmers learn one powerful query language and use it on systems ranging from PCs to the largest mainframe computers (see Figure below). Programmers and database users also find SQL valuable because SQL statements can be embedded into many program-ming languages, such as the widely used C++ and COBOL languages. Because SQL uses standardized and simplified procedures for retrieving, storing, and manipulating data in a database system, the popular database query language can be easy to understand and use.



Simple SQL Queries

The Syllabus Says you should be able to write:

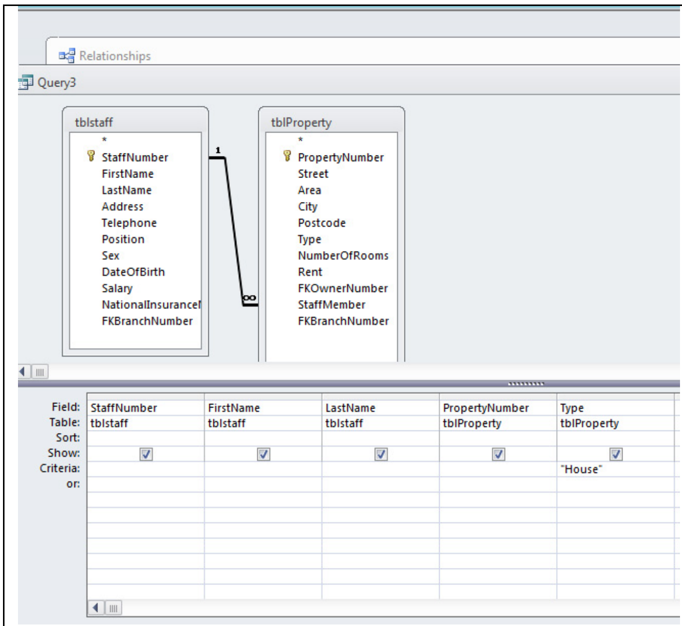
- simple queries using SQL (up to two tables), including insert, update and select queries. This also means you have to know about joins.
- queries across multiple tables using appropriate database tools, including the following: parameter, calculated field, concatenated field, aggregation, update, delete and make table

As previously discussed, SQL can be used as Manipulation Language or a definition langue. This means you can write queries in SQL to query the database and do CRUD (create, read, update, and delete) activities on the data (Select, Update, Delete) or you can use SQL to create data base objects – such a Make Table Queries.

Joins

An inner join in SQL means only include rows where the joined fields are equal.

For example if you wanted to list the HOUSES (Only) managed by STAFF, your QBD grid would like this:



Find ALL Staff who manage properties that are of type HOUSE.

SQL

SELECT tblstaff.StaffNumber, tblstaff.FirstName, tblstaff.LastName, tblProperty.PropertyNumber, tblProperty.Type
FROM tblstaff **INNER JOIN** tblProperty ON tblstaff.StaffNumber = tblProperty.StaffMember
WHERE (((tblProperty.Type)="House"));

So you need to ADD the **INNER JOIN** command **ON** the fields you want to be equal. If you wanted to find a **specific** staff member who managed houses it would look like this:

SELECT tblstaff.StaffNumber, tblstaff.FirstName, tblstaff.LastName, tblProperty.PropertyNumber, tblProperty.Type
FROM tblstaff **INNER JOIN** tblProperty **ON** tblstaff.StaffNumber = tblProperty.StaffMember
WHERE (((tblstaff.StaffNumber)="SG37") **AND** ((tblProperty.Type)="House"));

So you simply put the INNER JOIN on the fields that are linked and the criteria in the WHERE clause.

Quiz

1. Give an example of a dynamic website.
2. Give an example of a tool for creating a dynamic website.
3. What are the advantages for a large organisation to create a database driven website rather than a static website.
4. Do the Tutorial at

   https://www.w3schools.com/sql/sql_intro.asp
   Remember to need to be able to create;
   - simple queries using SQL (up to two tables), including insert, update and select queries. This also means you have to know about joins.
   - queries across multiple tables using appropriate database tools, including the following: parameter, calculated field, concatenated field, aggregation, update, delete and make table.

   https://www.w3schools.com/sql/sql_insert.asp

   https://www.w3schools.com/sql/sql_update.asp

   https://www.w3schools.com/sql/sql_select.asp

   https://www.w3schools.com/sql/sql_join.asp

5. What is SQL?
6. What can SQL do?
7. The best way to get an idea of how these work is to write queries in SQL to do the following:
   a. Write a select query with SQL to find staff who work at branch B3.
   b. Write an update query with SQL to add $1000 to every staff members salary.
   c. Write a delete query in SQL to delete a staff member provided by a user. (use []) a parameter query.

COMPUTER SCIENCE (Exam Question)

Question 8

Consider the folowing table extracts.

Contact (ContactID, SupplierID, Title, FistName, LastName, Department, DeptTitle, WorkPhone)

ContactID	SupplierID	Title	FistName	LastName	Department	DeptTitle	WorkPhone
1	gmag	Mr	Harrison	Taylor	Sales	Sales Manager	(08) 9955 5478
2	mag1	Miss	Mary	Blackson	Sales	Junior Sales Rep	(08) 9420 7890
3	mag1	Mr	Grant	Gilberts	Sales	Sales Manager	(08) 9420 7892
4	napp	Mrs	Gayle	Frampton	Sales	Sales Manager	(08) 9318 4442
5	napp	Miss	Lisa	Davidson	Accounts	Accountant	(08) 9318 4445
6	pb1	Miss	Great	Prigeon	Marketing	Manager	(08) 9360 5699
7	pb1	Mr	Harry	Spalding	Accounts	Manager	(08) 9378 1234
8	ppbk1	Miss	Shona	Jackson	Sales	Junior Rep	(08) 9246 7991

Supplier (SupplierID, SupplierName, SupplierAddress, SupplerSuburb, State, PostCode, City, Phone)

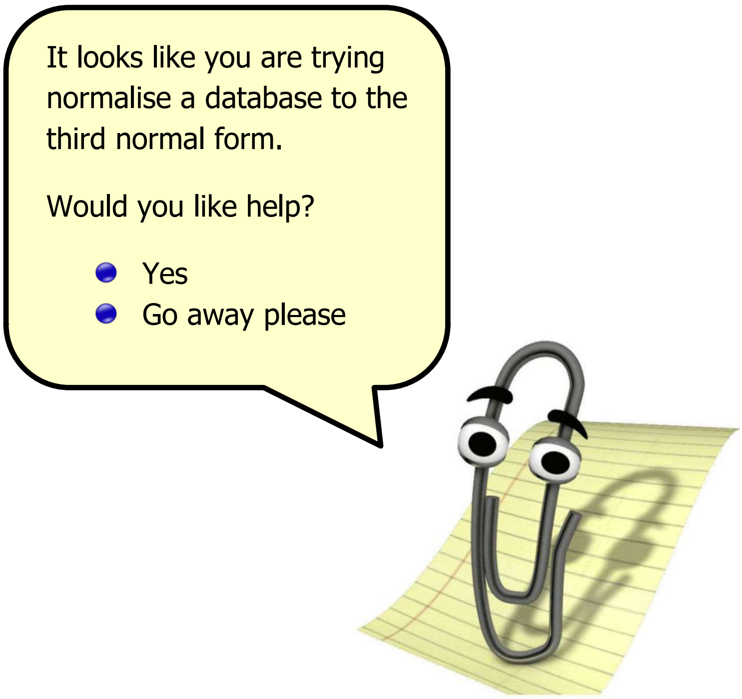
SupplierID	SupplierName	SupplierAddress	SupplerSuburb	State	PostCode	City	Phone
gmag	Global Magazines	116 Mount Street	North Perth	WA	6080	Perth	(08) 9955 5477
mag1	Magazine Extreme	18 Argyle Street	Victoria Park	WA	6200	Perth	(08) 9460 7899
napp	New Distributors	100 Elizabeth Street	North Perth	WA	6010	Perth	(08) 9318 4444
pb1	Presentation Books	10 Tewkesbury Avenue	South Perth	WA	6151	Perth	(08) 9360 5699
ppbk1	Paper Books	10 Albany Street	Bayswater	WA	6053	Perth	(08) 9646 7999

Using a Simple Query Language (SQL), write a query to list all of the first and last names of the Contacts that belong to the Suppliers that reside in North Perth.

References

http://create-website.org/concepts/datadriven.html
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC209517/
https://www.business2community.com/web-design/5-important-points-database-driven-website-01567201
https://en.wikipedia.org/wiki/Open_Database_Connectivity
https://searchoracle.techtarget.com/definition/MySQL

Database Documentation



Building systems that people can actually use and meet their needs is an important goal. On top of this people must be able to actually use the system that is delivered – ease of use depends upon a combination of:

- User Training
- Database documentation for users.

User Database documentation means documentation that helps people get the best out of the software. This documentation might take a tutorial approach, be organised into major concepts or just a reference to commands and what they do. With good Database documentation it should provide a tutorial aspect as well a reference manual.

Start With a Good Database Design

Good database design can make documenting your database a much easier task. Although it might not always be an option, if you have any influence over the database design, here are some key design areas that you should look out for:

- **Naming conventions** - The lack of naming conventions can result in very inconsistent and cryptic names for the tables, columns and other database objects making it more difficult to document them properly. Applying good naming conventions to the database design will promote better readability of the documentation.
- **Primary keys** - While every table should have a primary key that defines which columns uniquely identify a record, most databases do not require them. Not only does this affect database performance and data integrity, knowing what the primary key is for a table is key to understanding the data model and an essential part of the database documentation.
- **Table relationships** - Another key to understanding the data model is documenting the relationships between tables. And just like primary keys, they are not required to be defined.

And even if you do not have any control over the database design, a quick review will help you better estimate how much time it will take and set expectations for the completion of the documentation.

Comment What You Can and Annotate Everything Else

Depending on the commenting capabilities of your database, you should add comments to the following database objects:

- **Tables** - Most tables are typically used to store logical representations of real world objects. The comments for these types of tables should describe what that real world object is. Your database may also include tables, sometimes referred to as "link tables" that are used to create many-to-many relationships between tables. For these types of tables, your comment should describe the purpose of the relationship.
- **Views** - Most views represent stored database queries that answer a specific question. The comments for views should describe what those questions are.
- **Columns** - As tables typically represent real world objects, the columns within those tables are used to store properties related to those objects. Column comments should describe what those properties are and how the column represents them.
- **Stored Procedures and Parameters** - Stored procedures are commented much the same way that programmers comment source code. For documentation purposes, the most important comments are the description of the stored procedure and descriptions of each parameter.
- **Functions, Parameters and Return Values** - Functions are commented the same way as stored procedures. The only difference is that functions also have a return value that should also be described with a comment.

The descriptions that you add to these database objects should be concise and consist of one or two sentences. If there is additional information that is important to document, you should consider structuring your comments into two parts: a summary description followed by a remarks section. The remarks section can be used to include the additional information.

Comments and annotation can make a database much more useable but there are many instances where you are going to need actual separate database documentation. Here are a few examples:

- **System Documentation** - If you are creating a database for a client, a complete set of system documentation is typically a required deliverable.
- **End-User Documentation** - While your DBA and database programmer may have access to the database definition, you may allow some users restricted access that only allows them to execute database queries. These users typically will not have access or the skills necessary to read and understand a database definition. You will need to provide them with some form of documentation that will help them create their database queries. User documentation should also talk about access methods, levels of access and required procedures to avoid data being lost or stolen.
- **Design Reviews** - Not all participants in a design review are going to have the same level of database knowledge and expertise. Your subject matter experts may not be technical at all. To facilitate such a design review, you will need to provide documentation that anybody can read and understand.

Data Dictionary

A Data Dictionary provides detailed information about the data, such as standard definitions of data elements, their meanings, and allowable values. While a conceptual or logical Entity Relationship Diagram will focus on the high-level business concepts, a Data Dictionary will provide more detail about each attribute.

Essentially, a data dictionary provides a tool that enables you to communicate stakeholder requirements in such a way that your technical team can more easily design a relational database or data structure to meet those requirements.

The Key Elements of a Data Dictionary

A Data Dictionary provides information about each attribute, also referred to as fields, of a data model. An attribute is a place in the database that holds information. For example, if we were to create a Data Dictionary representing the articles of an online magazine, we'd potentially have attributes for article title, article author, article category, and the article content itself.

A Data Dictionary is typically organized in a spreadsheet format. Each attribute is listed as a row in the spreadsheet and each column labels an element of information that is useful to know about the attribute.

Let's look at the most common elements included in a data dictionary.

- **Attribute Name** – A unique identifier, typically expressed in business language, that labels each attribute.
- **Optional/Required** – Indicates whether information is required in an attribute before a record can be saved.
- **Attribute Type** – Defines what type of data is allowable in a field. Common types include text, numeric, date/time, enumerated list, look-ups, booleans, and unique identifiers.

While these are the core elements of a data dictionary, it's not uncommon to document additional information about each element, which may include the source of the information, the table or concept in which the attribute is contained, the physical database field name, the field length, and any default values. They also contain information about which Forms, Reports/ Queries use that data element so any changes to the structure of the element or the object can be judged and allowed for.

Example of a Data Dictionary

You are probably wondering how all of this comes together.

Here's a look at a simplified example data dictionary that contains the attribute from an online Magazine article example, along with critical information about each attribute.

Attribute Name	Required	Type	Field Length	Default Values	Constraints	Notes
Article Ref No	Yes	PK	4	Auto generated	Primary key	Auto generated
Article Title	Yes	Text	250	n/a	Not null	Can contain HTML
Article Author	Yes	Look-up	n/a	n/a	Not null	
Article Category	Yes	Look-up	n/a	Uncategorized	Not null	
Article Content	No	Text	Unlimited	n/a	Not null	Can contain HTML

As you can see, a data dictionary defines critical information about each attribute in a business-focused way. It also organizes information that might otherwise be scattered across multiple different documents and specs, making it easier for your database developer to design or update a database that meets business requirements.

Quiz

1. If you were writing a manual for a database what would be three important tasks or concepts to cover and how would you address them?
2. What is a Data dictionary and who would refer to this document and why?

References

https://techwriter.me/best-practices-guide/documenting-databases.aspx

https://www.bridging-the-gap.com/data-dictionary/

Privacy Act 1988

http://www.oaic.gov.au/privacy/privacy-resources/privacy-guides/

The legal aspects data and information security is handled by The Privacy Act 1988.

Organisations are required to take under the Privacy Act 1988 to protect the personal information they hold from misuse, interference, loss, and from unauthorised access, modification or disclosure.

The Privacy Act includes 13 Australian Privacy Principles (APPs) that regulate the handling of personal information by certain organisations and Australian Government agencies.

APP 11 requires APP entities to take active measures to ensure the security of personal information they hold and to actively consider whether they are permitted to retain this personal information.

Specifically, APP 11.1 states that an APP entity that holds personal information must take reasonable steps to protect the information from misuse, interference and loss, as well as unauthorised access, modification or disclosure.

Under APP 11.2, APP entities must also take reasonable steps to destroy or de-identify the personal information they hold once it is no longer needed for any purpose for which it may be used or disclosed under the APPs.

What is personal information security?

Under The Privacy Act defines 'personal information' as 'information or an opinion about an identified individual. This might include a person's name and address, medical records, bank account details, photos, videos and even information about what an individual likes, their opinions and where they work.

An important subset of personal information in the Privacy Act is 'sensitive information.' Sensitive information is defined in the glossary, and includes health information. The Privacy Act generally affords a higher level of privacy protection to sensitive information than to other personal information.

Whether information constitutes personal information under the Privacy Act will depend on whether an individual can be identified or is 'reasonably identifiable' in the particular circumstances.

Personal information security

'Information security' involves all measures used to protect any information generated by an entity or individual, that is not intended to be made publicly available, from compromise, loss of integrity or unavailability. This can include personal information, security classified information and commercially confidential information.

Why is it important?

Personal information security is about more than just ensuring compliance with the requirements of the Privacy Act. If you mishandle the personal information of your customers, it can cause a financial or reputational loss to the customer. In turn, this can also lead to a loss of trust and considerable harm to your reputation. A significant breach may result in a loss of customers or business partners and revenue.

If personal information that is essential to your functions or activities is lost or altered, it can have a serious impact on your ability to undertake business as usual.

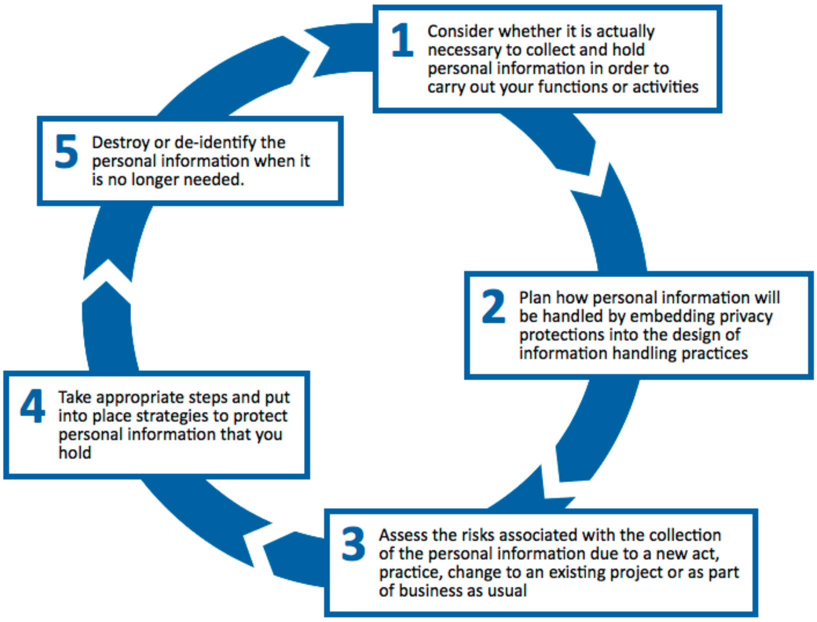
The benefits of applying personal information security to your business practices can include more efficient processes. It also reduces the risk of privacy breaches and the time and resources involved in addressing any breaches that do occur.

Many of the steps and strategies in this guide will also assist you to take reasonable steps to ensure good handling of other types of information, such as commercially confidential information.

The information lifecycle

If you handle personal information, you should consider how you will protect personal information during the stages of its lifecycle.

The information lifecycle



Circumstances that affect assessment of reasonable steps

What qualifies as reasonable steps to ensure the security of personal information depends on the circumstances, including the following:

- the nature of your entity
- the amount and sensitivity of the personal information held
- the possible adverse consequences for an individual in the case of a breach
- the practical implications of implementing the security measure, including the time and cost involved

Nature of the entity

The size of your entity, its resources, the complexity of its operations and the business model, are all relevant to determining what steps would be reasonable to protect the personal information you hold. For instance, a franchise or a business using outsourcing is likely to provide access to its personal information to third parties (franchisees and contractors). The reasonable steps it takes may be different to those it would take if it did not operate in this manner.

When you outsource any of your personal information handling to a third party (including to a cloud service provider), and you continue to 'hold' that information, you will still be subject to APP 11. Part B sets out steps to assist you when implementing information handling practices.

Amount and sensitivity of personal information held

Generally, as the amount and/or sensitivity of personal information that is held increases, so too will the steps that it is reasonable to take to protect it.

The community generally expects that their sensitive information will be given a higher level of protection than non-sensitive information. This expectation is reflected in the increased privacy protections which apply to the handling of sensitive information.

Although it is not defined as sensitive information under the APPs, people often expect that their financial information will be given a high level of protection. The protections in the Privacy Act in relation to credit reporting information and tax file numbers reinforce this.

Steps and strategies which may be reasonable to take

Appropriate security measures for protecting personal information need to be considered in regards to all of your entity's acts and practices. This section outlines examples of key steps and strategies you should consider under the nine broad topics listed below. It includes a number of questions to ask yourself when considering or implementing these steps and strategies.

- **Governance, culture and training.**
  Insufficient interest in personal information security from staff, in particular senior management including the board (or equivalent decision making body), can lead to threats to the security of personal information being ignored and not properly attended to. Appropriate training can assist in mitigating these issues and making staff aware of common personal information security threats (see 'Personnel security and training' section below).
- **Internal practices, procedures and systems.**
  For the purposes of APP 11, you should document the internal practices, procedures and systems that you use to protect personal information. Your documentation should outline the personal information security measures that are established and maintained against the risks and threats to personal information. These documents should be regularly reviewed and updated to ensure they reflect your current acts and practices.
- **ICT security.**
  Effective ICT security requires protecting both your hardware and software from misuse, interference, loss, unauthorised access, modification and disclosure. However, ICT security measures should also ensure that the hardware, software and personal information stored on it remain accessible and useful to authorised users.

  It is expected that entities regularly monitor the operation and effectiveness of their ICT security measures to ensure that they remain responsive to changing threats and vulnerabilities and other issues that may impact the security of personal information.
- **Access security.**
  Access security and monitoring controls help you protect against internal and external risks by ensuring that personal information is only accessed by authorised persons.

  You should also consider whether your cloud service provider should be required to have similar controls to those you might apply to your own systems, such as governance arrangements and controls relating to software security, access security and network security set out in the sections above.

- **Third party providers (including cloud computing).**
  Entities that outsource part or all of their personal information handling will need to consider whether they still 'hold' that personal information. If so, APP 11 will apply and you will need to take reasonable steps to comply with APP 11.
- **Data breaches.**
  In the event of a data breach, having a response plan that includes procedures and clear lines of authority can assist you to contain the breach and manage your response.
- **Physical security.**
  Physical security is an important part of ensuring that personal information is not inappropriately accessed. You need to consider what steps, if any, are necessary to ensure that physical copies of personal information are secure. Similarly, you should consider whether the workspace itself is designed to facilitate good privacy practices.
- **Destruction and de-identification.**
  Where an entity holds personal information it no longer needs for a purpose that is permitted under the APPs, it must ensure that it takes reasonable steps to destroy or de-identify the personal information.
- **Standards.**
  Standards are documents that set out requirements, specifications and procedures designed to ensure products, services and systems are safe, reliable and consistently perform in the way they are intended.

These steps and strategies are not intended to be prescriptive or exhaustive and it may not be necessary to take all the steps and strategies outlined.

Sensitive Information

Sensitive information is a sub-set of personal information and is given a higher level of protection under the National Privacy Principles

Sensitive information is defined in the Privacy Act to mean information or an opinion about an individual's:

- racial or ethnic origin;
- political opinions;
- membership of a political association;
- religious beliefs or affiliations;
- philosophical beliefs;
- membership of a professional or trade association;
- membership of a trade union;
- sexual preferences or practices; or
- criminal record.
- health information
- genetic information

Generally, an organisation is not allowed to collect sensitive information from an individual unless:

- the individual has consented;
- collection is required or authorised by law;
- the information is required to establish or defend a legal or equitable claim; or

Non – profit organisations, including charities, may collect sensitive information if:

- it relates solely to the members or the organisation, or people who have regular contact with it for the purpose of its activities; and
- the organisation undertakes to the individual that it will not disclose the information without consent.

Exercise

1. Name an organisation that has person information about you.
2. What fields of data will they have – for example name, home phone number etc.
3. Would the school for example have any information on students that could be regarded as sensitive information? If yes why would this information be considered sensitive information?
4. What law covers the security of data and information stored on people?
5. What would should happen in the event of a data breach of personal information in an organisation.

References

1. https://www.oaic.gov.au/
2. http://www.crowellphoto.com/
3. Black Hat: Misfits, Criminals, and Scammers in the Internet Age. John Biggs
4. http://time.com/3731150/vince-vaughn-dave-franco-istock-images/
   https://www.mediaandprivacyriskreport.com/locked-folder-462368213/
5. https://www.tokopedia.com/dojoryuninja/baju-ninjutsuninja-shozoku
6. https://www.shutterstock.com/image-illustration/bank-vault-door-255636325

Control Structures – For Loop

for i in range(x):	i =counter variable – will count from zero till the value before x. note you need to put in the : at the end or it won't run. eg for i in range(4) 0, 1, 2, 3
for i in range(start, beyondLast, step):	i =counter variable – will count from *start* till the value before *beyondLast*, stepping by *step* eg for i in range(1,3,12) 1, 4, 7, 10
for i in [x,y,z]:	i =counter variable – will cycle once through the list given. eg for i in ["hi", "how", "are", "you"] hi, how, are ,you
Note that anything indented directly under the for loop will be run with the loop.	for i in range(3): print ("We like Python's turtles") We like Python's turtles We like Python's turtles We like Python's turtles

Error Messages

ParseError

Parse errors happen when you make an error in the syntax of your program. Syntax errors are like making grammatical errors in writing. Most commonly you have missed a comma or bracket.

TypeError

TypeErrors occur when you you try to combine two objects that are not compatible. For example you try to add together an integer and a string.

NameError

Name errors almost always mean that you have used a variable before it has a value. Often NameErrors are simply caused by typos in your code.

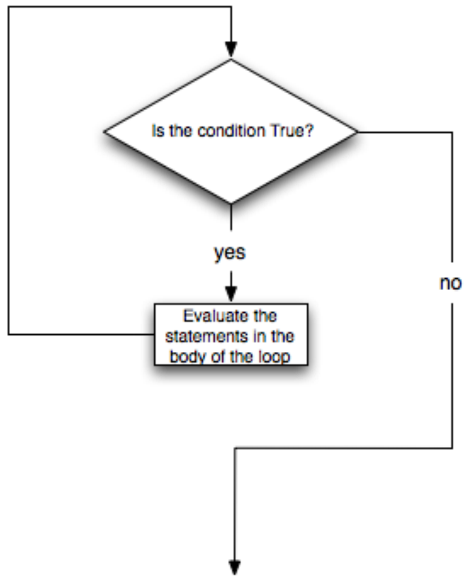
ValueError

Value errors occur when you pass a parameter to a function and the function is expecting a certain limitations on the values, and the value passed is not compatible.

While Loop

There is another Python statement that can also be used to build an iteration. It is called the while statement. The while statement provides a much more general mechanism for iterating. Similar to the if statement, it uses a boolean expression to control the flow of execution. The body of while will be repeated as long as the controlling boolean expression evaluates to True.

The following figure shows the flow of control.



We can use the while loop to create any type of iteration we wish, including anything that we have previously done with a for loop.

Use a for loop if you know the maximum number of times that you'll need to execute the body. For example, if you're traversing a list of elements, or can formulate a suitable call to range, then choose the for loop.

So any problem like "iterate this weather model run for 1000 cycles", or "search this list of words", "check all integers up to 10000 to see which are prime" suggest that a for loop is best.

By contrast, if you are required to repeat some computation until some condition is met, as we did in this 3n + 1 problem, you'll need a while loop.

Example

A woman has saved $200 to spend on Christmas presents. A program is needed to accept the values of presents which are to be bought and to show the amount of money remaining. The program must stop when the value of a present exceeds the money remaining.

PLANNING

How many presents can be bought with $200? The answer depends on the value of the presents. For example, if only small presents were being bought, there may be enough money to buy 25 of them. If large presents are being bought, there may only be enough money for 2 or 3.

Let us see how the algorithm might work. If the first present costs $40, there will be $160 left and the program must repeat. If the next present costs $100, there will be $60 left and still the program must continue. If the next present costs $80, there will not be enough money and so the program can stop and give a message.

In programming, there is a looping construct which will test a condition and continue to repeat a process while the condition is true. When the condition is not true, it will stop repeating. This is the DO – WHILE loop.

```
#Xmas Program Code

print("A program to keep track of Xmas spending")
money = 200
cost = float(input("What is the cost of the first present? $ "))
while  cost<money:
   money = money-cost
   print("Money left now is $ ",money)
   cost = float(input("What is the cost of the next present? "))
print("There is not enough money left for this present")
```

Exercises

1. A program is needed to accept numbers from the keyboard and to add these numbers to a running total. The program must stop and show the total when the number 999 is entered. Copy and complete the following program plan and write the code.
2. A program is needed to count the number of attempts it takes an athlete to long jump further than 6m. Copy and complete the following plan and write the code.
3. A group of volunteers have offered to assist in planting trees for the Green Australia campaign. There are 5000 trees to be planted. Plan and code a program to accept the number of trees taken by each volunteer. The program should stop when there are less than 1000 trees left.
4. Plan and code a program that asks the user to guess a number in the range 1.. 20. If the user has not guessed correctly in 5 attempts, the program should give a message and show the number. Note you will have to look up how to create random numbers.

Functions

Functions are bits of reusable code that you can “call”.

def name(parameters):

statements

return x

def	tells python you are creating a function
name	is the name of that function
parameters	are the variable you will pass into the function. There don’t have to be any.
statements	the code indented after the def is the code that will run when the function is called
return	you don’t have to have a return statement but if you do it will return a value (x or whatever) to the function.
local variable	is a variable contained within a function. It’s value is lost when the function finishes even if the same function is called again.
gobal variable	is a variable outside of a function

Note a function can call another function within its code.

Benefits of Functions

1. Gives you an opportunity to name a group of statements. Functions can simplify a program by hiding a complex computation behind a single command.
2. Creating a new function can make a program smaller by eliminating repetitive code.
3. Sometimes you can write functions that allow you to solve a specific problem using a more general solution.

Example

```
import turtle

#this defines the function "drawSquare" with the parameters t and sz.
#That is it is a function called drawSquare that expects two variables to be passed into it when it runs
def drawSquare(t, sz):
    for i in range(4):
    t.forward(sz)
    t.left(90)

wn = turtle.Screen()          # Set up the window and its attributes
wn.bgcolor("lightgreen")

alex = turtle.Turtle()        # create alex
drawSquare(alex, 50)        # Calls the function drawSware and passes alex and 50 to the function
```

Functions Statements Expressions Operators

print ()	• Prints to screen
type ()	• Returns the type of data eg integer, float or string
int ()	• Converts into an integer – discarding any decimals so 3.99 becomes 3 – it can also turn a string into an integer
float()	• Converts a sting or an integer into float
len()	• Length of a string
+	• Addition
-	• Subtraction
*	• Multiplication
/	• Division
**	• Exponentiation (to the power of) eg 2**3 = 2*2*2 = 8
//	• Divides discarding decimals eg 8 // 3 = 2
%	• Divides and returns remainder eg 5 % 3 = 2 (it goes in once with 2 left over)
input()	• Allows the user to provide a prompt string eg name = input("What is your name :")
import math	• imports more maths functions into python
math.pi	• π
import random	• imports random number functions into python
random.random()	• returns a floating point number in the range [0.0, 1.0]
random.randrange(x,y)	• The randrange function generates an integer between its lower and upper argument, the lower bound is included, but the upper bound is excluded. So (1,100) would include all numbers from and including 1 to and including 99.

Python Exercises Week One

Pseudo Code	Python

{A program to calculate the circumference of a
circle whose radius is 5.0 cm}
Set a Constant Pi to 3.14
{begin}
    Radius = 5.0
    Circumference = 2 * Pi * Radius
    Output(“the circumference is”, circumference :1 :2
)
{End of Module}
	#A program to calculate the circumference of a
	# circle whose radius is 5.0 cm
	
	Pi = 3.14 #sets a constant identifier.
	
	#begin
	Radius = 5.0
	Circumference = 2 * Pi * Radius
	print('the Circumference is %5.3f.' %Circumference)
	#End of Module

Code and run this program and then:
Note the following:

1. An identifier is a word used to name whatever can be named in a program. Identifiers need to be sensible / meaningful and reflect the data that they hold.
2. Python identifiers are case sensitive.
3. You cannot use reserved words as identifiers.
4. This program has 2 variables and 1 constant.
5. The # is used to apply comments that help explain the code- these are part of the Programs’ Internal Documentation.

Exercises

1. Write down the rules for Python Identifiers. Are python identifiers case sensitive? {use the Python Help in IDLE and search for the Lexical Analysis}
2. What is an alpha numeric character?
3. Is there any limit on the length of identifiers?
4. Where in a program can a comment appear?
5. Give 5 examples of Reserved words in Python.

Programming Exercises

1. Code the program simple.

File Edit Format Run Options Windows Help

```
#Program Simple


print(' Hello');
print(' How are  you.');
print(' I am your special friend.');
print(' For money.')
print(' Nit wit.');

print('12 % 5 is' , 12 % 5);
# end Module
#1. Apply the following labels to the code: Program Header, Statement,
#Block, Syntax error,Output Statement, Program Identifier}
#{2. Try and work out what DIV(//) and MOD( %) do by changing the 12 and/or the 5 to different values}
```

Expression	My Guess	Actual Output Value
print(2 + 3 * 4)		
print((2 + 3) * 4))		
print((2 + (3 * 4))		
print(12 + 9)		
print(91.5 + 8.5)		
print(1 * 10 + 3)		
print(1 + 10 * 3)		
print(“Leaping” + “ Lizards”)		
print(“43” + “ 7”)		
print(“43 + 7”)		
print(4.0/ 9.0)		
print 9 / / 4		
print 9 % 4		

Problem

1. Your friend has 175,200 M&M’s, how many plastic bags will she have to buy to put the M&M’s into bags of 37, and how many M&Ms will be left over. Code a Python program to solve this problem.
2. Read in 5 numbers and add them up. See if you can remember how to add up ONLY the positive numbers, count how many positive numbers, and get the average of the positive numbers.
3. Write the pseudo code for your programs in (1) and (2) above.

Python Exercises Week Two

Procedures / Modules

By dividing the program up into modules we can build a program using a divide and conquer technique. Solving each small sub problem and then putting the whole thing together at the end. This divides the program into two distinct parts:

- The **Declaration** part where you declare your constant, types, variables and procedures/ modules.
- And the **Statement** part or **main** program where you tell Python to execute the procedures/ modules declared above.

This makes for easier programming/ easier error detection, greater portability and re- usability. We can develop and test one module at a time and make sure that it is correct before moving onto the next module.

When we put the name of the module in the main line, we are calling or invoking the module – in other words getting the program to execute the statements of the module – and we can do this as many times as we like.

The programmer is responsible for calling the procedures/modules in the correct logical order.

