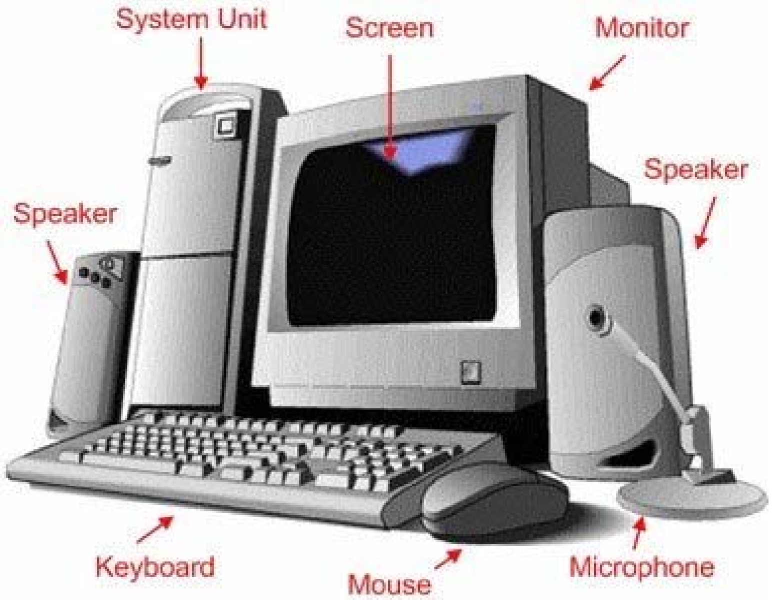
1.1 THE COMPONENTS OF A COMPUTER SYSTEM AND MODES OF USE

Types of hardware

Hardware

- Is the collective name given to all the physical,mechanical,electronic devices that make up a computer system.
- Some examples of computer hardware are shown in figure below:



- Basically these devices may be split up into:

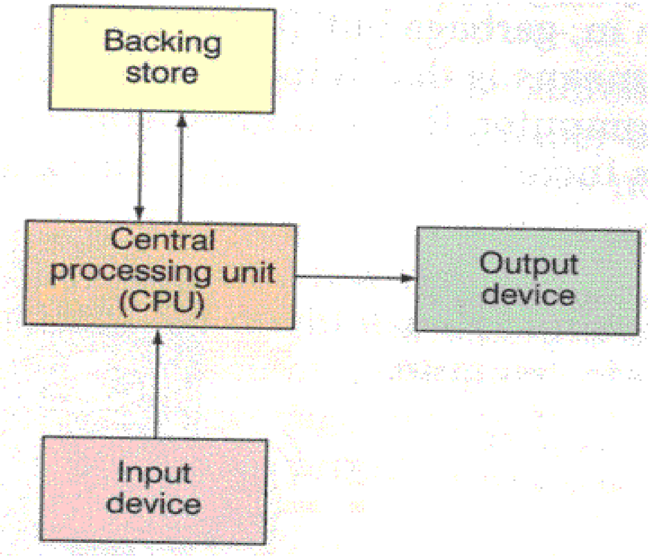
- **INPUT DEVICES**: which are used to get the data into the computer.

- **CPU (CENTRAL PROCESSING UNIT)**: - which is the brain of the computer.

- **BACKING STORAGE**: - which consists of the disk drives used to store data when the power is switched off.

- **OUTPUT DEVICES**: which include such units as printers and VDUs (visual display unit), which are used to provide output in the form of printouts, screen displays, etc.

- All devices external to the CPU and main memory are known as peripherals.
- A simple computer system is shown below:



- Below is a listing of different hardware devices in the computer and different methods of determining additional information about the hardware device.

1. CD-ROM drive
2. Floppy disk drive
3. Hard disk drive
4. Memory
5. Modem
6. Monitor
7. \Motherboard
8. \Network Card
9. Printer
10. Scanner
11. Sound Card
12. Video Card

Electronic point of sale (EPOS)

Definition

Self-contained, computerized equipment that performs all tasks of a store checkout counter. It allows payments by bank or credit cards, verifies transactions, provides sales reports, coordinates inventory data, and performs several other services normally provided by employees

Some hardware used at a EPOS include:

Input: Barcode reader

Output: Receipt printer

EFTPOS, *Electronic Funds Transfer* *Point of Sale*, refers to the technology that allows a retailer to directly debit a customer's bank account by using a debit card. The debit card, generally the same as an ATM card, is swiped through a reading device just like a credit card. The customer must enter his or her PIN number, generally requested once the amount of the sale has been entered into the EFTPOS device.

How EFTPOS works for stock control.

When goods are added to stock in the warehouse, its details are read using a barcode reader and added to stock.

At the time of sale, the barcodes are read again and reduced from the stock. Once the stock is below the required level, the user will be notified that stock is going low.

Other non-common input and output devices include:

Graphics plotter: it’s used by designers, architects and artists to output very large printouts.

Magnetic strip reader: used to read data from cards like debit and credit cards.

Optical mark reader: used to read marks from paper. For example multiple answer questions that is ticked or shaded.

Magnetic Ink Character Reader (MICR): used to read special characters especially in cheques.

- **OCR** devices are particularly well suited to applications such as credit card billing systems where data such as customer account number and amount due is encoded by the computer on a **turnaround document** - that is, a document which is produced by the computer and sent to the customer, who returns the bottom tear - off slip with their payment.

- A clerk checks that the payment enclosed is the same as the payment due, and manually keys in the amount paid if it is different.

- The machine readable document is then input tot he computer with no further keying required.

SCANNERS

- **Scanners** come in a variety of shapes and sizes, from **bar-code** scanners which scan bar codes on labels in supermarkets or libraries, to hand-held and page scanners which scan a complete page, pixel by pixel.

- A scanner works by shining a bright light onto the image being scanned while the scan head moves from the top tothe bottom of the document at an even rate. As it moves over each 'line' of the image, the scan head collects data by measuring the intensity of light that is reflected back from the document.

- Each scanned line therefore results in a stream of data which the scanner translates into digital information, with a certain number of bits representing each tiny area on the scanned picture. For a black-and-white image, only one bit will be required; for 256 shades of grey, 8 bits will be required.

- The resolution of the scanner is measured in dots per inch (dpi) along the x and y axes; the higher the resolution, the sharper the image, but the more memory it will occupy.

- With appropriate OCR software, scanned text can be turned into a form that the computer can process, making this an alternative to keyboard entry for long text documents, or for forms where hand-printed characters are written in specified boxes.

MAGNETIC INK CHARACTER RECOGNITION (MICR)

- **MICR** is used mainly by banks for processing cheques. Special characters encoded along the bottom of a cheque are used to identify the bank number and the customer's account number.

- When the cheque is processed, the amount is also encoded by a bank operator, using a special ink containing ferric oxide which can be magnetised during processing.

- MICR has several advantages for cheque processing:

1. speed: over 1000 cheques per minute can be processed;

2. smudged cheques can still be read;

3. the characters are hard to forge.

OPTICAL MARK RECOGNITION (OMR)

- An optical mark reader is a scanning device which can detect marks in preset positions on a special form.

- OMR is frequently used to score multiple-choice tests and for market research questionnaires.

MAGNETIC STRIPES AND SMART CARDS

- Cards with magnetic stripes are widely used as charge cards, telephone cards, railway tickets etc. A bank card used for withdrawing cash from an automated teller machine (ATM), for example, will have encoded in the stripe:

1. the customer's account number;

2. the personal identification number (PIN) in encrypted form;

3. the bank's sort code;

4. the customer's withdrawal limit;

5. the amount withdrawn in the last time period (e.g. day)

- To use the ATM, the customer inserts the card and is then requested to enter the PIN, which is checked against the encrypted PIN on the card and the PIN held in the customer's record on the bank's computer.

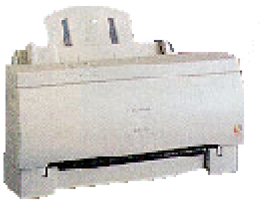
- The customer then presses a key to indicate the type of transaction desired, and if cash is required, types in the amount. This is checked against the account balance on the bank's computer and the information held on the card.

- A smart card can hold more information than a magnetic stripe card and contains a processing chip, making it extremely hard to forge or duplicate. They may eventually replace magnetic stripe cards.

Output Devices

The purpose of the output devices is to translate data and information from electrical impulses to human-readable format.

The output device which is necessary for the computer to display messages to the user is a monitor. If we want to keep the copy of the work on paper, we use printers. Plotters are devices that are more suitable for the output of high quality graphics.



VISUAL DISPLAY UNIT (VDU)

- A VDU has three basic attributes: size, colour and resolution. It has its own fixed amount of RAM associated with it to store the image being displayed on the screen, and the amount of RAM it has will determine the resolution and the maximum number of colours that can be displayed.

PRINTER

There are many different kinds of printer, inlcuding the following: -

1. **Laser printer.**

These printers give high-quality output are quiet in operation and relatively inexpensive to run, although they are fairly expensive to purchase. Major running costs include toner (powdered ink), occasional drum replacement and possibly a maintenance contract.

2. Inkjet printer.

These are inexpensive and can give good quality colour images. An inkjet printer would be a suitable choice for a school to purchase, for example for students projects, posters etc. The ink is expensive and special paper is required for best results.

3. Dot-matrix printer.

These are among the cheapest printers, and have the advantage that carbon copies can be produced, but the quality is not as good and they are noisy.

PLOTTER

Plotters are used to produce extremely accurate, high-quality drawings in, for example engineering or architectural aaplications. The two basic types of pen plotter are the drum plotter and the flatbed plotter, each of which have one or more pens that move over the surface of the paper under computer control to produce the drawing.

1.2. Types of Software

Software

- Is the general term for any program or set of instructions used to make a computer perform a task.

- Every processor, whether it is inside a mainframe computer, individual workstation or pocket camera, has to be given instructions in the form of a program before it can do anything at all.

Categories of Software:

1) **General purpose applications software**

2) **Special purpose applications software**

3) **Programming languages, compilers and interpreters**

4) **Operating systems**

5) **Utility programs**

GENERAL PURPOSE APPLICATIONS SOFTWARE

This category includes the common software packages that are found on most desktop computers most of which you will almost certainly use during your course. Examples:

- Word processing software (e.g. Word, WordPerfect) for producing and saving well laid out documents such as business letters, technical manuals, books, memos and reports.
- Desktop publishing software (e.g. PageMaker, Ventura, FrameMaker, PagePlus) for producing among other things like magazines, advertisements, newsletters and books.
- Spreadsheet Packages (e.g. MS Excel, Lotus 123, SuperCalc) for working with numbers, producing accounts and tabulated numerical information of all types.
- Database Packages (e.g. MS Access, Paradox, FoxPro) for information storage and retrieval. Databases are used in thousands of different applications from airline booking systems, mail order and invoicing systems to keeping tabs on distances flown racing pigeons.
- Graphics Packages (e.g. Paint, PaintBrush, CorelDraw) for producing artwork, 3 dimensional images, special textual effects.
- Computer-aided-design Packages (e.g. Turbo CAD, AutoCAD) for producing accurate engineering or architectural drawings.
- Multimedia Authoring Tools (e.g. Authorwave, Macromedia Director) which combine text, graphics, animation, sound and video for presentations, games, interactive tutorials etc.
- Telecommunications software (e.g. Internet Solution) which enables you to send and receive data over a wide area network via a modem, access the internet, send and receive electronic mail, browse the worldwide web.
- Expert Systems Software (e.g. Crystal, ELSIE) which can be programmed with the facts and rules about a certain 'domain' or field of knowledge such as

geological data around known oil fields The system can then be used to predict the likelihood of finding oil in a new location, given its geological profile.

SPECIAL PURPOSE APPLICATION SOFTWARE

- When an organisation wants to computerise some aspect of its business, it is often possible to buy an 'off the shelf' package to do more or less exactly what it wants.

- There are literally thousands of specialist applications readily available to perform such tasks for example:

- keeping business accounts
- stock control
- payroll
- general practice management and appointments
- theater booking

- The alternative to buying one of these packages is either to buy a database package and build a customised application to suit your exact requirements; OR write a suite of programs using a language such as PASCAL or C to perform the required tasks.

PROGRAMMING LANGUAGES

- All computers process instructions using machine code, which uses codes to represent basic operations such as load, move, add, subtract, compare and so on.
- Early programmers in 1940s and 1950s had to program using these numeric codes, a slow and tedious process.
- Later, programs were written which translated statements such as 'Add 1 to counter' into machine code so that programmers could write in something more closely resembling English, which was easier to learn, faster and quicker to debug.
- There are 100s of programming 'languages' and each of these languages comes with its own 'translation' program (one of two types: compilers and interpreter) which will take the code you write and turn it into machine executable form.
- Programmers who invent computer languages and write compilers or interpreters for them are called as system programmers.
- Bill Gates is the head of Microsoft who invented BASIC in 1971.

OPERATING SYSTEMS

- While you are using a computer to write a program or do some word processing, the operating system is working away in the background, following instructions that determine where in memory your program or document is stored, where on disk it will be saved and what to do if for example you press <Ctrl><Break> to interrupt the execution of your program.
- No computer can operate without an operating system.

- The essential parts of the operating system are loaded from disk into memory as soon as you switch your computer on. Other parts of the operating system (such as some utility programs) will be copied into memory when required.
- MS-DOS (Microsoft Disk Operating System) is a character-based operating system.
- The user has to communicate with the operating system via a command interface, which means you have to know exactly what to type to get it to do what you want, like display a list of which files are in a particular directory on your disk.

UTILITY PROGRAMS

- Utility programs perform common tasks such as formatting disks, copying and deleting files or repairing damaged files.
- Many utility programs are supplied with the operating system and can be called directly from it.
- Other utility programs can be purchased from a number of software manufacturers, such as

- PC Tools: a collection of utilities for PC system maintenance, management and protection.