It is a little like – “here is the description of what I what you to do when I say “Perform” and then, at the appropriate time saying: **PERFORM**.

- Consider the following program. Find the first unencumbered begin…… the first begin that is not preceded by the word “Module“ or “function”

```
Program payroll
{this program calculates the payroll for all employees}
    const
        normal = 37.5;
        threshold = 100;
        taxrate = 0.20;
    type
        codetype = 1000..9999;
    var
        code : codetype;
        rate, hours, gross, overtime, tax, net : real;
        employee, staffsize : 1..maxint;

Module Announce;
    begin
        OUTPUT(‘ This program calculates your employees” wages .’); {Announce}
    end: {of Announce}

Module FindStaffSize;
    begin
    OUTPUT(‘How many staff do you have.’); { Find Staff Size}

    INPUT( staffsize );
    end; {Of Find Staff Size}

Module GiveInstructions;
    begin
        OUTPUT(‘ For each in turn give me the code number, basic rate’);
        OUTPUT(‘ and hours worked on one line, separated by spaces.’);
    end;

Module GetWageDetails;
    begin
        INPUT (Code, rate, hours); {Get Wage Details}
    end;

Module CalculateOvertime;
    begin
        if hours > normal then {Calculate Overtime}
        overtime <<- hours – normal
        else
        overtime <<- 0;
    end; {of calculate overtime}
```

```
Module CalcGross;
    begin
        gross <<- hours * rate + overtime * 0.5 * rate; {Calculate Gross}
    end;

Module calctax;
    begin
        if gross <= threshold then {Calculate Tax }
            tax <<- 0
        else
            tax <<- taxrate * (gross – threshold );
    end; {Of CalcTax}

Module CalcNett;
    begin
        nett <<- gross – tax; {Calculate Nett wage}
    end;{Of CalcNett}

Module GiveWageDetails;
    begin
        OUTPUT('Employee' , code : 4, ' "s wage is $ ' , gross : 6 : 2); {output}
        OUTPUT(' and after tax of $' , tax : 6 : 2, ' this leaves $ ' , nett : 6 :2, ' . ');
    end;

Begin {Main Program}
Announce;
FindStaffSize;
GiveInstructions
    for employee : =1 to staffsize do
        begin {Loop on staff}
            GetWageDetails
            CalculateOvertime;
            CalcGross;
            Calctax;
            CalcNett;
            GiveWageDetails;
        End; {of loop on employee}
end.
```

Note the following

- The program is now longer.
- The program executes a more slowly.
- The Main line gives a good description of what the program does.
- The program has clear parts that are do with Input/ Processing / Output.
- The program has comments to aid readability and comprehension by other programmers.
- The program uses sensible variable names to aid readability and comprehension.
- The main line – listing the modules - can be written first and then each module developed in turn. This known as Top Down Programming.
- Each module represents a Process Bubble in a DFD.
- Modules can contain their own modules and functions.
- Module can contain their own variables/const and types and these are called LOCAL variables.

Exercises

1. Develop a modularized program to read in 3 numbers, add them up, and report which is biggest of the 3.
2. Develop 3 simple modules to Calculate (1) The area of a circle (2) the Area of a Square (3) the Area of a triangle. Find out from the user which Area they wish to calculate, then execute the modules to Input the necessary data, Calculate the Area and output the results. The blank line in the main program is a module that you have to develop.

The Main line may look as follows:

```
Begin {Main}
    Announce;
    GetChoice;
    Case choice of ## In Python you will need to use an extended if then else
    1: begin
        GetCircleData;
        CalcAreaCircle;
    ______________________

    End;
    2: begin
        getSquareData;
        CalcAreaSquare;
    End;
    ______________________

    3: begin
        GetTriangleData;
        CalcAreaTriangle;
    End;
    ______________________

    End;{ Of Case statement}
    End. {MAIN}
```

Python Exercises Week Three

Software Tools Selection & Parameters

If Statements, Boolean Expressions. Classes of Parameters (Val, Reference) Parameters: Actual & Formal. Trace Tables

Selection

The **selection** control structure gives the programmer the ability to force one group of statements or a statement to be executed rather than another, but not both. The choice hinges on the value of a **Boolean expression**. This is a special type of **logical** expression that can only be evaluated to true or false – no maybes.

The expression looks like this: A > B
and uses all of the math connectors that you know about > < > = <= <>,
as well **AND**, **OR**, & **NOT** to build complex Boolean expressions such as: (A > B) AND (B > C) or NOT(a = b) which is the same as (A <> B), the 2nd expression being a little riskier- why?

When built into a selection statement the Boolean expressions appear as follows:

```
    If (a > b) then
        StatementOne
    Else
        StatementTwo;
```

The "else" bit is optional of course.

When writing If statements in Pseudo Code, indenting is imperative to aid understanding.
For Example:

```
    if (a > B ) then output( A) else if ( a = b ) then output(b) else output( a)…….
```

is very difficult to follow unlike:

```
    If  (a > b) then
        Output (a)
    Else
        if ( a = b ) then
            output (b)
        else
            output(a);
        Endelse {of a = b}
    Endif {of a > b}
```

Parameters

Parameters are the variables that the modules (procedures/functions) use and / or change as a result of their processing. The also make modules more transportable and re-usable many times in the program.

Parameters can be classified as **Formal Parameters** which appear in the module header and the **Actual parameters** that appear in the module call in the mainline.

```
module GetNumbers ( a, b, c :); {formal Parameters}
Begin
   output('Give me 3 numbers, all on one line and press ENT or <RTN>');
   input(A, B, C);
 #End Module GetNumbers
```

And then

GetNumbers(num1, num2, num3); { actual Parameters-which will be reference parameters}

The actual parameters MUST exist as global (non local) variables to that procedure, whereas the Formal parameters are only used to tell the programmer WHAT is happening to the ACTUAL parameters that must be the same type as shown in the procedure header.

So use of parameters helps programmers see what procedures are doing as well making procedures reusable and portable.

We can call module GETNUMBERS again:

    Getnumbers(num4, num5, num6);

Indeed we can call it as many times as we like as long as can changing the actual parameters, the new data will be placed in those memory spaces listed in the procedure call.

How DO we tell if a Parameter is Value or Reference?

- Var or **reference** parameters are changed by the procedure as a reference parameter is linked directly to the global variable used as the AP.
- **Value** parameters use the data from the memory spaces during the module but any changes to the data are not passed back out to any global variable as the value parameters result in local copies of the global actual parameters being made and no link is made to the global memory space.
- **Formal** Parameters are in the module header.
- **Actual** parameters are in the module call.

Consider the pseudo code below and identify the **Formal** and **Actual** Parameters and the **Value** and **Reference** Parameters. You will need to study the code and see which parameters appear on the RHS of the = sign – if they do, then they are Reference as their value is being changed.

Pseudo Code	Comment/ Question
Module CalculatePay (CRate, CHours, CPay) CPay ⍈ CRate * CHours **End** CalculatePay	1. List the Formal Parameters for Calculate Pay: 2. Divide your list into Value and Reference Parameters
Module CalculateTax (CPay, TTax) YearlyPay ⍈ CPay * 52 **Case** YearlyPay of <= 6000 : YearlyTax ⍈ 0 <= 30000 : YearlyTax ⍈ (YearlyPay – 6,000) * 0.15 <= 75000 : YearlyTax ⍈ 3600 + (YearlyPay – 30,000) * 0.3 <= 150000 : YearlyTax ⍈ 17100 + (YearlyPay – 75,000) * 0.4 > 150000 : YearlyTax ⍈ 47100 + (YearlyPay – 150,000) * 0.45 **End** Case TTax ⍈ YearlyTax / 52 **End** CalculateTax	3. List the Formal Parameters for Calculate Tax 4. Divide your list into Value and Reference Parameters

Finally YearlyPay and YearlyTax are examples of **Local Variables** – they exist ONLY for the module to do its job and cannot be used by any other module unless that module is nested inside the module where the local var is declared.

The important principle for programmers is known as The **Principle** of **Deepest Declaration** – that means you should declare your **variables** as **deeply** as possible and only declare globally those vars/ const / types that are used by more than one module. This then avoids un intended side effects and variables being changed by accident by procedures that are not supposed to be accessing them.

To Do

- Explain what is meant by a Selection Control Structure and list the two methods of controlling selection in Python, explaining why and how they are different.
- Write out the pseudo code for Selection.
- Grading Program for students – demo selection and modules with Parameters.

Homework

1. Trace tables of nominated code/ pseudo code.
2. Define all of the terms in bold.

Sample Exam Question

List 3 Boolean operators used in software development**. (3 Marks)**
Clearly differentiate between Val and Reference parameters, using an example to illustrate your understanding. **(4 Marks)**
Conditions are used in algorithms. What could you add to the following condition to change its result? Give examples of what you would do. **(2 marks)**

Age>= 18 and Sex = “Male” or Qualification = “Degree”

Python Exercises Week Four

Repetition / Functions

Loops(For, While, Repeat), Data Types, Tracing, Case Statements, Ordinal / Enumerated Types, functions

This week we will be considering:

- Data types – ordinal/ enumerated.
- Tracing
- Control structures- Repetition, and Selection II – The case statement.

The **simple data types** that you know about can be classified as **ordinal** and non-ordinal types. The ordinal types have discrete values and are not approximations. The non-ordinal simple type is the ***real*** data type.

It is always best to try and define your types as tightly as possible as it helps make the program more robust and also assists in data validation and error trapping.

More advanced than this, you can declare your own types with your own values, these are called **Enumerated types**.

Eg

Type

DaysOfWeek = (mon, tues, wed, thus, fri ,sat, sun);

and then a

var

days : daysOfWeek;

which you can use in the internal workings of the program. This tells other programmers that this is an important concept and protects the program from accidentally giving variables the wrong value. It is also more self-documenting.

Eg

For days ← Mon **to** Fri **do**

{statements}

Is a loop that will run from Mon to Friday.

Now **Repetition** (as we have started it above)

There are three ways of forcing some statements to be repeated in a program, these are the **For – Do Loop** (fixed iteration) and the **While – do** and **Repeat** until loops (both variable iteration).

For Do Loops

These loops always run and always executes a fixed number of times.

Eg

```
For count ← 1 to 10 do
    Output('Hello');
```

Note the following

- Count is known as the Loop control variable and it must be an ordinal variable (ie integer)
- The LCV cannot/ should not be changed inside the loop
- It automatically only increments by one and only one.
- The starting values and final values of the loop must also be ordinal(integer)
- It can count **downto** ie for 10 downto 1 do ……
- As with any control structure (except repeat) if you have more than one statement to be executed, then you must make a block, encompassing the statements with #End.

  Eg

  For count ← 1 **to** 10 **do**

  S1;

  S2;

  S3;

  #End; {Of loop on count}

While Do Loops

These are also known ***as test first*** loops and execute (or not) depending upon the evaluation of a **Boolean expression**. Once the BE becomes false, execution of the loop ends.

Eg

While x > 10 **do**

S1;

X x – 1

Endwhile {Pseudo code}

Note the following

- Use this loop if the number of times the loop has to iterate always changes and may include zero iterations.
- The programmer must ensure that there is a statement in the loop to change the BE so that it may, at some stage, become false and so terminate the loop. Failure to do this will result in an infinite loop.
- Avoid using this loop where the loop is patently going to execute at least once – use a repeat loop instead.

EG

X ← 10;

While X <> 1 **do**

Begin

S1;

S2;

X ← x – 2;

End;

This loop clearly will execute at least once as x starts at ten.

However, there is a logic error – can you pick it up and what problem will it cause????

Repeat Until Loops

These loops are also **Conditional** iteration and depend upon the value of a BE, except that, as the BE is at the end of the loop, they are a **TEST LAST** loop and so are always done ***at least once***.

Eg

Pseudo Code	Python
X ← 10; **Repeat** S1; S2; X ⍰ x – 1; **Until** x = 0;	X = 10 **While** True; S1; S2; X = X- 1 **If** X = 0: Break
	You will ONLY get Pseudo Code in any exam. This is the only time you can use the ***break*** statement.

Note that in this case you might as well have used a for do loop as the repeat loop clearly has a fixed number of iterations.

Note the following

- The programmer must ensure that there is a statement in the loop to change the BE so that it may, at some stage, become false and so terminate the loop. Failure to do this will result in an infinite loop.
- You can use complex Boolean expressions such as: (x<= 3) or (found = true);
- Repeat loops are often used to validate the user input. Imagine what an ATM does in reading your attempt to enter a PIN. After 3 goes at remembering your Pin, it snatches the card and ruins your day.

Functions

A special class of module also exists in most programming languages, that specifically calculates a value. These are known as functions and closely resemble math functions. The main differences are:

1. Functions are a type eg **function** Sum(a, b : real) : real; { This functions take two pieces of input data and adds them up}
2. The function stores the result in a memory space called “sum” (in this case) that it sets up without telling or asking you – so it has to has a type.
3. It is activated by a function designator – which can be used in an expression rather than by “calling” it as you do procedures.

A function to calculate the greatest common divisor of two natural numbers(integers >=0) would look as follows:

Pseudo Code	Python
Function GCD (p, q) : natural; **Begin** Remainder ← p mod q; **While** remainder <> 0 do P ←Q; Q ← remainder; Remainder ← P **mod** Q; **End**while GCD ← Q; #**End**; { of Function GCD}	**def** CalcGCD(p,q):# p, q are value parameters or input to the function rem = p % q # rem is a local var **while** rem != 0: p = q q = rem rem = p % q #endwhile **return** q # What the function Calculates / returns to the NAME of the function #endCalcgcd

To operate the function you could do either of the following:

Method One	Method Two
print(CalcGCD **(72, 18)**; { quite often used to test functions}	Input (p, q); GlobalGCD = GCD(p, q); print(GlobalGCD) #Where GlobalGCD is a global variable

Python vs The Syllabus

In Python, every block of code (or Module) starts with the same structure:

Def # Module Name

For our purposes, as we are learning to deal with a syllabus in Pseudo Code…

1. If the **def** returns a value by calling **return** x. The value is given to the name of the def and we will consider it a **function**.
2. If the def is called by just using its name somewhere else without passing its value to a variable then we will consider it a **module** not a function. Generally these will access a **GLOBAL** variable instead…

EG

Python Function	Python Module / Procedure

n = int(input('Enter a value for n!'))
def Nfactorial(n):
p = 1
for count **in** range (1,n+1): p = p * count;
return p
#endfunction
 	
Nfact= Nfactorial(n)# Invoke the function
print("Nfactorial of",n,' is{:5d}'.format(Nfact))
	global NFact
	n = int(input('Enter a value for n!'))
	def Nfactorial(n):
	global NFact
	p = 1
	for count in range (1,n+1):
	p = p * count;
	NFact = p # Give the result to the global variable
	print('{:10d}'.format(NFact))
	#**end module**
	
	Nfactorial(n)# Call the Module
	print("Nfactorial of",n,' is{:5d}'.format(NFact))

See Also Appendix B: Local and Globals in

Python Homework

Stair & Reynolds 171 -175 – read- see Moodle
Tracing and trace tables exercises.

Sample Exam Question

Explain the difference between a completed module and a stub. (2 Marks) Explain the difference between a function and a module. (2 Marks) What is a subrange type – give a clear example of such a type declaration. (2 Marks)

While Do Loops Exercises

1. What is meant by conditional execution?
2. What is an infinite loop?
3. Identify the error in the following code fragment (it is not pseudo code)
```
Readln(number);
While not (number= flagvalue) do
    Writeln(number);
    Readln(number);
```
4. Design a program, and proves that it works by tracing it, that returns whether or not a number entered by the user is a Prime Number.
5. Code the nFactorial Algorithm.
6. What is the problem in the following:
```
Readln(X):
While x >0.0 do
Begin
    Y := sqrt(x);
    Writeln(x :10:5 , y:10:7);
End;{ of while x > 0.0}
```

   Fix the problem end then draw the pseudo diagram for the code.
7. The following has a logical error:
```
Readln(x);
While not (x= 0) do
Begin
    Readln(x);
    Sum := sum + x;
End;
```

Python Exercises 4.1 Repetition – For Loops

Problem

You want the machine to write out the multiplication table of a number you enter. The table ends at 20 times the number

Design

The usual strategy for designing loops is:

1. Determine what has to be repeatedly done. This gives the body of the loop. In this case we want to write out the result of counter * number 20 times.
2. Determine under what conditions it is to be repeated. This determines the type of loop (for-do, while, or repeat) and the condition for ending the loop. This may mean you have to add some instructions in the loop to increase a counter. If the statements in the loop are to be repeated a ***known*** number of times (such as 20, above) then we use a for-do loop. If the loop must be done ***at least once*** use a **repeat**-until loop and if the loop ***may not be done*** at all then use a **while**-do loop. This **While** loop does a test first and if the test fails skip the loop and goes to the next statement.

The general rules for choosing a form of repetition are

1. Use the for-do loop if you know before entering the loop how many repetitions you want.
2. The body of a repeat statement will always be repeated at least once before the test is carried out. This makes it inappropriate when you may want no repetitions to take place.
3. The while statement requires the test condition to already exist. If the test condition is only generated in the body of the loop, the repeat statement is the most appropriate.

The basic Python code looks like this:

Python	Comment

` #Program TimesTable;`
`#Begin Module`
 	
`print ('What number do you want a table for?');`
`number = int(input())`
`for counter in range(20):`
`    product = number * counter`
`    print (product)`
`#End Module`
	User input
	Counter starts at 0 and goes to 19
	So does 20 iterations

Enter the code and run it and see what it does.

Exercises

Run each of the following fragments of Python code and write down what happens and why (or why something does not happen)

Python	Comment / Output/ Why?
for i in range (10): print(i)	
for i in range (1,10): print(i)	
for i in range (10,1): print(i)	
for i in range (10,1,-1): print(i)	
for i in range(3,25,3): print(i)	
for i in range(3,25,-3): print(i)	
for i in range(25,3,-3): print(i)	
for i in range (10.0): print(i)	
for i in range (a,b): print(i)	
a = 1 b= 10 for i in range (a,b): print(i)	
a = 1 b= 10 for i in range (a,b,0): print(i)	

For the basic program above, make the following amendments.

Note the following:

1. Any calculations must be processed and stored in variable names, you must not do the processing in a print statement.
2. Use the Format Commands from Week 2 and 3 to properly format the output.
3. You will need to use the POW and SQRT functions.

To Do

1. Modify the output so the table has two columns, one for the counter, and a second one for the product.
2. Produce a four column table with columns for counter, counter times number, square of counter, and square root of counter. (Refer to the standard functions earlier in this course to find how to determine the square and square root of a number.)
3. Write headings over the four columns and allow the user to set the upper and lower limits of the counter. i.e. the user should be able to have a table that starts at 4* number and ends at 54 (or whatever)

Questions

1. What type must the for loop control variable be? 2 Can the for do loop count up as well as down?
2. Why is indenting so important?
3. Can the initial value and final values of the for loop be variables? If so what type must they be?

A new programmer has typed the following, but complains it only adds up one number or does almost nothing, correct the errors in each code fragment

(i) Sum = 0 For count in range(5): number = int(input('enter a value')) sum = sum + sum # endfor print(sum);	**(ii)** Sum := 0 For count in range(5): number = int(input('enter a value')) sum = sum + number #endfor print(sum);
(iii) Sum = 0 For count in range(5,1,1): number = int(input('enter a value')) sum = sum + number #endfor print(sum);	**(iv)** sum = 0 for count in range(1, 5): number = int(input('enter a value')) sum = sum + number #endfor print(sum)

To Do

1. Draw the correct pseudo code for your final program above that generates a times table.
2. The user enters a Final value for the loop that is greater than the starting value.. what can you do? Design a solution with a flow chart, before you try and code it.
3. Develop a Python Program that asks the user how many numbers that have to enter and then allows them to enter those numbs are maintain a running total, reporting the total so far and the final sum.

The dialog should go like this:

```
>>> ================================ RESTART ==========
>>>
This program will add up as many values as you like
How many values do you have?3
enter a value1
The total so far is,  1
enter a value2
The total so far is,  3
enter a value3
The total so far is,  6
6
>>>
```

4. Further modify the code so that it only adds up the positive numbers
5. Calculate the average of the positive numbers. Please use the format commands to control your output.

Software Dev 4.2 Python Exercises Repetition: While Problem

You want the computer keep on accepting numbers till the user enters a negative number. The computer should then give a total of all the positive numbers entered so far. Accepting the numbers is a repetitive process.

Consider the following Python code below:

Python	Comment
File Edit Format Run Options Windows Help	
`# Demo The Use of a while Loop`	
`sum = 0`	Set sum to 0.
`number = int(input('enter a value, type -999 to stop'))`	Get input.
`while number != -999:`	Test to see if it is the terminating condition.
`    sum = sum + number`	Add it up.
`    number = int(input('enter a value, type -999 to stop'`	Get the next one
`#Endwhile`	
`print("The final total is","{:10d}".format(sum));`	Report the sum.

Notes

Our program requires the machine to remember two things, the number 'number' and the number ‘sum’. These are the things that are going to change as a result of the program executing.

To set the sum to 0 and later add each number to the total, use the statement at the top of the statement syntax diagram. This is called the assignment statement. The = symbol is read as 'becomes equal to'. The assignment statement assigns the value to the right of the = symbol to the variable space given by the name on the left of the:= symbol. It is important to set variables to 0 so that the contents of the memory space are known before starting.

Lastly, as we want to do more than one thing after the while statement, we indent. This creates a compound statement. A compound statement is also called a block and is treated as if it was one statement.

This algorithm uses WHILE – DO loop – also known as a test first loop and has the ability to execute 0 times. If you do not control the loop properly, it can also execute an infinite number of times..
Try taking out the statement # number= int(input (number)) inside the body of the loop and see what happens.

Control C will help you at this point.

Note: 1. Blank lines are used to separate the main sections of the program.
2. How indentation shows how far the control of a statement extends.

Activities

1. Code the Program a second time, but allow FLOAT numbers to be entered.
2. Also keep a count of how many numbers were read in and write this out together with the total of the numbers entered.
3. Change the program so it writes out an mean (average) and not a total at the end.

Modify the program so that it adds up 10 positive numbers and does not stop until 10 numbers have been added up.

Hint you will a compound Boolean test at the start – while (number>0) and (counter <= 10):

1. Develop an algorithm that allows a user up to 3 guesses at their PIN (which should be a constant) and then report to them – if they guessed it correctly or if they have had their card confiscated.
4. Write your Python code out using Pseudo Code.
5. Trace your algorithms using a trace table.

Python Exercises Week Five

Structured Types: Arrays

Arrays, Parameters, Review, Data Structures, Structured Types

Structured types or Complex Types are those data types that allow you to create a structure that holds more than one piece of data. All of the **simple** types only hold one piece of data in their single memory space. This course considers **Arrays, records** and **Files** as the important structured types that we can use to create a **Data Structure** that most resembles the nature of the real world processing problem that we are trying to solve.

An **Array**, which we will consider this week, is a logically contiguous **indexed data type** that contains more than one piece of data **all of the same type**. It is a monoculture. It allows for Random access via the index, is held totally in RAM (so is volatile) and is limited in size.

It is declared as follows:

Const
    Max = 10;
Type
    Datatype = integer;
    Numbertype = **array** [1..max] of datatype;
VarNumbers : Numbertype;

Python	Comment

#Program DemoArray
import random
#Python calls arrays lists
Max = 5 # Set constant
#initialise
userdata = [0,0,0,0,0] # Sets a List of size 5
	The first item in the list in Python is at location [0], not [1]. Beware of this when using pseudo code.

The variable numbers is referred to as indexed variable as to access any of the ten memory spaces we have created, we have to use the index.
Eg Numbers [2] := 12;

Note

- The index must be an ordinal type.
- The array size = in this case *max* is set at run time and cannot be changed.
- Any attempt to read/ write assign to Numbers[11....... n + 1] will cause an error.
- Arrays are often processed using loops. Eg for
  count := 1 to max do
      Readln(numbers[count]);{ this also shows why const are so important}

Python List Processing

Python (Follows on from above)	Comment

for i in range(0,Max):
    userdata[i] = int(input("enter a value"))
#Echo the data
for i in range(0,Max):
    print(userdata[i])
 	
print(userdata)
#assign new data and find the maximum
scores = [34,56,78,98,2]
L = scores[0]
for count in range(1,5):
    if scores[count]> L:
        L = scores[count]
print(L)
	# Use the const the finish the loop
	# Loop start at 0 as the index starts at 0
	
	Assume that the first item in the array is the largest unless you find otherwise.

To DO

1. Recast the code above so that it uses Modules to get data from the user and to Echo the input.
2. Create functions to Discover the Largest and Smallest items in the lists (arrays).
3. Enhance your function so it also returns the position of the largest and smallest item in the list (respectively)

You may use the following code to fill the array with random numbers to save you a lot of typing. #Arrays

```
import random
userdata=[]
for i in range(0,7):
   #n = int(input("enter a value"))
   n = random.randint(1,10)     #Generate random value for n
   userdata.insert(i, n)    # Insert at Pos i, item n
```

Why are Data Structures so important?

- **A Data Structure** is a set of rules for storing and manipulating data. Your data structure is the combination of **const**, **types** and **variables** that you declare and should, as closely as possible, emulate the real world problem.
- It is important to think carefully about the data structure because:
- We don't often have to find unknown answers, we just need to implement a solution that employs simple math and i/o routines. A quality data structure + good algorithm makes the program more portable, robust and reusable and easy to modify
- A carefully thought out DS can minimise the programmer effort and make the program shorter and easier to debug/ maintain/ update.
- The entire character of the program can be changed by a small change to the DS. So type definitions are much more important than their written size suggests.
- An algorithm that compensates for a simple DS can become unwieldy and complex.

To Do

- How do we go about testing processing statements, modules/ Procedures and entire programs
- How we generate valid and useful test data?
- There is debate about whether or not to pass arrays as ***global*** (reference) parameters no matter what. Why is there the debate???
- Define data types for
  - an array of cricket scores for a team of cricketers;
  - a class (of up to 32 students) of raw numeric marks eg. 72.

An array to store the rainfall in mm for each month of the year.

Python Exercises SD5

Data Types - Arrays - Structured Types Problem

You want to store the data entered by the user for use later in the program. To do this we have introduce a Structured Type, known as an indexed variable - referred to in Python as a List.
Graphically / logically a list looks like this:

userdata

[0]	[1]	[2]	[3]

: INT

This sets up an array/list that can contain integers, were each individual item in the array is accessed via the index. i.e userdata[1] = 7 # Puts number 7 in the array at location / index [1]
In Python you codethe array / list by coding
userdata =[0,0,0,0]
The short Python code below will give you an introduction to arrays / lists in python.

<table>
<tr><th>Python</th><th>Comment</th></tr>
<tr><td>

ArrayDemo1.py - C:/Python27/ArrayDemo1.py (2.7.16)

File Edit Format Run Options Window Help

```
# Program DemoArray
# Python call lists
Max = 5 # Set constant
#initialise
user data = [0,0,0,0,0] # Sets of Lists of size 5
#get Data

for i in range(0,Max):
    userdata[i] = int(input("enter a value "))
#Echo the data
for i in range(0,Max):
    print(userdata[i])
```

</td><td></td></tr>
</table>

Arrays & Loops

Array processing is often done with loops - as you can see above. The controlling variable of the loop(which must be an ordinal type of course) is used as the index element for the array. For example

Exercises

1. Draw a diagram to represent the following data structures
(a) Var score : array [1..5] of INT
(b) var country : array [l..8] of string;
(c) var height : array [1..7] of FLOAT;
2. Consider the following array

Age[l]	7
Age[2]	8
Age[3]	3
Age[4]	6
Age[5]	9

Show the effect of the following program segment:-
count = 4;
index = 1;
print(age[count]); print(age[index]);
print(age[index + count]);

3. Write the type and variable declarations for the following data
   (a) 50 car types
   (b) 100 golf scores
   (c) 25 student grades
   (d) 30 student names
4. Assuming that the declarations have been made,

   max = 10; # A constant

   numbers = [1,2,3,4,5]
   counter : integer;
   What is the problem with the following code sequence?
   for counter in range(0,7]):
       (numbers[counter])= int(input("Enter a value"))
   (i) When will the problem be discovered?
   (ii) How could this problem be avoided?
   (iii) Why is the declaration of a constant(max) a useful one?

5. Design and write the code sequence to read in 5 names and scores in two separate arrays – 1 called
   names and 1 called score.
6. Define a data structure which could be used to accept the names and marks of 20 students. Design code with flowcharts to accept 20 names and marks and then write them back out again in table. Code your finished design at a computer.
   Now do Worksheet Array Output #1.

Python Exercises Week Six

Source Code, Byte Code, Exe code, Compilers, Interpreters, Structure Charts, Scope

Byte Code

Programming code that, once compiled, is run through a virtual machine instead of the computers processor. By using this approach, source code can be run on any platform once it has been compiled and run through the virtual machine.

Bytecode is the compiled format for Java programs. Once a Java program has been converted to bytecode, it can be transferred across a network and executed by Java Virtual Machine (JVM). Bytecode files generally have a .class extension.

Source Code

Program instructions in their original form. The word source differentiates code from various other forms that it can have (for example, object code and executable code).

Initially, a programmer writes a program in a particular programming language. This form of the program is called the source program, or more generically, source code. To execute the program, however, the programmer must translate it into machine language, the language that the computer understands. The first step of this translation process is usually performed by a utility called a compiler. The compiler translates the source code into a form called object code. Sometimes the object code is the same as machine code; sometimes it needs to be translated into machine language by a utility called an assembler.

Source code is the only format that is readable by humans. When you purchase programs, you usually receive them in their machine-language format. This means that you can execute them directly, but you cannot read or modify them. Some software manufacturers provide source code, but this is useful only if you are an experienced programmer.

Exe Code

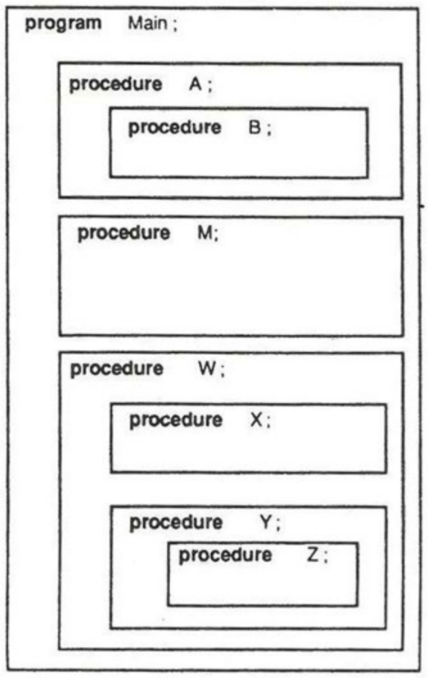
Code that has passed through a compiler or an assembler is called exe code. It is machine code that is directly executable by the processor and cannot be read by humans. Each processor requires a different compiler or assembler as, for each processor, its native machine code is different hence the need to compile the code for each environment. Pascal can be written with a text editor as source code and then compiled for a number of different processors as long as you have a compiler to do that job for you.

Scope

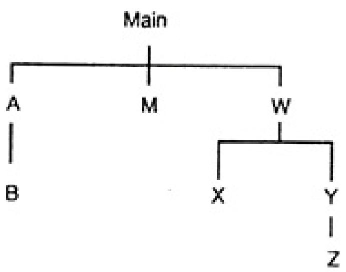
Scope is to do with the variables declared in procedures versus those in the mainline and where they can be accessed from – to be changed, updated, deleted or otherwise. The scope of a variable is defined as follows: " …to be the set of statements in the program in which the variable can be accessed. In Pascal the scope of any variable is the procedure in which it is declared together with all the procedures nested directly or indirectly within it, except those procedures in which the identifier has been reused." [Rohl & Barrett, p229]

The important principle for programmers is known as The Principle of Deepest Declaration – that means you should declare your variables as deeply as possible and only declare globally those vars/ const / types that are used by more than one module. This then avoids un intended side effects and variables being changed by accident by procedures that are not supposed to be accessing them.

Eg Always declare Loop control variables as local variables in modules. Consider the diagram below:



We consider that Procedure B is nested in within Procedure A and A is nested within the Main line. B is Indirectly nested within the Main Line. See the interpretation below:



This is just a more graphic representation of the same information; three procedures, *A*, *M* and *W* are *directly nested* within the main program; *B* is directly nested within *A*; and so on. All procedures are nested within the main program. Those not directly nested (*B*, *X*, *Y* and *Z*) are *indirectly nested*.

Study the Layout and the chart above and try to complete the following table – assume that there is no such thing as recursion.

Statements within :	May access variables declared in
Main	Main
A	
B	
M	
W	
X	
Y	Y, W, Main
Z	

Variables accessed by :	Should be declared in :
B	
A, A & B	
M	
X	
Z	
Y Y & A	
W, W & X, W & Y , W & Z, W & Y & Z	
Main, Main & A etc	Main

Questions

1. What is byte code?
2. Why use byte code instead of EXE code?
3. Which operator environment gives you access to the source code as part of the "rules" of using that environment?
4. Define " Scope"

Homework

1. How do software bugs arise?
2. Identify and discuss two type of user interfaces provided by an OS.
3. **Define** the term Utility Software and give an example
4. What is an Application Service Provider and what issues may arrive in using one?

Sample Exam Question/s

When implementing new Software using the SDC, there are steps you need to cover. Four stages/steps are listed below, describe each term in detail. (8 marks)

- Analyse detailed requirements (2 marks):
- Design data and algorithms (2 marks):
  Implement and test with live data (2 marks):
- Evaluate performance of the program (2 marks):

Program 6 Structured Types: Records

Strings, Records, String Functions

Unlike an **Array**, a **Record**, can hold multiple types of data in the fields defined by the programmer. For example

```
Type
        studentRectype = record
        name : String [ 25]; { A 25 char special case of an array}
        age : 11..19; { Subrange type}
        Gender : ( M, F );
        Mentor : real;
        Paid : Boolean;
    End; { of the type definition}
Var
    Student : StudentRectype; {One memory space for one record of this type}
```

This sets up **ONE** memory space that can store data about **one** student. The individual elements of the record type declaration are called fields and are ***accessed*** as follows:

(i) **Dot Notation**. The syntax is Record Variable Name DOT field name – eg

Student. Age := 12;

This notation can then be used like any other variable, you can compare fields, assign data, read data into the variable, write data out of the variable, and erase or initialise the **variable.fieldname** as you would any other variable.

(ii) **With Statement: with** rec Var Name **do {This is NOT a loop}**

```
            Begin
                {access fields}
            End;
```

Eg with student do

```
    Begin
        Name : = "george"; Age : = 14;
        Writeln('What is their mentor group?');
        Readln( Mentor );
    End;
```

Homework

Error Checking

Test Data

Tracing Activities

Discuss what is meant by the term Generations of Programming Languages and give examples of a "language" at each generation from 1 GL to 4th GL such as SQL.

Sample Exam Question

Compare and contrast Arrays and Records in terms of data stored, potential types and sizes. Explain the differences between 3rd and 4th Generation Programming Languages.

Python Exercises Week Seven

Records

Unlike an Array, a Record, can hold multiple types of data in the fields defined by the programmer.
For example

```
Type
        studentRectype = record
                name : String [25]; { A 25 char special case of an array}
                age : 11..19; { Subrange type}
                Gender : (M, F);
                Mentor : real;
                Paid : Boolean;
        End; {of the type definition}
Var
        Student : StudentRectype; { One memory space for one record of this type}
```

This sets up ONE memory space that can store data about one student. The individual elements of the record type declaration are called fields and are accessed as follows:

(i) Dot Notation . The syntax is Record Variable Name DOT field name – eg

```
Student. Age := 12;
```

This notation can then be used like any other variable, you can compare fields, assign data, read data into the variable, write data out of the variable, and erase or initialise the variable.fieldname as you would any other variable.

(ii) With Statement: with rec Var Name do {This is NOT a loop}

```
        begin
                {access fields}
        End;
```

```
Eg with student do
        Begin
                Name : = " george"; Age : = 14;
                Writeln(' What is their mentor group?');
                Readln(Mentor);
        End;
```

Homework

Discuss what is meant by the term Generations of Programming Languages and give examples of a "language" at each generation from 1 GL to 4^{th} GL such as SQL.

Sample Exam Question

Compare and contrast Arrays and Records in terms of data stored, potential types and sizes. Explain the differences between 3^{rd} and 4^{th} Generation Programming Languages.

Python

In Python we have to introduce the concept of a CLASS- classes are a little different from boring old records as classes are Objects that can also contain the METHODS (or things that we can do to the object) For our syllabus it will suffice just to declare a simple class without any methods.

```python
class Student(object):
    """A student  of XYZ School. Students have the
    following properties:

    Attributes: or Fields
        name: A string representing the students's name.
        age - the students age.
        Gender
        Paid . Paid their fees or not
        Mentor. Form class
    """
# Set up a student definition

    def __init__(self, name, age, gender):
        """Return a Student object whose name, age and gender fields
                are mandatory, and  we are assigning them to the fields of a
                a new record. SELF allows you set up the instance of the object"""
        if gender not in ('M','F'):
            raise Exception("Invalid Gender")
        self.name = name
        self.age = age
        self.gender = gender
        self.paid = False
        self.mentor = None
    #end of  student definition
#Now create the object

Jeremy = Student("Jeremy",14, 'M')# Creates an instance called
                                    #Jeremy of the Student object.
#We can use the dot notation to access data fields of Jeremy
Jeremy.paid = True
Jeremy.mentor = 11.15

#or output the data
print(Jeremy.name, Jeremy.age,Jeremy.mentor, Jeremy.gender)
Stud171 = Student(" ", 0, "M")
Stud171.name = input("please enter a name")
Stud171.age = int(input("please enter an age"))
Stud171.gender = input("please enter the gender")
print(Stud171.name, Stud171.age, Stud171.gender)
```

Python file length : 1471 lines : 40 Ln : 40 Col : 49 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

Python Exercises Week 8 Documentation

Internal Documentation

Computer software is said to have Internal Documentation if the notes on how and why various parts of code operate is included within the source code as comments.

The following is considered to be part of ID:

- Comments in the code to explain steps in the algorithm
- Sensible & Meaningful identifiers for constants, variables and module names that comply with Corporate Standards for Identifiers
- code layout with appropriate use of indentation and white space.

There are even International Standards for Internal Documentation such as IEEE 1063-2001 "Standard for Software User Documentation" written by IEEE, ISO/IEC 18019-2004 and ISO/IEC TR 9294.

External Documentation

External documentation is written in a place where people who need to use the software can read about how to use the software. External documentation can be broken down into library documentation, which describes tools that a programmer can use, and user documentation, which is intended for users of an application. {Note that User Documentation – User Manuals are now all on line and so far harder to use! IMHO}

The line between internal and library documentation is not clear cut because the trend is to write library documentation inside a program as comments, relying on software that extracts the documentation and puts it into a form suitable for people who only want to use the library. For example, the extensive Java library documentation is created by software called javadoc that reads Java programs, including comments, and writes documentation.

Why Have Documentation?

Can we fix it? Well yes we can, but only if it is documented in the first place…. Both the software developer and the maintainer (the person who has to fix it or edit later) has to be able to gain a relatively fast understanding of the software, its design philosophy, assumptions and goals. To do this requires quality documentation.

Importance of documentation for the developer

Each function in a piece of software solves a specific problem. Before you try to solve any problem, you should have a good understanding of exactly what the problem is. It makes no sense just to start writing and then, afterwards, look at what you have come up with to see whether it solves any useful problem!

Inexperienced computer programmers imagine that they can keep all problem descriptions in their heads. Experience has shown that they can't. Three issues come up.

1. When writing a function definition without written documentation, you only have a rough idea of what it is supposed to do. While you write, the idea morphs in your head. A simple interruption can cause the idea to lose what focus it has. You start thinking about the program as a whole instead of thinking of just the function that you are working on, and the function starts to take on responsibilities that it should have nothing to do with.
2. Suppose that you test the function and find that it does not work. So you need to fix it. But during the process of fixing it, you have nothing but your memory telling you want the function is supposed to do. It is difficult to keep that in your head along with the details of how the function is supposed to work, and the process of fixing a function definition takes the function further away from its original intent.
3. Later, when you need to use that function, you have forgotten just what it does. Unwilling to reverse-engineer it, you make a guess based on what you remember. You often forget important details, and your software does not work because of it. You are faced with laborious debugging to find out what is going on.

Importance of documentation for the maintainer

You might have heard of "self-documenting code". The idea is for functions to be written in a readable form so that, to find out what a function does, you just read the function's definition. For very small pieces of software that can be achieved. But imagine a larger piece of software, say with about 1000 functions. Such software is built up function upon function; one function typically uses a few others that are defined in the same collection of 1000 functions, with the exception of the bottom-level functions that only use the library.

Suppose that the software has no internal documentation, and relies on "self-documenting code". Now you want to understand what a particular function does. But it uses 3 other undocumented functions, so you need to understand what they do first. Each of those uses 2 undocumented functions, so you must read their definitions too. It goes on and on. You find yourself reading thousands of lines of code to understand a single function whose body is only ten lines long.

The only way that anyone can work with undocumented software is to reverse-engineer the whole thing and add documentation that should have been written by the developer. Most of the time, that is too difficult. Undocumented software is often just thrown away as unmaintainable.

http://www.cs.ecu.edu/karl/3300/spr14/Notes/Documentation/documentation.html

To Do

1. Copy a fragment of your Python code from any longer program that you have written and explain which aspects of it are examples of good ID. Clearly label the parts that are good examples. Explain why some it needs improvement and is not compliant with acceptable standards for ID.
2. List the components of good ID. (See the Appendix C for help)
3. External Documentation is meant for whom and whom does it assist and why?

Documentation: Structure Charts

Once the flow of data and control in the system is decided using tools like DFDs and CFDs, the system is given shape through programming. Prior to this, the basic infrastructure of the program layout is prepared based on the concepts of modular programming.

In **modular** programming, the complete system is coded as small **independent** interacting modules. Each module is aimed at doing one specific task. The design for these modules is prepared in the form of structure charts.

A structure chart is a design tool that pictorially shows the relation between **processing** modules in computer software. Describes the hierarchy of components modules and the data are transmitted between them. Includes analysis of input-to-output transformations and analysis of transaction.

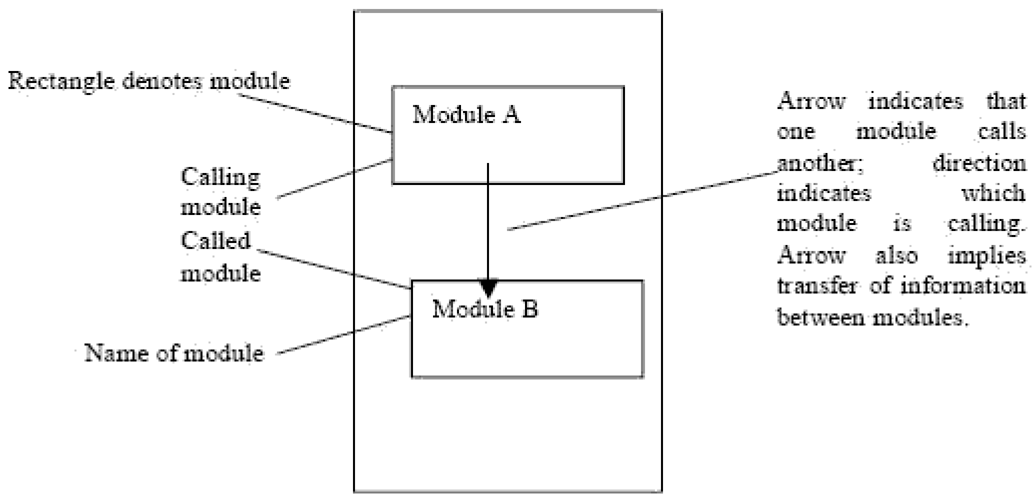
Structure charts show the relation of processing modules in computer software. It is a design tool that visually displays the relationships between program modules. It shows which module within a system interacts and graphically depicts the data that are communicated between various modules.

Structure charts are developed prior to the writing of program code. They identify the data passes existing between individual modules that interact with one another.

They are not intended to express procedural logic. This task is left to flowcharts and pseudocode. They don't describe the actual physical interface between processing functions. Notation

Program modules are identified by rectangles with the module name written inside the rectangle.

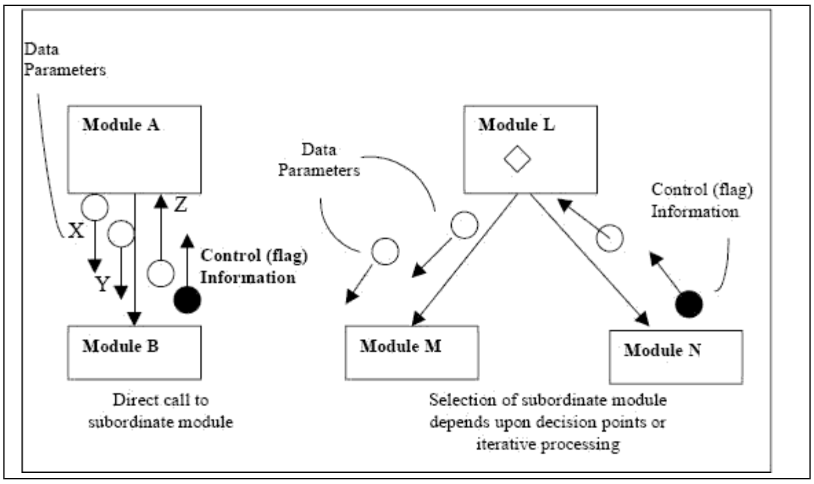
Arrows indicate calls, which are any mechanism used to invoke a particular module.



Notation used in structure charts.

Annotations on the structure chart indicate the parameter that are passed and the direction of the data movement. In fig. 6.3, we see that modules A and B interact. Data identified as X and Y are passed to module B, which in turn passes back Z.

Fig 6.3 - Annotations and data passing in structure charts



A calling module can interact with more than one subordinate module. Fig. 6.3 also shows module L calling subordinate modules M and N. M is called on the basis of a decision point in L (indicated by the diamond notation), while N is called on the basis of the iterative processing loop (noted by the arc at the start of the calling arrow.

Data passing

When one module calls another, the calling module can send data to the called module so that it can perform the function described in its name. The called module can produce data that are passed back to the calling module.

Two types of data are transmitted. The first, parameter data, are items of data needed in the called module to perform the necessary work. A small arrow with an open circle at the end is used to note the passing of data parameters. In addition, control information (flag data) is also passed. Its purpose is to assist in the control of processing by indicating the occurrence of, say, errors or end-of-conditions. A small arrow with a closed circle indicates the control information. A brief annotation describes the type of information passed.

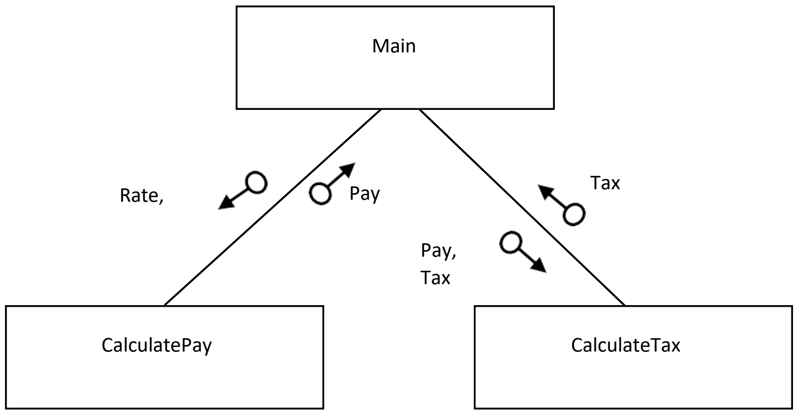
Structure chart is a tool to assist the analyst in developing software that meets the objectives of good software design.

http://www.freetutes.com/systemanalysis/sa6-structure-charts.html

To Do

Create a structure chart for the Payroll program from earlier this Term.

Sample Structure Chart from SCSA:



Structure charts represent modules graphically. The data parameters passed between modules are included.

Rate and Hours are **value parameters** that send a value to the module, but do not return any value.

Two-way parameters **pay** and **tax** would be variable parameters. This indicates that any changes made to the values are passed back to the calling module.

The structure chart is showing the **actual** parameters from Module **Main**, not the formal parameters shown in Module CalculatePay and Module CalculateTax. See MAIN below

Module Main

Input (Rate) Input (Hours)

Call CalculatePay (Rate, Hours, Pay)# Calculate the gross pay

Call CalculateTax (Pay, Tax) #Calculates Tax

NettPay ← Pay - Tax

Output (NettPay)

End Main

Programming Documentation Requirements

When you submit this program it should also have two pieces of documentation accompanying it. These will be the Program Description Report and the User Manual.

Program Description Report

The Program Description Report should include a verbal description of what the program does and more importantly how it does it. It is chiefly used by someone who may wish to extend of modify the program or who needs to know the code should a fault be found. You should assume that the person has a reasonable knowledge of the programming language involved but knows nothing at all about what you have written.

The program description report should include the following sections:

a) **Problem Description:** Introduce the problem and state what the program does towards solving it.
b) **Program Outline:** This section should describe the broad structure of the program. It should help show how the separate sections fit together, as well as giving the overall design philosophies of the program. To do this it may well be necessary to briefly describe the input and output formats or the data structures used.
c) **Description of the Algorithms:** This should discuss the components in (b) above. For example, how invalid data is trapped and explain in detail, using diagrams where appropriate, what each section does, how it does it, and why it has been implemented in a given manner.
d) **Description of the Data Structures:** That is describe how the required data is stored, not a description of the input formats and not a list of all the variables used in the program. You should explain why a particular data structure was chosen and what its bounds are, if this is relevant.
e) **Diagrams:** You should include one design Pseudo Code form.
f) **Program Listing:** Complete with sample output, also if a program was not complete a list of alterations that you would make if time permitted.

User Manual

This should provide the information required by someone wishing to use the program, the user is not a programmer but knows how to use computers. The manual should include the following sections;

a) **Introduction:** This describes what the program does.
b) **Program Access:** How to run the program- a brief description of the system is required.
c) **Data Format:** Saying how the data is to be prepared, what format and any bounds that apply.
d) **Output:** A description of the output, how it is formatted and how it is interpreted.
e) **Error Recovery:** A description of how the program handles invalid data and an explanation of what to do in each case. List error messages and explain their meaning.
f) **Required Data Format:** If your program assumes correct data then the documentation should say so.

Clearly this is not a trivial exercise. Discussion with you colleagues is encouraged but the project must be essentially your own work. The projects will be compared and any two found to be substantially the same will SHARE the lowest mark of the two. (Note “Every programmer has a ‘signature’.”) It is your responsibility to keep sufficient back-up copies of your work in order to guard against electronic disaster. Lost files/corrupted disks will not be accepted as excuses for lateness.

I. **"External" Documentation (or Program Information):** In programming courses, the comprehensive set of documents that detail the design, development, and structure of a program are usually condensed into a comparatively brief 'block comment' at the top of the source code. This "external" documentation will minimally include:

a. Author(s) name, the course name/number, assignment name/number, instructor's name, and due date.
b. Detailed description of the problem the program was written to solve, including the algorithm used to solve the problem.
c. The program's operational requirements, such as the programming language, special compilation information, and the input information.
d. Required features of the assignment that author(s) were not able to complete, and/or information about the existing bugs.

II. **Documentation about the "Classes":** When writing the code for a class in an object–oriented programming language, it should be preceded by a block comment minimally containing the following:

a. The class name, (author(s) name in team projects,) the names of any external packages upon which the class depends, the name of the package for the classes containing this class (if any), and the inheritance information.
b. An explanation of the purpose of the class.
c. Brief descriptions of the class and instance constants and variables.
d. Brief descriptions of constructors as well as the implemented class and instance methods.

III. **"Internal" Documentation (or in-program documentation):** The details of the program are explained by comments and placed within the code. The internal documentation should minimally include the following:

a. A 'block comment' which should be placed at the head of every method (also known as the function or subprogram). This will include the method name; the purpose of the method; the method's pre– and post–conditions; the method's return value (if any); and a list of all parameters, including direction of information transfer (into this method, out from the method back to the calling method, or both), and their purposes.
b. Meaningful identifier names. Traditionally, simple loop variables may have single letter variable names, but all others should be meaningful. Never use nonstandard abbreviations. If the programming language has a naming convention for variables, methods, classes, etc., then those conventions should be used.
c. Each variable and constant must have a brief comment immediately after its declaration that explains its purpose. This applies to all variables, as well as to fields of structure declarations.
d. Complex sections of the program that need some more explanations should have comments just before or embedded in those program sections.

IV. **Miscellaneous / Optional Requirements:**

a. Write programs with appropriate modularity; that is, create classes when appropriate, write methods that accomplish limited, well-defined tasks, etc.
b. Global/public variables should be avoided in programs, unless it is required.
c. Use "white spaces" (blank lines) to set apart logically related sections of code.
d. Indent bodies of methods, loops, and "if" statements, and do so with a single, consistent style.
e. Unconditional branching (such as the "goto" statement) should be avoided in programs unless it is required for that specific language (such as the assembly language).

Notes. There is a number of standards and tools for program documentation, such as IEEE 1063-2001 "Standard for Software User Documentation" written by IEEE, ISO/IEC 18019-2004 and ISO/IEC TR 9294 written by the International Standards Organization (ISO) and the International Electrotechnical Commission (IEC).

Tools such as Doxygen, javadoc, ROBODoc, and TwinText can be used to auto-generate the code documents. Hence, these tools add more capabilities for document preparation. For example, they are able to extract the comments from the source code and create reference manuals in such forms as text or HTML files.

References

1. O. McCann. "Toward Developing Good Programming Style". http://www.cs.arizona.edu/people/mccann/style.html, [accessed Jan 17, 2012]
2. P. DePasquale. http://www.comtor.org/ [accessed Jan 17, 2011]
3. O. Paull, "The Importance of Software Documentation", http://www.ehow.co.uk/about_6706857_importance-software-documentation.html [accessed Jan 17, 2012]
4. Dimitri van Heesch: "Doxygen Documentation. Generate documentation from source code", 2012, http://www.stack.nl/~dimitri/doxygen/ [accessed Jan 17, 2012]

Python Exercises Week 9 Test Data

Testing and Debugging

Alas even the best of programs and algorithms will rarely run first time, they usually (inevitably?) have some bugs in- indeed there is a case for arguing that there is at always one bug left in the program.

The Aim of testing therefore is the execution of a program with the intent of finding errors.

Testing

- can only demonstrate the presence of bugs not their absence.
- must be as comprehensive as possible.
- must be structured.

1. Testing Assignment Statements
    - do not use just 1 or O
    - test the statement at its limits.
    - write out the contents of memory immediately the statement has been executed.
2. Conditional Statements
    - test values either side of the criterion.
    - use values that produce known results.
3. Must Test
    - that the algorithm does not treat incorrect data as though it was correct.
    - handles data correctly
      i.e. to print A, B ,C. in ascending order there are 20 tests, 9 decision points, and 29 combinations of logic paths. However, if we were to proceduralise this action we would reduce this to only 24 tests in total, plus one or two to check out communication between procedures.
4. Testing Modules
    - put in statements to indicate that modules are being executed.
      i.e. "Procedure X now running"
      Strategy
        - each set of test data must have expected results already calculated.
        - do not test too many things at once.
        - inspect results thoroughly, do not assume if something is correct once that it will remain so.
        - use test data designed by someone other than the author.
        - in initial development test one feature at a time.
        - make sure that procedures do not interfere with each other.
        - use unusual but valid data to test exception handling.
        - test incorrect data to test error handling.

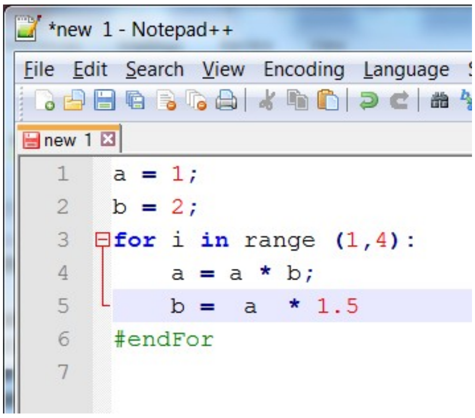
Common Errors

1. Variables not assigned or initialized.
2. Local variables corrupted by other procedures.
3. Flags not being cleared /reset after their use.
4. Arrays over bounds.
5. Loops not terminating.
6. Loops not handling the lst, middle or last case.
7. Contradictory logic.
   i.e. if sex not = 'M' or not = 'F' then output error. will reject every value of sex. It should have been expressed as - if sex not = 'M' and not ='F' then write error

   When you have completed this process there is a reasonable chance that you will have eliminated nearly or all of the bugs but the chances are that some piece of data or combination of data will crash your program. Do not let this depress you, it is almost a fact of life for the programmer.

Trace Tables

These tables are an important part of your arsenal of weapons for testing your program. They allow you to calculate on paper what your program should produce. You can then test your program with known data. They are also useful highlighting mistakes For example, we have some Python code as follows:-



Line	a	b	i	
1	1			
2		2		
3			1	
4	2			
5		3		
3			2	

And so on

You can then compare known output to the actual and begin to trace the error. Other tools include:

- Putting Trace statements in the code the print values of variables at critical points this is called debug printing;
- Tracing the execution of the program by putting in statements such as " I am now entering the while loop and the value of x was true."
- If the programming language offers the facility, you can also use such things as the Observe window in Python

Note

Even simple programs, with statements in sequence require extensive testing to try and discover logical errors – such as our Simple payroll program that contained a logic error when people worked less than standard hours (37.5). The programmer had assumed that the all staff work 37.5 hours or more when this was not the case.

As soon as you introduce if then elseif statements you have to create test data to test all of the pathways.

To Do

So, given that we should know what syntax, run-time and logic errors are:

1. Define and give an example of each type of error both from the web and using your own python code.
2. Generate a list of test data for the following code:

GCD_test.py - C:/Users/E0401679/Desktop/GCD

File Edit Format Run Options Window

```
#gcd Calculator
p = int(input("enter a value"))
q = int(input("enter a value"))
r = p % q
while r !=0:
    p = q
    q = r
    r = p % q
#end while
GCD = q
print (GCD)
```

Test Data	What is tested	Why?
p = 144 q =96	GCD calculation	GCD = 48. Correct Logic
Q = 144, p =96	GCD	Make sure that Order is unimportant
P, q = 40	Does r = 0?	Skips the while loop(test for 0 iterations)

3. Generate test data for the following code.

```python
#Program Is Leap

year  = int(input(" enter a year"))

Leap = ((year % 4 == 0) and( year % 100 != 0 )) or(year % 400 ==0)
print(Leap)

#end
```

Test Data	What is tested	Why?

4. Generate test data for the following code

Python 3.4.1: Perfect.py - H:\10DB&Programming2015\ProgrammingTaskSolutions\Perfect.py

File Edit Format Run Options Windows Help

```python
#Program Perfect Numbers
#Test if n entered by the uiser is Perfect

#n= int(input("Enter a value"))


def IsPerfectN(n):
    sum = 0
    for i in range(1,n): # bad algorithm does not stop soon enough
        if n % i == 0:
            sum = sum + i
    Perfect = sum == n
    return Perfect
#end Function
count = 0
n= 2
while count <=3:
    if IsPerfectN(n) == True:
        count = count + 1
        print(n)
    n = n + 1
#end of trying to find ten Perfect Numbers
```

Data Validation

Syllabus Says:

- **use** data validation techniques, including range checking and type checking
- **design** and implement data validation techniques

Before your carefully designed and developed algorithm does any processing, it is up to you to ensure that the data is at least valid- this means it must be of the correct type for the data structure and the algorithm to handle.

The data validation process often entails one or all of the following:

1. **Type** Validation – is that data INT or STR or FLOAT and is that what the algorithm and data structure expect?
2. **Range** Validation – is the data in an acceptable range for the program – ie is x> 20 and x <40 or is ch between 'a' and 'z'?
3. **Limit** Validation – Does the data exceed a certain limit. For examples some ATM's will not allow you to exceed a fixed amount for withdrawal and with some accounts you cannot exceed a withdrawal limit. Ie WD <= 400

The programmer must make sure that the program does not crash (run time error) just because the user has misunderstood the instructions and entered data in an incorrect format or is out of range/ over the limit.

Range Checking

Given that the data is the correct type, range checking is simple and quite often involves a repeat loop:

Pseudo Code	Comment
Repeat Output("please enter a value between 100..400") Input(number) If not (number in[100..400] then Output('error') **Until** number in [100..400]	Repeat commonly used to check range of data as it must be entered at least once or, in this case, n times until the user gets it right.

An important part of the process is providing feedback so that the user understands the nature of the mistake and can correct it.

It is also important to use BOOL variables so you can easily tell if the data was valid or not. This makes for more easily readable code that is easy to debug later.

Data Validation Python Exercises

1. Study the code that follows and decide what it is trying to validate

Python 3.4.1: CheckDigitbyJezza.py - H:\ComputerScience\ATARUnit2CSC_2015\WeeklyWorkFiles\CheckDigitbyJezza.py

File Edit Format Run Options Windows Help

```
def get_data():
    string = input("Enter your 13 digits on one line sep by spaces: ")
    print("")

    # split the input on each space
    strings = string.split(' ')

    # ensure we have 13 digits
    if len(strings) != 13:
        print("invalid number of digits!")
        return None

    data = []
    # check that each element is a single digit
    for item in strings:
        if not item.isdigit():
            print("{:s} isn't a digit!".format(item))
            return None

        if len(item) != 1:
            print("{:s} isn't a single digit!".format(item))
            return None

        data.append(ord(item) - ord('0'))

    return data

print(get_data())
```

2. Look at the next block of code and answer the questions (that are comment) in the code

Python 3.4.1: CheckDigitWoutIsDigit.py - H:\ComputerScience\ATARUnit2CSC_2015\WeeklyWorkFiles\CheckDigitWoutIsDigit.py

File Edit Format Run Options Windows Help

```
def get_data():
    string = input("Enter your 13 digits on one line sep by spaces: ")
    print("")

    # split the input on each space
    strings = string.split(' ')

    # What is being tested here?
    if len(strings) != 13:
        print("invalid number of digits!")
        return None

    data = []
    # What is being checked next?
    for item in strings:
        if len(item) != 1:
            print("{:s} isn't a single character!".format(item))
            return None

        if ord(item) < ord('0') or ord(item) > ord('9'):
            print("{:s} isn't a digit!".format(item))
            return None

        data.append(ord(item) - ord('0'))# What is this doing?

    return data # Returns the digits in the list DATA if they are valid

print(get_data())
```

3. Add this code to your Checkdigit Algorithm so that it does the error checking that your CD Algorithm does not.
4. Specify if this range, type or limit validation?
5. Design an Algorithm that reads in a number as a string of digits. ie ‘1’ ‘2’ ‘3’ and then converts it to the equivalent INT.

This bit of PYTHON will get you started:

```
# read a line of input

string = input("{}: ".format("Enter your digits on one line separated by spaces"))

# split the input on each space data = string.split(' ') print(data)

Length = len(data) # Calculates the length of the digits
```

The effect is that you will have list, called data, that holds (for example) ['1', '2', '3'] with a Length of 3.

In English the Algorithm is as follows:

Set a variable called Multiplier to 1

Set Number equal to 0 # to hold the eventual INT

While Length is still bigger than or equal to 1

    Convert the LAST item in the list to a Digit

    Add the new Digit to Number after Multiplying it by the Multiplier Multiply the Multiplier by 10

    Decrease Length by 1

Output the Final Number # In this case it should be 123 as an INT

6. Build your data validation algorithm (from above) into your new program so that it rejects any attempt to type a non numeric and anything but a single digit character.
7. Write Pseduo code to check if an INT is between 6000 and 6999

Data Validation: Python Exercises

1. Name the 3 types of data validation.
2. Study the code that follows and decide what it is trying to validate.
   Also sate the type of the function in each case and List the Formal and Actual Parameter.

<table>
<tr><th>Code</th><th>Validation?</th><th>Formal Parameter</th><th>Actual Parameter</th></tr>
<tr><td>*Python 3.4.1: DataValidationRoutines.py - H:/ComputerScience,
File Edit Format Run Options Windows Help
Data Validtion Routines

def Testch(ch):
    Valid = ch in ['a','z']
    return Valid

#Main
print(Testch('a'))</td><td></td><td></td><td></td></tr>
<tr><td>def TestAnother(n):
    Valid = False
    if n >=1 and n <= 100:
        Valid = True
    return Valid

print(TestAnother(75))</td><td></td><td></td><td></td></tr>
<tr><td>def AndAnother(n):
    Valid = n <500
    return Valid
print(AndAnother(500))</td><td></td><td></td><td></td></tr>
</table>

3. Write PSEUDO CODE to accept a PIN, compare it to a constant known as STORED PIN and validate that the PIN is correct. The user is allowed 3 attempts at the PIN.
4. Write a function DaysInMonth to take Year and Month as Input(value parameters) and return the correct DAYS in the month for that year and month. It should call a function LEAP that returns true if the given year is LEAP.

Part B

I know that many of you have not attempted this, so:

1. Design an Algorithm that reads in a number as a string of digits. Ie ‘1’ ‘2’ ‘3’ and then converts it to the equivalent INT.

This bit of PYTHON will get you started:

```
# read a line of input

string = input("{}: ".format("Enter your digits on one line separated by spaces"))


# split the input on each space

data = string.split(' ')

print(data)

Length = len(data) # Calculates the length of the digits
```

The effect is that you will have list, called data, that holds (for example) ['1', '2', '3'] with a Length of 3.

In English the Algorithm is as follows:

Set a variable called Multiplier to 1
Set Number equal to 0 # to hold the eventual INT
While Length is still bigger than or equal to 1
    Convert the LAST item in the list to a Digit
    Add the new Digit to Number after Multiplying it by the Multiplier
    #Multiply the Multiplier by 10 * ? Logic error?
    Decrease Length by 1
Output the Final Number # In this case it should be 123 as an INT

2. Build your data validation algorithm (from above) into your new program so that it rejects any attempt to type a non numeric and anything but a single digit character.
3. Generate test data for your final algorithm
4. Write Pseudo code to check if an INT is between 6000 and 6999

Python Exercises Week 10 Strings

A **string** is usually a bit of text you want to display to someone, or "export" out of the program you are writing. **Python** knows you want something to be a **string** when you put either " (double-quotes) or ' (single-quotes) around the text.

learn**python**thehardway.org/book/ex6.html

A string is actually a special case of an array (or list) or characters. It is stored in a similar way to a list and can be processed in the same way. All programming languages, including Python, have a list as long as your arm of standard functions with which to process strings.

Conceptually a string looks like this:

WORD	I	A	M	A	W	O	R	D	
Index	0	1	2	3	4	5	6	7	

Note that in Python String- in this case word[0] is used for a character. In some languages location [0] is reserved as the length of the string.

String Processing

There are literally hundreds of methods available in Python to manipulate strings, for example:

Python	Comment

Python 3.4.1: StringProcessing.py - H:/ComputerScience/ATAR_Units3&4_2016/Unit4Programming&Comms/StringProc
File Edit Format Run Options Windows Help
`word = "hello i am your special friend"`
 	
`word = word.title()`
`print(word)`
	Given any string variable as input, the .title method converts the whole sentence to a Title Case:
	`>>>`
	`Hello I Am Your Special Friend`
	`>>>`

To Do

It is nice to build your own functions to emulate what is built in as it helps your understanding of strings and Lists. So

1. Write a function to take 1 char as input and convert it to its uppercase Equivalent. If it is already uppercase, no conversion will be required.
2. Write a function take a word as input and sort it into ascending order of letters.
3. Read In two words and test to see if they are Anagrams.