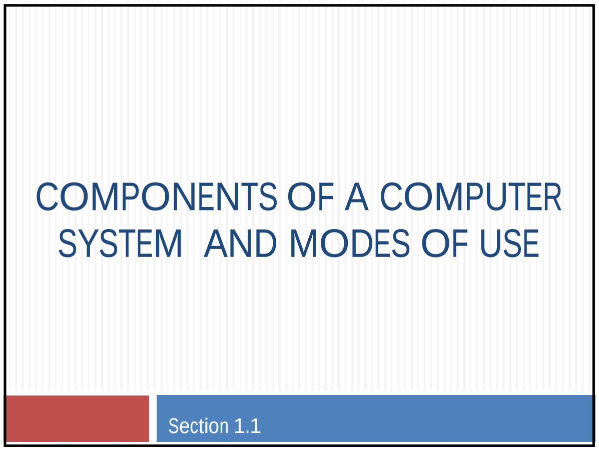
- Dr Solomon's Anti-Virus Toolkit: a virus detection and protection system.

- After DarkScreenSaver: screen displays that appear when you leave the computer alone for a few minutes.

Resources:

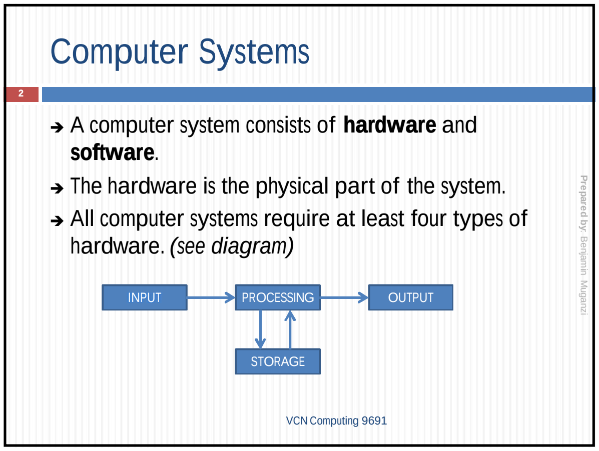
(1) Stephen D., [***Information Systems for you, 3rd. Edition***], Nelson Thornes Ltd., 2001.

(2) [***http://www.computerhope.com/index.htm***], Computer Hope (tm), 1998.



COMPONENTS OF A COMPUTER SYSTEM AND MODES OF USE

Section 1.1

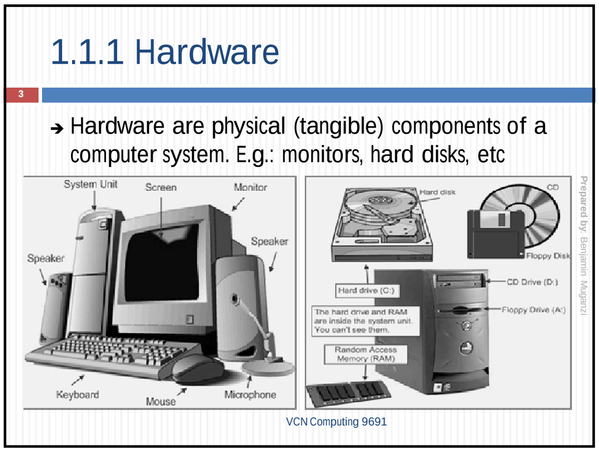


Computer Systems

2

- A computer system consists of **hardware** and **software**.
- The hardware is the physical part of the system.
- All computer systems require at least four types of hardware. *(see diagram)*

VCN Computing 9691

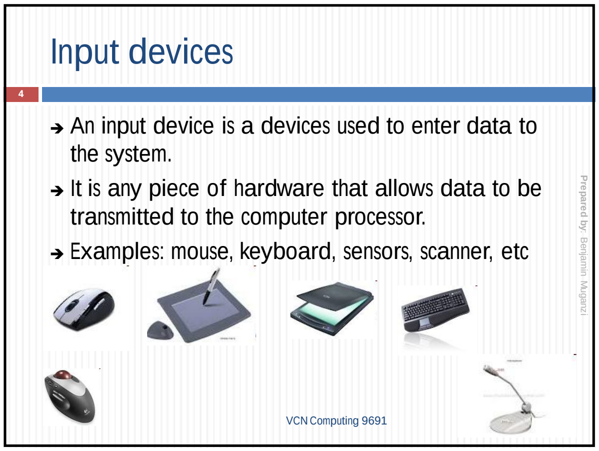


1.1.1 Hardware

3

- Hardware are physical (tangible) components of a computer system. E.g.: monitors, hard disks, etc

VCN Computing 9691



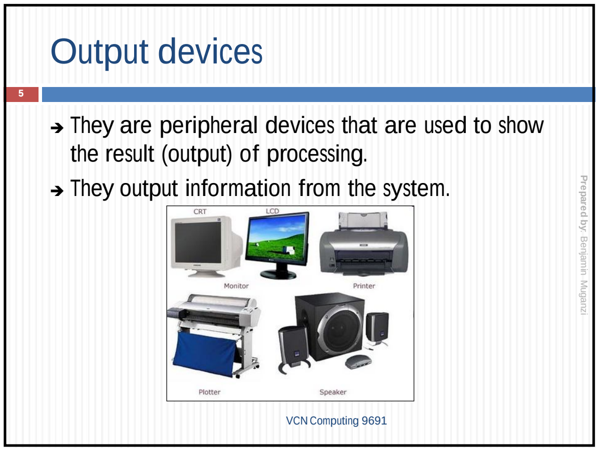
Input devices

4

- An input device is a devices used to enter data to the system.
- It is any piece of hardware that allows data to be transmitted to the computer processor.
- Examples: mouse, keyboard, sensors, scanner, etc

Prepared by: Benjamin Muganzi

VCN Computing 9691



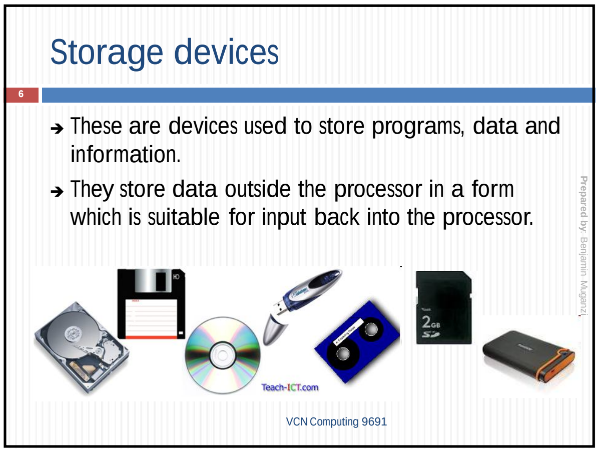
Output devices

5

- They are peripheral devices that are used to show the result (output) of processing.
- They output information from the system.

Prepared by: Benjamin Muganzi

VCN Computing 9691



Storage devices

6

- These are devices used to store programs, data and information.
- They store data outside the processor in a form which is suitable for input back into the processor.

Prepared by: Benjamin Muganzi

VCN Computing 9691

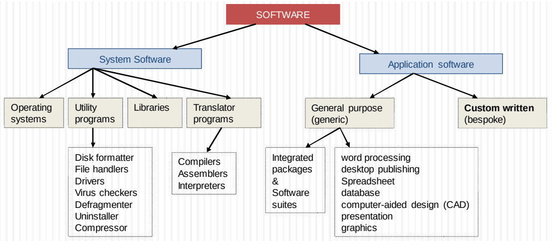
1.1.2 Software

7

- Software are the logical components of a computer system.
- This is the various sets of instructions which tell the system how to do things.
- Without software, hardware cannot function

Types of software

8



SOFTWARE

System Software

- Operating systems
- Utility programs
  - Disk formatter
File handlers
Drivers
Virus checkers
Defragmenter
Uninstaller
Compressor
- Libraries
- Translator programs
  - Compilers
Assemblers
Interpreters

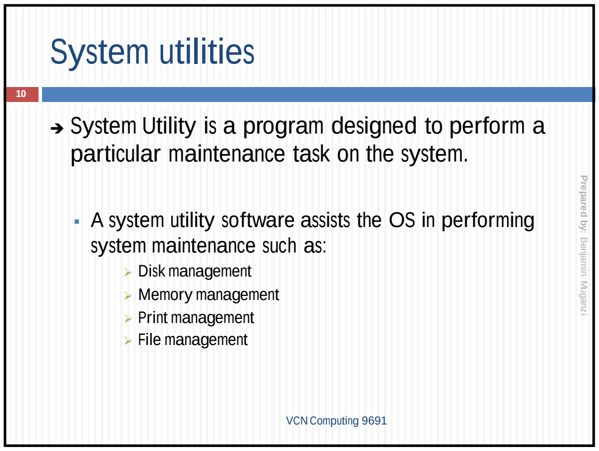
Application software

- General purpose (generic)
  - Integrated packages & Software suites
  - word processing
desktop publishing
Spreadsheet
database
computer-aided design (CAD)
presentation
graphics
- **Custom written** (bespoke)

Operating Systems

9

- Operating system is a software program which controls the operations of the computer system.
  - It provides a user interface.
  - It controls how the computer responds to user's requests
  - It controls how the hardware communicate with each other
  - It provides an environment in which application software can be executed



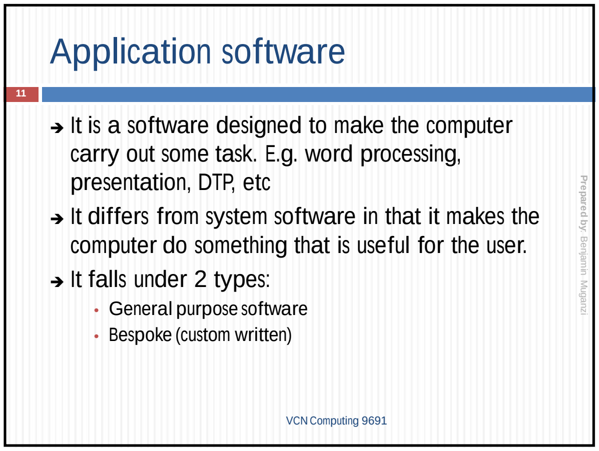
System utilities

10

- System Utility is a program designed to perform a particular maintenance task on the system.
  - A system utility software assists the OS in performing system maintenance such as:
    - Disk management
    - Memory management
    - Print management
    - File management

Prepared by: Benjamin Muganzi

VCN Computing 9691



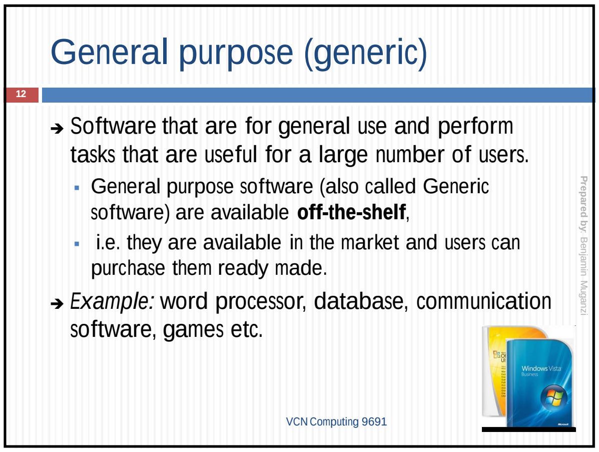
Application software

11

- It is a software designed to make the computer carry out some task. E.g. word processing, presentation, DTP, etc
- It differs from system software in that it makes the computer do something that is useful for the user.
- It falls under 2 types:
  - General purpose software
  - Bespoke (custom written)

Prepared by: Benjamin Muganzi

VCN Computing 9691



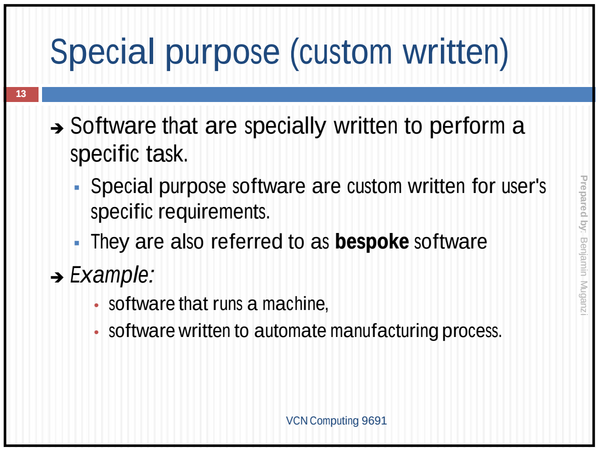
General purpose (generic)

12

- Software that are for general use and perform tasks that are useful for a large number of users.
  - General purpose software (also called Generic software) are available **off-the-shelf**,
  - i.e. they are available in the market and users can purchase them ready made.
- *Example:* word processor, database, communication software, games etc.

Prepared by: Benjamin Muganzi

VCN Computing 9691



Special purpose (custom written)

13

- Software that are specially written to perform a specific task.
  - Special purpose software are custom written for user's specific requirements.
  - They are also referred to as **bespoke** software
- *Example:*
  - software that runs a machine,
  - software written to automate manufacturing process.