Work Through the STRINGS exercises at :
http://interactivepython.org/runestone/static/ShentonCSC/Strings/StringsRevisited.html

Software Licenses

Network (Per Seat)

A Per-seat license is a software license model based on the number of individual users who have access to a digital service or product. For example, 50-user per-seat license would mean that up to 50 individually named users can access the program.

Imagine a classroom with 100 chairs, where the classroom represents the software and the chairs literally are the Per Seat users. The classroom, or software, can only hold 100 students, or users. If the 101st students comes along, they cannot register for that class because all the seats are taken. Until one of those chairs becomes available, no other student can register for, let alone attend that class.

Enterprise License.

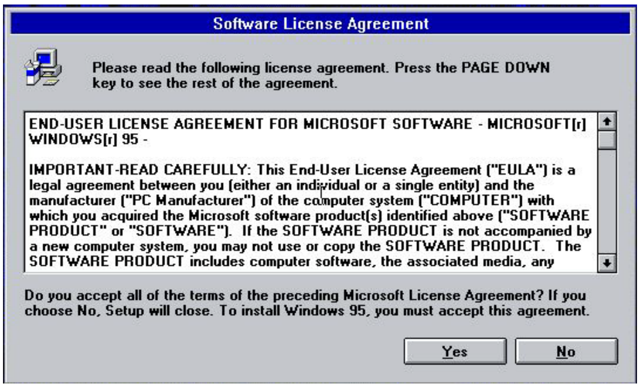
A software site license that is issued to a large company. It typically allows unlimited use of the program throughout the organization, although there may be restrictions and limitations. It usually foregoes the need to register the software each time it is installed on another computer; however, there might be a master password that is required to activate each copy.

Proprietary Software

Proprietary software consists of software that is licensed by the copyright holder under very specific conditions. In general, you can use the software, but you are not allowed to modify the software or distribute it to others.

The original source code for the software is not available, which means you can't see the actual code written by the programmers. Proprietary software is, therefore, also referred to as closed-source software. This is done on purpose to protect the intellectual property invested in software development. If the source code were released, even with copyright restrictions, competitors could benefit from using this code.

End-user license agreement



An End User License Agreement (EULA) is a legal contract between a software application author or publisher and the user of that application. The EULA, often referred to as the "software license," is similar to a rental agreement; the user agrees to pay for the privilege of using the software, and promises the software author or publisher to comply with all restrictions stated in the EULA. The user is asked to indicate they that "accept" the terms of the EULA by opening the shrink wrap on the application package, breaking the seal on the CD case, sending a card back to the software publisher, installing the application, executing a downloadable file, or by simply using the application. The user can refuse to enter into the agreement by returning the software product for a refund or clicking "I do not accept" when prompted to accept the EULA during an install.

Factors Affecting Software Development

Vocabulary

- user interface (UI) = is the way humans interact or engage with a computing device, handheld, laptop or desktop
- device interface = every computing device has an interface that people use in order to 'work' the device
- interaction with the device could be clicking a mouse, sliding a finger across a screen, talking to or listening to the device
- usability = how easy it is to use
- inclusivity = a sense of belonging, can participate. Include community, culture as well as
- accessibility = "To allow people with and without disabilities to benefit from the same services" source

User Needs

A good set of user requirements are needed for any project, especially computer system projects, to be successful. To deliver the right product, you need to articulate your requirements early on, if you don't obtain user requirements through efficient means, you won't deliver the right product.

The problem is that requirements are not just sitting around waiting to be written down. It takes a lot of effort and skill to elicit, analyze and verify them.

- The software must do what the user expects the software to do.
- If you make software for an ATM Kiosk, then the user is expecting to be able get money out of the kiosk.
- The user needs must be considered in the planning stages and the user must be given an opportunity to look at the design and to provide feedback on how good it will work for them.

User Interface (UI)

- UI includes input controls; buttons, text input boxes, radio buttons, check boxes, drop down lists
- UI includes navigation controls; breadcrumbs, sliders, search fields
- UI includes information components; tool tips, progress bar, message box

Graphical user interface (GUI) suitable for target audience

- if a target audience is 5 years old, large pics, not much text, bright colours
- if a target audience is 55 years old, small pics, text, normal colours
- match the interface to the users (target audience)

Logical and hierarchical organisation of content

- UI should be well set out, not cluttered, easy to follow, large sites should have a sitemap for hierarchical organisation
- UI should have a flow that is easy to understand. Small sites should have a menu system that is named using common names (logical organisation, eg About Us, Contact Us, Home etc
- Let things go where they are expected to go.

Relevant help features of a graphical user interface

- usability; people will be able to use the website more effectively with the following;
    - a search function
    - a site map
    - breadcrumbs
- inclusivity;
    - language choice,
    - cultural sensitivity (images in one culture may not be liked in another culture),
    - gender neutral or specific
- accessibility;
    - font resizeable,
    - alternate text for images,
    - screen readers (software that vision impaired people buy to be able to 'read' the screen)
    - choose colour blind friendly colours (not red and green together)

Processing Efficiency

- Software has different memory and cpu intensity.
- Rendering 3D on a computer uses much more CPU and memory than typing into a small text editing program.
- It would be unwise to run 'high end' software on 'low end' hardware as it will crash.
- When making software, it must be designed to match particular hardware. These are listed in the technical specifications when purchasing the software.

Development Time

- In making the software how much time is needed.
- How many programmers are working on the project.
- Testing must be in this development time to ensure that the software works as intended.
- Keep track of all timing using project management scheduling software such as GANTT charts.

Technical Specifications

- It is important to match the software to the hardware.
- If software is made that demands 8 Gb of RAM and it is on a computer that has only 2 Gb of RAM, there will be problems.
- This is called hardware/software compatibility.
- When advertising the software, the technical specifications must be listed.

Professional Ethics of Developers

There is a dot point on professional ethics of developers of new software hence these notes. This covers some possible issues to be aware of but it's not a section you need to know backwards and forwards like the fetch execute cycle.

The Association for Computing Machinery (ACM) is the world's largest educational and scientific computing society. It has its own Code of Ethics and another set of ethical principles that were also approved by the IEEE as the standard for teaching and practicing software engineering. These codes are Code of Ethics and Professional Conduct and the Software Engineering Code of Ethics and Professional Practice, respectively, and some of their guidelines are presented below:

From the Code of Ethics and Professional Conduct (ACM):

- **Contribute to society and human well-being.** Programmers should work to develop computer systems that can reduce negative consequences to society, such as threats to safety and health, and that can make everyday activities and work easier. It is "an obligation to develop to high standards" (Savage).
- **Avoid harm to others.** Computer systems have an indirect impact on third parties. They can cause loss of information and resources that might result severely harmful for users, the general public, or employers. Therefore, software developers should minimize the risk of harming others due to coding errors, or security issues, by following standards to design and test systems (Code of Ethics and Professional Conduct).
- **Be honest and trustworthy.** This principle encourages programmers to be honest and aware of their limitations in knowledge and education when writing computer systems. Also, if a programmer knows there is something wrong with a computer system, he or she should report it immediately to avoid undesirable consequences.
- **Give proper credit for intellectual property**. It is mandatory for every software developer to never use and take credit for someone else's work, even when it has not been protected by a copyright law, patent, etc. They must recognize and fully credit other people's works, and they should use their own ideas to develop software.
- **Respect the privacy of others.** Computer systems are wrongly used by some people to violate the privacy of others. Software developers should write programs that can protect users' private information and that can avoid other undesired people to have unauthorized access to it (Code of Ethics and Professional Conduct).
- **Honour confidentiality.** Unless required by law or any other ethical guideline, a programmer must keep secret any additional information related to his or her employer that arises from working in a project.

From Software Engineering Code of Ethics and Professional Practice (IEEE, ACM):

- **Approve software only if they have a well-founded belief it is safe and meets specifications.** Programmers cannot assume that a system is ready to use only because it performs the tasks needed. They should make sure these systems are also safe and meet every specification required by the user. If programs are not safe, users are unprotected from hackers that could steal important information or money. Therefore, several tests should be performed in order to ensure a system's security before approving it.

- **Accept full responsibility for their own work.** If a program presents errors, the software developer should accept full responsibility for his or her work, and should work on revising, correcting, modifying, and testing it.
- **Not knowingly use software that is obtained or retained either illegally or unethically.** If a computer system will be used as a base for the creation of another, then permission to do so should be asked by the programmer. This principle prohibits using any other software for any purpose if the way it was gotten is not clear or is known to be illegal or unethical.
- **Identify, define, and address ethical, economic, cultural, legal and environmental issues related to work projects.** If a programmer notices and identifies that working on a project will lead to any kind of problems, then the programmer should report it to his or her employer before continuing.
- **Ensure that specifications for software on which they work satisfy the users' requirements and they have the appropriate approvals.** Software developers should come to their employers to ask for the correspondent approval to the system they are creating before continuing working on the next part. If it doesn't meet the requirements, then a modification to the source code of the system should be made.
- **Ensure adequate testing**, debugging and review of software. Programmers should perform the appropriate tests to the pieces of software they work with and should check for errors and system security holes to make sure that the programs are well implemented.
- **Not engage in deceptive financial practices such as bribery, double billing, or other improper financial practices.** Programmers are exposed to be participants in illegal activities to get money. They get involved in them due to threats, economic issues, or simply because they want to obtain easy money by taking advantage of their knowledge about how computer systems work. This guideline prohibits programmer involvement in such unlawful actions.
- **Improve their ability to create safe, reliable, and useful quality software.** Since technology advances faster year by year, and so does virtual criminality, the need of well-structured and designed programs is increasing. Computer systems get old and limited by new ones and new devices. Programmers should "further their knowledge of developments in the analysis, specification, design, development, maintenance, and testing software and related documents" (Software Engineering Code of Ethics and Professional Practice) in order to create better pieces of software.

Accessibility

Another issue would be **Accessibility**. That is allowing your software to be used by people with disabilities, who speak a different language, limited access to technology, cultural issues or any other factor that might limit their ability to use your software. This can be a legal requirement under anti-discrimination laws for some software. So, for example if you were developing an app to allow people to buy tickets to an event there would have to be a way for people with disabilities to buy tickets also.

Ownership of Intellectual Property

The laws governing Intellectual property include The Intellectual Property Laws Amendment Act 1998, The Copyright Act 1968, the Designs Act 1906, the Patents Act 1990 and the Trade Marks Act 1995.

If you write software for a company who owns the property rights?

- If a company is paying you to develop software, they own the software, not you.
- You as a software developer cannot make money from giving these ideas to a different company.
- To keep confidentiality, some companies make their developers sign a non-disclosure agreement (NDA).
- All countries have laws that protect intellectual property.

If it is in your contract they might own the rights to all software you develop while working there. So if you developed a program outside of work they might also have a claim to that. It would have to be in a contract (that you have signed) however not a “policy” many organisations will have policies about this and that these are guidelines they are not enforceable by law.

Activities

Choose a piece of software that you use.

1. What are the user needs for your software?
2. Briefly describe the UI.
3. How is it licenced?
4. What are its’ technical specifications?
5. Has it addressed Accessibility? If yes how, if no how do you think it could?

References

1. https://en.wikipedia.org/wiki/Per-seat_license
2. https://www.micropact.com/blog/detail/concurrent-vs.-per-seat-licenses-the-analogy
3. https://www.ikea.com/au/en/catalog/categories/departments/living_room/16239/
4. https://trekmovie.com/2013/05/14/doug-drexler-bringing-his-uss-enterprise-to-star-trek-continues-exclusive-first-look/
5. https://study.com/academy/lesson/software-licensing-proprietary-and-free-and-open-source-licenses.html
6. https://www.thecomputingteacher.com/csc/index.php/sdc/development-factors
7. https://www.coleyconsulting.co.uk/require.htm
8. https://searchsoftwarequality.techtarget.com/tip/Software-requirements-Using-models-to-understand-users-needs
9. http://www.cpushack.com/2013/01/26/cpu-of-the-day-new-logo-old-processor-intel-486-dx2-66/
10. https://www.mobygames.com/game/dos/doom/cover-art/gameCoverId,3354/
11. https://www.acm.org/code-of-ethics
12. "Software Engineering Code of Ethics and Professional Practice." Association for Computing Machinery Advancing Computing as a Science & Profession. Association for Computing Machinery (ACM), Institute of Electrical and Electronic Engineers (IEEE). Web. 28 Sep 2013.

Software Development Cycle

Web and software developers all over the world use the software development ~~life~~ cycle (SD~~L~~C) to implement tech projects successfully. The process dot pointed in the curriculum is called the Software Development Cycle rather than Life Cycle that the rest of the world uses so I'll drop the L from now on. The software industry follows the SDC to design, develop, and test projects before deployment. When carried out effectively, the SDC produces high-quality software that meets customer expectations and reaches completion within time and cost estimates. Because it defines which tasks must be performed at each step in the software development process, the SDC is a method of quality control and a way to ensure that tech development teams stay on the same page.

Analyse Detailed Requirements

- What is required of this software?
- Who is going to use it?
- Why are they going to use it?
- When will they use it?
- What problem will the software overcome?

Requirements analysis, is the process of determining user expectations for a new or modified product. These features, called requirements, must be quantifiable, relevant and detailed. In software engineering, such requirements are often called functional specifications. Requirements analysis is an important aspect of software development.

Requirements analysis involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users, avoidance of feature creep and documentation of all aspects of the project development process from start to finish. Energy should be directed towards ensuring that the final system or product conforms to client needs rather than attempting to mould user expectations to fit the requirements.

Requirements analysis is a team effort that demands a combination of hardware, software and human factors engineering expertise as well as skills in dealing with people.

Design Data and Algorithms

- Plan a solution to the questions above.
- Create flowcharts and show the client to see if these will provide a workable solution.

The primary purpose of most computer programs is not to perform calculations, but to store and retrieve information—usually as fast as possible. For this reason, the design of data structures and the algorithms that manipulate them is at the heart of computer science.

Any program design process looking at Data Structures and Algorithms will need to:

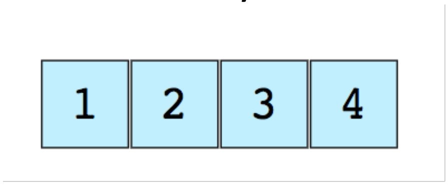
- Develop commonly used data structures and algorithms. These form a programmer's basic "toolkit". For many problems, some data structure or algorithm in the toolkit will provide a good solution.
- Look at trade-offs, between the costs and benefits associated with every data structure or algorithm. This is done by describing, for each data structure, the amount of space and time required for typical operations.
- Measure the effectiveness of a data structure or algorithm. Only through such measurement can you determine which data structure in your toolkit is most appropriate for a new problem. The techniques presented also allow you to judge the merits of new data structures that you or others might invent.

Code Data Structures and Instructions

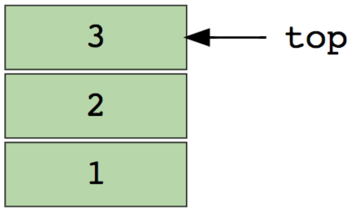
A data structure is a container that stores data in a specific layout. This “layout” allows a data structure to be efficient in some operations and inefficient in others. The goal is to pick the data structure that’s most optimal for the problem at hand.

Some data structures could include;

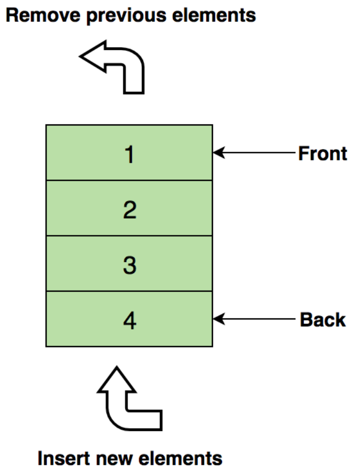
Arrays



Stacks



Queues



In python data structures include Lists, Tuples, Sets etc.

Instructions is referring the coding or programming. Commenting within the code is important so down the track you can understand exactly what the code does. Do this in small chunks at a time.

Debug Syntax and Logic Errors

As part of the programming process you need to fix errors in the code

Syntax Errors are errors in the syntax of a sequence of characters or tokens that is intended to be written in a particular programming language. So errors in the use of language.

For compiled languages, syntax errors are detected at compile-time. A program will not compile until all syntax errors are corrected. For interpreted languages, however, a syntax error may be detected during program execution.

Runtime Errors (not mentioned in the syllabus) are errors that occur while the program is running. These include errors like division by zero errors, overflow errors (calculations produces a result that is greater than what a given register can store or represent), etc.

Logical Errors are a type of runtime errors where there is a bug in a program that causes it to operate incorrectly, but not to terminate abnormally (or crash). A logic error produces unintended or undesired output or other behaviour, although it may not immediately be recognized as such.

Test to Meet Specifications

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing is used to evaluate:

- meets the requirements that guided its design and development,
- responds correctly to all kinds of inputs,
- performs its functions within an acceptable time,
- it is sufficiently usable,
- can be installed and run in its intended environments, and
- achieves the general result its stakeholders desire.

As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources. As a result, software testing typically (but not exclusively) attempts to execute a program or application with the intent of finding software bugs (errors or other defects). The job of testing is an iterative process as when one bug is fixed, it can illuminate other, deeper bugs, or can even create new ones.

Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors.

- Have a report showing details of how the program ran as part of this testing. Try it on different operating systems and hardware.

Document Internally and Externally

Software documentation is written text or illustration that accompanies computer software or is embedded in the source code. It either explains how it operates or how to use it, and may mean different things to people in different roles.

Documentation is an important part of software engineering. Types of documentation include:

1. Embedded – Comments within programming code. Programmers include annotation in the source code of a computer program. They are added with the purpose of making the source code easier to understand and are ignored by compilers and interpreters.
2. Requirements – Statements that identify attributes, capabilities, characteristics, or qualities of a system. This is the foundation for what will be or has been implemented.
3. Architecture/Design – Overview of software. Includes relations to an environment and construction principles to be used in design of software components.
4. Technical – Documentation of code, algorithms, interfaces, and APIs.
5. End user – Manuals for the end-user, system administrators and support staff.
6. Marketing – How to market the product and analysis of the market demand.

- Make a report to show the commenting on your code (internal documentation).
- Make a user guide to show the end user (secretary or other) for how to actually use the software in their location.

Implement and test with live data

- Start the program in the client's work environment with actual data to see if it works.
- Do it over again on different computers to make sure it works for the client on all machines.

Alpha Testing

Alpha testing is a type of acceptance testing; performed to identify all possible issues/bugs before releasing the product to everyday users or the public. The focus of this testing is to simulate real users by using a black box and white box techniques. The aim is to carry out the tasks that a typical user might perform. Alpha testing is carried out in a lab environment and usually, the testers are internal employees of the organization. To put it as simple as possible, this kind of testing is called alpha only because it is done early on, near the end of the development of the software, and before beta testing.

Beta Testing

Beta Testing of a product is performed by "real users" of the software application in a "real environment" and can be considered as a form of external User Acceptance Testing.

Beta version of the software is released to a limited number of end-users of the product to obtain feedback on the product quality. Beta testing reduces product failure risks and provides increased quality of the product through customer validation.

It is the final test before shipping a product to the customers. Direct feedback from customers is a major advantage of Beta Testing. This testing helps to tests the product in customer's environment.

Evaluate performance of the program

- Survey the staff or audience using the program to see if it works as expected.
- Make note of what changes should be made and fix those.
- Consider sending out an update of the software after you have again tested to make sure the changes have been successful.

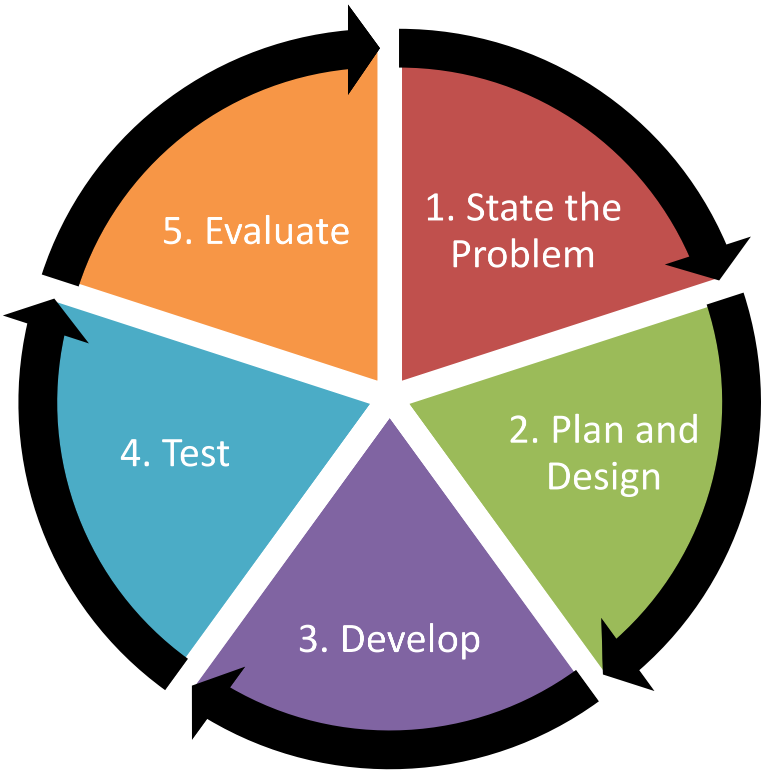
Quiz

Consider the stages of the Software Development Cycle as we learnt them in year 11

Stages of the software development cycle

These are the stages of the SDL that are identified in the curriculum.

- state the problem
- plan and design
- develop
- test
- evaluate



1. Where do the 8 Year Twelve stages fit into the 5 Year 11 Stages?
2. What additional information has been given added to the Software Development ~~Life~~ Cycle?

References

https://online.husson.edu/software-development-cycle/
https://airbrake.io/blog/sdlc/what-is-the-software-development-life-cycle
http://istqbexamcertification.com/what-are-the-software-development-life-cycle-sdlc-phases/
https://projectpals.com/wp-content/uploads/ProblemRepresentation-1.png
https://www.bildnow.com/wp-content/uploads/Blueprint_Large1.jpg
https://www.teachermagazine.com.au/files/Computing%2C_programming_and_coding_in_schools.jpg
https://www.dreamscloud.com/images/mosaics/bcanstockphoto5299794.jpg
https://canateam.com/images/inline/mm-1.jpg
https://opendsa-server.cs.vt.edu/ODSA/Books/CS2/html/IntroDSA.html

https://medium.freecodecamp.org/the-top-data-structures-you-should-know-for-your-next-coding-interview-36af0831f5e3
https://en.wikipedia.org/wiki/Software_testing
https://www.guru99.com/alpha-beta-testing-demystified.html

Data Types

Integer

A whole number. The following are integers:

0

1

-125

144457

In contrast, the following are ***not*** integers:

5.34

-1.0

1.3E4

Real (floating-point) number

A real number (that is, a number that can contain a fractional part). The following are floating-point numbers:

3.0

-111.5

½

3E-5

Boolean data type

In computer science, the Boolean data type is a data type, having two values (usually denoted true and false), intended to represent the truth values of logic and Boolean algebra.

An expression that results in a value of either TRUE or FALSE. For example, the expression

2 < 5 (2 is less than 5)

is a Boolean expression because the result is TRUE. All expressions that contain relational operators , such as the less than sign (<), are Boolean. The operators -- AND, OR, XOR, NOR, and NOT -- are Boolean operators.

Boolean expressions are also called comparison expressions, conditional expressions, and relational expressions.

Character data type

In computer software, any symbol that requires one byte of storage. This includes all the ASCII and extended ASCII characters, including the space character. In character-based software, everything that appears on the screen, including graphics symbols, is considered to be a character.

ASCII

American Standard Code for Information Interchange, is a character encoding standard for electronic communication. ASCII codes represent text in computers, telecommunications equipment, and other devices. Most modern character-encoding schemes are based on ASCII, although they support many additional characters.

String

In computer science, the Boolean data type is a data type that represent text rather than numbers. It is comprised of a set of characters that can also contain spaces and numbers. For example, the word "hamburger" and the phrase "I ate 3 hamburgers" are both strings. Even "12345" could be considered a string, if specified correctly. Typically, programmers must enclose strings in quotation marks for the data to recognized as a string and not a number or variable name.

Arrays

An array is a series of memory locations – or 'boxes' – each of which holds a single item of data, but with each box sharing the same name. All data in an array must be of the same data type.

For example, imagine that a score table in a game needs to record ten scores. One way to do this is to have a variable for each score:

score_0
score_1
score_2
score_3
score_4
score_5
score_6
score_7
score_8
score_9

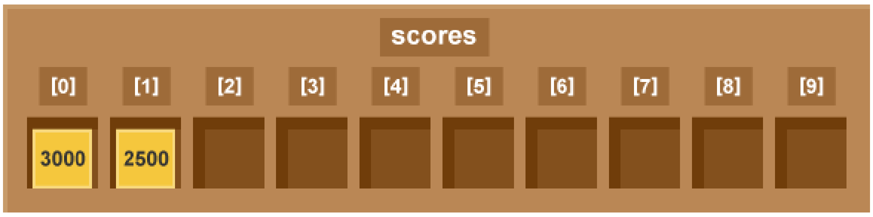
This would work, but there is a better way. It is much simpler to keep all the related data under one name. We do this by using an array.

Instead of having ten variables, each holding a score, there could be one array that holds all the related data:

score(9)

By using this array, all 10 data items can be stored in one place. You have defined 10 variables - score(0), score(1),….score(9)

It helps to think of an array as a row of cells, like the ones found in a table. Each cell represents an element:



The individual values, or array elements, are numbered 0 to 9 because computers start counting at 0.

This describes a one-dimensional array.

Naming arrays

Arrays are named like variables. The number in brackets determines how many data items the array can hold. The array score(9) would allow ten data items to be stored.

To store a data item in an array, the element that the data will be stored in needs to be referenced.

The following would store the score 2500 in the second element:

score[1] = 2500

Record

Programming languages allow you to define a special data structure called a record. Generally, a record is a combination of other data objects. For example, a record might contain three integers, a floating-point number, and a character string.

Records are distinguished from arrays by the fact that their number of fields is typically fixed, each field has a name, and that each field may have a different type.

A record type is a data type that describes such values and variables. Most modern computer languages allow the programmer to define new record types. The definition includes specifying the data type of each field and an identifier (name or label) by which it can be accessed. In type theory, product types (with no field names) are generally preferred due to their simplicity, but proper record types are studied in languages such as System F-sub. Since type-theoretical records may contain first-class function-typed fields in addition to data, they can express many features of object-oriented programming.

Programming Data Types

Constants

In computer programming, a constant is a value that cannot be altered by the program during normal execution, i.e., the value is constant. This is contrasted with a variable, which is an identifier with a value that can be changed during normal execution, i.e., the value is variable.

```
const float PI = 3.1415927;
```

Variable

A symbol or name that stands for a value. For example, in the expression

x+y

x and y are variables. Variables can represent numeric values, characters, character strings, or memory addresses.

Local Variable

A local variable is a variable that is given local scope. Local variable references in the function or block in which it is declared override the same variable name in the larger scope. In programming languages with only two levels of visibility, local variables are contrasted with global variables.

So the variable exists in a function for example but only in that functions. Other functions might have the same variable name with a different value and the data inside that variable can't be accesses outside of that function.

Global Variables

A global variable is a variable with global scope, meaning that it is visible (hence accessible) throughout the program. In compiled languages, global variables are generally static variables, whose extent (lifetime) is the entire runtime of the program, though in interpreted languages (including command-line interpreters), global variables are generally dynamically allocated when declared, since they are not known ahead of time.

Parameter

A parameter is a special kind of variable in computer programming language that is used to pass information between functions or procedures. The actual information passed is called an argument.

Naming Conventions

In computer programming, a naming convention is a set of rules for choosing the character sequence to be used for identifiers which denote variables, types, functions, and other entities in source code and documentation.

Reasons for using a naming convention (as opposed to allowing programmers to choose any character sequence) include the following:

To reduce the effort needed to read and understand source code;

To enable code reviews to focus on more important issues than arguing over syntax and naming standards.

To enable code quality review tools to focus their reporting mainly on significant issues other than syntax and style preferences.

The choice of naming conventions can be an enormously controversial issue, with partisans of each holding theirs to be the best and others to be inferior. Colloquially, this is said to be a matter of dogma. Many companies have also established their own set of conventions.

Potential benefits

Some of the potential benefits that can be obtained by adopting a naming convention include the following:

- to provide additional information (i.e., metadata) about the use to which an identifier is put;
- to help formalize expectations and promote consistency within a development team;
- to enable the use of automated refactoring or search and replace tools with minimal potential for error;
- to enhance clarity in cases of potential ambiguity;
- to enhance the aesthetic and professional appearance of work product (for example, by disallowing overly long names, comical or "cute" names, or abbreviations);
- to help avoid "naming collisions" that might occur when the work product of different organizations is combined (see also: namespaces);
- to provide meaningful data to be used in project handovers which require submission of program source code and all relevant documentation;
- to provide better understanding in case of code reuse after a long interval of time.

Readability

Well-chosen identifiers make it significantly easier for developers and analysts to understand what the system is doing and how to fix or extend the source code to apply for new needs.

For example, although the statement

```
a = b * c;
```

is syntactically correct, its purpose is not evident. Contrast this with:

```
weekly_pay = hours_worked * pay_rate;
```

which implies the intent and meaning of the source code, at least to those familiar with the context of the statement.

Common elements

The exact rules of a naming convention depend on the context in which they are employed. Nevertheless, there are several common elements that influence most if not all naming conventions in common use today.

Length of identifiers

- shorter identifiers may be preferred as more expedient, because they are easier to type
- extremely short identifiers (such as 'i' or 'j') are very difficult to uniquely distinguish using automated search and replace tools
- longer identifiers may be preferred because short identifiers cannot encode enough information or appear too cryptic
- longer identifiers may be disfavoured because of visual clutter

Letter case and numerals

Some naming conventions limit whether letters may appear in uppercase or lowercase. Other conventions do not restrict letter case, but attach a well-defined interpretation based on letter case. Some naming conventions specify whether alphabetic, numeric, or alphanumeric characters may be used, and if so, in what sequence.

Multiple-word identifiers

A common recommendation is "Use meaningful identifiers." A single word may not be as meaningful, or specific, as multiple words. Consequently, some naming conventions specify rules for the treatment of "compound" identifiers containing more than one word.

As most programming languages do not allow whitespace in identifiers. One approach is to delimit separate words with a nonalphanumeric character. The two characters commonly used for this purpose are the hyphen ("-") and the underscore ("_"); e.g., the two-word name "two words" would be represented as "two-words" or "two_words". Another approach is to indicate word boundaries using medial capitalization, called "camelCase", "Pascal case", and many other names.

Activities

1. Why might you use integers as your data type rather than real numbers?
2. Why is Boolen data so important is how computers function?
3. What are some other common terms used to refer to real, and character data?
4. Why are Parameters important for Functions to be useful in programming?
5. You are writing a program to calculate the daily profit at a coffee shop. Devise a naming scheme for your program and list and describe some of the variables you would use in this program. You must include an example of variables of each variable type outlined in the notes.

References

https://www.bbc.com/bitesize/guides/zy9thyc/revision/1
https://en.wikipedia.org/wiki/Record_(computer_science)
https://en.wikipedia.org/wiki/Global_variable
https://www.techopedia.com/definition/3725/parameter-param

Coding

Source Code

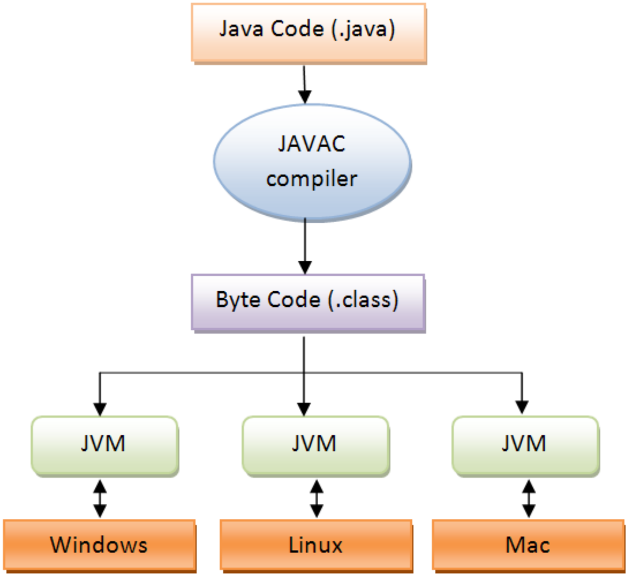
In computing, source code is any collection of code, written using a human-readable programming language. It's also a surprisingly good 2011 sci-fi time travel movie you totally should see it. The big red button in the end is pretty ridiculous – you'll know what I mean when you see it.

Bytecode

When converting the programming language from source code to machine code, some programming languages convert the source code to an intermediate code known as bytecode.

Programming code that, once compiled, is run through a virtual machine instead of the computer's processor. By using this approach, source code can be run on any platform once it has been compiled and run through the virtual machine.

Java is one of the major programming languages that uses the bytecode. The process of converting the source code to bytecode is as follows.



In Java, there is a virtual machine called Java Virtual Machine (JVM) that helps to run Java programs. A virtual machine is similar to an operating system installed on the system. When running the Java program, the compiler converts the Java program or the source code to a Java bytecode. Then the JVM converts the bytecode to machine code. Machine code is directly executed by the computer. The bytecode is written for JVM. It is not specific to the machine. Therefore, the bytecode is executed by various platforms such as Windows, Linux and Mac. The bytecode has numeric codes, constants and references that encode the result of parsing and semantic analysis.

Executable Code

A file in a format that the computer can directly execute. Unlike source files, executable files cannot be read by humans. To transform a source file into an executable file, you need to pass it through a compiler or assembler.

It typically refers to machine language, which is the set of native instructions the computer carries out in hardware. Executable files in the DOS/Windows world use .EXE and .COM file extensions, while executable files in Unix and Mac do not require specific extensions. They are identified by their file structure.

Interpreter Vs Compiler

We generally write a computer program using a high-level language. A high-level language is one which is understandable by us humans. It contains words and phrases from the English (or other) language. But a computer does not understand high-level language. It only understands program written in 0's and 1's in binary, called the machine code. A program written in high-level language is called a source code. We need to convert the source code into machine code and this is accomplished by compilers and interpreters. Hence, a compiler or an interpreter is a program that converts program written in high-level language into machine code understood by the computer.

The difference between an interpreter and a compiler is given below:

Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.
It takes less amount of time to analyse the source code but the overall execution time is slower.	It takes large amount of time to analyze the source code but the overall execution time is comparatively faster.
No intermediate object code is generated, hence are memory efficient.	Generates intermediate object code which further requires linking, hence requires more memory.
Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.	It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.
Programming language like Python, Ruby use interpreters.	Programming language like C, C++ use compilers.

Programming Errors

A large part of programming involves finding and correcting errors in the code. That is parts of the code that stop it functioning as intended. There are three types of programming errors identified in the curriculum.

Syntax Errors

Syntax Errors are errors in the language you have used. You messed up some language's syntax. Maybe you forgot to put a colon or a semicolon somewhere in a Python script or maybe, in Java or C++, you forgot a semicolon. Pretty much every programming language goes through a parser, and all syntax errors can be detected by the parser.

Let's look at some examples of syntax errors. In the first example below, the print statement would have worked in the old version of Python (python 2), but one of the big changes that came with Python 3 was the new requirement to put parenthesis around the argument.

1.

```
#  python 3 Syntax Error
message = "Penjee is for Python"
print message

#the above code would have worked in Python 2 before parenthesis became required
```

2.

```
// JAVA Syntax error
// no semicolon on following line
int x = 3
int y =  4 ;
```

You'll get it once you've programmed for a while…

Syntax errors are much easier to catch because your compiler/interpreter will be able to catch them for you.   In fact, many IDE's will give you a warning about syntax errors, no need to even run the program.

Run-Time Errors

A runtime error is a program error that occurs while the program is running. The term is often used in contrast to other types of program errors, such as syntax errors and compile time errors.

There are many different types of runtime errors. One example is a logic error, which produces the wrong output. For example, a miscalculation in the source code or a spreadsheet program may produce the wrong result when a user enters a formula into a cell. Another type of runtime error is a memory leak. This type of error causes a program to continually use up more RAM while the program is running. A memory leak may be due to an infinite loop, not deallocating unused memory, or other reasons.

A program crash is the most noticeable type of runtime error, since the program unexpectedly quits while running. Crashes can be caused by memory leaks or other programming errors. Common examples include dividing by zero, referencing missing files, calling invalid functions, or not handling certain input correctly.

Logical Error

In computer programming, a logic error is a bug in a program that causes it to operate incorrectly, but not to terminate abnormally (or crash). A logic error produces unintended or undesired output or other behaviour, although it may not immediately be recognized as such.

Logic errors occur in both compiled and interpreted languages. Unlike a program with a syntax error, a program with a logic error is a valid program in the language, though it does not behave as intended. The only clue to the existence of logic errors is the production of wrong solutions.

Documentation

Why Have Documentation?

Can we fix it? Well yes we can, but only if it is documented in the first place…. Both the software developer and the maintainer (the person who has to fix it or edit later) has to be able to gain a relatively fast understanding of the software, its design philosophy, assumptions and goals. To do this requires quality documentation.

Internal Documentation

Computer software is said to have Internal Documentation if the notes on how and why various parts of code operate is included within the source code as comments.

The following is considered to be part of ID:

- Comments in the code to explain steps in the algorithm
- Sensible & Meaningful identifiers for constants, variables and module names that comply with Corporate Standards for Identifiers
- code layout with appropriate use of indentation and white space.

There are even International Standards for Internal Documentation such as IEEE 1063-2001 "Standard for Software User Documentation" written by IEEE, ISO/IEC 18019-2004 and ISO/IEC TR 9294.

External Documentation

External documentation is written in a place where people who need to use the software can read about how to use the software. External documentation can be broken down into library documentation, which describes tools that a programmer can use, and user documentation, which is intended for users of an application. {Note that User Documentation – User Manuals are now all on line and so far harder to use! IMHO}

The line between internal and library documentation is not clear cut because the trend is to write library documentation inside a program as comments, relying on software that extracts the documentation and puts it into a form suitable for people who only want to use the library. For example, the extensive Java library documentation is created by software called javadoc that reads Java programs, including comments, and writes documentation.

Importance of documentation for the developer

Each function in a piece of software solves a specific problem. Before you try to solve any problem, you should have a good understanding of exactly what the problem is. It makes no sense just to start writing and then, afterwards, look at what you have come up with to see whether it solves any useful problem!

Inexperienced computer programmers imagine that they can keep all problem descriptions in their heads. Experience has shown that they can't. Three issues come up.

1. When writing a function definition without written documentation, you only have a rough idea of what it is supposed to do. While you write, the idea morphs in your head. A simple interruption can cause the idea to lose what focus it has. You start thinking about the program as a whole instead of thinking of just the function that you are working on, and the function starts to take on responsibilities that it should have nothing to do with.
2. Suppose that you test the function and find that it does not work. So you need to fix it. But during the process of fixing it, you have nothing but your memory telling you want the function is supposed to do. It is difficult to keep that in your head along with the details of how the function is supposed to work, and the process of fixing a function definition takes the function further away from its original intent.
3. Later, when you need to use that function, you have forgotten just what it does. Unwilling to reverse-engineer it, you make a guess based on what you remember. You often forget important details, and your software does not work because of it. You are faced with laborious debugging to find out what is going on.

Importance of documentation for the maintainer

You might have heard of "self-documenting code". The idea is for functions to be written in a readable form so that, to find out what a function does, you just read the function's definition. For very small pieces of software that can be achieved. But imagine a larger piece of software, say with about 1000 functions. Such software is built up function upon function; one function typically uses a few others that are defined in the same collection of 1000 functions, with the exception of the bottom-level functions that only use the library.

Suppose that the software has no internal documentation and relies on "self-documenting code". Now you want to understand what a particular function does. But it uses 3 other undocumented functions, so you need to understand what they do first. Each of those uses 2 undocumented functions, so you must read their definitions too. It goes on and on. You find yourself reading thousands of lines of code to understand a single function whose body is only ten lines long.

The only way that anyone can work with undocumented software is to reverse-engineer the whole thing and add documentation that should have been written by the developer. Most of the time, that is too difficult. Undocumented software is often just thrown away as unmaintainable.

http://www.cs.ecu.edu/karl/3300/spr14/Notes/Documentation/documentation.html

Activities

1. What type of programming error will allow your program to run but not give you the results you want.
2. Copy a fragment of your Python code from any longer program that you have written and explain which aspects of it are examples of good Internal Documentation. Clearly label the parts that are good examples. Explain why some it needs improvement and is not compliant with acceptable standards for Internal Documentation.
3. List the components of good Internal Documentation. (See the Appendix C for help)
4. External Documentation is meant for whom and whom does it assist and why?

References

https://en.wikipedia.org/wiki/Source_code

https://www.differencebetween.com/difference-between-source-code-and-vs-bytecode/

https://www.webopedia.com/TERM/B/bytecode.html

https://www.programiz.com/article/difference-compiler-interpreter

Code Testing

Trace Tables

A trace table is a technique used to test algorithms, in order to make sure that no logical errors occur whilst the algorithm is being processed. The table usually takes the form of a multi-column, multi-row table; With each column showing a variable, and each row showing each number input into the algorithm and the subsequent values of the variables.

Trace tables are typically used in schools and colleges when teaching students how to program. They can be an essential tool in teaching students how a certain algorithm works and the systematic process that is occurring when the algorithm is executed. They can also be useful for debugging applications, helping the programmer to easily detect what error is occurring, and why it may be occurring.

A trace table is a technique used to test algorithms, in order to make sure that no logical errors occur whilst the algorithm is being processed. The table usually takes the form of a multi-column, multi-row table; With each column showing a variable, and each row showing each number input into the algorithm and the subsequent values of the variables.

Trace tables are typically used in schools and colleges when teaching students how to program. They can be an essential tool in teaching students how a certain algorithm works and the systematic process that is occurring when the algorithm is executed. They can also be useful for debugging applications, helping the programmer to easily detect what error is occurring, and why it may be occurring.

```
number = 3
PRINT number
FOR i from 1 to 3:
 number = number + 5
 PRINT number
PRINT "?"
```

Algorithm

```
1  number = 3
2  PRINT number
3  FOR i from 1 to 3:
4      number = number + 5
5      PRINT number
6  PRINT " ? "
```

Trace Table

Line	number	i	OUTPUT

Algorithm

```
1  number = 3
2  PRINT number
3  FOR i from 1 to 3:
4      number = number + 5
5      PRINT number
6  PRINT " ? "
```

Trace Table

Line	number	i	OUTPUT
1	3		
2			3
3		1	
4	8		
5			8
3		2	
4	13		
5			13
3		3	
4	18		
5			**18**

Range Checking

- A range check is used to make sure our data falls within the "*boundaries*" that we set
- We can look at it as the *scope* of values that our data can fall within
- Range checking is most commonly used on data that involves numbers and figures

Example:

Type	Description	Data Validation
Maximum	The maximum grade possible for Computer Science is 100%	$<= 100$
Minimum	The minimum grade possible for Computer Science is 0%	$>= 0$
Range	To achieve an 'A' grade in the Computer Science course you must score between 75% and 100%.	$>= 75$ AND $<= 100$

Type Checking

- Type checking is the process of verifying and enforcing the constraints of types of data, and it can occur either at compile time (i.e. statically C and Java do this) or at runtime (i.e. dynamically python does this).
- Type checking is all about ensuring that the possibility of type errors is kept to a minimum. A type check is a simple check on the type of data being put in to or pulled out from a function
- We do these checks to ensure that the program will not "break" from an incompatible data type
- These are often used for date / text fields (including mail addresses) to validate the address or date entered is correct

Example:

Type	Description	Type Validation
Number	The number entered must be in numerical form. E.g. 10 not "ten"	$10 \neq$ ten
Email	The email address must follow correct syntax and resolve to a real domain	email@domain.tld
Date	The date must be valid (e.g. there is no 34th day or 13th month)	34/13/2012

Programming Concepts

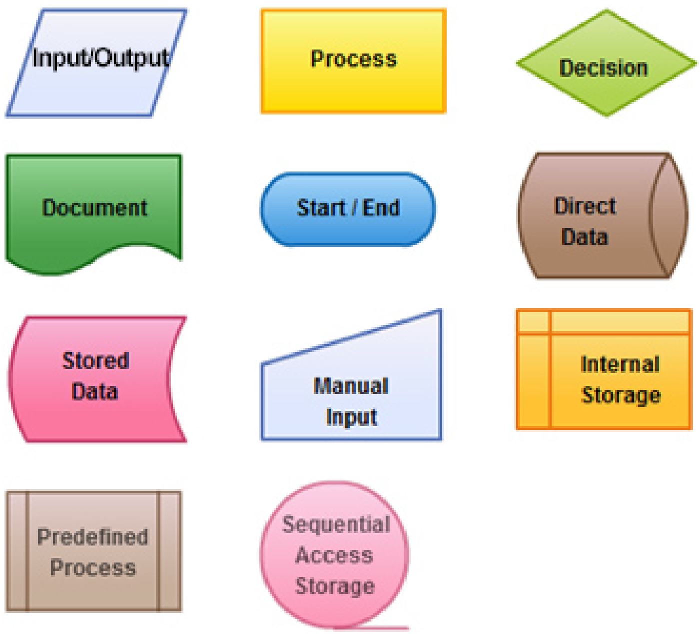
Flow Charts

A flowchart is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields.

Flowcharts are used in designing and documenting complex processes or programs. Like other types of diagrams, they help visualize what is going on and thereby help the people to understand a process, and perhaps also find flaws, bottlenecks, and other less-obvious features within it. There are many different types of flowcharts, and each type has its own repertoire of boxes and notational conventions. The two most common types of boxes in a flowchart are:

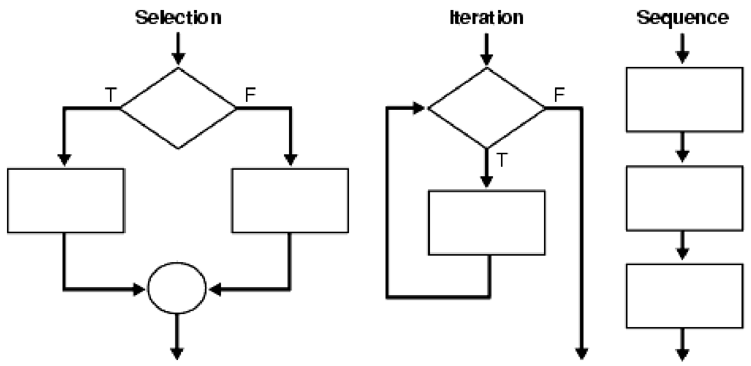
- a processing step, usually called activity, and denoted as a rectangular box
- a decision, usually denoted as a diamond.

Flow chart symbols



Control Structures

A control structure is a block of programming that analyzes variables and chooses a direction in which to go based on given parameters. The term flow control details the direction the program takes (which way program control "flows"). Hence it is the basic decision-making process in computing; flow control determines how a computer will respond when given certain conditions and parameters.



Sequence

A sequence is a piece of code that runs from top to bottom without branching or looping.

Selection One Way (if then)

IF-THEN

The simplest form of IF statement associates a condition with a sequence of statements enclosed by the keywords THEN and END IF (not ENDIF), as follows:

```
IF condition THEN
  sequence_of_statements
END IF;
```

The sequence of statements is executed only if the condition is true. If the condition is false or null, the IF statement does nothing. In either case, control passes to the next statement. An example follows:

```
IF sales > quota THEN
  compute_bonus(empid);
  UPDATE payroll SET pay = pay + bonus WHERE empno = emp_id;
END IF;
```

You might want to place brief IF statements on a single line, as in

```
IF x > y THEN high := x; END IF;
```

Selection Two Way (if then else)

The second form of IF statement adds the keyword ELSE followed by an alternative sequence of statements, as follows:

```
IF condition THEN
  sequence_of_statements1
ELSE
  sequence_of_statements2
END IF;
```

The sequence of statements in the ELSE clause is executed only if the condition is false or null. Thus, the ELSE clause ensures that a sequence of statements is executed. In the following example, the first UPDATE statement is executed when the condition is true, but the second UPDATE statement is executed when the condition is false or null:

```
IF trans_type = 'CR' THEN
  UPDATE accounts SET balance = balance + credit WHERE ...
ELSE
  UPDATE accounts SET balance = balance - debit WHERE ...
END IF;
```

Selection Multi Way (case, nested if)

The THEN and ELSE clauses can include IF statements. That is, IF statements can be nested, as the following example shows:

```
IF trans_type = 'CR' THEN
  UPDATE accounts SET balance = balance + credit WHERE ...
ELSE
  IF new_balance >= minimum_balance THEN
    UPDATE accounts SET balance = balance - debit WHERE ...
  ELSE
    RAISE insufficient_funds;
  END IF;
END IF;
```

Case Example

Case statements provides a list of possible outcomes.

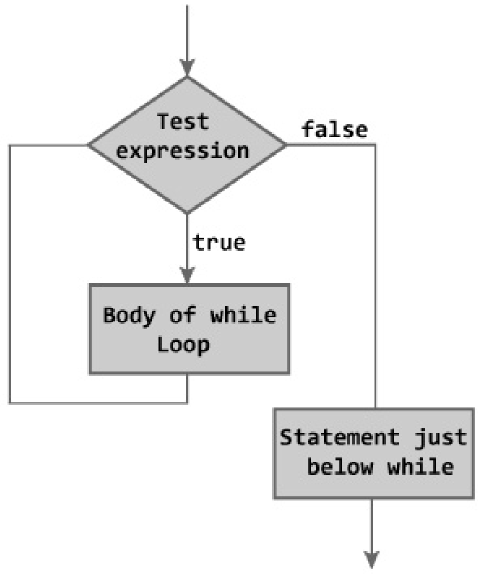
```
INTEGER :: Class

SELECT CASE (Class)
  CASE (1)
    WRITE(*,*)  'Freshman'
  CASE (2)
    WRITE(*,*)  'Sophomore'
  CASE (3)
    WRITE(*,*)  'Junior'
  CASE (4)
    WRITE(*,*)  'Senior'
  CASE DEFAULT
    WRITE(*,*)  "Hmmmm, I don't know"
END SELECT
WRITE(*,*)  'Done'
```

The programming language C uses a switch to provide case selection. Python doesn’t have case selection but uses dictionaries to provide the same functionality.

```
options = {0 : zero,
           1 : sqr,
           4 : sqr,
           9 : sqr,
           2 : even,
           3 : prime,
           5 : prime,
           7 : prime,
}

def zero():
   print "You typed zero.\n"

def sqr():
   print "n is a perfect square\n"

def even():
   print "n is an even number\n"

def prime():
   print "n is a prime number\n"
```

Test First (while)

The WHILE-LOOP statement associates a condition with a sequence of statements enclosed by the keywords LOOP and END LOOP, as follows:




```
WHILE condition LOOP
  sequence_of_statements
END LOOP;
```

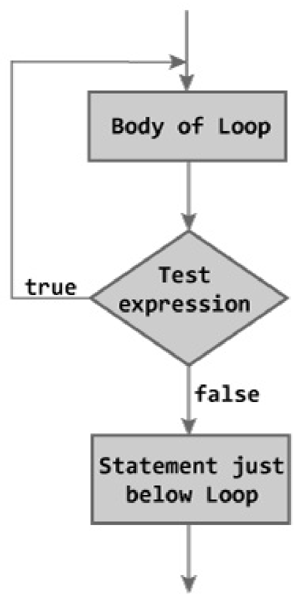
Before each iteration of the loop, the condition is evaluated. If the condition is true, the sequence of statements is executed, then control resumes at the top of the loop. If the condition is false or null, the loop is bypassed and control passes to the next statement. An example follows:

```
WHILE total <= 25000 LOOP
  ...
  SELECT sal INTO salary FROM emp WHERE ...
  total := total + salary;
END LOOP;
```

Test Last (repeat until)

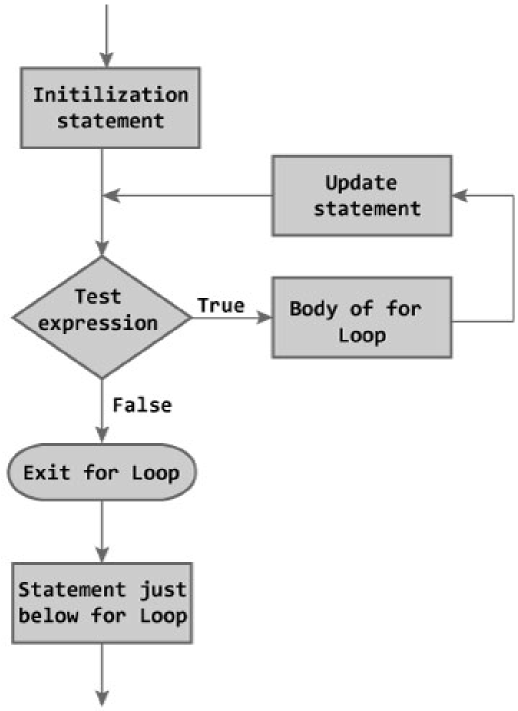
Unlike for and while loops, which test the loop condition at the top of the loop, the repeat ... until loop checks its condition at the bottom of the loop.

A repeat ... until loop is similar to a while loop, except that a repeat ... until loop is guaranteed to execute at least one time.



Fixed (for)

Like the while loop the for loop is a programming language statement, i.e. an iteration statement, which allows a code block to be repeated a certain number of times.




```
for x in range(0, 3):
   print "We're on time %d" % (x)
```

Stub

A stub is a routine that doesn't actually do anything other than declaring itself and the parameters it accepts and returning something that is usually the values expected in one of the "happy scenarios" for the caller. Stubs are used commonly as placeholders for implementation of a known interface, where the interface is finalized/known but the implementation is not yet known/finalized. The stub contains just enough code to allow it to be compiled and linked with the rest of the program. In RMI nomenclature, a stub communicates on the server-side with a skeleton.

A stub may simulate the behavior of existing code (such as a procedure on a remote machine, such methods are often called mocks) or be a temporary substitute for yet-to-be-developed code. Stubs are therefore most useful in porting, distributed computing as well as general software development and testing.

An example of a stub in pseudocode might be as follows:

```
BEGIN
   Temperature = ThermometerRead(Outside)
   IF Temperature > 40 THEN
       PRINT "It's HOT!"
   END IF
END

BEGIN ThermometerRead(Source insideOrOutside)
    RETURN 28
END ThermometerRead
```

Statement

In computer programming, a statement is a syntactic unit of an imperative programming language that expresses some action to be carried out. A program written in such a language is formed by a sequence of one or more statements. A statement may have internal components (e.g., expressions).

Functions

A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for your application and a high degree of code reusing. You have already seen various functions in python like type() and int(). These are called built-in functions provided by the language itself, but you can write your own functions as well.

Modularisation

Modular programming is the process of subdividing a computer program into separate sub-programs.

A module is a separate software component. It can often be used in a variety of applications and functions with other components of the system. Similar functions are grouped in the same unit of programming code and separate functions are developed as separate units of code so that the code can be reused by other applications.

Object-oriented programming (OOP) is compatible with the modular programming concept to a large extent. Modular programming enables multiple programmers to divide up the work and debug pieces of the program independently.

The benefits of using modular programming include:

- Less code has to be written.
- A single procedure can be developed for reuse, eliminating the need to retype the code many times.
- Programs can be designed more easily because a small team deals with only a small part of the entire code.
- Modular programming allows many programmers to collaborate on the same application.
- The code is stored across multiple files.
- Code is short, simple and easy to understand.
- Errors can easily be identified, as they are localized to a subroutine or function.
- The same code can be used in many applications.
- The scoping of variables can easily be controlled.

You have already used modules in python, the turtle and maths modules. You can write your own modules.

https://www.datacamp.com/community/tutorials/modules-in-python

Scope and Lifetime

The difference between lifetime and scope is quite simple.
Lifetime - Refers to how long or when the variable is valid (i.e. how long will it retain its value for).
Scope - Refers to where the variable can be accessed. The difference between lifetime and scope is quite simple. Lifetime is how long the variable retains its value for. The scope refers to where the variable can be used.

The scope controls the lifetime and visibility of the variable. It is good programming practice to use the most restrictive scope possible for your variables.

Not all variables are accessible from all parts of our program, and not all variables exist for the same amount of time. Where a variable is accessible and how long it exists depend on how it is defined. We call the part of a program where a variable is accessible its scope, and the duration for which the variable exists its lifetime.

A variable which is defined in the main body of a file is called a global variable. It will be visible throughout the file, and also inside any file which imports that file. Global variables can have unintended consequences because of their wide-ranging effects – that is why we should almost never use them. Only objects which are intended to be used globally, like functions and classes, should be put in the global namespace.

A variable which is defined inside a function is local to that function. It is accessible from the point at which it is defined and exists for as long as the function is executing. The parameter names in the function definition behave like local variables, but they contain the values that we pass into the function when we call it. When we use the assignment operator (=) inside a function, its default behaviour is to create a new local variable.

Here is an example of variables in different scopes:

```
# This is a global variable
a = 0

if a == 0:
   # This is still a global variable
   b = 1

def my_function(c):
   # this is a local variable
   d = 3
   print(c)
   print(d)

# Now we call the function, passing the value 7 as the first and only parameter
my_function(7)

# a and b still exist
print(a)
print(b)

# c and d don't exist anymore -- these statements will give us name errors!
print(c)
print(d)
```

Parameter Passing

Parameter Passing is the mechanism used to pass parameters (variables) to a procedure (subroutine) or function. The most common evaluation strategy when passing arguments to a function has been call by value and call by reference:

Call by Value

The most common strategy is the call-by-value or pass-by-value. In call-by-value the result is bound to the corresponding variable in the function. A local copy of its value will be used and will be unchanged when the function returns. So, you call a function using a variable this function has access to the value of that variable but can't change it outside of the function.

Call by Reference

In call-by-reference or pass-by-reference, a function gets an implicit reference to the argument, rather than a copy of its value. As a consequence, the function can modify the value of the variable outside of the function.

Activities

1. Create a flow chart on asking a group of people if they want a tea or coffee. You need to allow for multiple people, tea or coffee, milk and sugar.

2. Describe the scope of the variables a, b, c and d in this example:

```
def my_function(a):
   b = a - 2
   return b

c = 3

if c > 2:
   d = my_function(5)
   print(d)
```

   What is the lifetime of these variables? When will they be created and destroyed?

   Can you guess what would happen if we were to assign c a value of 1 instead?

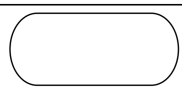
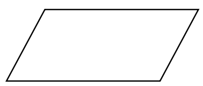
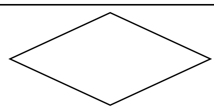
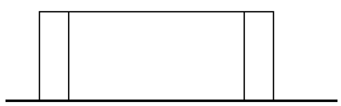
   Why would this be a problem? Can you think of a way to avoid it?

References

https://docs.oracle.com/cd/A87860_01/doc/appdev.817/a77069/03_struc.htm
https://cdn.programiz.com/sites/tutorial2program/files/c-do-while-loop.jpg
https://en.wikipedia.org/wiki/Method_stub
https://en.wikipedia.org/wiki/Statement_(computer_science)
https://www.techopedia.com/definition/25972/modular-programming
https://www.encyclopedia.com/computing/dictionaries-thesauruses-pictures-and-press-releases/parameter-passing

Unit 2B – Program Components and Simple Algorithms

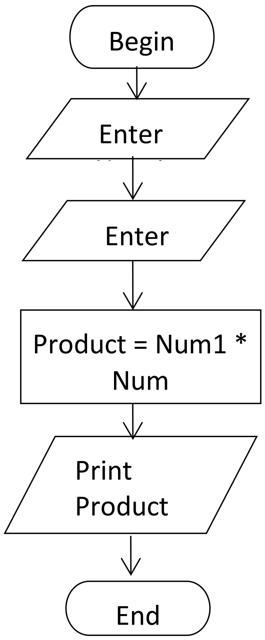
Flow chart symbols and pseudocode

Symbol	Meaning
	Terminal: begin and end
	Input or output
	Process: the description of an action or process
	Decision: one line comes in at the top and two lines leave it
	Sub-program or module: a portion of code that performs a particular task. The Stage 2 syllabus does not include modules, but modular code should be encouraged as early as possible, as this promotes efficiency of code.

Sequence

The instructions are processed in order.

Flowchart



Pseudocode

```
Input (Num1)
Input (Num2)
Product ← Num1 * Num2
Output (Product)
```

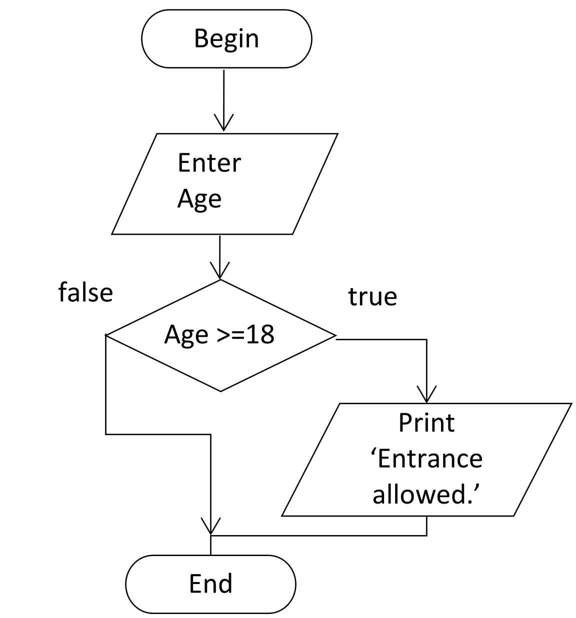
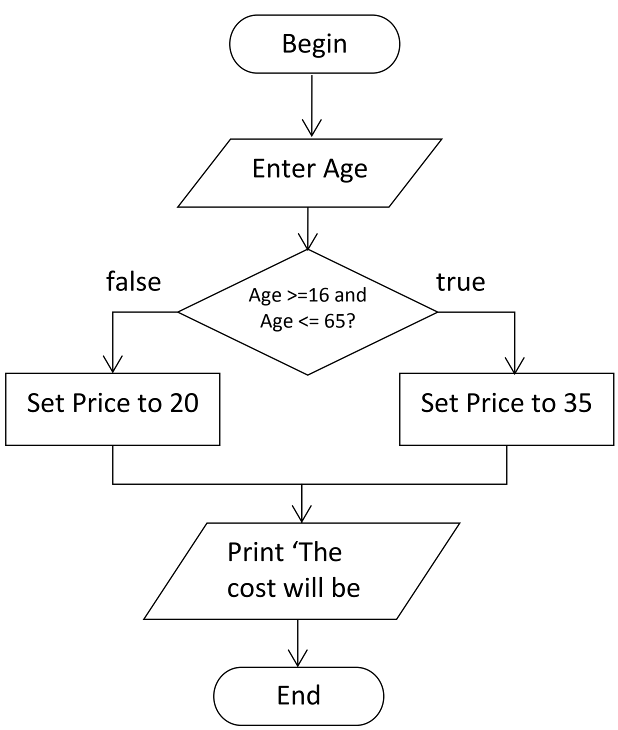
OR

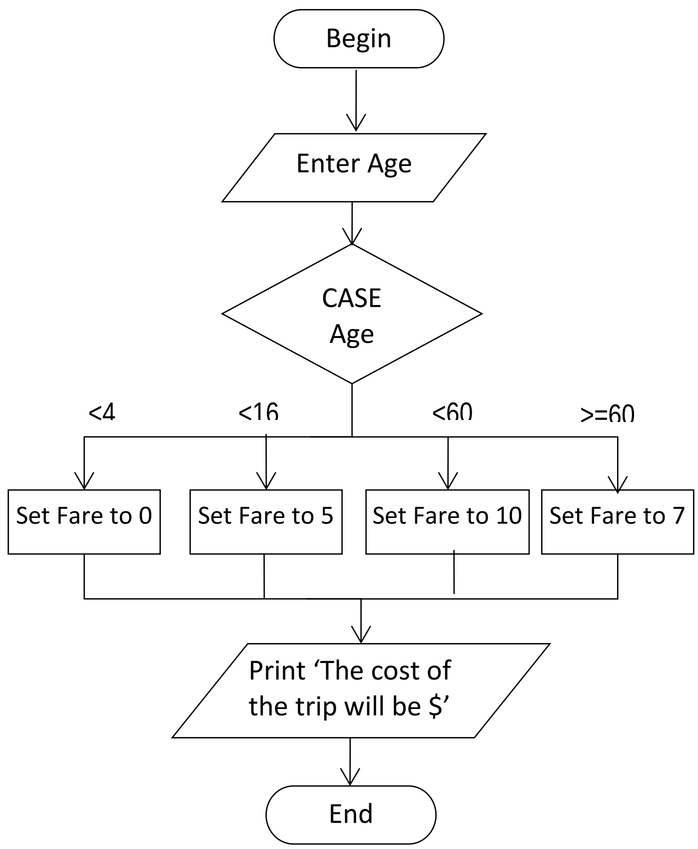
```
Read (Num1)
Read (Num2)
Product ← Num1 * Num2
Write (Product)
```

Note: Read and Write can be used in place of Input and Output

Selection

A condition is tested to determine which branch or path is followed.

Flowchart	Pseudocode
	One way selection If *condition* then Input (Age) If Age >= 18 then Output ('Entrance allowed') End If
	Two-way selection If *condition* then .. else Input (Age) If (Age >= 16) and (Age <= 65) then Price $\leftarrow$ 35 Else Price $\leftarrow$ 20 End If Output ('The cost will be \$', Price)

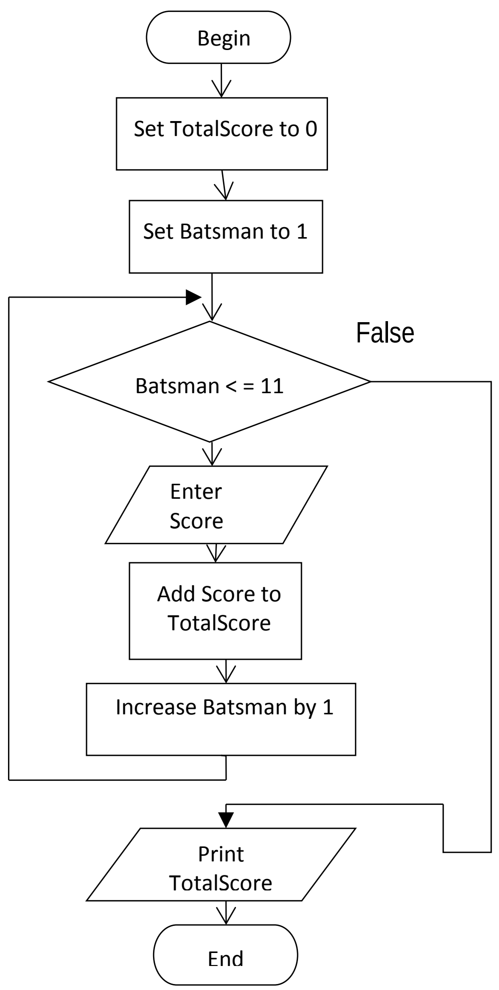
Selection (continued) A condition is tested to determine which branch or path is followed.	
Flowchart	**Pseudocode**
	**Multi way selection (Case)** Input (Age) Case Age of < 4 : Fare ← 0 < 16 : Fare ← 5 < 60 : Fare ← 10 >= 60 : Fare ← 7 End Case Output ('The cost of the trip will be $', Fare)

Repetition also commonly called iteration or looping

Repeating an action or series of actions a number of times.