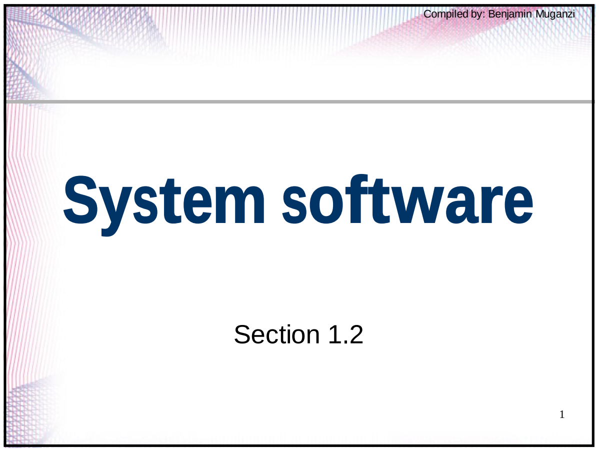
Prepared by: Benjamin Muganzi

VCN Computing 9691

14

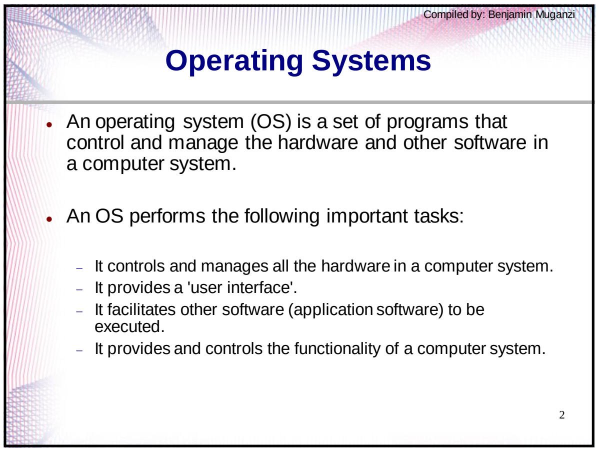
Prepared by: Benjamin Muganzi

VCN Computing 9691



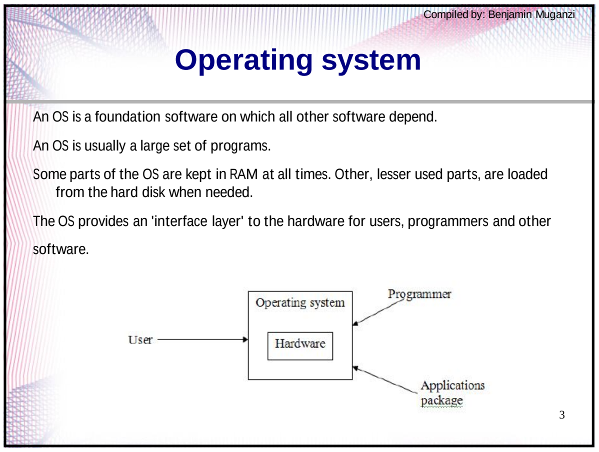
System software

Section 1.2



Operating Systems

- An operating system (OS) is a set of programs that control and manage the hardware and other software in a computer system.
- An OS performs the following important tasks:
  - It controls and manages all the hardware in a computer system.
  - It provides a 'user interface'.
  - It facilitates other software (application software) to be executed.
  - It provides and controls the functionality of a computer system.



Operating system

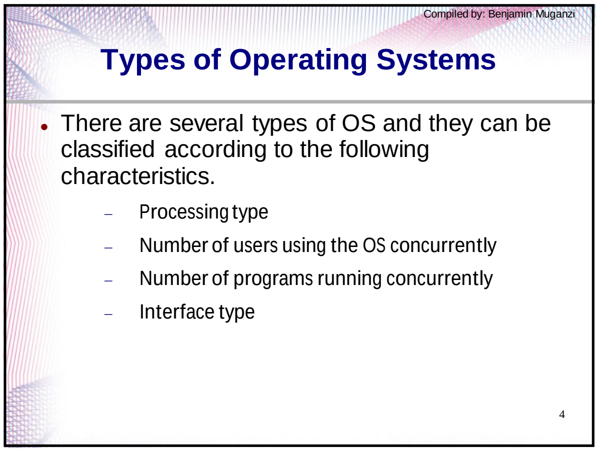
An OS is a foundation software on which all other software depend.

An OS is usually a large set of programs.

Some parts of the OS are kept in RAM at all times. Other, lesser used parts, are loaded from the hard disk when needed.

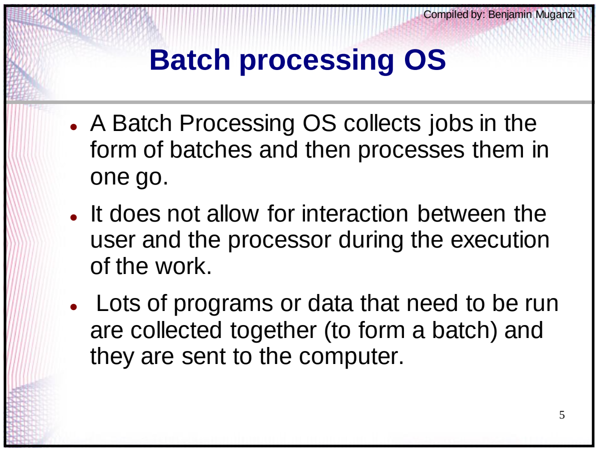
The OS provides an 'interface layer' to the hardware for users, programmers and other software.

Operating system
Hardware
User
Programmer
Applications package



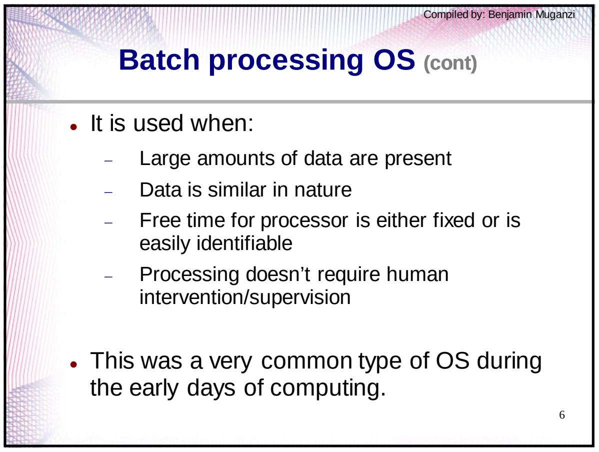
Types of Operating Systems

- There are several types of OS and they can be classified according to the following characteristics.
  - Processing type
  - Number of users using the OS concurrently
  - Number of programs running concurrently
  - Interface type



Batch processing OS

- A Batch Processing OS collects jobs in the form of batches and then processes them in one go.
- It does not allow for interaction between the user and the processor during the execution of the work.
- Lots of programs or data that need to be run are collected together (to form a batch) and they are sent to the computer.



Batch processing OS (cont)

- It is used when:
  - Large amounts of data are present
  - Data is similar in nature
  - Free time for processor is either fixed or is easily identifiable
  - Processing doesn’t require human intervention/supervision
- This was a very common type of OS during the early days of computing.

Batch processing OS (cont)

Examples where batch processing can be used:

- Printing of Bank Statements
- Producing Utility Bills (electricity, gas, telephone etc.)
- Compiling of large and complex high level programs.

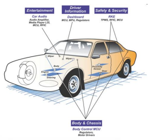
Batch processing OS (cont)

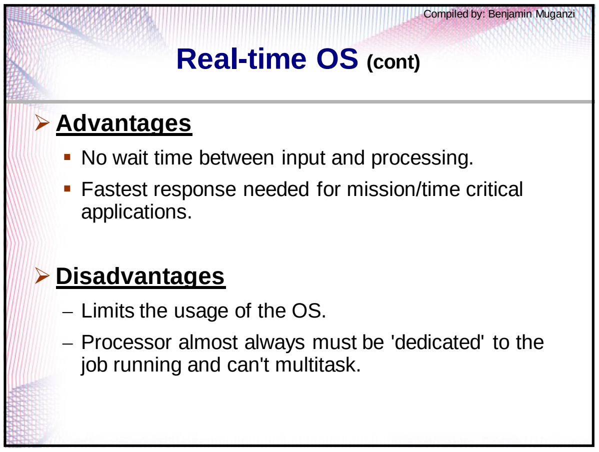
- **Advantages:**
- No need for an administrator to be present (unattended processing).
- Processing is done at off-peak hours to save processor time.
- Solves the “speed mismatch” problem. Takes out the slowest component of a computer system – human being.

- **Disadvantages:**
- Doesn’t produce immediate processing output.
- Process may run into a problem and stop.
- Errors are not immediately identifiable.

Real-time Operating systems

- This is an OS in which the requests are executed immediately and can therefore produce a response within a specified short time.
- Input is processed without any delay and output is produced quickly so that the user can decide what to do next.

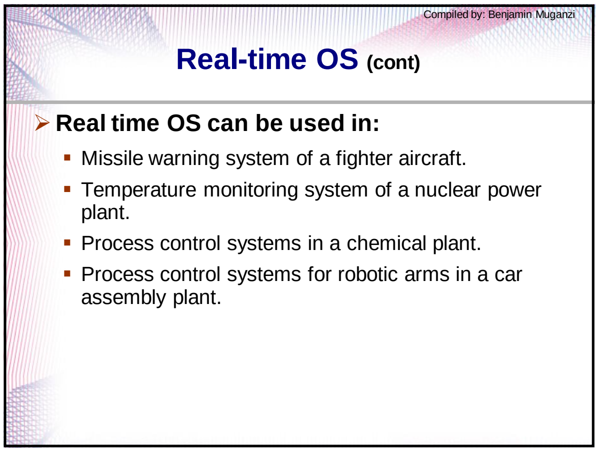




Compiled by: Benjamin Muganzi

Real-time OS (cont)

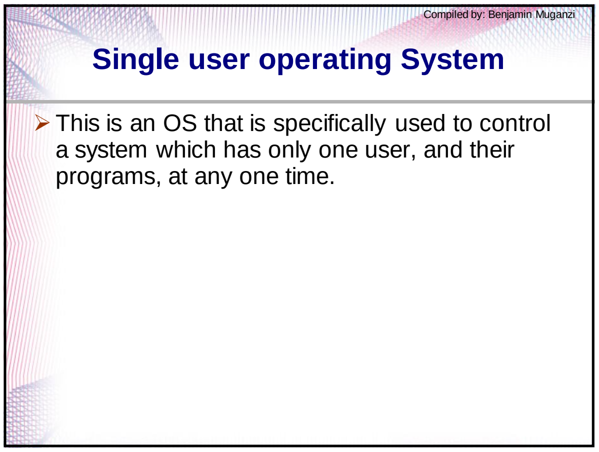
- **Advantages**
  - No wait time between input and processing.
  - Fastest response needed for mission/time critical applications.
- **Disadvantages**
  - Limits the usage of the OS.
  - Processor almost always must be 'dedicated' to the job running and can't multitask.



Compiled by: Benjamin Muganzi

Real-time OS (cont)

- **Real time OS can be used in:**
  - Missile warning system of a fighter aircraft.
  - Temperature monitoring system of a nuclear power plant.
  - Process control systems in a chemical plant.
  - Process control systems for robotic arms in a car assembly plant.



Compiled by: Benjamin Muganzi

Single user operating System

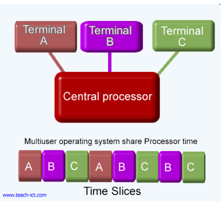
- This is an OS that is specifically used to control a system which has only one user, and their programs, at any one time.

Single user OS (cont)

- **Advantages**
  - Exclusive access to processing and storage facility for a single user.
- **Disadvantages**
  - A user usually doesn't use the full processing power of a computer. Processor stays
  - idle most of the times.
- **Example:**
  - Your home computer operating system.

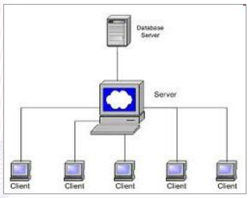
Multi user operating system

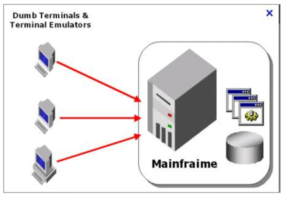
- A multi user OS can be used by more than one user concurrently.

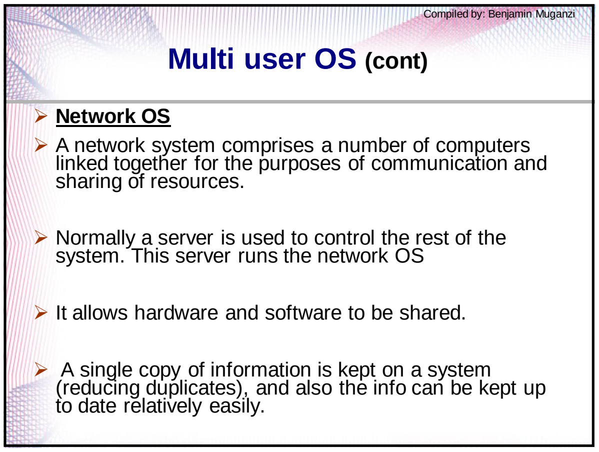


Multi user OS (cont)

- There are 2 types:
  - Network OS
  - Time sharing OS

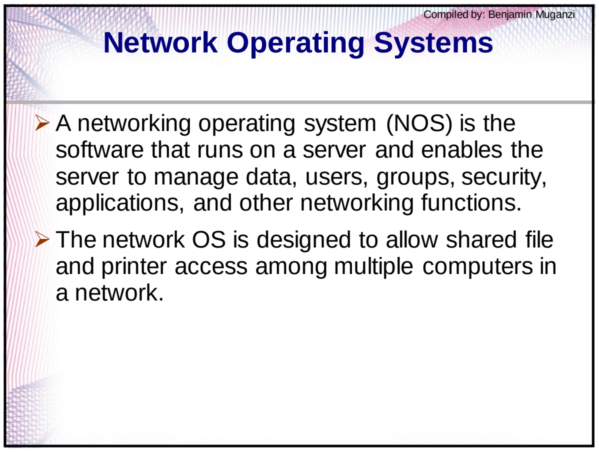






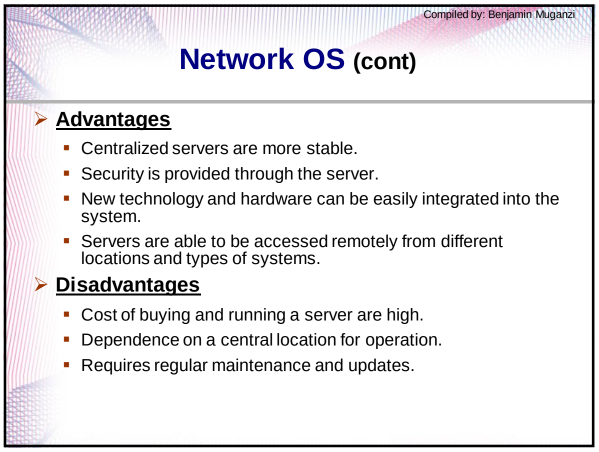
Multi user OS (cont)

- **Network OS**
- A network system comprises a number of computers linked together for the purposes of communication and sharing of resources.
- Normally a server is used to control the rest of the system. This server runs the network OS
- It allows hardware and software to be shared.
- A single copy of information is kept on a system (reducing duplicates), and also the info can be kept up to date relatively easily.