FOR: Fixed or counted loop

This loops or repeats a counted or fixed number of times.
The number of repetitions is known when the loop begins.

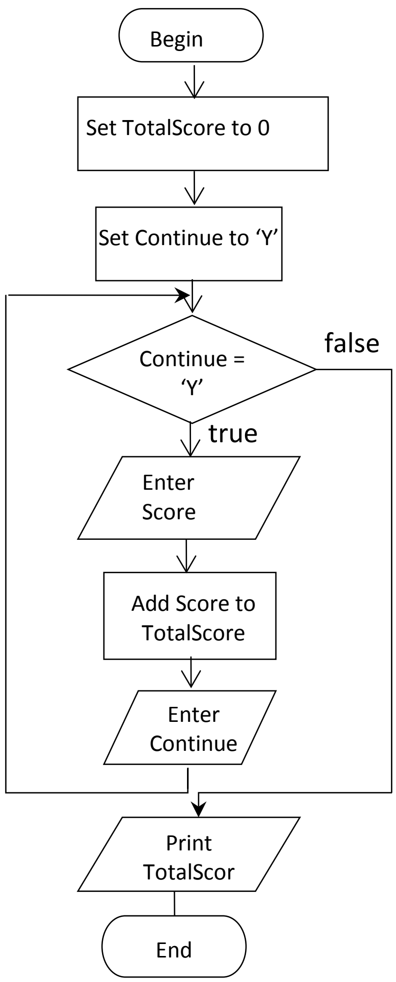
Flow chart	Pseudocode
	TotalScore ← 0 For Batsman ← 1 to 11 Input (Score) TotalScore ← TotalScore + Score End For Output (TotalScore)

While: test first or pre-test loop

This loops a variable number of times.

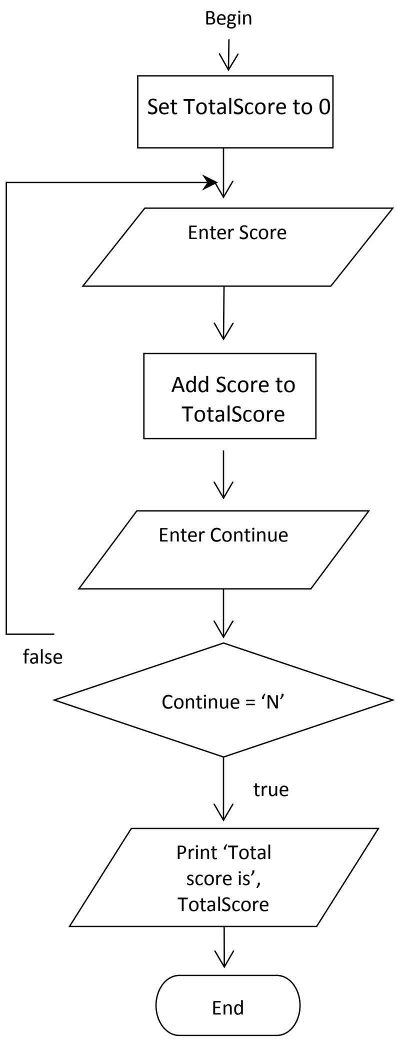
The number of repetitions is not known when the loop begins. This is tested before the loop is entered—test first—it is possible that the loop is executed zero times.

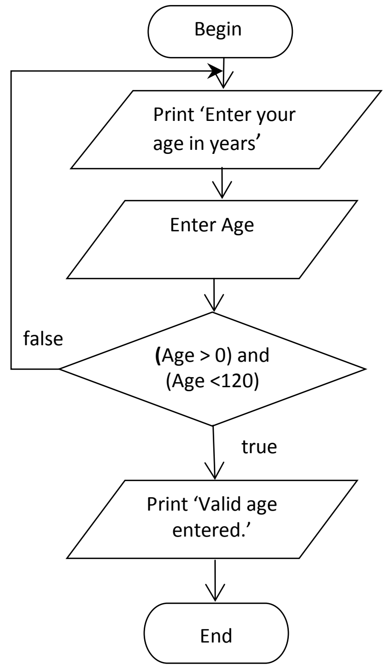
Flow chart



Pseudocode

```
TotalScore ← 0
Continue ← ‘Y’
While Continue = ‘Y’
        Input (Score)
        TotalScore ← TotalScore + Score
        Input (Continue)
End While
Output (TotalScore)
```

Repeat ... Until: test last of post-test loop	
This loops a variable number of times. The number of repetitions is not known when the loop begins. This is tested at the end of the loop—test last—and therefore must be executed at least once.	
Flow chart	**Pseudocode**
	TotalScore ← 0 Repeat Input (Score) TotalScore ← TotalScore + Score Input (Continue) Until Continue = 'N' Output ('Total score is ', TotalScore)




```
Repeat
        Output ('Enter your age in years')
        Input (Age)
Until (Age > 0) and (Age <120)
Output ('Valid age entered')
```

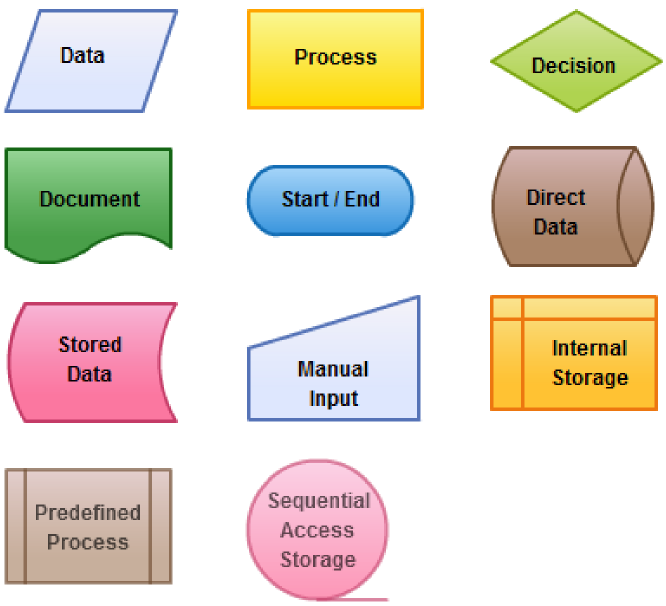
Flow Charts

A flowchart is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields.

Flowcharts are used in designing and documenting complex processes or programs. Like other types of diagrams, they help visualize what is going on and thereby help the people to understand a process, and perhaps also find flaws, bottlenecks, and other less-obvious features within it. There are many different types of flowcharts, and each type has its own repertoire of boxes and notational conventions. The two most common types of boxes in a flowchart are:

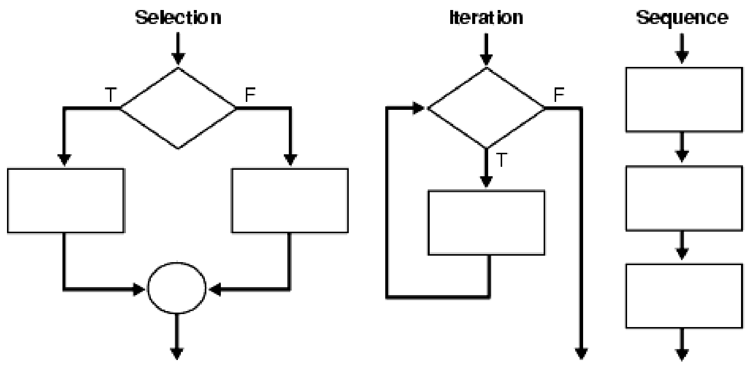
- a processing step, usually called activity, and denoted as a rectangular box
- a decision, usually denoted as a diamond.

Flow chart symbols



Control Structures

A control structure is a block of programming that analyzes variables and chooses a direction in which to go based on given parameters. The term flow control details the direction the program takes (which way program control "flows"). Hence it is the basic decision-making process in computing; flow control determines how a computer will respond when given certain conditions and parameters.



Sequence

A sequence is a piece of code that runs from top to bottom without branching or looping.

Selection One Way (if then)

IF-THEN

The simplest form of IF statement associates a condition with a sequence of statements enclosed by the keywords THEN and END IF (not ENDIF), as follows:

```
IF condition THEN
  sequence_of_statements
END IF;
```

The sequence of statements is executed only if the condition is true. If the condition is false or null, the IF statement does nothing. In either case, control passes to the next statement. An example follows:

```
IF sales > quota THEN
  compute_bonus(empid);
  UPDATE payroll SET pay = pay + bonus WHERE empno = emp_id;
END IF;
```

You might want to place brief IF statements on a single line, as in

```
IF x > y THEN high := x; END IF;
```

Selection Two Way (if then else)

The second form of IF statement adds the keyword ELSE followed by an alternative sequence of statements, as follows:

```
IF condition THEN
  sequence_of_statements1
ELSE
  sequence_of_statements2
END IF;
```

The sequence of statements in the ELSE clause is executed only if the condition is false or null. Thus, the ELSE clause ensures that a sequence of statements is executed. In the following example, the first UPDATE statement is executed when the condition is true, but the second UPDATE statement is executed when the condition is false or null:

```
IF trans_type = 'CR' THEN
  UPDATE accounts SET balance = balance + credit WHERE ...
ELSE
  UPDATE accounts SET balance = balance - debit WHERE ...
END IF;
```

Selection Multi Way (case, nested if)

The THEN and ELSE clauses can include IF statements. That is, IF statements can be nested, as the following example shows:

```
IF trans_type = 'CR' THEN
  UPDATE accounts SET balance = balance + credit WHERE ...
ELSE
  IF new_balance >= minimum_balance THEN
    UPDATE accounts SET balance = balance - debit WHERE ...
  ELSE
    RAISE insufficient_funds;
  END IF;
END IF;
```

Case Example

Case statements provides a list of possible outcomes.

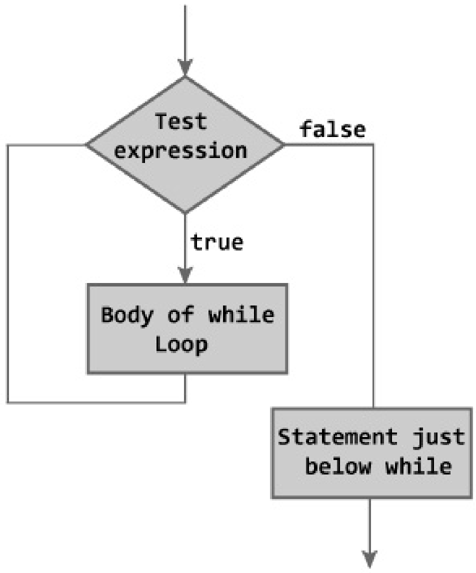
```
INTEGER :: Class

SELECT CASE (Class)
  CASE (1)
    WRITE(*,*)  'Freshman'
  CASE (2)
    WRITE(*,*)  'Sophomore'
  CASE (3)
    WRITE(*,*)  'Junior'
  CASE (4)
    WRITE(*,*)  'Senior'
  CASE DEFAULT
    WRITE(*,*)  "Hmmmm, I don't know"
END SELECT
WRITE(*,*)  'Done'
```

The programming language C uses a switch to provide case selection. Python doesn't have case selection but uses dictionaries to provide the same functionality.

```
options = {0 : zero,
           1 : sqr,
           4 : sqr,
           9 : sqr,
           2 : even,
           3 : prime,
           5 : prime,
           7 : prime,
}

def zero():
   print "You typed zero.\n"

def sqr():
   print "n is a perfect square\n"

def even():
   print "n is an even number\n"

def prime():
   print "n is a prime number\n"
```

Test First (while)

The WHILE-LOOP statement associates a condition with a sequence of statements enclosed by the keywords LOOP and END LOOP, as follows:




```
WHILE condition LOOP
  sequence_of_statements
END LOOP;
```

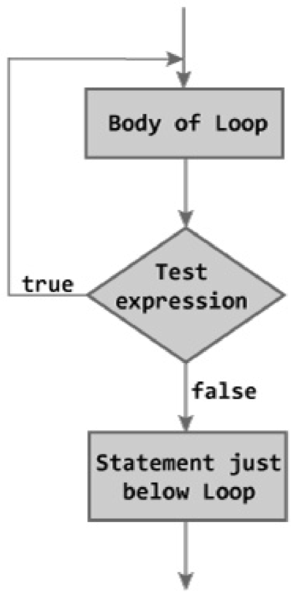
Before each iteration of the loop, the condition is evaluated. If the condition is true, the sequence of statements is executed, then control resumes at the top of the loop. If the condition is false or null, the loop is bypassed and control passes to the next statement. An example follows:

```
WHILE total <= 25000 LOOP
  ...
  SELECT sal INTO salary FROM emp WHERE ...
  total := total + salary;
END LOOP;
```

Test Last (repeat until)

Unlike for and while loops, which test the loop condition at the top of the loop, the repeat ... until loop checks its condition at the bottom of the loop.

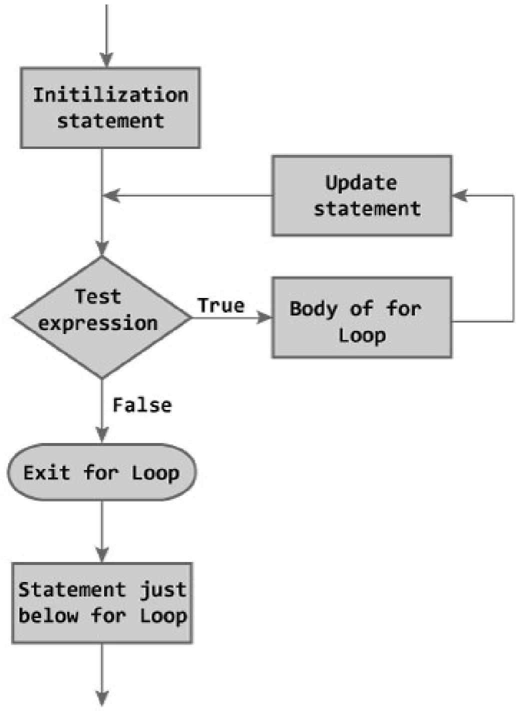
A repeat ... until loop is similar to a while loop, except that a repeat ... until loop is guaranteed to execute at least one time.



Fixed (for)

Like the while loop the for loop is a programming language statement, i.e. an iteration statement, which allows a code block to be repeated a certain number of times.

```
for x in range(0, 3):
   print "We're on time %d" % (x)
```



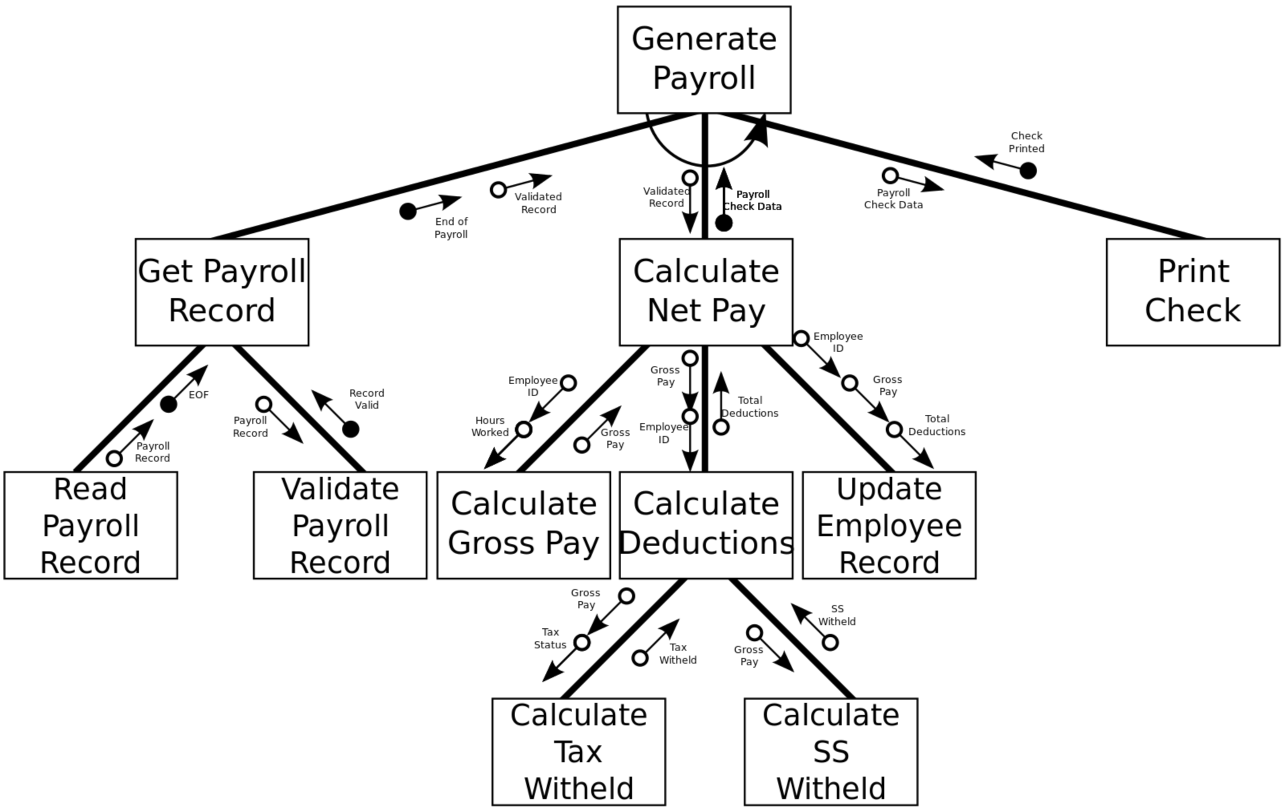
Structure Charts using Yourdon and Constantine method

A Structure Chart (SC) in software engineering and organizational theory is a chart which shows the breakdown of a system to its lowest manageable levels. They are used in structured programming to arrange program modules into a tree. Each module is represented by a box, which contains the module's name. The tree structure visualizes the relationships between modules.

- A structure shows movement of data between modules in a program.
- Visual analysis of a program helps understand how that program works.
- It is very useful when the developer is talking to the client as a tool to help explain how the program will work.

How are these drawn?

- Rectangles are drawn and joined by lines.
- The top rectangle is often called the 'main' module.
- The rectangles under this are other modules or sub-routines.
- These other modules or sub-routines get data from the 'main' module and sometimes return data to it.
- These data flows look a bit odd, they have a small circle at the end of an arrow.
- A white circle is data coupling (data shared by a couple of items/modules)
- A black circle is control coupling (returns a boolean value indicating control). eg Has check been printed = True. Therefore it is a control couple.



Exercise

Create a flow chart on asking a group of people if they want a tea or coffee. You need to allow for multiple people, tea or coffee, milk and sugar.

Draw a structure chart that represents the algorithm below.

```
Module Inputnums
        Input (num1)
        Input (num2)
End Module

Function Average (num1,num2)
        (num1+num2) /2
End Function

Module Outputavg (average)
        Output (average)
End Module

Main Module
        Call Inputnums
        Call Average (num1,num2)
        Call Outputavg
End Module
```

References

https://docs.oracle.com/cd/A87860_01/doc/appdev.817/a77069/03_struc.htm
https://cdn.programiz.com/sites/tutorial2program/files/c-do-while-loop.jpg

Networks

Contents

Table of Contents

Networking

LAN

A local area network (LAN) is a network that connects computers and devices in a limited geographical area such as a home, school computer laboratory, office building or closely positioned group of buildings. Each computer or device on the network, called a node, often shares resources such as printers, large hard disks and programs.

Advantages of Networking Computers

Speed

Sharing and transferring files within networks is very rapid and can save time while maintaining the integrity of the files.

Cost

Individually-licensed copies of many popular software programs can be costly. Site-licensed (shared) versions are available at considerable savings. Programs stored on a network's server allow for easy upgrading.

Security

Sensitive files and programs on a network are easily password-protected and a series of drives can be established for specific directories to restrict access to authorised users. This can also assist in controlling illegal software copying.

Peripheral equipment sharing

Resources such as printers, fax machines and modems can be shared.

Electronic mail

Email helps in personal and professional communication. Electronic mail on a LAN allows an organisation's staff to communicate within the building without leaving their desks.

Centralised software management

Software can be loaded on one computer (the file server), eliminating the need to spend time and energy installing updates and tracking files on independent computers throughout the building.

Workgroup computing

Workgroup software allows many users to work on a document or project concurrently

Disadvantages of networking computers

- The cabling, network cards, file servers, etc that are required to set up a network are expensive.
- If one computer, cable or network card fails, the entire network may stop operating.
- Viruses can spread from one computer to another over a network.
- Files stored on computer networks can be accessed, stolen or edited more easily than files stored on a non-networked computer. Appropriate security measures must be implemented to prevent this.
- If the server develops a fault, users may not be able to run the application programs. A fault in the network can cause users to lose data.
- Decisions on resource planning tend to become centralised.
- Networks that have grown with little thought or appropriate planning can be inefficient in the long term.
- As traffic increases on a network, performance declines unless it is properly designed.

Factorings Influencing Network Performance

Bandwidth

In networking, bandwidth is the maximum rate of data transfer across a given path. It is the result of the medium being used and any hardware bottlenecks slowing down the speed.

- Throughput is the actual rate that information is transferred
- Latency the delay between the sender and the receiver decoding it

Network Design

Network design can have a great influence on the speed. When the traffic load grows on your network, how do you steer around congestion and reduce collisions? By using network switches. Proper planning can avoid bottlenecks. The design should ensure that the areas that need the best bandwidth have it available.

Data Collisions

As networks grow the will be more data collisions as multiple devices try to send data at the same time. Technologies like switches can greatly reduce the amount of data collisions and thus the impact on speed of an expanding network.

Excess broadcast traffic

Excessive broadcast traffic can sometimes create a broadcast storm. A broadcast storm occurs when messages are broadcast on a network and each message prompts a receiving node to respond by broadcasting its own messages on the network. This, in turn, prompts further responses that create a snowball effect. The LAN is suddenly flooded with packets, creating unnecessary traffic that leads to poor network performance or even a complete loss of network service.

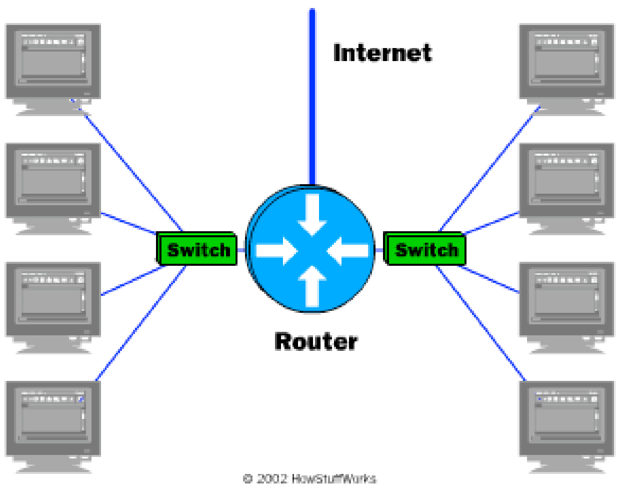
Networking Hardware

Router

A router is a networking device that forwards data packets between computer networks. Routers perform the traffic directing functions on the Internet. Data sent through the internet, such as a web page or email, is in the form of data packets. A packet is typically forwarded from one router to another router through the networks that constitute an internetwork until it reaches its destination node.

A router is connected to two or more data lines from different networks. When a data packet comes in on one of the lines, the router reads the network address information in the packet to determine the ultimate destination. Then, using information in its routing table or routing policy, it directs the packet to the next network on its journey.

The most familiar type of routers are home and small office routers that simply forward IP packets between the home computers and the Internet. An example of a router would be the owner's cable or DSL router, which connects to the Internet through an Internet service provider (ISP). More sophisticated routers, such as enterprise routers, connect large business or ISP networks up to the powerful core routers that forward data at high speed along the optical fiber lines of the Internet backbone. Though routers are typically dedicated hardware devices, software-based routers also exist.



Switch

A network switch is a computer networking device that connects devices on a computer network by using packet switching to receive, process, and forward data to the destination device. Switches are another fundamental part of many networks because they speed things up. Switches allow different nodes (a network connection point, typically a computer) of a network to communicate directly with one another in a smooth and efficient manner.

Essentially, a LAN switch creates a series of instant networks that contain only the two devices communicating with each other at that particular moment.

Firewall

A firewall monitors and controls incoming and outgoing network traffic based on predetermined security rules. A firewall typically establishes a barrier between a trusted internal network and untrusted external network, such as the Internet.

A firewall is recognized as the first line of defence in securing sensitive information. For better safety, the data can be encrypted.

A network's firewall builds a bridge between an internal network that is assumed to be secure and trusted, and another network, such as the Internet, that is not assumed to be secure and trusted

Without a firewall in place, a computer can be accessible to anyone on the Internet.

Modem

Your modem serves as a bridge between your local network and the Internet. Historically, the term “modem” is shorthand for modulator-demodulator. Modems were used to modulate the signals on telephone lines so that digital information could be encoded and transmitted over them and then demodulated—and decoded—on the other end. Though modern broadband connections don’t really work the same way, we kept using the term “modem” because it’s a device people were already familiar with and associated with connecting to the Internet.

When you connect to the Internet you are generally connecting through a digital subscriber line (DSL) connection. DSL is a very high-speed connection that uses the same wires as a regular telephone line. ADSL is just a type of DSL where downloading speed is faster than uploading speed.

Some ISPs offer a modem and router in a single device. That device has the electronics and software in it to provide both functions, acting as a modem that communicates with your ISP and functioning as a router to create a home network. Some ISPs also bundle a phone interface into the same box so you can use their VOIP offerings.

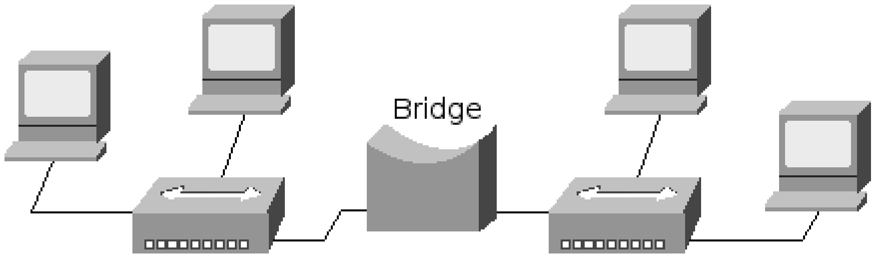
Network Interface Card

Often abbreviated as NIC, an expansion board you insert into a computer so the computer can be connected to a network. These days network connectivity is handled by the motherboard.

WAP

In computer networking, a wireless access point (WAP), is a networking hardware device that allows a Wi-Fi device to connect to a wired network. The WAP usually connects to a router (via a wired network) as a standalone device, but it can also be an integral component of the router itself. An WAP is differentiated from a hotspot, which is the physical location where Wi-Fi access to a WLAN is available.

Bridge
A network bridge is a computer networking device that creates a single aggregate network from multiple communication networks or network segments. Bridging is distinct from routing, as routing allows multiple different networks to communicate independently while remaining separate whilst bridging connects two separate networks as if they are only one network (hence the name "bridging"). In the OSI model, bridging is performed in the first two layers, below the network layer (layer 3). If one or more segments of the bridged network are wireless, the device is known as a wireless bridge and the function as wireless bridging.



Types of Network Bridges
Several different kinds of bridge devices exist, each designed for specific kinds of networks including

- Wireless bridges - support Wi-Fi wireless access points
- Wi-Fi Ethernet bridges - allows connecting Ethernet clients and interfacing them to a local Wi-Fi network, useful for older network devices that lack Wi-Fi capability

Wireless Bridging
Bridging is especially popular on Wi-Fi computer networks. In Wi-Fi, wireless bridging requires access points communicate with each other in a special mode that supports the traffic needing to flow between them. Two access points that support wireless bridging mode work in pairs. Each continues to support their own local networks of connected clients while additionally communicating with the other to handle bridging traffic.

Bridges vs. Repeaters
Bridge and network repeater devices share a similar physical appearance; sometimes, a single unit performs both functions. Unlike bridges, however, repeaters do not perform any traffic filtering and do not join two networks together but instead pass along all traffic they receive. Repeaters serve primarily to regenerate traffic signals so that a single network can reach longer physical distances.

Bridges vs. Switches and Routers
In wired computer networks, bridges serve a similar function as network switches. Traditional wired bridges support one incoming and one outgoing network connection (accessible through a hardware port), whereas switches usually offer four or more hardware ports. Switches are sometimes called multi-port bridges for this reason.
Likewise, bridges lack the intelligence of network routers: Bridges do not understand the concept of remote networks and cannot redirect messages to different locations dynamically but instead support only one outside interface.

Gateway

A gateway is a network node that connects two networks using different protocols together. While a bridge is used to join two similar types of networks, a gateway is used to join two dissimilar networks.
The most common gateway is a router that connects a home or enterprise network to the internet. In most IP-based networks, the only traffic that doesn't go through at least one gateway is traffic flowing among nodes on the same local area network (LAN) segment -- for example, computers connected to the same switch.

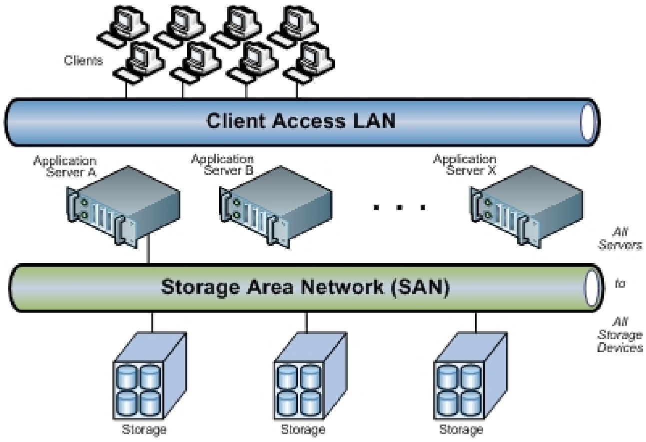
Repeater

In telecommunications, a repeater is an electronic device that receives a signal and retransmits it. Repeaters are used to extend transmissions so that the signal can cover longer distances or be received on the other side of an obstruction.
Some types of repeaters broadcast an identical signal, but alter its method of transmission, for example, on another frequency or baud rate.
There are several different types of repeaters; a telephone repeater is an amplifier in a telephone line, an optical repeater is an optoelectronic circuit that amplifies the light beam in an optical fiber cable; and a radio repeater is a radio receiver and transmitter that retransmits a radio signal.
A broadcast relay station is a repeater used in broadcast radio and television.
When an information-bearing signal passes through a communication channel, it is progressively degraded due to loss of power. For example, when a telephone call passes through a wire telephone line, some of the power in the electric current which represents the audio signal is dissipated as heat in the resistance of the copper wire. The longer the wire is, the more power is lost, and the smaller the amplitude of the signal at the far end. So with a long enough wire the call will not be audible at the other end. Similarly, the farther from a radio station a receiver is, the weaker the radio signal, and the poorer the reception. A repeater is an electronic device in a communication channel that increases the power of a signal and retransmits it, allowing it to travel further. Since it amplifies the signal, it requires a source of electric power.

Storage Area Network (SAN)

A Storage Area Network (SAN) is a specialized, high-speed network that provides block-level network access to storage. Each block in a block-level storage system can be controlled as an individual hard drive, and the blocks are managed by a server operating system. SANs are typically composed of hosts, switches, storage elements, and storage devices that are interconnected using a variety of technologies, topologies, and protocols. SANs may also span multiple sites.

A SAN presents storage devices to a host such that the storage appears to be locally attached. This simplified presentation of storage to a host is accomplished through the use of different types of virtualization.



Network Attached Storage (NAS)

A network-attached storage (NAS) device is a server that is dedicated to nothing more than file sharing. Network-attached storage does not provide any of the activities that a server in a server-centric system typically provides, such as email, authentication or file management. NAS systems are networked appliances which contain one or more storage drives, often arranged into logical, redundant storage containers or RAID.

Network-attached storage removes the responsibility of file serving from other servers on the network. NAS allows more hard disk storage space to be added to a network that already utilizes servers without shutting them down for maintenance and upgrades.

With a NAS device, storage is not an integral part of the server. Instead, in this storage-centric design, the server still handles all of the processing of data but a NAS device delivers the data to the user. A NAS device does not need to be located within the server but can exist anywhere in a LAN and can be made up of multiple networked NAS devices.

SAN vs NAS

Storage area networks (SANs) and network attached storage (NAS) both provide networked storage solutions. A NAS is a single storage device that operates on data files, while a SAN is a local network of multiple devices.

The differences between NAS and SAN can be seen when comparing their cabling and how they're connected to the system, as well as how other devices communicate with them. However, the two are sometimes used together to form what's known as a unified SAN.

A NAS unit includes a dedicated hardware device that connects to a local area network, usually through an Ethernet connection. This NAS server authenticates clients and manages file operations in much the same manner as traditional file servers, through well-established network protocols.

To reduce the costs that occur with traditional file servers, NAS devices generally run an embedded operating system on simplified hardware and lack peripherals like a monitor or keyboard and are instead managed through a browser tool.

A SAN commonly utilizes Fibre Channel interconnects and connects a set of storage devices that are able to share data with one another.

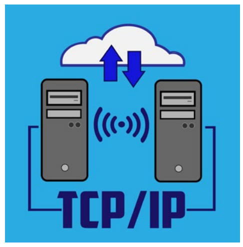
Protocols

A protocol is an agreed-upon format for transmitting data between two devices. The protocol determines the following:

- the type of error checking to be used
- data compression method, if any
- how the sending device will indicate that it has finished sending a message
- how the receiving device will indicate that it has received a message

The benefit of protocols is your computer will work with other computers and devices easily because they all communicate on the same agreed to protocols.

Department of Defence (DoD) TCP/IP



The Department of Defence Four-Layer Model was developed in the 1970s for the DARPA Internetwork Project that eventually grew into the Internet. The core Internet protocols adhere to this model, although the OSI Seven Layer Model is justly preferred for new designs.