Network Operating Systems

- A networking operating system (NOS) is the software that runs on a server and enables the server to manage data, users, groups, security, applications, and other networking functions.
- The network OS is designed to allow shared file and printer access among multiple computers in a network.



Network OS (cont)

- **Advantages**
  - Centralized servers are more stable.
  - Security is provided through the server.
  - New technology and hardware can be easily integrated into the system.
  - Servers are able to be accessed remotely from different locations and types of systems.
- **Disadvantages**
  - Cost of buying and running a server are high.
  - Dependence on a central location for operation.
  - Requires regular maintenance and updates.

Multi user OS (cont)

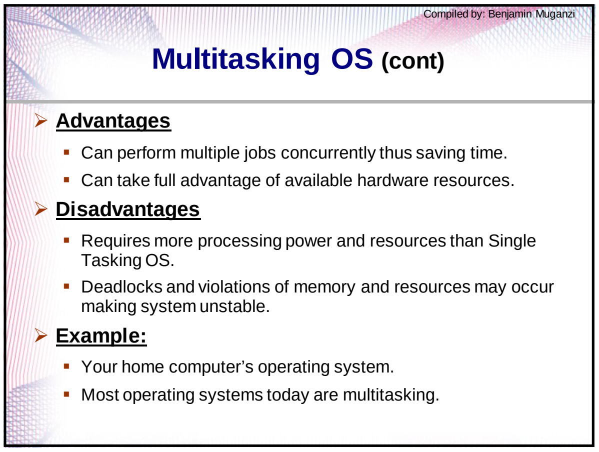
- **Time-sharing system**
- Has a single powerful computer which is connected up to a number of terminals.
- These terminals have very limited amount of processing power or none at all.
- This also allows sharing of hardware and software across the system, and also allows communication between users.

Multi user OS (cont)

- **Advantages**
  - Allows processing and storage capacity to be utilized to full extent.
  - Lowers the cost of ownership.
  - Sharing of hardware resources and software.
  - Easy to maintain one computer than many.
- **Disadvantages**
  - Increased risk of system failure.
  - Data security becomes a concern.
  - Performance may become slow if number of user are not managed or if user are using
  - processing intensive applications.

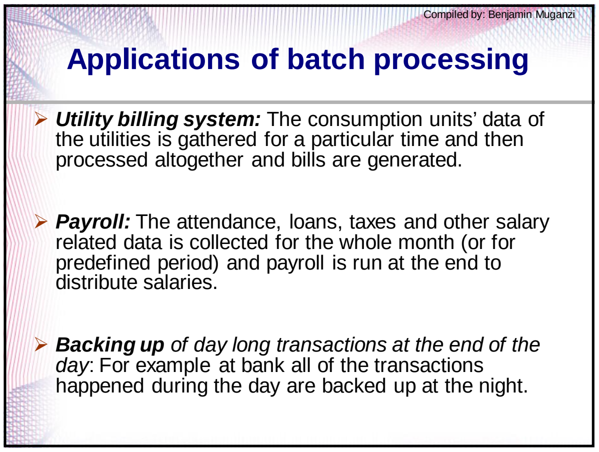
Multi tasking Operating System

- A multitasking OS can run many programs at the same time (apparently).
  - *In actuality, the OS is assigning small portions of time to each program, creating an illusion that all programs are running at the same time.*



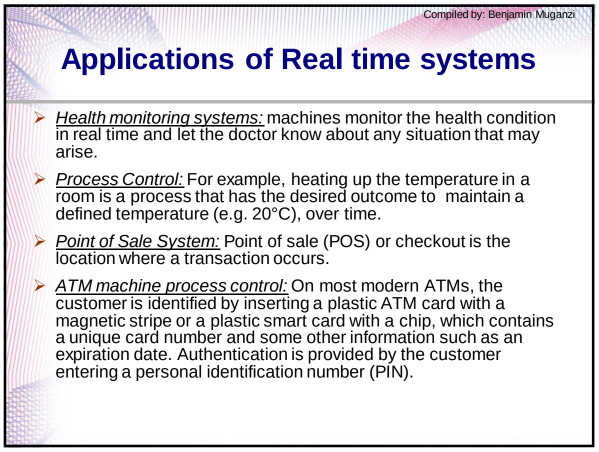
Multitasking OS (cont)

- **Advantages**
  - Can perform multiple jobs concurrently thus saving time.
  - Can take full advantage of available hardware resources.
- **Disadvantages**
  - Requires more processing power and resources than Single Tasking OS.
  - Deadlocks and violations of memory and resources may occur making system unstable.
- **Example:**
  - Your home computer's operating system.
  - Most operating systems today are multitasking.



Applications of batch processing

- ***Utility billing system:*** The consumption units' data of the utilities is gathered for a particular time and then processed altogether and bills are generated.
- ***Payroll:*** The attendance, loans, taxes and other salary related data is collected for the whole month (or for predefined period) and payroll is run at the end to distribute salaries.
- ***Backing up** of day long transactions at the end of the day*: For example at bank all of the transactions happened during the day are backed up at the night.



Applications of Real time systems

- *Health monitoring systems:* machines monitor the health condition in real time and let the doctor know about any situation that may arise.
- *Process Control:* For example, heating up the temperature in a room is a process that has the desired outcome to maintain a defined temperature (e.g. 20°C), over time.
- *Point of Sale System:* Point of sale (POS) or checkout is the location where a transaction occurs.
- *ATM machine process control:* On most modern ATMs, the customer is identified by inserting a plastic ATM card with a magnetic stripe or a plastic smart card with a chip, which contains a unique card number and some other information such as an expiration date. Authentication is provided by the customer entering a personal identification number (PIN).

2.1. OPERATING SYSTEM

WHAT IS AN OPERATING SYSTEM?

- It is a program or a set of programs that manages the operation of the computer.
- The most frequently used instructions in the operating system must be stored in main memory and remain there whilst other programs such as application programs are being executed.
- This portion of the operating system is called by many different names such as control program, nucleus, monitor, supervisor or executive.
- See figure 63.1.

LOADING AN OPERATING SYSTEM

- On all large computers and most micros, the operating system is held on disk and has to be loaded into main memory once the computer has been switched on, before any other programs can be run.
- The process of loading the operating system is called booting the system.
- On a microcomputer system, the operating system is usually held on the hard disk if the system has one. Otherwise it will be held on a floppy disk, which will be known as a system disk.
- On all computers a small program held in ROM (the 'loader') will tell the computer where to look for the operating system and give instructions for loading at least part of it into memory.
- Once part of the operating system has been loaded, more instructions can be executed to load the rest of the nucleus. This method of 'pulling itself up by its own bootstraps' is where the expression 'booting' comes from.
- In smaller hand-held computers the control program is permanently held in ROM.

MODES OF OPERATION

Operating systems vary considerably in their capabilities, from relatively simple single user microcomputer systems, to sophisticated mainframe computers. The modes of operation include:

1) **Single program operation.**

- The OS supervises the loading and running of one program at a time and the input and output of data from and to peripheral devices.

2) **Multiprocessing.**

- In larger systems one computer system may have more than one processor but may still share all or some of the same memory.

3) **Multiprogramming.**

- This means that two or more programs are being run concurrently.
- The operating system has to allocate resources to each program, allowing each one a small amount of processor time before moving on to the next one.

4) **Batch processing.**

- There is no interaction between the user and the computer during batch processing.
- One or more programs are submitted to the computer as a job and put in a batch queue until it is their turn to be run.

5) **Real-time processing.**

- Real-time systems can be of different types; process control, information storage and retrieval and transaction processing.
- In any of these systems, the data input to the computer must be processed immediately. A fast response is required but whereas a few seconds delay may be acceptable in an information retrieval system, response must be instantaneous in a process control system which might for example be monitoring a chemical reaction and so on.
- For example, commercial airlines use real-time processing in their on-board computers and some airlines have a continuously updated display in the passenger areas for the interest of the passengers. The on-board computer constantly updates it estimate of arrival time, etc based on the current readings for wind velocity, direction and so on.

THE FUNCTIONS OF AN OPERATING SYSTEM

Obviously not all operating systems carry out precisely the same tasks; a single-programming micro will not need to perform all the tasks of a multiprogramming, multi-user mainframe computer.

The functions listed below apply to a greater or lesser extent to many computers and will be explained in more depth in subsequent Units:

1. **Resource allocation and scheduling.**

- All operating systems must have the ability to load programs into memory and start them running; more complex operating systems have to allocate CPU time, memory and I/O (input/output) resources to different processes running concurrently.

2. **Memory Management.**

    - Some or all the computer's memory will be 'partitioned', with different programs running in different partitions.
    - The operating system decides on how these partitions are organised.
    - Even in a micro which can only run one program at a time, some memory management is required.
    - the operating system for example will occupy a different partition from the user's program. See fig. 63.3 - memory partitioning.

3. **Backing-store Management.**

    - This includes supervising the creation, updating and deletion of files.
    - A directory of files has to be maintained so that they can be quickly located when required.

4. **Input/Output Control.**

    - Reading from and writing to the various peripherals is controlled by the operating system.

5. **Interrupt handling.**

    - An interrupt may be caused deliberately, as for example in a time-sharing system where a real-time clock generates interrupts at regular time intervals to give the next user processor time.
    - Interrupts may also be caused by program errors, hardware malfunctions, or external events such as pressing the 'Break' key.

6. **Operator Interface.**

    - The OS accepts commands from the operator and responds to them.

7. **Security.**

    - Most multi-user systems provide the ability to assign login codes and passwords to different users.

8. **Utilities.**

    - Most OS contain utility programs for file management, editing files, reorganising disk space, making backups and so on.

9. **Accounting facilities.**

- Large systems will keep a record of facilities used by each person such as processor time, number of pages printed and disk space used.

Topic 2.2: USER INTERFACES

INTRODUCTION

- User interface is also called as a 'human computer interface (H.C.I.)' or 'man machine interface'. It is used to describe the interaction between a user and a computer, in other words, the method by which the user tells the computer what to do and the responses which the computer makes.

THE IMPORTANCE OF GOOD INTERFACE DESIGN

A good interface design can help to ensure that users carry out their tasks:

- safely
- effectively
- efficiently
- enjoyably

Well designed systems can improve the output of employees, improve the quality of life and make the world safer and more enjoyable place to live in.

DESIGNING USABLE SYSTEMS

In order to design a usable interface, the designer has to take into considerations the following factors:

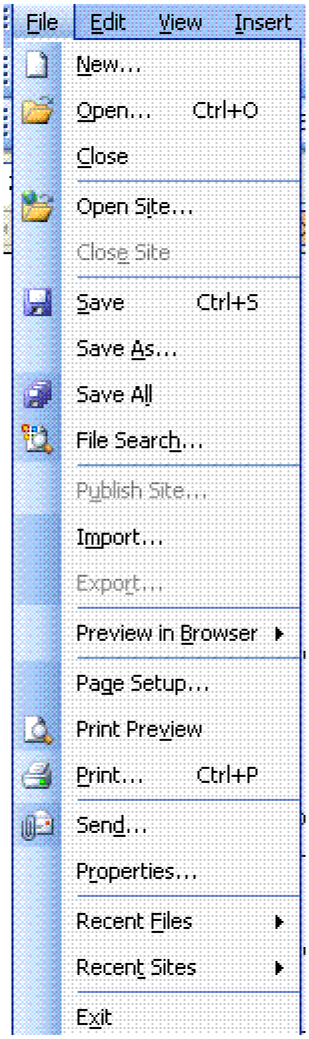
- **WHO** will use the system. For example, will the users be the computer professionals or members of the general public who may be wary of computers? For an educational program, will the users be young, for example primary school children, or teenagers on an A Level course? Will the system have to cater for both beginners and experienced users?
- **WHAT TASKS** the computer is performing. Is the task very repetitive? Does the task require skill and knowledge? Do tasks vary greatly from one occasion to the next? A travel agent who spends most of the day making holiday bookings will require a different interface from an office worker who needs to be able to switch between word processing, accounts and accessing the company database.
- **THE ENVIRONMENT** in which the computer is used. Will the environment be hazardous, noisy, or calm and quiet?
- **WHAT IS TECHNOLOGICALLY FEASIBLE**. Is it possible to simply dictate a letter to a word processor instead of typing it in?

MENU INTERFACE

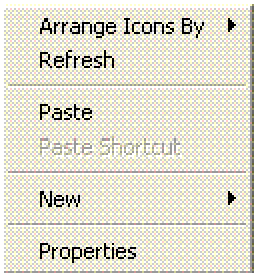
There are several different types of menu interface as the following:

1) **Full Screen Menu**. This type of menu is often used as the 'front end' of an application. It stays on screen until the user makes a choice. Try on your own!!

2) **Pull-Down Menu**. This type of menu is displayed along the top of the screen, and when the user clicks on an item, a sub-menu appears. The menu is always present whatever screen the user is looking at in the application.



3) **Pop-up Menu**. The menu pops up in response to, say, a click of the right mouse button on a particular area of the screen.

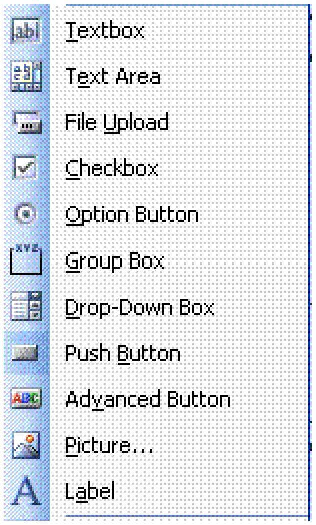


WIMP INTERFACE

- WIMP stands for Windows, Icons, Mouse and Pull-down menus.
- A window is an area on the screen through which a particular piece of software of a data file may be viewed. The window may occupy the whole screen, or the user can choose to have several windows on the screen with a different application running in each one. Windows can be moved, sized, stacked one top of the other, opened and closed.
- An icon is a small picture representing an item such as a piece of software, a file, storage medium (such as disk) or command. By pointing with a mouse at particular icon and clicking the mouse button the user can select it.

FORMS

When a user is required to enter data such as sales invoices or customer names and addresses, it is common to have a 'form' displayed on the screen for the user to fill in.



The following should be noted when designing forms of this type:

- The display should be given a title to identify it.

- The form should not be too cluttered. Spaces and blanks in a display are important.
- It should give some indication of how many character can be entered in each field of data.
- The user should be given a chance to go back and correct any field before the data is accepted.
- Items should appear in a logical sequence to assist the user.
- Default values should wherever possible be pre-written onto the form so that a minimum of data entry is required.
- Full exit and help facilities should be provided. For example, the user could enter '?' in a field if he requires more information.
- Lower case in a display is neater and easier to read than upper case.
- 'Attention-getting' devices such as blinking cursors, underlining etc should not be over-used.

DIALOGUE BOXES

- Dialogue boxes are a special type of form associated with the Windows environment: an example: open MS WORD, <FILE>, <PRINT>. Try on your own (DIY). Or open MS FrontPage, <INSERT>, <FORM>, <CHECKBOX> OR <OPTION BUTTON>.

COMMAND-DRIVEN INTERFACES

- With this type of interface very little help is given to the user who has to type a command such as, for example using MSDOS PROMPT: <DELETE FILE1> to delete a file. DIY!
- Commands enable a user to quickly and concisely instruct the computer what to do but they do require the user to have a knowledge of the commands available and the syntax for using them.

Topic 2.3: UTILITY SOFTWARES

1) DISK FORMATTING

2) FILE HANDLING (DELETING, COPYING, MOVING, SORTING)

3) HARDWARE DRIVERS

- **Drivers** are programs that control a device. Every device, whether it is a printer, disk drive, or keyboard, it must have a driver program. Devices such as the video or soundcard need to have their drivers installed when the product is installed.

4) FILE COMPRESSION

- Reduces chances to see your program being cracked and spreaded everywhere.
- Protect your executables files where executable file always contains superfluous codes. It allows compress exe-file up to 30-60% original.

5) VIRUS CHECKERS

- Removes viruses, Trojan horses and worms that are available in any file in the PC
- back up your files, and prevent computer problems
- Update with up-to-date viruses

Resources:

P M Heatcote, *[A level Computing, 3rd. edition]*, Ashford Colour Press Ltd, Gosport, Hants, 1996.

1.3 Data; its representation, structure and management

Bits and Bytes

- The word “bit” is derived from two words, **bi**nary digi**t.**
- Computers only understand machine language which is comprised of bits.
- Bits are rarely used alone in computers. Normally we refer to a set of 8 bits which make a **byte**.
- With 8 bits in a byte, you can represent 256 values ranging from 0 to 255, as shown here:

    0 = 00000000
    1 = 00000001
    2 = 00000010
    ...
    254 = 11111110
    255 = 11111111

How to represent characters

A character set is the set of symbols that the computer’s software can recognize.

The character set is represented in the computer by using bit patterns.

When you press an 'a' on the keyboard, for example, a particular pattern of 0s and 1s is stored in the keyboard buffer to represent that character

In order for different hardware manufacturers to have the same binary patterns, standard character sets have been formed. Examples of such character sets are **ASCII**, **Unicode**, etc

American Standard Code for Information Interchange (ASCII)

- ASCII system is used to represent the data generated by the computer keyboard.
- Each binary value between 0 and 127 is given a specific character.
- Most computers extend the ASCII character set to use the full range of 256 characters available in a byte.
- The upper 128 characters handle special things like accented characters from common foreign languages.
- A byte is used for each character.

Below is a list of some ASCII character codes. (There is no need to cram or master them)

Character	ASCII
space	00100000
!	00100001
"	00100010
#	00100011
$	00100100
%	00100101
/	00101111
0	00110000
1	00110001
2	00110010
3	00110011
4	00110100
5	00110101
6	00110110
7	00110111
8	00111000
9	00111001
?	00111111
@	01000000
A	01000001
B	01000010
C	01000011
D	0100100
a	01100001
b	01100010
c	01100011
d	01100100

Unicode standard:

The Unicode Standard is a character coding system designed to support the worldwide interchange, processing, and display of the written texts of the diverse languages and technical disciplines of the modern world. In addition, it supports classical and historical texts of many written languages.

This includes support for languages such as Hebrew or Arabic, which write from right to Left.

NOTE:

ASCII uses 8 bits so there are 256 different codes that can be used and hence 256 different characters.
A problem arises when the computer retrieves a piece of data from its memory. Imagine that the data is **01000001**. Is this the number 65, or is it A?
They are both stored in the same way, so how can it tell the difference?

- The answer is that characters and numbers are stored in different parts of the memory, so it knows which one it is by knowing whereabouts it was stored

Data Types:

- The computer needs to use different types of data in the operation of the system.
- All of these different types of data have to be stored as binary numbers.
- The computer can distinguish one type of data from another by seeing the location in memory where it is stored.

There are 5 major data types that the computer normally deals with:

- Numeric (Integers)
- Alphanumeric (Characters)
- Boolean
- Date/time
- Currency

Numeric data:

- The computer must be able to handle numbers. Whole numbers are called Integers. They can be positive or negative, and the computer must store these as binary.
- When two bytes are used, then we have stored a short integer and when 4 bytes are used, it is a long integer.
- The difference between the two is that a long integer stores more numbers.

NB: (How the computer stores fractions or negative numbers shall be looked at in A2.)

Boolean data:

- Data which can only have two states e.g. [*Yes/No], [true/false], [on/off], etc* is known as BOOLEAN data.
- Since there are only two options, the computer uses a single bit (which can be either 0 or 1) to store the data. It is stored as this with 1 standing for true and 0 standing for false, 1 for yes and 0 for No, etc

Character data:

- This is the data type representing characters on the keyboard, strings, special symbols, etc (When several characters are grouped together, they form a string.)
- Text or string data type is assigned to data which cannot be used in mathematical calculations. Examples are:-
  - School admission number such as "LW12345"
  - An email address such as john-smith_colombo@yahoo.com
  - Telephone number

Date/time and Currency data

- These are special types of data normally handled at the operating system level unlike the other data types which are handled at the application level.
- The computer has rules that govern such data types and it simply checks the data that is input against these rules.
- In databases, these data types are normally used.

REPRESENTATION OF INTEGERS

- When the characters 15863 are pressed on the keyboard, the binary codes for 1, 5, 8, 6 and 3 are sent to the keyboard buffer.
- If these digits represent, say, a telephone number, they can be stored in this format in 5 consecutive bytes.
- If, on the other hand, the digits represent an integer (whole number) that is to be used in a calculation, they need to be converted to a **binary integer.**
- Our ordinary number system is called the **denary** system and uses ten digits 0 to 9.
- In this system, the number 387 represents:

  3 X 100 + 8 X 10 + 7 X 1

- This is a base 10 number system and as we move from right to left, each digit is worth 10 times as much as the previous one.

TRANSLATING FROM BINARY TO DENARY

- In the binary system, only the two digits 0 and 1 are used, and as we move from right to left, each digit is worth twice as much as the previous one.
- Thus the binary number 01011101 can be set out under column headings as follows:

128	**64**	**32**	**16**	**8**	**4**	**2**	**1**
0	1	0	1	1	1	0	1

This represents 64 + 16 + 8 + 4 + 1

TRANSLATING FROM DENARY TO BINARY

- A programmer generally has some control over how many bytes are to be used to hold an integer value.

BINARY ARITHMETIC

When we learn arithmetic, we use Base **10** numbers. (**denary** or **decimal** system).

1. The first method is by dividing the number by 2 consecutively

2. The number **1583** means 1 'thousand', 5 'hundreds', 8 'tens' and 3 'units'...

1000	100	10	1
1	**5**	**8**	**3**

The **Binary** number system is a Base **2** system.

Consider the binary number 10010110.

We can set it out under headings....

128	64	32	16	8	4	2	1
1	**0**	**0**	**1**	**0**	**1**	**1**	**0**

So the binary number **10010110** represents the denary number 128 + 16 + 4 + 2 = **150.**

DATA STRUCTURES

INTRODUCTION TO DATA STRUCTURES

- All data processing on a computer involves the manipulation of data.
- This data can be organised in the computer's memory in different ways according to how it is to be processed, and the different methods of organising data are known as data structures.
- Computer languages such as Pascal have built-in elementary data types (such as *integer, real, Boolean* and *char*) and some built-in structured or composite data types (data structures) such as *record, array* and *string.* These composite data types are made up of a number of elements of a specified type such as *integer* or *real*.

STATIC AND DYNAMIC DATA STRUCTURES

- Data structures such as arrays are called '**static data structures**' because they are defined within the program as being of a certain size, and the space is reserved in memory to hold the array.
- Other data structures such as stacks, queues, lists, linked lists and binary trees are said to be '**dynamic data structures**' because they can get bigger or smaller during the execution of the program.

Static data structures

ARRAYS

- An array is defined as a finite ordered set of homogeneous elements.
- Finite means that there is a specific number of elements in the array.
- An array is a set of variables of the same type, such as integer or real.
- The number may be large or small but it is fixed and it must be at least one.
- Thus an array is a *static data structure*.
- Ordered implies that there is a first, second, third etc element of the array.

- An array x consisting of fifty integers may be declared in Pascal as follows:

An **array** is a '**list**' of data items which are all the **same data type**.

The data items in an array are called its **elements**.
Each element has a **subscript** to identify it (usually a number...but not necessarily).

A **linear list** is a one-dimensional array, where each item is identified by one subscript.

Example : A linear list (one-dimensional array) of names called '**NAME**' .

Subscript	Name
1	Julie Stephenson
2	Amanda Platt
3	John Kingsley
4	Nita Cardigan
5	Linda Hanson
6	Oscar Nominee

NAME[**2**] denotes the element of the array **NAME** which has subscript **2** - ie 'Amanda Platt'.

- Aray Declaration consists of
- Array name
- Number of elements/Subscript
- Dimension

MULTI-DIMENSIONAL ARRAYS

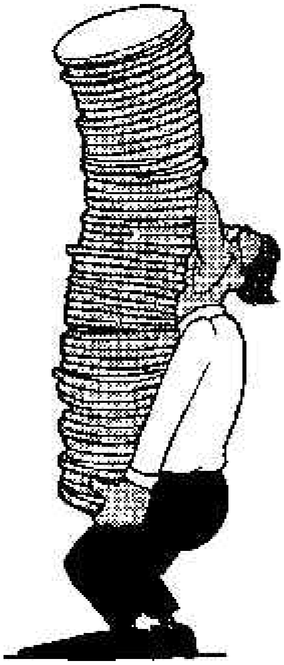
- The concept of a one-dimensional array can be extended to two or more dimensions.
- A two dimensional array may be declared in Pascal as, for example,

x : **array**[0..10] [1..6] **of integer**;

STACKS

- A stack is a dynamic data structure which has the special property that it can only be accessed at one end, like a stack of plates in a cafeteria.
- It is known as a ***last-in first-out (LIFO) data structures.***

Mark	**← Top of Stack**
Daniel	
Roxanne	



A stack is a **LIFO** (Last In First Out) Data structure **-** like a stack of plates being washed **-** a new plate gets placed on the top of the stack, and if one is removed, it is also removed from the top.

Adding a new element to a stack is called **pushing**... ...

......removing one is called **popping**.

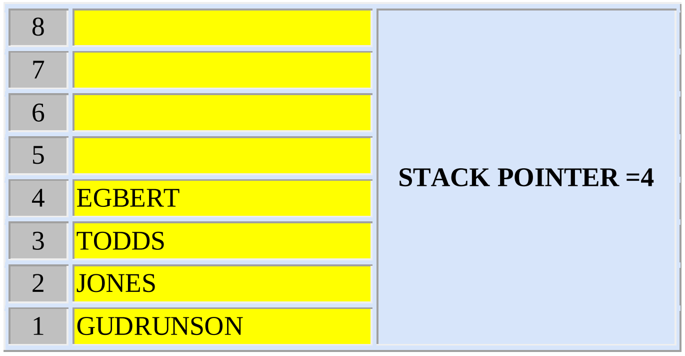
With every stack there is a **stack pointer** showing where the top of the stack is.

(When implementing a stack in a high level language you will need an **array** to hold the elements of the stack and an integer variable '**TopOfStack**')

Example : The diagram shows a stack with three names already in it. :

8		**STACK POINTER = 3**
7		
6		
5		
4		
3	TODDS	
2	JONES	
1	GUDRUNSON	

If a new name '**EGBERT**' is added to (pushed onto) the stack, the new stack looks like......