TCP/IP DoD Model

Application Layer. This layer sends and receives data for particular applications, such as Domain Name System (DNS), Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP) and Simple Mail Transfer Protocol (SMTP). The application layer itself has layers of protocols within it. For example SMTP encapsulates the Rest for comments (RFC) 2822 message syntax, which encapsulates Multipurpose Internet Mail Extensions (MIME), which encapsulate other formats such as Hypertext Markup Language (HTML).

Transport Layer. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are commonly used transport layer protocols. TCP has emerged as the dominant protocol used for the bulk of internet connectivity due to its ability to break large data sets into individual packets, check for and resend lost packets, and reassemble packets in the correct sequence. But these additional services come at a cost in terms of additional data overhead and delays called latency.

In contrast, UDP just sends the packets, which means that it has much lower bandwidth overhead and latency. With UDP, packets may take different paths between sender and receiver and, as a result, some packets may be lost or received out of order. UDP is an ideal protocol for network applications in which perceived latency is critical, such as in gaming and voice and video communications, which can suffer some data loss without adversely affecting perceived quality.

Internet Layer. The Internet layer in the TCP/IP reference model is responsible for transferring data between the source and destination computers. The Internet layer accepts data from the Transport layer and passes the data to the Network Interface layer. The Internet Protocol unsurprisingly operates at this layer. The following are the functions of the Internet layer:

- Transmitting data to the Network Interface layer.
- Routing the data to the correct destination. This layer takes care of sending the data through the shortest route if more than one route is available. In addition, if a route through which a datagram is to be sent has problems, the datagram is sent through an alternate route.

Network Layer. This layer is where the packets of data are delivered to the computers/users. This data is called frames in this layer. Media Access Control (MAC) addresses are the physical addresses for the network interface cards. These are also added to the frame, Logic Link Control (LLC) adds information about internet protocol is to be used. This is also added to the frame. This is also the physical layer which uses protocol 802.3 protocol for wired conversion of frames and 802.11 for wireless protocol. The most common network is an ethernet network - wired or wireless.

The hardware connected to the Network Interface is listed below:

- Network medium: This is usually twisted pair cabling. In networking today, wireless has become attractive.
- Network interface card (NIC): The NIC has the following addresses:
  - MAC address, which is a physical address
  - IP address, which is a logical address

Transmission Control Protocol/Internet Protocol, TCP/IP is a network standard, that defines how data is routed from one end of a network to the other end, ensuring the data arrives correctly. It describes rules for;

- dividing data into small pieces, called packets
- providing addresses for each packet,
- checking for and detection of errors,
- sequencing packets and,
- regulating the flow of messages along the network

TCP (Transmission Control Protocol)

TCP (Transmission Control Protocol) is a standard that defines how to establish and maintain a network conversation via which application programs can exchange data. TCP works with the Internet Protocol (IP), which defines how computers send packets of data to each other. Together, TCP and IP are the basic rules defining the Internet.

TCP is a connection-oriented protocol, which means a connection is established and maintained until the application programs at each end have finished exchanging messages. It determines how to break application data into packets that networks can deliver, sends packets to and accepts packets from the network layer, manages flow control, and—because it is meant to provide error-free data transmission—handles retransmission of dropped or garbled packets as well as acknowledgement of all packets that arrive. In the Open Systems Interconnection (OSI) communication model, TCP covers parts of Layer 4, the Transport Layer, and parts of Layer 5, the Session Layer.

For example, when a Web server sends an HTML file to a client, it uses the HTTP protocol to do so. The HTTP program layer asks the TCP layer to set up the connection and send the file. The TCP stack divides the file into packets, numbers them and then forwards them individually to the IP layer for delivery. Although each packet in the transmission will have the same source and destination IP addresses, packets may be sent along multiple routes. The TCP program layer in the client computer waits until all of the packets have arrived, then acknowledges those it receives and asks for the retransmission on any it does not (based on missing packet numbers), then assembles them into a file and delivers the file to the receiving application.

Retransmissions and the need to reorder packets after they arrive can introduce latency in a TCP stream. Highly time-sensitive applications like voice over IP (VoIP) and streaming video generally rely on a transport like User Datagram Protocol (UDP) that reduces latency and jitter (variation in latency) by not worrying about reordering packets or getting missing data retransmitted.

IP (The Internet Protocol)

IP - The Internet Protocol is the principal communications protocol in the Internet protocol suite for relaying datagrams across network boundaries. Its routing function enables internetworking, and essentially establishes the Internet.

IP has the task of delivering packets from the source host to the destination host solely based on the IP addresses in the packet headers. For this purpose, IP defines packet structures that encapsulate the data to be delivered. It also defines addressing methods that are used to label the datagram with source and destination information.

Historically, IP was the connectionless datagram service in the original Transmission Control Program introduced by Vint Cerf and Bob Kahn in 1974, which was complemented by a connection-oriented service that became the basis for the Transmission Control Protocol (TCP). The Internet protocol suite is therefore often referred to as TCP/IP.

IP4 vs IP6

The first major version of IP, Internet Protocol Version 4 (IPv4), is the dominant protocol of the Internet. Its successor, Internet Protocol Version 6 (IPv6), has been growing in adoption, reaching almost 25% of all Internet traffic as of October, 2018.

An Internet Protocol address (IP address) is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. An IP address serves two principal functions: host or network interface identification and location addressing.

Internet Protocol Version 4 (IPv4) is the fourth revision of the Internet Protocol and a widely used protocol in data communication over different kinds of networks. IPv4 is a connectionless protocol used in packet-switched layer networks, such as Ethernet. It provides the logical connection between network devices by providing identification for each device.

Internet Protocol version 4 (IPv4) defines an IP address as a 32-bit number. However, because of the growth of the Internet and the depletion of available IPv4 addresses, a new version of IP (IPv6), using 128 bits for the IP address, was developed. IPv6 deployment has been ongoing since the mid-2000s.

IP addresses are usually written and displayed in human-readable notations, such as 172.16.254.1 in IPv4, and 2001:db8:0:1234:0:567:8:1 in IPv6.

Dynamic Host Configuration Protocol (DHCP)

The Dynamic Host Configuration Protocol (DHCP) is a network management protocol used on networks whereby a DHCP server dynamically assigns an IP address and other network configuration parameters to each device on a network, so they can communicate with other IP networks. A DHCP server enables computers to request IP addresses and networking parameters automatically from the Internet service provider (ISP), reducing the need for a network administrator or a user to manually assign IP addresses to all network devices. In the absence of a DHCP server, a computer or other device on the network needs to be manually assigned an IP address, which will not enable it to communicate outside its local subnet.
DHCP can be implemented on networks ranging in size from home networks to large campus networks and regional Internet service provider networks. A router or a residential gateway can be enabled to act as a DHCP server. Most residential network routers receive a globally unique IP address within the ISP network. Within a local network, a DHCP server assigns a local IP address to each device connected to the network.

Domain Name Server

Domain Name Servers (DNS) are the Internet's equivalent of phone book. They maintain a directory of domain names and translate them to Internet Protocol (IP) addresses.
This is necessary because, although domain names are easy for people to remember, computers or machines, access websites based on IP addresses.
Information from all the domain name servers across the Internet are gathered together and housed at the Central Registry. Host companies and Internet Service Providers interact with the Central Registry on a regular schedule to get updated DNS information.
When you type in a web address, e.g., www.jimsbikes.com, your Internet Service Provider views the DNS associated with the domain name, translates it into a machine friendly IP address (for example 216.168.224.70 is the IP for jimsbikes.com) and directs your Internet connection to the correct website.
After you register a new domain name or when you update the DNS servers on your domain name, it usually takes about 12-36 hours for the domain name servers world-wide to be updated and able to access the information. This 36-hour period is referred to as propagation.

Ethernet

Ethernet is the most popular network standard for LANs because it is relatively inexpensive and easy to install and maintain.

Ethernet is based on the bus topology, but Ethernet networks can wired in a star pattern.

Ethernet specifies no central computer or device on the network should control when data can be transmitted; that is each node attempts to transmit data when it determines the network is available to receive communications. If two computers on an Ethernet network attempt to send data at the same time, a collision will occur, and the computers must attempt to send their message again.

802.3

The Ethernet (IEEE 802.3) standard for Local Area Networks (LANs). The 802.3 standard (also called Twisted Pair Ethernet) uses a twisted-pair cable for distances up to 100m.

- 10Base-T - operates at 10 Mbps - uses a twisted-pair cable with maximum lengths of 100 meters.
- 100Base-T – operates at 100 Mbps – often referred to as fast internet – can use – two or four paris of twisted wires or fibre optic cables.
- 1000Base-T – operates 1Gb/s – CAT-5 cabling
- 10GbE – operates 10Gb/s – copper (CAT6a or better) or fibre cabling

WiFi

http://www.howstuffworks.com/wireless-network.htm/printable

If you've been in an airport, coffee shop, library or hotel recently, chances are you've been right in the middle of a wireless network. Many people also use wireless networking, also called WiFi or 802.11 networking, to connect their computers at home, and some cities are trying to use the technology to provide free or low-cost Internet access to residents

WiFi has a lot of advantages. Wireless networks are easy to set up and inexpensive. They're also unobtrusive -- unless you're on the lookout for a place to use your laptop, you may not even notice when you're in a hotspot. In this article, we'll look at the technology that allows information to travel over the air. We'll also review what it takes to create a wireless network in your home.

First, let's go over a few WiFi basics.

What Is WiFi?

A wireless network uses radio waves, just like cell phones, televisions and radios do. In fact, communication across a wireless network is a lot like two-way radio communication. Here's what happens:

1. A computer's wireless adapter translates data into a radio signal and transmits it using an antenna.

2. A wireless router receives the signal and decodes it. The router sends the information to the Internet using a physical, wired Ethernet connection.

802.11

IEEE 802.11 is a set of standards for implementing wireless local area network (WLAN) computer. They are created and maintained by the IEEE LAN/MAN Standards Committee.

IEEE 802.11 is part of the IEEE 802 set of LAN protocols, and specifies the set of media access control (MAC) and physical layer (PHY) protocols for implementing wireless local area network (WLAN). All the 802.11 specifications use the Ethernet protocol and Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) for path sharing.

The protocols are typically used in conjunction with IEEE 802.2, and are designed to interwork seamlessly with Ethernet, and are very often used to carry Internet Protocol traffic.

Building a Wireless Network

If you already have several computers networked in your home, you can create a wireless network with a wireless access point. If you have several computers that are not networked, or if you want to replace your Ethernet network, you'll need a wireless router. This is a single unit that contains:

1. A port to connect to your cable or DSL modem
2. A router
3. An Ethernet hub
4. A firewall
5. A wireless access point

A wireless router allows you to use wireless signals or Ethernet cables to connect your computers to one another, to a printer and to the Internet. Most routers provide coverage for about 100 feet (30.5 meters) in all directions, although walls and doors can block the signal. If your home is very large, you can buy inexpensive range extenders or repeaters to increase your router's range.

Security

Security is an important part of a home wireless network, as well as public WiFi hotspots. If you set your router to create an open hotspot, anyone who has a wireless card will be able to use your signal.

WiFi Protected Access (WPA) is now part of the 802.11i wireless network security protocol. It uses encryption and involves signing on with a password. Most public hotspots are either open or use WPA or 128-bit WEP technology (Wired Equivalency Privacy – an older form of security).

Wireless Application Protocol

Wireless Application Protocol (WAP) is a technical standard for accessing information over a mobile wireless network. The syllabus refers to this as cellular network even though that is an American term that isn't used in Australia. A WAP browser is a web browser for mobile devices such as mobile phones that uses the protocol.
Before the introduction of WAP, mobile service providers had limited opportunities to offer interactive data services, but needed interactivity to support Internet and Web applications such as:

- Email by mobile phone
- Tracking of stock-market prices
- Sports results
- News headlines
- Music downloads

Bluetooth

A Bluetooth connection is wireless and automatic, and it has a number of interesting features that can simplify our daily lives. Bluetooth takes small-area networking to the next level by removing the need for user intervention and keeping transmission power extremely low to save battery power.

The big draws of Bluetooth are that it is wireless, inexpensive and automatic.

There are other ways to get around using wires, including infrared communication. Infrared is used in most television remote control systems. Infrared communications are fairly reliable and don't cost very much to build into a device, but there are a couple of drawbacks. First, infrared is a "line of sight" technology. For example, you have to point the remote control at the television or DVD player to make things happen. The second drawback is that infrared is almost always a "one to one" technology. You can send data between your desktop computer and your laptop computer, but not your laptop computer and your PDA at the same time.

RFID

Radio-frequency identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to objects. The tags contain electronically stored information. Passive tags collect energy from a nearby RFID reader's interrogating radio waves. Active tags have a local power source (such as a battery) and may operate hundreds of meters from the RFID reader. Unlike a barcode, the tag need not be within the line of sight of the reader, so it may be embedded in the tracked object. RFID is one method of automatic identification and data capture (AIDC).
RFID tags are used in many industries. For example, an RFID tag attached to an automobile during production can be used to track its progress through the assembly line; RFID-tagged pharmaceuticals can be tracked through warehouses; and implanting RFID microchips in livestock and pets enables positive identification of animals. Tags can also be used in shops to expedite checkout, and to prevent theft by customers and employees.
Since RFID tags can be attached to cash, clothing, and possessions, or implanted in animals and people, the possibility of reading personally-linked information without consent has raised serious privacy concerns. These concerns resulted in standard specifications development addressing privacy and security issues with the use on-chip cryptography methods for untraceability, tag and reader authentication, and over-the-air privacy.

Carrier Sense Multiple Access/Collision Detect (CSMA/CD) (Wired)

Carrier-sense multiple access with collision detection (CSMA/CD) is a media access control method used most notably in early Ethernet technology for local area networking. It uses carrier-sensing to defer transmissions until no other stations are transmitting. This is used in combination with collision detection in which a transmitting station detects collisions by sensing transmissions from other stations while it is transmitting a frame. When this collision condition is detected, the station stops transmitting that frame, transmits a jam signal, and then waits for a random time interval before trying to resend the frame.

CSMA/CD is a modification of pure carrier-sense multiple access (CSMA). CSMA/CD is used to improve CSMA performance by terminating transmission as soon as a collision is detected, thus shortening the time required before a retry can be attempted.

CSMA/CD is specified in the IEEE 802.3 standard.

Carrier-sense Multiple Access with Collision Avoidance (CSMA/CA) (Wireless)

Carrier-sense multiple access with collision avoidance (CSMA/CA) in computer networking, is a network multiple access method in which carrier sensing is used, but nodes attempt to avoid collisions by beginning transmission only after the channel is sensed to be "idle". When they do transmit, nodes transmit their packet data in its entirety.

It is particularly important for wireless networks, where the collision detection of the alternative CSMA/CD is not possible due to wireless transmitters desensing (degradation in sensitivity due to noise sources) their receivers during packet transmission.

CSMA/CA is unreliable due to the hidden node problem. In wireless networking, the hidden node problem or hidden terminal problem occurs when a node can communicate with a wireless access point (AP), but cannot directly communicate with other nodes that are communicating with that AP. This leads to difficulties in medium access control sublayer since multiple nodes can send data packets to the AP simultaneously, which creates interference at the AP resulting in neither packet getting through.

CSMA/CA is a protocol that operates in the Data Link Layer (Layer 2) of the OSI model.

Parity Bit

A parity bit, or check bit, is a bit added to a string of binary code to ensure that the total number of 1-bits in the string is even or odd. Parity bits are used as the simplest form of error detecting code.

There are two variants of parity bits: even parity bit and odd parity bit.

In the case of even parity, for a given set of bits, the occurrences of bits whose value is 1 is counted. If that count is odd, the parity bit value is set to 1, making the total count of occurrences of 1s in the whole set (including the parity bit) an even number. If the count of 1s in a given set of bits is already even, the parity bit's value is 0. So basically there is an extra bit that changes so that if you count all the ones it should always add up to an even number.

In the case of odd parity, the coding is reversed. For a given set of bits, if the count of bits with a value of 1 is even, the parity bit value is set to 1 making the total count of 1s in the whole set (including the parity bit) an odd number. If the count of bits with a value of 1 is odd, the count is already odd so the parity bit's value is 0. So basically there is an extra bit that changes so that if you count all the ones it should always add up to an odd number.

So let's say we are sending 7 bits of data. We add one extra bit to ensure the number of ones always add up to an even number with even parity or an odd number with odd parity.

7 bits of data	**(count of 1 bits)**	**8 bits including parity**	
		even parity	**odd parity**
0000000	0	00000000	10000000
1010001	3	11010001	01010001
1101001	4	01101001	11101001
1111111	7	11111111	01111111

Checksum

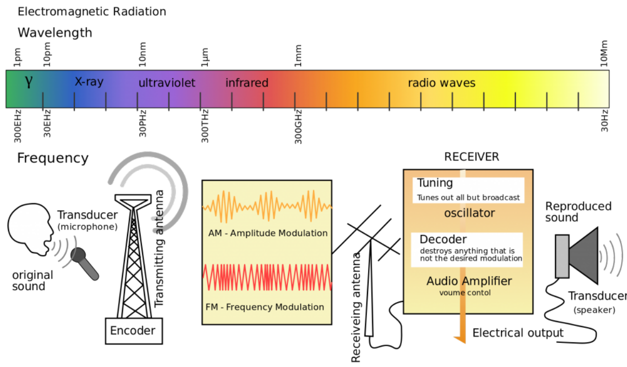
A checksum is an error-detection method where the transmitter computes a numerical value according to the number of set or unset bits in a message and sends it along with each message frame. At the receiver end, the same checksum function (formula) is applied to the message frame to retrieve the numerical value. If the received checksum value matches the sent value, the transmission is considered to be successful and error-free. A mismatched checksum shows that the entire message has not been transmitted.

Transmission

Radio Waves

Radio waves are a type of electromagnetic radiation with wavelengths in the electromagnetic spectrum longer than infrared light. Radio waves have frequencies as high as 300 gigahertz (GHz) to as low as 30 hertz (Hz).

Broadcast Radio



I bet your wondering why we are covering this in Computer Science. Me too but it's in the syllabus so here we go.

Free music, news, and chat wherever you go! Until the Internet came along, nothing could rival the reach of radio—not even television. A radio is a box filled with electronic components that catches radio waves sailing through the air and converts them back into sounds your ears can hear. Radio was first developed in the late-19th century and reached the height of its popularity several decades later. Although radio broadcasting is not quite as popular as it once was, the basic idea of wireless communication remains hugely important: in the last few years, radio has become the heart of new technologies such as wireless Internet, mobile phones, and RFID (radio frequency identification) chips. Meanwhile, radio itself has recently gained a new lease of life with the arrival of better-quality digital radio sets.

Line of Sight

Line of sight is a type of propagation that can transmit and receive data only where transmit and receive stations are in view of each other without any sort of an obstacle between them. FM radio, microwave and satellite transmission are examples of line-of-sight communication.

Long-distance data communication is more effective through wireless networks but geographical obstacles and the curvature of the earth bring limitations to line-of-sight transmission. However, these issues can generally be mitigated through planning, calculations and the use of additional technologies.

For example, mobile phones use a modified line-of-sight transmission, which is made possible through a combination of effects like diffraction, multipath reflection, local repeaters and rapid handoff.

AM

AM stations were the earliest broadcasting stations to be developed. AM refers to amplitude modulation, a mode of broadcasting radio waves by varying the amplitude of the carrier signal in response to the amplitude of the signal to be transmitted. The medium-wave band is used worldwide for AM broadcasting.

The signal is subject to interference from electrical storms (lightning) and other electromagnetic interference (EMI). One advantage of AM radio signal is that it can be detected (turned into sound) with simple equipment. If a signal is strong enough, not even a power source is needed; building an unpowered crystal radio receiver was a common childhood project in the early decades of AM broadcasting.

FM

FM refers to frequency modulation. FM radio came to overcome the problem of radio-frequency interference (RFI), which plagued AM radio reception. At the same time, greater fidelity was made possible by spacing stations further apart in the radio frequency spectrum.

Satellite Transmission

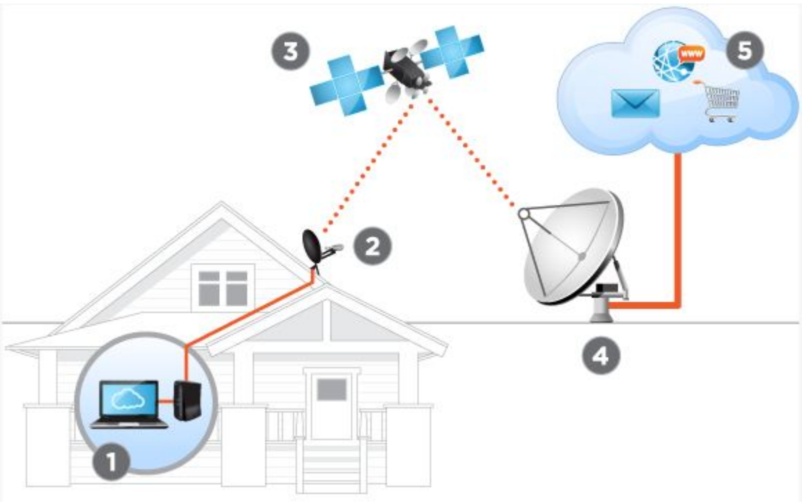
A communication satellite receives a radio wave signal from a ground station on earth, then bounces the signal back down to a different ground station on earth. The satellite not only bounces it back down, but is like a repeater in that it amplifies the signal, so it can keep going further. This means that a radio wave can go much further in distance than just 'line of sight' on the ground. Remote areas often use satellites to get communications because it is costly with other choices.

Satellite Networking

Satellite internet equipment is made up of three main components: a geostationary satellite in space, a satellite dish mounted on the premises and a modem that transmits internet signals from the dish to the computer or LAN.

Satellite internet is obtained through a satellite broadband provider. The provider provides the satellite dish and required modem.

How Does it Work?



Satellite internet uses geostationary satellites, rather than telephone lines or cable systems, to deliver an internet signal directly. Geostationary means the satellite remains in a fixed position relative to a point on the Earth by matching its rotation velocity as the satellite orbits the planet. The satellite provides two-way data communications between a satellite dish mounted on the exterior of the premises and the hub of the satellite internet provider. When a webpage is accessed, the request is sent from the computer to the satellite. The satellite then transmits the signal to the satellite internet service's hub, where the specific website you've requested is located and beamed back to the satellite. The data is then sent to the dish and the site loads on the computer through a modem connected to the dish. While the internet signal travels a staggering distance back and forth, it only takes a handful of seconds for this entire process to occur. With satellite internet, you can connect all the computers and internet-enabled devices in a LAN to the web.

In rare instances, extreme weather may affect the signal.

It may not be the fastest internet connection in the world, but with speeds approximately ten times faster than dial-up, satellite internet is connecting the most rural parts of the country to the World Wide Web.

Microwave Transmission

Microwave communication is method of wirelessly sending data. It is very similar to radio technology. Microwaves are right next to radio waves on the electromagnetic spectrum.

What is microwave used for?

Modern microwave systems are used in telephone networks (both wireless and wireline) and ISPs. They're used by power utilities to remotely manage the power grid. They're used by public safety agencies (ex. police, fire) for remote monitoring and management. Many industries used microwave, and they do it for one important reason:

Microwave is a powerful tool wherever data must be transmitted a long distance without physical wires. This is common in rural mountainous regions, where installing physical transmission lines is difficult and expensive. Microwave systems minimize installations and maintenance, as a signal microwave tower can transmit data across dozens of miles.

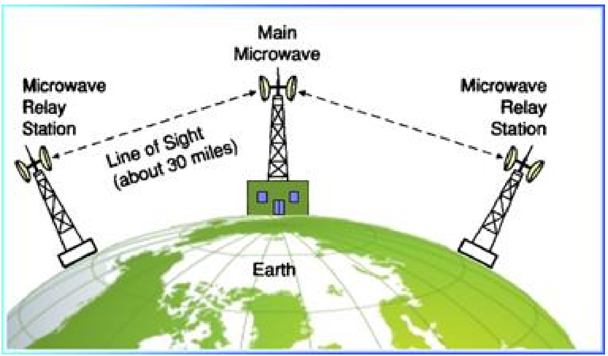
Radio waves are normally omnidirectional. When an antenna transmits radio waves, they are propagated in all directions. This means that the sending and receiving antennas do not have to be aligned. The omnidirectional characteristics of radio waves make them useful for multicasting, in which there is one sender but many receivers. Our AM and FM radio stations, mobile phones and televisions are examples of multicasting.

Electronic waves with frequencies between 1 GHz to 300 GHz are normally called microwaves. Unlike radio waves, microwaves are unidirectional, in which the sending and receiving antennas need to be aligned.

Microwaves propagation is line-of-sight therefore towers with mounted antennas need to be in direct sight of each other. Due to the unidirectional property of microwaves, a pair of antennas can be placed aligned together without interfering with another pair of antennas using the same frequency.

High-frequency microwaves cannot penetrate walls. This is why receiving antennas cannot be placed inside buildings.

Why are microwave transmitters mounted on towers?



Just like visible light, microwaves are blocked by obstacles. They need a clear path to reach their destination. Mounting transmitters and receivers high on a tower offers a clear line-of-sight to the next tower. Also, taller towers reduce the impact of the Earth's curvature. Taller towers can be spaced farther apart and still see one another.

Microwave communication is the sending of signals via radio using a series of microwave towers. It's a form of "line of sight" communication. There must be nothing obstructing transmission of data between these towers. That's why microwave towers are frequently placed on mountaintops. When positioned on a tall peak, a tower has lines of sight to valleys below on all sides and to other mountaintop towers. The increase elevation also reduces the impact of the Earth's curvature on line of sight.

Cellular Network

A cellular network or mobile network (as we call it in Australia) is a communication network where the last link is wireless. The network is distributed over land areas called cells, each served by at least one fixed-location transceiver, but more normally three cell sites or base transceiver stations. These base stations provide the cell with the network coverage which can be used for transmission of voice, data, and other types of content. A cell typically uses a different set of frequencies from neighbouring cells, to avoid interference and provide guaranteed service quality within each cell.

When joined together, these cells provide radio coverage over a wide geographic area. This enables a large number of portable transceivers (e.g., mobile phones, tablets and laptops equipped with mobile broadband modems, pagers, etc.) to communicate with each other and with fixed transceivers and telephones anywhere in the network, via base stations, even if some of the transceivers are moving through more than one cell during transmission.

Cellular networks offer a number of desirable features:

- More capacity than a single large transmitter, since the same frequency can be used for multiple links as long as they are in different cells
- Mobile devices use less power than with a single transmitter or satellite since the cell towers are closer
- Larger coverage area than a single terrestrial transmitter, since additional cell towers can be added indefinitely and are not limited by the horizon

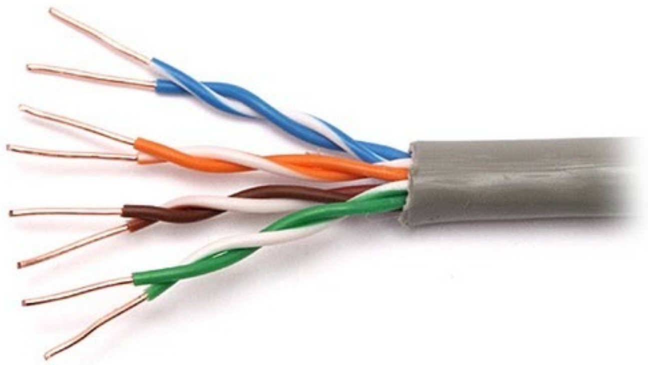
Major telecommunications providers have deployed voice and data cellular networks over most of the inhabited land area of Earth. This allows mobile phones and mobile computing devices to be connected to the public switched telephone network and public Internet.

Twister Pair

Twisted pair is the ordinary copper wire that connects home and many business computers to the telephone company. To reduce crosstalk or electromagnetic induction between pairs of wires, two insulated copper wires are twisted around each other. Each connection on twisted pair requires both wires. Since some telephone sets or desktop locations require multiple connections, twisted pair is sometimes installed in two or more pairs, all within a single cable. For some business locations, twisted pair is enclosed in a shield that functions as a ground. This is known as shielded twisted pair (STP). Ordinary wire to the home is unshielded twisted pair (UTP).

Unshielded Twisted Pair (UTP)

Unshielded twisted pair (UTP) is a ubiquitous type of copper cabling used in telephone wiring and local area networks (LANs). There are five types of UTP cables -- identified with the prefix CAT, as in category -- each supporting a different amount of bandwidth.



Alternatives to UTP cable include coaxial cable and fiber optic cable. There are benefits and tradeoffs to each type of cabling, but broadly speaking, most enterprises favour UTP cable due to its low cost and ease of installation.

How UTP cables work: Twisted pair design

Inside a UTP cable is up to four twisted pairs of copper wires, enclosed in a protective plastic cover, with the greater number of pairs corresponding to more bandwidth. The two individual wires in a single pair are twisted around each other, and then the pairs are twisted around each other, as well. This is done to reduce crosstalk and electromagnetic interference, each of which can degrade network performance. Each signal on a twisted pair requires both wires.

Shielded Twisted Pair (STP)

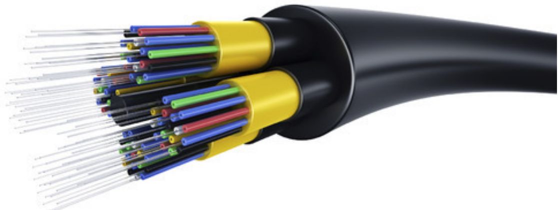
Shielded twisted pair is a special kind of copper telephone wiring used in some business installations. An outer covering or shield is added to the ordinary twisted pair telephone wires; the shield functions as a ground. Twisted pair is the ordinary copper wire that connects home and many business computers to the telephone company. To reduce crosstalk or electromagnetic induction between pairs of wires, two insulated copper wires are twisted around each other. Each signal on twisted pair requires both wires. Since some telephone sets or desktop locations require multiple connections, twisted pair is sometimes installed in two or more pairs, all within a single cable. Shielded twisted pair is often used in business installations. The more common kind of wire that is installed to your home is unshielded twisted pair.

Ethernet Cabling

An Ethernet cable is one of the most popular forms of network cable used on wired networks. Ethernet cables connect devices on local area networks such as PCs, routers and switches. Unshielded twisted-pair (UTP) cable is the most common networking media.

UTP Category	Purpose	Transfer Rate
Category 5	Data	100 Mbps
Category 5e	Data	1 Gbps
Category 6	Data	1/10 Gbps

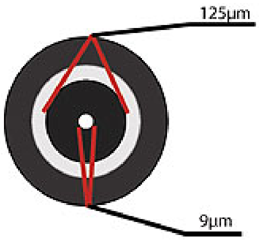
Fibre Optic Cable



A fibre optic cable is a network cable that contains strands of glass fibres inside an insulated casing. These cables are designed for long distance and very high bandwidth (gigabit speed) network communications.

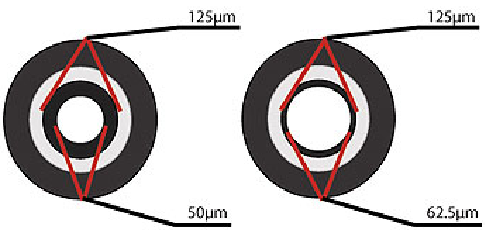
Fibre optic cables carry communication signals using pulses of light. While expensive, these cables are increasingly being used instead of traditional copper cables, because fibre offers more capacity, is less susceptible to electrical interference, low attenuation loss over long distances and is an electrical insulator (so can be laid down along with high voltage cables).

Single Mode Fibre Optic Cable



Single Mode fibre optic cable has a small diametral core that allows only one mode of light to propagate. Because of this, the number of light reflections created as the light passes through the core decreases, lowering attenuation and creating the ability for the signal to travel further. This application is typically used in long distance, higher bandwidth runs by Telcos, CATV companies, and Colleges and Universities.

Multimode Fibre Optic Cable



Multimode fibre optic cable has a large diametral core that allows multiple modes of light to propagate. Because of this, the number of light reflections created as the light passes through the core increases, creating the ability for more data to pass through at a given time. Because of the high dispersion and attenuation rate with this type of fibre, the quality of the signal is reduced over long distances. This application is typically used for short distance, data and audio/video applications in LANs. RF broadband signals, such as what cable companies commonly use, cannot be transmitted over multimode fibre. Multimode cables are considered to be the "domestic" fibre as they are used for local-area network.

Network Performance

Bandwidth

Bandwidth describes the maximum data transfer rate of a network or Internet connection. It measures how much data can be sent over a specific connection in a given amount of time. For example, a gigabit Ethernet connection has a bandwidth of 1,000 Mbps, (125 megabytes per second). An Internet connection via cable modem may provide 25 Mbps of bandwidth.

While bandwidth is used to describe network speeds, it does not measure how fast bits of data move from one location to another. Since data packets travel over electronic or fibre optic cables, the speed of each bit transferred is negligible. Instead, bandwidth measures how much data can flow through a specific connection at one time.

When visualizing bandwidth, it may help to think of a network connection as a tube and each bit of data as a grain of sand. If you pour a large amount of sand into a skinny tube, it will take a long time for the sand to flow through it. If you pour the same amount of sand through a wide tube, the sand will finish flowing through the tube much faster. Similarly, a download will finish much faster when you have a high-bandwidth connection rather than a low-bandwidth connection.

Data often flows over multiple network connections, which means the connection with the smallest bandwidth acts as a bottleneck. Generally, the Internet backbone and connections between servers have the most bandwidth, so they rarely serve as bottlenecks. Instead, the most common Internet bottleneck is your connection to your ISP.

NOTE: Bandwidth also refers to a range of frequencies used to transmit a signal. This type of bandwidth is measured in hertz and is often referenced in signal processing applications.