8		**STACK POINTER =4**
7		
6		
5		
4	EGBERT	
3	TODDS	
2	JONES	
1	GUDRUNSON	

(Note that the Stack Pointer has been incremented).

USES OF STACKS

Stacks are very useful data structures in computing.. Uses include:

- to store return addresses, parameters and register contents when subroutines are called. When the subroutine ends, the address at the top of the stack is popped and the computer continues execution from that address;
- in evaluating mathematical expressions held in reverse Polish notation, i.e. a form of notation used by compilers as an intermediate step in translating expressions such as A := (B*C) + D/E.

QUEUE

- A queue is a **first-in first-out (FIFO) data structure**; new elements are added tot he rear of the queue, and elements leave from the front of the queue.
- A queue can be implemented as an array with a pointer to the front of the queue and a pointer to the rear of the queue.

A **queue** is a **FIFO** (First In First Out) data structure. The first item to be added is the first one to be removed. (Think of a queue at a supermarket checkout!)

The implementation of a queue involves **two pointers**:

- **Front** - which points to the front of the queue (the item which has been in the queue longest).
- **Back** - which points to the back of the queue (the item most recently added) The diagram shows a queue after two names have been added :

8		**Front Pointer = 1** **Back Pointer = 2**
7		
6		
5		
4		
3		
2	Harry	
1	William	

If '**Jonathan**' and '**Hayley**' are then added to the queue:

8		**Front Pointer = 1** **Back Pointer = 4**
7		
6		
5		
4	Hayley	
3	Jonathan	
2	Harry	
1	William	

Then a name is removed and '**Paul**' is added....

8		**Front Pointer = 2** **Back Pointer = 5**
7		
6		
5	Paul	
4	Hayley	
3	Jonathan	
2	Harry	
1		

An **example** of a queue in a computer system would be the **printer queue** ie spooled documents waiting for output to a printer. The documents would be printed in the order in which they were spooled.

USES OF QUEUE

Queues are used in a variety of applications such as:

- holding jobs waiting to be run by the computer;
- a keyboard buffer, to allow a whole line to be typed and edited while the processor is busy doing something else;
- spooling output on to a disk to await printing.

Files, Records and Fields

A **field** is an item of stored data. A field could be a name, a date, an address, a description, a quantity, etc. When a field is defined it is given a name (identifier) and a type, that defines the type of data that will be stored in that field. This is exactly the same as defining a variable within a program.

Examples of field definitions

```
String firstName;
int quantityInStock;
```

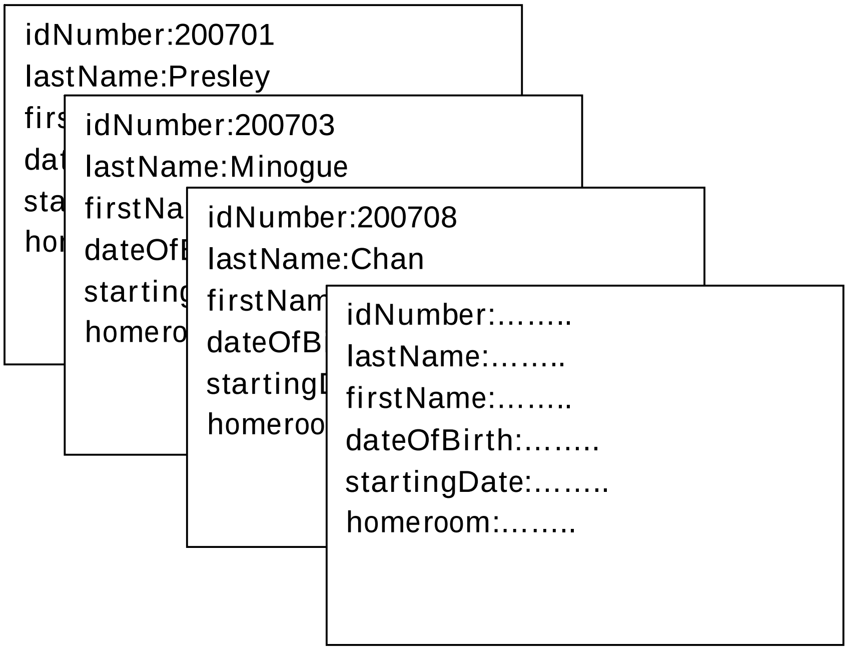
A **record** is the collection of fields that relate to a single entity. For example, we could have a student record that includes fields for the student's name, address, homeroom, date of birth, etc. A product record could include fields for the serial number, description, cost price, quantity in stock, etc.

Example of a student record

```
public class StudentRecord
{
     private int idnumber;
     private String lastName;
     private String firstName;
     private Date birthDate;
     private Date startingDate;
     private String homeroom;
     ...............................
}
```

A **file** is a collection of related records. For example, a student file might include all of the records of students enrolled at a school. A police department might keep a file of criminal records, which includes details about all known criminals. Files are stored on secondary storage devices such as hard disks, CD-ROMs etc.

Within a file, all records usually have the same structure. That is, every record in the file contains the same fields. Only the data stored in the fields of different record will be different.



Conceptual diagram of a student file

A **database** is a collection of related files. For example, a library database might include files for books, borrowers and circulation (borrowing and returning of books).

Key Fields

A key field is a field in a data file containing information capable of uniquely identifying a record in relation to all other records in the same data file.

The main purpose for storing data in files is so that is can be retrieved and processed at a later time. The information stored in a key field has to be unique (at least within the file) so that each record in a file can be identified. A key field is often an identity number, a serial number, a social security number, a product number or some other unique value. No two records within a file can have the same key value.

In the above student file example, the idNumber field is the key field.

Fixed and Variable Length Records

The size (length) of a record is the number of bytes of storage spece required to store the record. It can be calculated by adding together the size of each field in a record. Recall that, in Java programs, each `int` value is stored in 4 bytes, each `double` in 8 bytes, etc. However, `Strings` may be of different lengths.

A file of **fixed length records** contains records that are **all exactly the same size**.

A file of **variable length records** contains records of **different sizes**.

Does this mean that if we store `string` data, then we can only store them in variable length records? The answer is “No”. We can store `strings` in fixed length records if we make sure that, for a particular `string` field, the length of the data items is always the same.

For example, we could make all of the `firstName` fields in our student record contain names that are , say, 15 characters long. for short names, like “Joe” we could add trailing spaces or null characters to make the string exactly 15 characters long. Thus “Joe” would be stored as ”Joe□□□□□□□□□□□□”, “Bartholomew” would be stored as “Bartholomew□□□□”, etc. If a student’ first name was longer than 15 characters, then it would be truncated to make it exactly 15 characters long.

Note
When you are designing fixed length fields, you must make sure that you do not make them too small, so that important data is lost through truncation. You must also make sure that you do not make them too large, as this will waste too much storage space. You need to know about the range of sizes of string that you are going to store – it is time consuming to change the size later!

Record Type	**Advantages for Files**	**Disadvantages for Files**
Fixed length	All records are exactly the same size. This makes programming for processing the file easier. It also makes it possible to more accurately predict how much storage space will be required for the file.	There will be wasted space in many of the records – storage will be less efficient.
Variable length	There is little wasted space in the files – storage is more efficient.	Programming the processing of files is more complex.

File Size

To work out the size of a record, you need to add together the number of bytes required for each field.

To calculate the size of a file you multiply the record size by the number of records.

Review Question

A file has the following record structure:

```
int productID;
String productDescription;
double cost;
int quantityInStock;
```

If the `productDescription` field has a fixed length of 56 bytes,

1. Calculate the size of one record.
2. Calculate the size of a file containing 8000 records.
3. Suggest a suitable key for this file. Justify your choice.

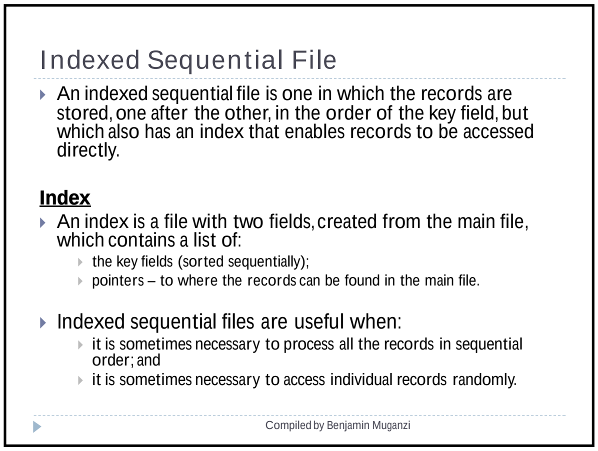
Modes of File Access

Serial File

- A serial file is one in which records are stored, one after the other, in the order in which they are added – not in order of a key field.
- This means that new records are stored at the end of the file.
- Both disks and tapes can be used to store a file serially.

Sequential File

- A sequential file is one in which the records are stored, one after the other, in the order of the key field.
- As with a serial file, both tape and disks can be used to store a file sequentially and access to the records must take place from the beginning of the file.
- **Benefits**
- Sequential files allow the records to be displayed in the order of the key field – this makes the process of adding a record slower, but significantly speeds up searches.



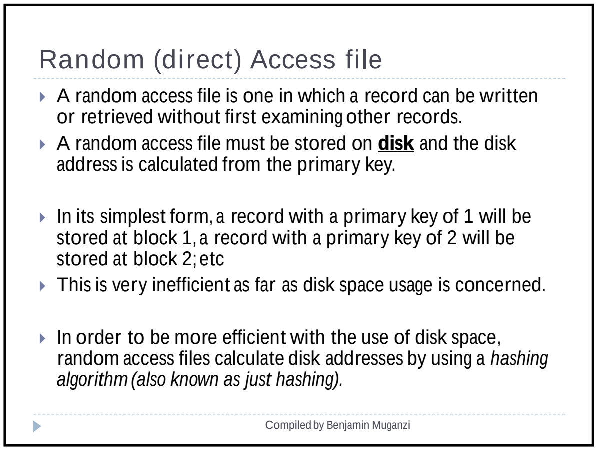
Indexed Sequential File

- An indexed sequential file is one in which the records are stored, one after the other, in the order of the key field, but which also has an index that enables records to be accessed directly.

Index

- An index is a file with two fields, created from the main file, which contains a list of:
  - the key fields (sorted sequentially);
  - pointers – to where the records can be found in the main file.
- Indexed sequential files are useful when:
  - it is sometimes necessary to process all the records in sequential order; and
  - it is sometimes necessary to access individual records randomly.

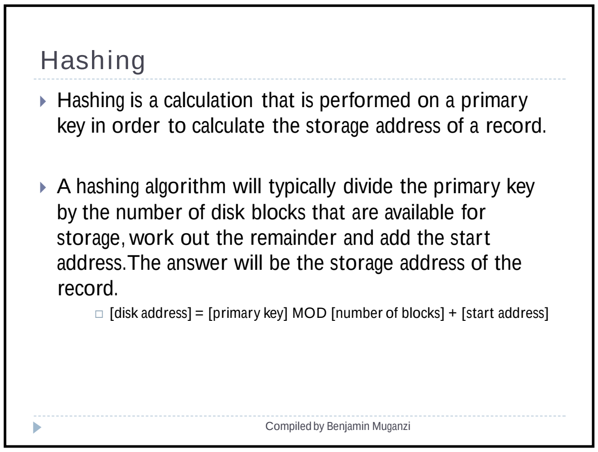
Compiled by Benjamin Muganzi



Random (direct) Access file

- A random access file is one in which a record can be written or retrieved without first examining other records.
- A random access file must be stored on **disk** and the disk address is calculated from the primary key.
- In its simplest form, a record with a primary key of 1 will be stored at block 1, a record with a primary key of 2 will be stored at block 2; etc
- This is very inefficient as far as disk space usage is concerned.
- In order to be more efficient with the use of disk space, random access files calculate disk addresses by using a *hashing algorithm (also known as just hashing).*

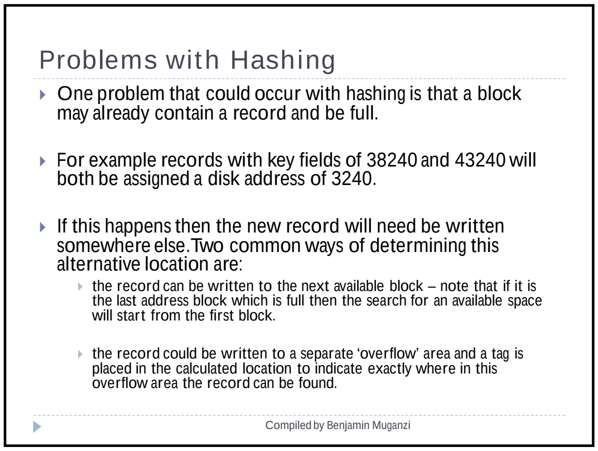
Compiled by Benjamin Muganzi



Hashing

- Hashing is a calculation that is performed on a primary key in order to calculate the storage address of a record.
- A hashing algorithm will typically divide the primary key by the number of disk blocks that are available for storage, work out the remainder and add the start address.The answer will be the storage address of the record.
  - □ [disk address] = [primary key] MOD [number of blocks] + [start address]

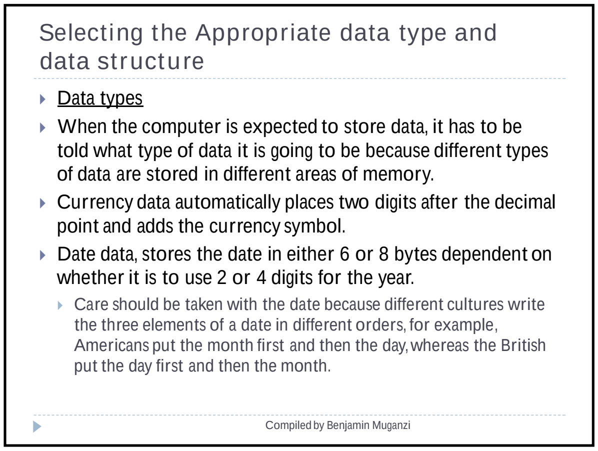
Compiled by Benjamin Muganzi



Problems with Hashing

- One problem that could occur with hashing is that a block may already contain a record and be full.
- For example records with key fields of 38240 and 43240 will both be assigned a disk address of 3240.
- If this happens then the new record will need be written somewhere else.Two common ways of determining this alternative location are:
  - the record can be written to the next available block – note that if it is the last address block which is full then the search for an available space will start from the first block.
  - the record could be written to a separate ‘overflow’ area and a tag is placed in the calculated location to indicate exactly where in this overflow area the record can be found.

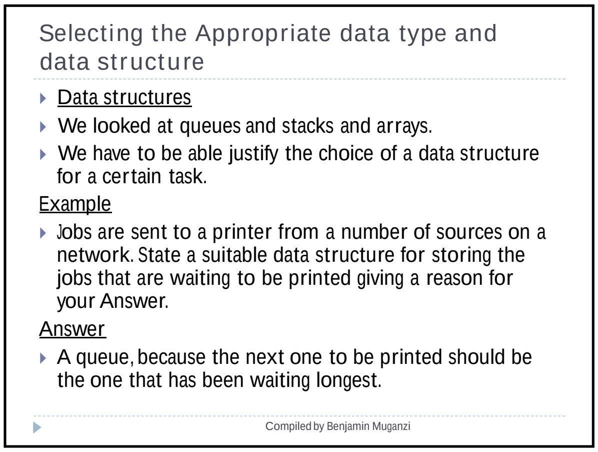
Compiled by Benjamin Muganzi



Selecting the Appropriate data type and data structure

- Data types
- When the computer is expected to store data, it has to be told what type of data it is going to be because different types of data are stored in different areas of memory.
- Currency data automatically places two digits after the decimal point and adds the currency symbol.
- Date data, stores the date in either 6 or 8 bytes dependent on whether it is to use 2 or 4 digits for the year.
  - Care should be taken with the date because different cultures write the three elements of a date in different orders, for example, Americans put the month first and then the day, whereas the British put the day first and then the month.

Compiled by Benjamin Muganzi



Selecting the Appropriate data type and data structure

- Data structures
- We looked at queues and stacks and arrays.
- We have to be able justify the choice of a data structure for a certain task.

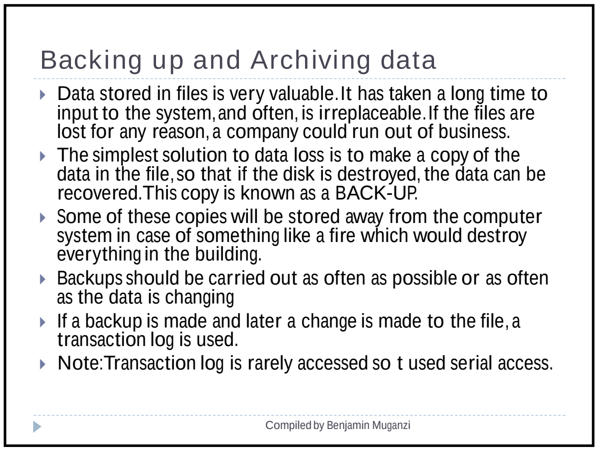
Example

- Jobs are sent to a printer from a number of sources on a network. State a suitable data structure for storing the jobs that are waiting to be printed giving a reason for your Answer.

Answer

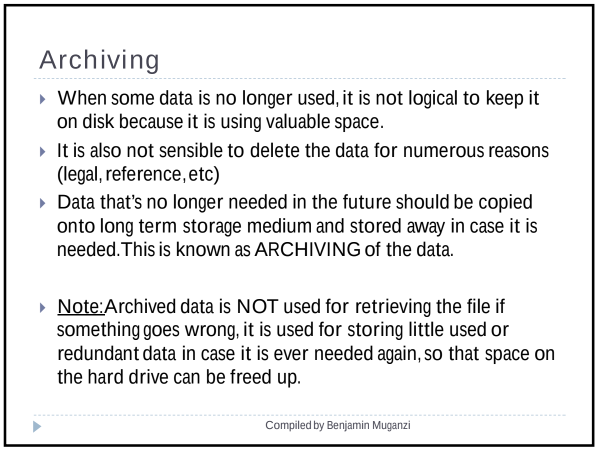
- A queue, because the next one to be printed should be the one that has been waiting longest.

Compiled by Benjamin Muganzi



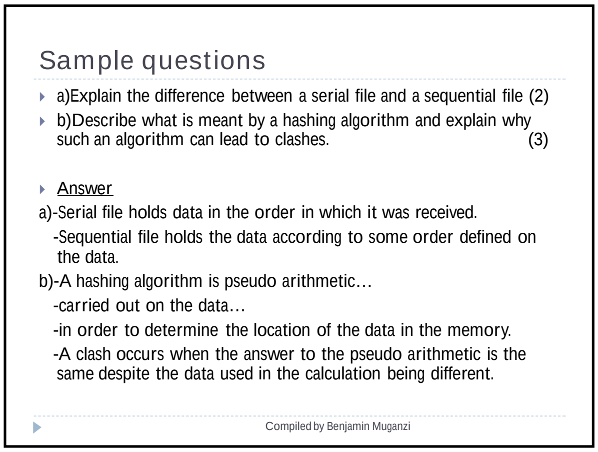
Backing up and Archiving data

- Data stored in files is very valuable. It has taken a long time to input to the system, and often, is irreplaceable. If the files are lost for any reason, a company could run out of business.
- The simplest solution to data loss is to make a copy of the data in the file, so that if the disk is destroyed, the data can be recovered. This copy is known as a BACK-UP.
- Some of these copies will be stored away from the computer system in case of something like a fire which would destroy everything in the building.
- Backups should be carried out as often as possible or as often as the data is changing
- If a backup is made and later a change is made to the file, a transaction log is used.
- Note:Transaction log is rarely accessed so t used serial access.



Archiving

- When some data is no longer used, it is not logical to keep it on disk because it is using valuable space.
- It is also not sensible to delete the data for numerous reasons (legal, reference, etc)
- Data that's no longer needed in the future should be copied onto long term storage medium and stored away in case it is needed. This is known as ARCHIVING of the data.
- Note:Archived data is NOT used for retrieving the file if something goes wrong, it is used for storing little used or redundant data in case it is ever needed again, so that space on the hard drive can be freed up.



Sample questions

- a)Explain the difference between a serial file and a sequential file (2)
- b)Describe what is meant by a hashing algorithm and explain why such an algorithm can lead to clashes. (3)

- Answer

a)-Serial file holds data in the order in which it was received.

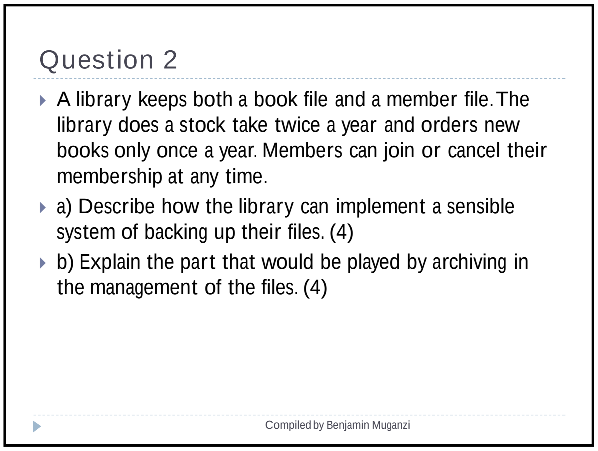
-Sequential file holds the data according to some order defined on the data.

b)-A hashing algorithm is pseudo arithmetic…

-carried out on the data…

-in order to determine the location of the data in the memory.

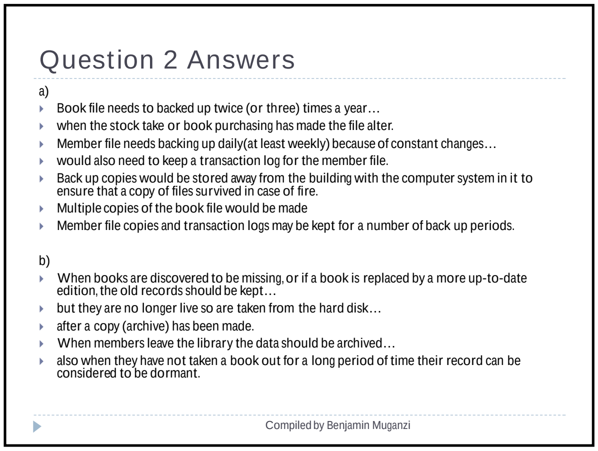
-A clash occurs when the answer to the pseudo arithmetic is the same despite the data used in the calculation being different.



Question 2

- A library keeps both a book file and a member file. The library does a stock take twice a year and orders new books only once a year. Members can join or cancel their membership at any time.
- a) Describe how the library can implement a sensible system of backing up their files. (4)
- b) Explain the part that would be played by archiving in the management of the files. (4)

Compiled by Benjamin Muganzi



Question 2 Answers

a)

- Book file needs to backed up twice (or three) times a year…
- when the stock take or book purchasing has made the file alter.
- Member file needs backing up daily(at least weekly) because of constant changes…
- would also need to keep a transaction log for the member file.
- Back up copies would be stored away from the building with the computer system in it to ensure that a copy of files survived in case of fire.
- Multiple copies of the book file would be made
- Member file copies and transaction logs may be kept for a number of back up periods.

b)

- When books are discovered to be missing, or if a book is replaced by a more up-to-date edition, the old records should be kept…
- but they are no longer live so are taken from the hard disk…
- after a copy (archive) has been made.
- When members leave the library the data should be archived…
- also when they have not taken a book out for a long period of time their record can be considered to be dormant.

Compiled by Benjamin Muganzi

PROCESSOR COMPONENTS

At the heart of every computer is the **CPU** (**Central Processing Unit**) - the 'brain' of the computer. It refers to that part of the processor which 'processes' the data and instructions.

All arithmetic is performed here, every program instruction is interpreted and executed here, and every item to be placed in memory or retrieved from memory passes through the CPU.

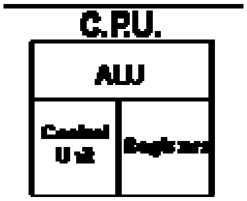
However, the processor would not be much use without **memory** in which to hold the data and the programs which process the data.

The CPU and the main memory of the computer are located in the same physical unit, which is often referred to as the processor unit.

INSIDE THE CPU

A computer consists of the processor, memory, input and output units.

The processor itself consists of two main components:



The main parts of the CPU are

1. The **ALU** (Arithmetic and Logic Unit) - which performs the calculations on the data.
2. The **Control Unit** - which coordinates all operations inside the computer.
3. Some **Memory** storage. A **register** is a place where a single item of data can be stored.

. All computers will have at least one **input device** (eg keyboard, mouse), and at least one **output device** (eg monitor).

All computers have some **memory**. The **program** a computer is running as well as the **data** are all stored in memory.
(Eg When you write a letter using a word processor then both the **word processing program** and the **letter** are stored in memory).

MEMORY

Computer memory, or storage, can be classified into two main divisions:

1. **Main Memory/Primary Memory.**
2. **Backup memory/Secondary memory.**

RANDOM ACCESS MEMORY

- Memory may be one of two types, known as RAM and ROM respectively.

- RAM stands for Random Access Memory, which means that any memory location can be directly accessed by referring to its address.

- Data and programs are transferred into and out of RAM, and the data in RAM is processed according to the instructions in the program.

- There are two types of RAM memory chips: **static RAM** and **dynamic RAM**.

- Static RAM retains its information as long as the power is switched on.

- Dynamic RAM chips are smaller and simpler in design, but generally slower in operation, and the information slowly leaks away and has to be refreshed by special regenerator circuits in the CPU.

- The main memory of most computers uses dynamic RAM chips, with the more expensive and faster static RAM being used for cache memory.

- Both these types of memory lose all information as soon as the power is switched off.

MEMORY MAPPING

- Some RAM is given a special function.

- For example, part of RAM is reserved for the operating system, and some is reserved to hold addresses and parameters used, say, when calling subroutines.

- This is called a **stack**. This allocation of memory to different functions is referred to as **memory mapping**, and a typical memory map for a microcomputer is shown in figure 51.2.

READ ONLY MEMORY

- **ROM (Read Only Memory)** retains information even when the power is switched off, but the information has to be permanently recorded into the circuits when the chip is manufactured, and cannot be changed.

- **ROM** is used to store some of the operating system commands, and the instruction set of the computer.

- In special purpose computers used in video recorders, washing machines and cars, the program instructions are stored in ROM.

- It is also possible to buy Erasable Programmable Read Only Memory Chips (EPROM), which can be cleared by subjecting them to ultraviolet light, and then reprogrammed.

CACHE MEMORY

- A **cache** is a small fast memory usually between 1Kb and 256Kb, too expensive to be used for the whole of RAM that acts as an intermediate store between the CPU and the main memory, and is used to improve the overall speed of the computer.

- Cache memory can return data to the CPU faster than main memory because:

  - It is closer to the CPU;
  - It may use a dedicated control bus;
  - It may use higher-speed components;

- In systems which use cache memory, the most active portions of a program are kept in cache, from where instructions and data can be retrieved faster than ordinary RAM.

- Computers using cache memory are exploiting two principles:

  - An instruction or item of data that has been accessed once is likely to be accessed again the near future. (This is known as **temporal locality**).

  - If an item (instruction or data) is addressed at a given memory location, it is likely that other items with addresses near to this one will be accessed in the near future (**spatial locality**).