Network Design

Network design factors like category of the caballing, wifi hardware, the topology of network (bus vs star) can hugely effect speed of network.

There are many parts of the chain in a network can cause a bottle neck. Where the faster technology is having to wait for the slower technology to pass on the data.

Data Collisions

A data collision is the result of simultaneous data packet transmission between two or more network domain devices or nodes. Data collision packets break into fragments and retransmitted.

A node checks for network availability when attempting to transmit a data packet to another node. More than one network node may check the network simultaneously to verify that there are no active network transmissions and that signals are not in a channel transmission state. This mechanism is known as channel sensing.

Channel sensing facilitates simultaneous packet transmission by multiple nodes, which causes data collision and network noise. Other nodes that sense this noise use a backoff algorithm to avoid data collision.

The best remedy for collisions is to use a switched environment (star network using switches).

Excess Broadcast Traffic

Excessive broadcasts over a network will cause slow down. How can we reduce the impacts of network traffic?

- **Monitor the network traffic**. This provides insight about where possible congestion may lie. What this means is that you can make network adjustments to problem areas. The only way to understand if slow network speeds are caused by congestion is to monitor.
- **Network Segmentation.** The process is to divide your network into smaller sub-networks. The benefit of segmenting your network is to group assets and groups into specific areas.
- **Network Redundancy.** What redundancy does is ensure that if one router or network route becomes congested, a second route is used in its place. This would make sure there is no packet loss or a time out due to congestion.
- **Prioritization of traffic.** Preventing congestion can be changing a router setting. It can ensure there is no one application that is a bandwidth-hog. Also, by adjusting router settings, you can reduce congestion before it begins.

There are many more strategies that and reduce the effect of excessive traffic but in the context of this course you don’t need a deep understanding of this issue.

LAN, WLAN and WANs

A local area network (LAN) is a computer network that interconnects computers within a limited area such as a residence, school, laboratory, university campus or office building.

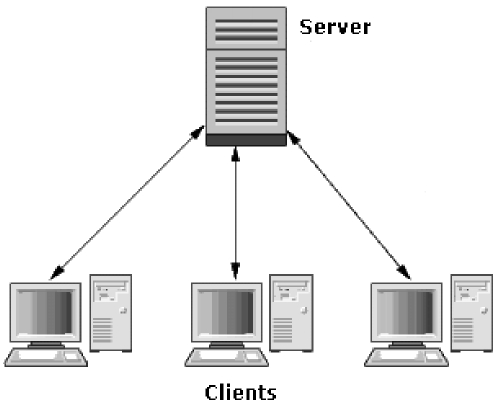
A wireless LAN (WLAN) is a wireless computer network that links two or more devices using wireless communication to form a local area network (LAN) within a limited area such as a home, school, computer laboratory, campus, office building etc.

A wide-area network (WAN) is any telecommunications network or computer network that extends over a large geographical distance/place.

Network Architectures

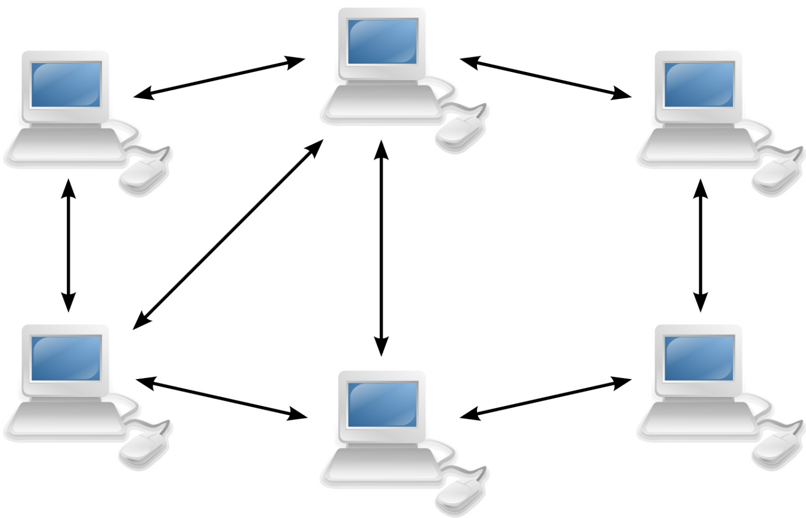
Client/Server

On a client/server network, one ***or more*** computers act as a server, and the other computers on the network request services from the server. The other computers and mobile devices on the network are referred to as clients.



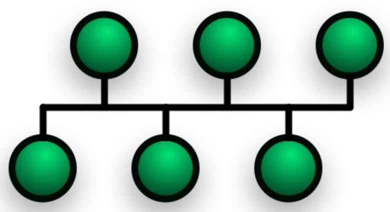
Peer-to-Peer

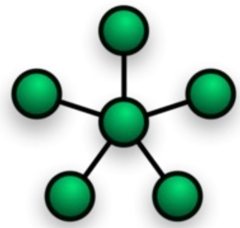
Peer-to peer network is a simple inexpensive network that typically connects fewer than 10 computers. Each computer called a peer, has equal responsibilities and capabilities, sharing hardware (such as a printer), data etc.



Network Topologies

A network topology refers to the layout of the computers and devices in a communications network. Three commonly used network topologies are bus, ring and star. Networks usually use combinations of these topologies.





Bus Network

A bus network consists of a single central cable, to which all computers and other devices (nodes) connect. The bus is the physical cable that connects the computers and other devices. It transmits data, instructions, and information in both directions. When a sending device transmits data, the address of the receiving device is included with the transmission so the data is routed to the appropriate receiving device.

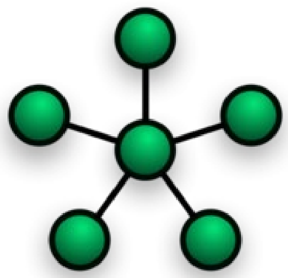
Bus networks are popular on LANs because they are inexpensive and easy to install. Computers and other devices can be attached and detached at any point on the bus without disturbing the rest of the network. The main disadvantage is the limit in size of the bus.

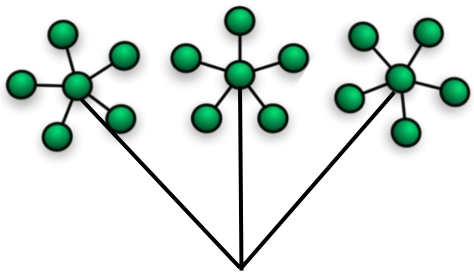
Star Network

On a star network, all of the computers and devices (nodes) on the network connect to a central device, thus forming a star. Two types of device that provide a common central connection point on the network are a hub and a switch.

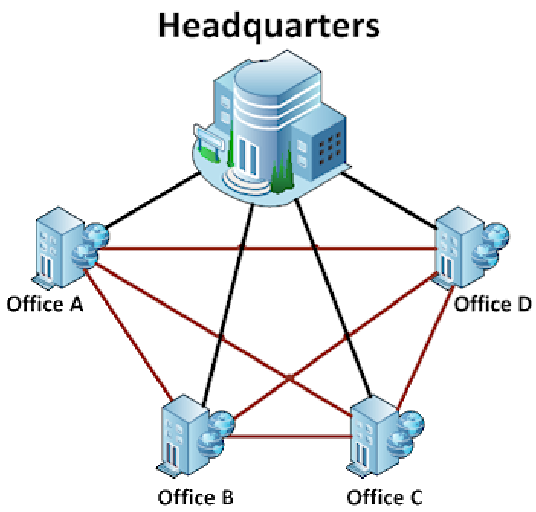
Star networks are fairly easy to install and maintain. Nodes can be added and removed from the network no disruption to the network.

If one node fails, only that node is affected. If the hub/switch fails the entire network is inoperable. Most star networks consist of multiple switches though an in that case only the part of the network connected to that switch will fail.





WAN



A WAN (wide area network) is a communications network that spans a large geographic area such as across cities, states, or countries. They can be private to connect parts of a business or they can be more public to connect smaller networks together.

The easiest way to understand what a WAN is to think of the internet as a whole, which is the world's largest WAN. The internet is a WAN because through the use of ISPs, it connects lots of smaller local area networks (LANs) or metro area networks (MANs).

On a smaller scale, a business may have a WAN that's comprised of cloud services, its headquarters, and smaller branch offices. The WAN in this case would be used to connect all of those sections of the business together.

No matter what the WAN joins together or how far apart the networks are, the end result is always intended to allow different smaller networks from different locations to communicate with one another.

WANs are often used by larger corporations or organizations to facilitate the exchange of data, and in a wide variety of industries corporations with facilities at multiple locations have embraced WANs. Increasingly, however, even small businesses are utilizing WANs as a way of increasing their communications capabilities.

Although WANs serve a purpose similar to that of local area networks (LANs), WANs are structured and operated quite differently. The user of a WAN usually does not own the communications lines that connect the remote computer systems; instead, the user subscribes to a service through a telecommunications provider. Unlike LANs, WANs typically do not link individual computers, but rather are used to link LANs. WANs also transmit data at slower speeds than LANs. WANs are also structurally similar to metropolitan area networks (MANs), but provide communications links for distances greater than 50 kilometers.

WANs have existed for decades, but new technologies, services, and applications have developed over the years to dramatically increase their efficacy for business. WANs were originally developed for digital leased-line services carrying only voice, rather than data. As such, they connected the private branch exchanges (PBXs) of remote offices of the same company. WANs are still used for voice services, but today they are used more frequently for data and image transmission (such as video conferencing). These added applications have spurred significant growth in WAN usage, primarily because of the surge in LAN connections to the wider networks.

How WANs Work

WANs are either point-to-point, involving a direct connection between two sites, or operate across packet-switched networks, in which data are transmitted in packets over shared circuits. Point-to-point WAN service may involve either analog dial-up lines, in which a modem is used to connect the computer to the telephone line, or dedicated leased digital telephone lines, also known as "private lines." Analog lines, which may be either part of a public-switched telephone network or leased lines, are suitable for batch data transmissions, such as nonurgent order entry and point-of-sale transactions. Dedicated digital phone lines permit uninterrupted, secure data transmission at fixed costs.

Point-to-point WAN service providers include both local telephone companies and long-distance carriers. Packet-switched network services are typically chosen by organizations which have low volumes of data or numerous sites, for which multiple dedicated lines would be too expensive. Depending on the service, WANs can be used for almost any data-sharing purpose for which LANs can be used. Slower transmission speeds, however, may make some applications less practical for WANs. The most basic uses of WANs are for electronic mail and file transfer, but WANs can also permit users at remote sites to access and enter data on a central site's database, such as instantaneously updating accounting records. New types of network-based software that facilitate productivity and production tracking, such as groupware and work-flow automation software, can also be used over WANs. Using groupware, workers at dispersed locations can more easily collaborate on projects. WANs also give remote offices access to a central office's other data communications services, including the Internet.

Activities

1. What is the difference between a router and a switch?
2. What is the difference between a router and a bridge?
3. What is the difference between SAN and NAS?
4. What are the differences (I'm going on a bit on a theme here) between the DoD TCP/IP four layers and the OSI seven layered model
5. What is the difference between IP4 and IP6? What do each look like?
6. What protocol can be used to automatically assign and configure an IP address?
7. What is the benefit of a Client Server network? (You will need to research)
8. What security encryption is part of 802.11?
9. What is the difference between CSMA/CD and CSMA/CA?
10. What does parity bit do?
11. Why do we need so many mobile phone towers?
12. What is the difference between shielded and unshielded twisted pair wiring.
13. What is broadcast traffic in regard to a computer network?
14. What's a WLAN?
15. What hardware is needed for a WiFi Network.
16. What is the similarities and differences between Bluetooth and WiFi.
17. Compare the transfer rates between WiFi, Ethernet and Firbre Optic.
18. Using the Cisco Symbols (see powerpoint) create a networking diagram for the following scenarios.
    a. A home network with a PC with wired connection, laptop, tablet and phones with wireless and all with internet access.
    b. A small business with a client server network with several wired connected PCs, a shared printer/photocopier, internet and local cloud storage.

References

1. https://www.lifewire.com/definition-of-pan-817889
2. https://en.wikipedia.org/wiki/Router_(computing)
3. https://www.howtogeek.com/234233/whats-the-difference-between-a-modem-and-a-router/
4. https://en.wikipedia.org/wiki/Repeater
5. https://www.slashfilm.com/bill-murray-repeating-groundhog-day/
6. https://en.wikipedia.org/wiki/Internet_protocol_suite#Comparison_of_TCP/IP_and_OSI_layering
7. https://learningnetwork.cisco.com/thread/66569
8. https://en.wikipedia.org/wiki/Radio-frequency_identification
9. https://www.techopedia.com/definition/5367/internet-protocol-version-4-ipv4
10. https://www.techopedia.com/definition/5368/internet-protocol-version-6-ipv6
11. http://www.networksolutions.com/support/what-is-a-domain-name-server-dns-and-how-does-it-work/
12. http://searchnetworking.techtarget.com/definition/TCP
13. https://www.techopedia.com/definition/28503/dns-server
14. https://www.snia.org/education/storage_networking_primer/san/what_san
15. https://techterms.com/definition/parity_bit
16. https://en.wikipedia.org/wiki/Carrier-sense_multiple_access_with_collision_detection
17. https://searchnetworking.techtarget.com/definition/Unshielded-Twisted-Pair
18. https://en.wikipedia.org/wiki/Line-of-sight_propagation
19. https://www.agilebroadcast.com.au/types-of-radio-broadcasting/
20. https://tenplay.com.au/channel-ten/family-feud
21. https://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol
22. https://internetofthingsagenda.techtarget.com/definition/gateway
23. http://edtechchris.com/2017/02/15/centos-7-firewall-cli-commands/
24. https://s3-ap-southeast-2.amazonaws.com/wc-prod-pim/JPEG_300x300/DLDWR116_d_link_3g_and_4g_router_n300.jpg
25. https://techterms.com/definition/bandwidth
26. http://compass.xbox.com/assets/ea/f9/eaf9cfd8-0c54-4bc9-9d88-c3f93955120f.jpg?n=X1S_gallery_1056x594_10.jpg
27. https://www.techopedia.com/definition/1792/checksum
28. https://www.lifewire.com/san-vs-nas-818005
29. https://en.wikipedia.org/wiki/Cellular_network
30. https://www.jamaudio.com/media/catalog/product/cache/7/image/1100x/9df78eab33525d08d6e5fb8d27136e95/h/x/hx-hp420-rd-3l_2.png
31. http://www.three.co.uk/static/images/device_pages/MobileVersion/Apple/iPhone_SE/Space_Grey/desktop/1.jpg
32. https://c.s-microsoft.com/en-au/CMSImages/surface_L_Pivot-Blue-pos2_V2.jpg?version=b1e71f22-8b53-6e70-531d-2cf47d8d129e
33. https://www.lifewire.com/wide-area-network-816383
34. http://www.encyclopedia.com/science-and-technology/computers-and-electrical-engineering/computers-and-computing/wide-area
35. http://computer.howstuffworks.com/wimax.htm
36. http://www.toptenreviews.com/services/articles/satellite-internet-101-how-does-it-work/
37. http://www.dpstele.com/network-monitoring/microwave/index.php
38. http://www.datacom2u.com/ENWirelessMedia.php
39. https://en.wikipedia.org/wiki/File_Transfer_Protocol

40. http://searchnetworking.techtarget.com/definition/TCP
41. http://www.ryanseslow.com/wp-content/uploads/2016/11/tcp-ip.jpg
42. https://www.cisco.com/c/en/us/products/wireless/small-business-500-series-wireless-access-points/index/_jcr_content/Grid/subcategory_atl/layout-subcategory-atl/anchor_info/image.img.jpg/1480395496493.jpg
43. https://www.lifewire.com/how-network-bridges-work-816357
44. https://www.multicominc.com/training/technical-resources/single-mode-vs-multi-mode-fiber-optic-cable/

Data Security

Security

Firewalls

A firewall is simply a program or hardware device that filters the information coming through the Internet connection into your private network or computer system. If an incoming packet of information is flagged by the filters, it is not allowed through.

A network's firewall builds a bridge between an internal network that is assumed to be secure and trusted, and another network, such as the Internet, that is not assumed to be secure and trusted

Without a firewall in place, a computer can be accessible to anyone on the Internet.

A good firewall allows you to choose access permissions for each program on your computer. When one of these programs tries to contact the outside world, your firewall will block the attempt and give you a warning unless it recognizes the program and verifies that you have given it permission to make that sort of connection. This is largely to prevent existing malware from spreading viruses or inviting hackers into your computer. In this regard, a firewall provides both a second line of defence and an early-warning system that might help you recognize when your computer's security is being threatened.

Hardware Firewalls

Hardware firewalls is hardware that sits between a computer and the Internet. They typically use packet filtering, which means they scan packet headers to determine their source, origin, destination addresses and check with the existing user defined rules to make an allow/deny decision.

Key advantages of hardware firewall.

1. **Speed**: Hardware firewalls are tailored for faster response times, so it can handle more traffic loads.
2. **Security**: A firewall with its own operating system is less prone for attacks. This in turn reduces the security risk and in addition, hardware firewalls have enhanced security controls.
3. **No Interference**: Since the hardware firewall is an isolated network component, it can be managed better, and does not load or slowdown other applications. The firewall can be moved, shutdown, or reconfigured with minimal interference to the network.

If you are using multiple servers for the hosting purpose, hardware firewall might be a better option, since it’s able to handle the traffic for multiple servers and can differentiate between which traffic is allowed to one server but not to another.

Passwords

http://www.google.com/goodtoknow/online-safety/passwords/

Passwords are the first line of defence against cyber criminals. It's crucial to pick strong passwords that are different for each of your important accounts and it is good practice to update your passwords regularly. Follow these tips to create strong passwords and keep them secure.

Use a unique password for each of your important accounts like email and online banking

Choosing the same password for each of your online accounts is like using the same key to lock your home, car and office – if a criminal gains access to one, all of them are compromised. So don't use the same password for an online newsletter as you do for your email or bank account. It may be less convenient, but picking multiple passwords keeps you safer.

Use a long password made up of numbers, letters and symbols

The longer your password is, the harder it is to guess. So make your password long to help keep your information safe. Adding numbers, symbols and mixed-case letters makes it harder for would-be snoops or others to guess or crack your password. Please don't use '123456' or 'password,' and avoid using publicly available information like your phone number in your passwords. It's not very original, and it isn't very safe!

Try using a phrase that only you know

One idea is to think of a phrase that only you know, and make it be related to a particular website to help you remember it. For your email you could start with "My friends Tom and Jasmine send me a funny email once a day" and then use numbers and letters to recreate it. "MfT&Jsmafe1ad" is a password with lots of variations. Then repeat this process for other sites.

Set up your password recovery options and keep them up-to-date

If you forget your password or get locked out, you need a way to get back into your account. Many services will send an email to you at a recovery email address if you need to reset your password, so make sure your recovery email address is up-to-date and an account you can still access.

Viruses

http://computer.howstuffworks.com/virus.htm

There are many different forms of electronic infection. The most common are:

- **Viruses:** A virus is a small piece of software that piggybacks on real programs. For example, a virus might attach itself to a program such as a spreadsheet program. Each time the spreadsheet program runs, the virus runs, too, and it has the chance to reproduce (by attaching to other programs) or wreak havoc.
- **E-mail viruses:** An e-mail virus travels as an attachment to e-mail messages, and usually replicates itself by automatically mailing itself to dozens of people in the victim's e-mail address book. Some e-mail viruses don't even require a double-click -- they launch when you view the infected message in the preview pane of your e-mail software.
- **Trojan horses:** A Trojan horse is simply a computer program. The program claims to do one thing (it may claim to be a game) but instead does damage when you run it (it may erase your hard disk). Trojan horses have no way to replicate automatically.
- **Worms:** A worm is a small piece of software that uses computer networks and security holes to replicate itself. A copy of the worm scans the network for another machine that has a specific security hole. It copies itself to the new machine using the security hole, and then starts replicating from there, as well.

Anti-virus software

Antivirus or anti-virus software sometimes known as anti-malware software, is computer software used to prevent, detect and remove malicious software.

Antivirus software was originally developed to detect and remove computer viruses, hence the name. However, with the proliferation of other kinds of malware, antivirus software started to provide protection from other computer threats. In particular, modern antivirus software can protect from: malicious Browser Helper Objects, browser hijackers, ransomware, keyloggers, backdoors, rootkits, trojan horses, worms, malicious LSPs, dialers, fraudtools, adware and spyware. Some products also include protection from other computer threats, such as infected and malicious URLs, spam, scam and phishing attacks, online identity (privacy), online banking attacks, social engineering techniques, Advanced Persistent Threat, botnets, DDoS attacks.

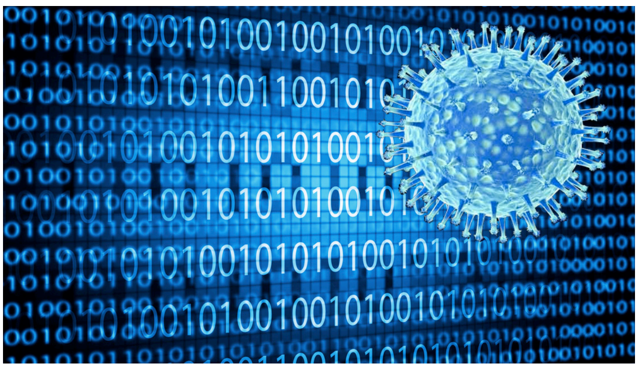
There are several methods which antivirus engine can use to identify malware:

Signature-based detection: is the most common method. To identify viruses and other malware, the antivirus engine compares the contents of a file to its database of known malware signatures. These are like a fingerprint for the virus.

Heuristic-based detection: is generally used together with signature-based detection. It detects malware based on characteristics typically used in known malware code.

Behavioural-based detection: is similar to heuristic-based detection and used also in Intrusion Detection System. The main difference is that, instead of characteristics hardcoded in the malware code itself, it is based on the behavioural fingerprint of the malware at run-time. Clearly, this technique is able to detect (known or unknown) malware only after they have starting doing their malicious actions.

Cloud-based detection: identifies malware by collecting data from protected computers while analysing it on the provider's infrastructure, instead of performing the analysis locally. This is usually done by capturing the relevant details about the file and the context of its execution on the endpoint, and providing them to the cloud engine for processing. The local antivirus agent only needs to perform minimal processing. Moreover, the vendor's cloud engine can derive patterns related to malware characteristics and behaviour by correlating data from multiple systems. In contrast, other antivirus components base decisions mostly on locally observed attributes and behaviours. A cloud-based engine allows individual users of the antivirus tool to benefit from the experiences of other members of the community.



Authentication

The process of identifying an individual usually based on a username and password. In security systems, authentication is distinct from authorization, which is the process of giving individuals access to system objects based on their identity. Authentication merely ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual.

User authentication occurs within most human-to-computer interactions other than guest accounts, automatically logged-in accounts and kiosk computer systems. Generally, a user has to enter or choose an ID and provide their password to begin using a system. User authentication authorizes human-to-machine interactions in operating systems and applications as well as both wired and wireless networks to enable access to networked and Internet-connected systems, applications and resources.

Machines need to authorize their automated actions within a network too. Machine authentication can be carried out with machine credentials much like a users' ID and password only submitted by the device in question. They can also use digital certificates issued and verified by a Certificate Authority (CA) as part of a public key infrastructure to prove identification while exchanging information over the Internet, like a type of digital password.

Encryption

http://computer.howstuffworks.com/encryption.htm/

When we use the Internet, we're not always just clicking around and passively taking in information, such as reading news articles or blog posts -- a great deal of our time online involves sending others our own information. Ordering something over the Internet, whether it's a book, a CD or anything else from an online vendor, or signing up for an online account, requires entering in a good deal of sensitive personal information. A typical transaction might include not only our names, e-mail addresses and physical address and phone number, but also passwords and personal identification numbers (PINs).

The incredible growth of the Internet has excited businesses and consumers alike with its promise of changing the way we live and work. It's extremely easy to buy and sell goods all over the world while sitting in front of a laptop. But security is a major concern on the Internet, especially when you're using it to send sensitive information between parties.

Let's face it, there's a whole lot of information that we don't want other people to see, such as:

- Credit-card information
- Social Security numbers
- Private correspondence
- Personal details
- Sensitive company information
- Bank-account information

Information security is provided on computers and over the Internet by a variety of methods. A simple but straightforward security method is to only keep sensitive information on removable storage media like portable flash memory drives or external hard drives. But the most popular forms of security all rely on encryption, the process of encoding information in such a way that only the person (or computer) with the key can decode it.

Computer encryption is based on the science of cryptography, which has been used as long as humans have wanted to keep information secret. Before the digital age, the biggest users of cryptography were governments, particularly for military purposes.

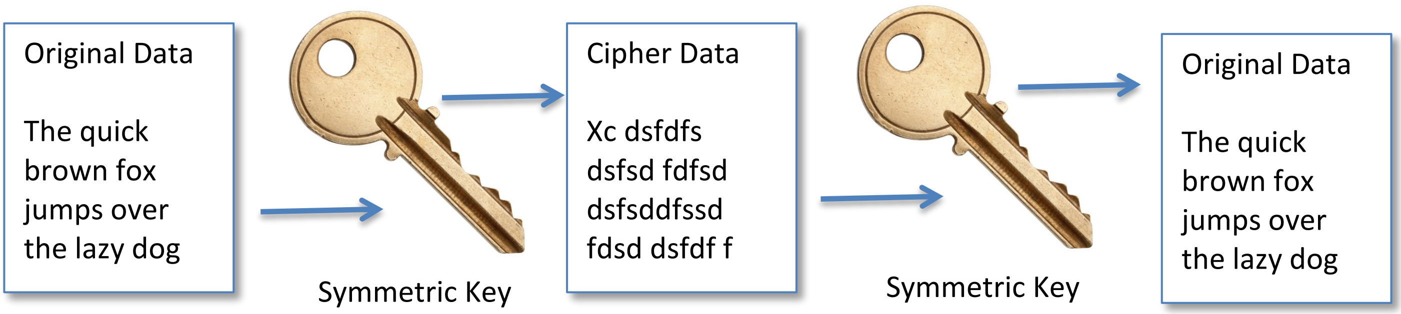
Most forms of cryptography in use these days rely on computers, simply because a human-based code is too easy for a computer to crack.

Computer encryption systems generally belong in one of two categories:

- Symmetric-key encryption
- Public-key encryption

Symmetric Key

In symmetric-key encryption, each computer has a secret key (code) that it can use to encrypt a packet of information before it is sent over the network to another computer. Symmetric-key requires that you know which computers will be talking to each other so you can install the key on each one. Symmetric-key encryption is essentially the same as a secret code that each of the two computers must know in order to decode the information. The code provides the key to decoding the message.



The first major symmetric algorithm developed for computers in the United States was the Data Encryption Standard (DES), approved for use in the 1970s. The DES uses a 56-bit key.

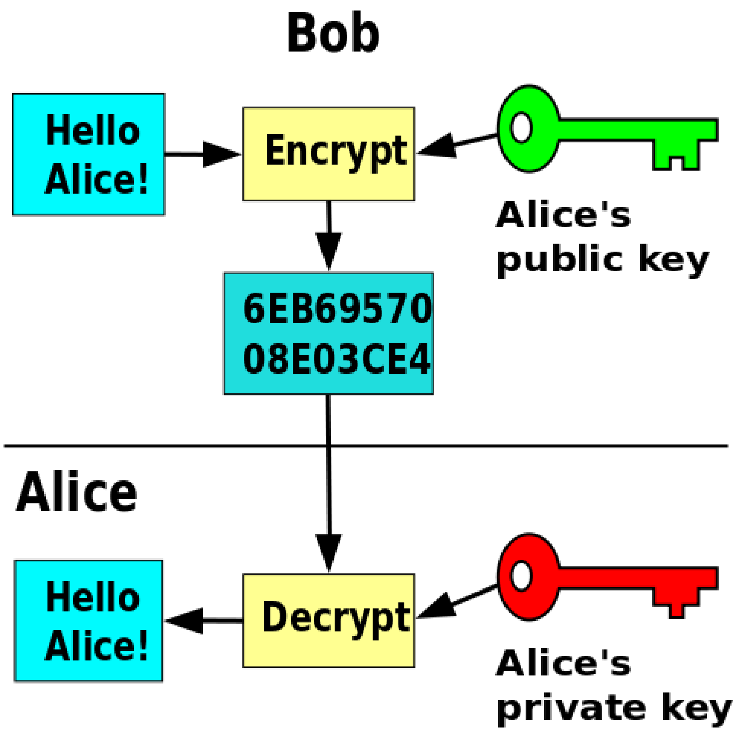
Because computers have become increasingly faster since the '70s, security experts no longer consider DES secure -- although a 56-bit key offers more than 70 quadrillion possible combinations (70,000,000,000,000,000), an attack of brute force (simply trying every possible combination in order to find the right key) could easily decipher encrypted data in a short while. DES has since been replaced by the Advanced Encryption Standard (AES), which uses 128-, 192- or 256-bit keys. Most people believe that AES will be a sufficient encryption standard for a long time coming: A 128-bit key, for instance, can have more than 300,000,000,000,000,000,000,000,000,000,000,000 key combinations.

Public Key Encryption

Public-key cryptography, also known as asymmetric cryptography, requires two separate keys, one of which is secret (or private) and one of which is public. Although different, the two parts of this key pair are mathematically linked.

How does it work?

Lets say Bob wants to send a message to Alice. Bob will use Alice's public key (which she can give him) to encrypt a message. Bob will then send the encrypted message where Alice will use her private key to decrypt. It works because the public key is basically an encrypted version of her private key – they are linked mathematically but you can't work out the private key from the public one.



Denial-of-Service Attack

In computing, a denial-of-service attack (DoS attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled.

In a distributed denial-of-service attack (DDoS attack), the incoming traffic flooding the victim originates from many different sources. This effectively makes it impossible to stop the attack simply by blocking a single source.

A DoS or DDoS attack is analogous to a group of people crowding the entry door of a shop, making it hard for legitimate customers to enter, disrupting trade.

Criminal perpetrators of DoS attacks often target sites or services hosted on high-profile web servers such as banks or credit card payment gateways. Revenge, blackmail and activism can motivate these attacks.

Backdoor

A backdoor is a method, often secret, of bypassing normal authentication or encryption in a computer system, a product, or a device (e.g. a home router). Backdoors are often used for securing remote access to a computer or obtaining access to plaintext in cryptographic systems.

A backdoor may take the form of a hidden part of a program, a separate program, code in the firmware of the hardware, or parts of an operating system such as Windows. Trojan horses can be used to create vulnerabilities in a device. A Trojan Horse may appear to be an entirely legitimate program, but when executed, it enacts an activity that may install a backdoor. Although some are secretly installed, other backdoors are deliberate and widely known. These kinds of backdoors have "legitimate" uses such as providing the manufacturer with a way to restore user passwords. The backdoor may be used to gain access to passwords, delete data on hard drives, or transfer information within the cloud.

Many systems that store information within the cloud fail to create accurate security measures. If many systems are connected within the cloud, hackers can gain access to all other platforms through the most vulnerable system.

Default passwords (or other default credentials) can function as backdoors if they are not changed by the user. Some debugging features can also act as backdoors if they are not removed in the release version.

IP Spoofing

IP Spoofing is a technique used to gain unauthorized access to machines, whereby an attacker illicitly impersonates another machine by manipulating IP packets. IP Spoofing involves modifying the packet header with a forged (spoofed) source IP address, a checksum, and the order value. Internet is a packet switched network, which causes the packets leaving one machine may be arriving at the destination machine in different order. The receiving machine resembles the message based on the order value embedded in the IP header. IP spoofing involves solving the algorithm that is used to select the order sent values, and to modify them correctly.

This process usually starts by identifying your host and finding the IP address trusted by your host so that you can send data packets and the host will see them as originating from a trusted IP address but that's not the case.

Hackers use IP spoofing to perform activities that are malicious and illegal. Some of the activities that can be performed include Service denial and man in the middle attacks. These two malicious acts are used by hackers to cause drama or havoc over the internet while hiding their identity.

Phishing

Phishing is the fraudulent act of acquiring private and sensitive information, such as credit card numbers, personal identification and account usernames and passwords. Using a complex set of social engineering techniques and computer programming expertise, phishing websites lure email recipients and Web users into believing that a spoofed website is legitimate and genuine. In actuality, the phishing victim later discovers his personal identity and other vital information have been stolen and exposed.

Phishing uses link manipulation, image filter evasion and website forgery to fool Web users into thinking that a spoofed website is genuine and legitimate. Once the user enters vital information, he immediately becomes a phishing victim.

Fortunately, phishing victimization is preventable. The following security precautions are recommended:

- Use updated computer security tools, such as anti-virus software, spyware and firewall.
- Never open unknown or suspicious email attachments.
- Never divulge personal information requested by email, such as your name or credit card number.
- Double check the website URL for legitimacy by typing the actual address in your Web browser.
- Verify the website's phone number before placing any calls to the phone number provided via email.

References

1. http://clipartall.com/subimg/robber-robber-clipart-1600_1532.png
2. http://science.howstuffworks.com/
3. http://bte.co.ug/wp-content/uploads/2016/01/digital-signature1.jpg
4. http://static.bbc.co.uk/history/img/ic/640/images/resources/topics/enigma.jpg
5. http://dreamicus.com/data/key/key-01.jpg
6. https://en.wikipedia.org/wiki/Public-key_cryptography
7. https://fcw.com/~/media/GIG/FCWNow/Topics/Identity/login%20screen.jpg
8. https://img04.deviantart.net/397d/i/2013/357/d/4/spy_vs_spy_wallpaper_by_etschannel-d6z5pdc.png
9. https://bobcares.com/blog/hardware-vs-software-firewall-a-brief-comparison/
10. https://en.wikipedia.org/wiki/Denial-of-service_attack
11. https://en.wikipedia.org/wiki/Backdoor_(computing)
12. https://www.iplocation.net/ip-spoofing
13. https://www.techopedia.com/definition/4049/phishing