- There are good reasons to expect temporal locality. For example, the existence of loops in a program means that the same instruction swill have to be fetched from memory over and over again, and furthermore the data that is operated on may well be the same in many of the instructions.

- Cache memory can also be used as a temporary storage for part of a disk.

- Disk drives are mush slower at storing and retrieving information than memory, so the information that is most likely to be required next from disk is stored in cache from where it can be quickly retrieved.

PRIMARY AND SECONDARY STORAGE

Introduction.

Secondary storage devices hold files that are not currently being used. For a file to be used it must first be copied to main memory first. After any modifications files must be saved to secondary storage. It is advisable to save your data files at the regular intervals as you work on them as data can be lost unexpectedly because of various reasons like interruption in power supply, memory management problems, freezing keyboard, etc.

As secondary storage media can be damaged and files on them become corrupted, it is suggested to make back-up copies of valuable files on a regular basis. Lots of people skip the last but very important step in the backup procedure - check that the backup copy of files is not damaged.

The Difference Between Primary and Secondary Storage.

- Primary storage is *volatile*: when the power is off, all contents of RAM are lost. That is why data from RAM is saved as files on secondary storage which is *non-volatile* and almost permanent (It wears out eventually or becomes out of dated technology)
- *Size*. Secondary storage is virtually infinite: when you run out of space on one disk, you use another. On the contrary there is a limited amount of RAM that can be accessed by the CPU. Some programs will not run on a particular computer system because there is not enough RAM available.
- *Access time*. It takes few nanoseconds for the CPU to access RAM but it takes several milliseconds to access secondary storage. The reasons for RAM being faster are:

1. RAM chips are located on the motherboard so the distance the electrical signals have to travel from the CPU to RAM or in the opposite direction is much shorter compared to the distance between the CPU and secondary storage devices. The shorter the distance, the faster the processing.

2. Also working with the secondary storage involves mechanical operations like spinning

- Cost. Secondary storage is cheaper than RAM in terms of cost per unit of data.

SECONDARY STORAGE DEVICES

- Secondary storage devices are used as a more permanent form of data storage, in contrast with primary storage (main memory) which loses its contents when power is switched off.
- Storage media include:

  1. Magnetic disk (hard disks and floppy disks);

2. Magnetic tape;
3. CD-ROM;
4. WORM disks;
5. Magneto-optical disks.

HARD DISKS

- There are two types of hard disk: **exchangeable** and **fixed**.
- As the names suggest, exchangeable disks can be removed from the drive and another disk inserted, whereas fixed disks, like those in a microcomputer, cannot be removed except for repair or replacement.
- Exchangeable disks are gradually being superseded by fixed disks which are faster and more reliable.
- A hard disk unit consists of several platters each of which is attached to a central spindle and has two recording surfaces.
- There is a read-write head for each surface, with all the read-write heads being mounted on a single unit and moving in and out together.
- Each surface has a number of concentric **tracks**, with each track being divided into a number of **sectors**.
- A sector (typically 256 bytes) is the smallest unit that can be read or written in a single operation.
- All the tracks that are accessible from a single position of the read-write heads form a **cylinder**; data is recorded cylinder to minimise movement of the read-write heads, thereby minimising access time.
- A hard disk typically has a storage capacity of between 512 Kb and several gigabytes.

FLOPPY DISKS

- **Floppy disks** consist of a thin piece of mylar plastic coated with ferric oxide, enclosed in a protective jacket.
- The disk is read through an opening in the jacket, protected in the standard 3 1/2" disk by a metal sleeve until the disk is inserted in the drive and accessed.
- The disk has a small hole in the corner, which when open causes the disk to be **write-protected** so that the data on the disk cannot be accidentally overwritten.
- Floppy disks, like hard disks are divided into tracks and sectors.
- A high-density disk can store 1.44 Mb of data.
- Before being used, a disk has to be formatted, a process which checks the disk for unusable areas, marks out tracks and sectors and creates the disk directory.

CD ROM

- **CD ROM (Compact Disk Read Only Memory)** is one of the three main forms of optical disk, the other two being WORM disks and magneto-optical disks.
- A CD is created using a mastering facility (just as audio CDs are created in a recording studio) and copies of master are duplicated or 'pressed' at a factory.
- They are used for many different applications including:

- software distribution (e.g. word for windows, windows 95);
- games;
- reference material such as encyclopedias, telephone directories, past copies of newspapers, art catalogues, dictionaries;
- advertising, such as the free CDs commonly distributed with computer magazines.

- The capacity of a CD ROM is around 650Mb, about 500 times as much as a high-density floppy disk.
- Their high capacity allows storage of photographic images, sound and video, making them ideal for **multimedia** applications.
- When the master disk is created, a laser beam burns tiny holes in the surface of the disk, which has a single spiral track divided into sectors.
- To read data from the disk, a laser beam is reflected off the surface of the disk, detecting the presence or absence of pits which represent binary digits.

WORM DISKS

- **Write Once, Read Many** optical laser disks (**WORM disks)** look similar to CD ROM disks, but they are often gold rather than silver in colour.
- An end user company can use these disks to write their own material, typically for archiving or storing say, graphic or photographic images which will not be changed.
- These disks are also widely used for pirated software; whereas silver CDs are pressed in factories, gold CDs are usually written one at a time on PCs in garages and back bedrooms.
- However, because there is a lot of competition among pirates, these CDs sometimes carry viruses which can cause havoc on a hard drive.

MAGNETO-OPTICAL DISKS

- **Magneto-optical disks** integrate magnetic and laser technology to enable read and write magnetic storage.
- A 5 1/2" disk can store up to 1000Mb.
- These disks may in the future replace current magnetic disks, but at present the technology is still developing and the disks are too expensive, slow and unreliable

1.5 Data transmission and networking

Network:

A network is a collection of computers, printers, scanners, routers, bridges, switches and other devices which are connected to each other by some transmission medium.

The devices connected in the network are called nodes of the network.

Local Area network (LAN)

This is a high-speed communications system designed to link computers and other data processing devices together within a **small geographic area**, such as a workgroup, department, or building.

This allows users to electronically share vital computing resources, such as expensive hardware (e.g. printers and CD-ROM drives), application programs, and information.

Wide Area Network (WAN)

This is a computer network that covers a far wider area than a LAN (Local Area Network). WANs cover cities, countries, continents and the whole world.

A **WAN** is formed by linking LANs together. For example, several major LANs in a city can connect together forming a WAN.

When all WANs in the world connect forming a global network, we call it **The Internet**. That's why the Internet is always written with a capital I. It is the biggest WAN we have!

How LANs differ from WANs

LAN	WAN
LAN exists in a small geographical area.	WAN exists in a wider geographical area.
LAN is usually hard-wired (uses cables to connect computers)	WAN requires other transmission media to connect devices except for cables
Data in a LAN is easier to keep secured	Data in a WAN is prone to hacking
Computers in a LAN can share hardware such as printers	Computers in a WAN cannot share hardware such as printers

To summarize:

- Computers can be linked together to form networks
- If the distances are short the network is called a LAN, if longer the network is a WAN
- Networks allow computers to communicate

- Networks allow the sharing of both hardware and software.

Hardware and software needed for communication in a LAN

- Network interface card which is needed by the computers to get connected to each other
- Cables which are used as the transmission medium
- Hub which is used to concentrate the cables connected to the computers
- Network operating system which is needed by the computers to get connected to each other and communicate data and information
- Hardware drivers for the network interface card and the modem which are needed to convert data and information to a format that the processors of the communicating computers can understand

Hardware and software needed for communication in a WAN

- Network interface card which is used to connect computers by cables
- Modem, ISDN or ADSL adapter which is needed to access the telephone system
- Dial up software which is needed to connect to a Web server or to access the telephone lines

Network Topologies

A network topology is the physical layout of network nodes. The following are the most widely used topologies of local area networks.

- Bus (Linear, Linear Bus)
- Star
- Ring

Linear/ Bus Topology

A linear bus topology consists of a main run of cable with a terminator at each end (See fig. 1). All nodes (file server, workstations, and peripherals) are connected to the linear cable. Ethernet networks use a linear bus topology.

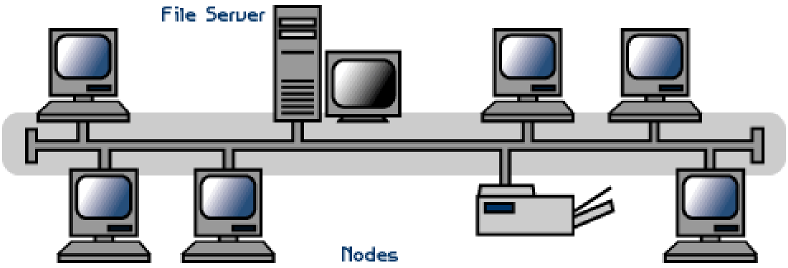


***Fig. 1**. Linear Bus topology*

Advantages of a Linear Bus Topology

- Easy to connect a computer or peripheral to a linear bus.
- Requires less cable length than a star topology.

Disadvantages of a Linear Bus Topology

- Entire network shuts down if there is a break in the main cable.
- Terminators are required at both ends of the backbone cable.
- Difficult to identify the problem if the entire network shuts down.
- Not meant to be used as a stand-alone solution in a large building.

Star Topology

A star topology is designed with each node (file server, workstations, and peripherals) connected directly to a central network hub or concentrator (See fig. 2).

Data on a star network passes through the hub or concentrator before continuing to its destination. The hub or concentrator manages and controls all functions of the network. It also acts as a repeater for the data flow. This configuration is common with twisted pair cable; however, it can also be used with coaxial cable or fiber optic cable.

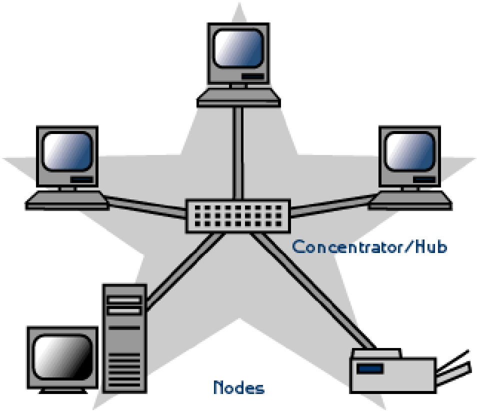


Fig. 2. Star topology

Advantages of a Star Topology

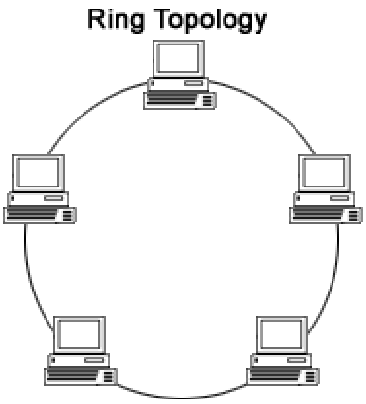
- Easy to install and wire.
- No disruptions to the network then connecting or removing devices.
- Easy to detect faults and to remove parts.

Disadvantages of a Star Topology

- Requires more cable length than a linear topology.
- If the hub or concentrator fails, nodes attached are disabled.
- More expensive than linear bus topologies because of the cost of the concentrators

Ring Topology

In a ring network, the two ends of the bus are joined up. This creates a loop so that data can be sent in both directions along the cable. If there is a break in the cable, the network will continue to operate because the data can go in the other direction.



Advantages

- Very orderly network where every device has access to the token and the opportunity to transmit
- Performs better than a star topology under heavy network load
- Can create much larger network using Token Ring
- Does not require network server to manage the connectivity between the computers

Disadvantages

- One malfunctioning workstation or bad port in the MAU can create problems for the entire network
- Moves, adds and changes of devices can affect the network
- Network adapter cards are much more expensive than Ethernet cards and hubs
- Much slower than an Ethernet network under normal load

Data Transmission:

Data transmission is the conveyance of any kind of information from one space to another.

In computer terms, it means sending a stream of bits or bytes from one location to another using any number of technologies, such as copper wire, optical fiber, laser, radio, or infra-red light. We have the following types of transmission:

Serial transmission:

Serial transmission is the transmission of data one bit behind another along a single wire.

- Serial transmission has the advantages of being simple and reliable because the next bit is not transmitted until the current one has arrived at its destination.
- However, because only one bit can be transmitted at a time, the speed of transmission is slow.

Parallel transmission:

Multiple wires are used and transmit multiple bits simultaneously.

- Parallel transmission of data is obviously faster than serial because all the bits are travelling at the same time,
- But because of the fine tolerances in the transmission, it is less reliable as the bits can become muddled up.

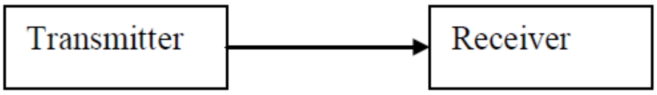
Examples of different types of data transmission

- The data transmission in a network using cables is serial
- The Data transmission inside the computer using the bus system is parallel. This is because of the very high speed of data transmission needed by the computer's processor in order to process them. For example the data transmission between the primary memory and the hard drive of the computer is parallel because the processor requires the data to be downloaded to the memory as quickly as possible
- The data transmission between the computer and the printer is parallel because the computer needs to send a big chunk of data to the printer's buffer quickly so that the processor will be free to attend to other tasks of the computer

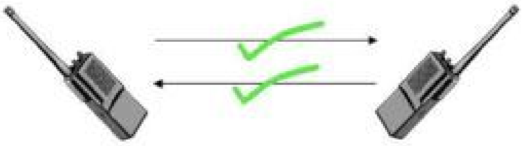
Modes of Transmission:

If data is to be transmitted between devices, there are three different modes of transmission possible.

a) **Simplex mode.** If data travelling between 2 devices can only travel in one direction then it is known as a SIMPLEX transmission. A good example is teletext information which is passed to a television receiver, but there is no way to send data in the other direction.

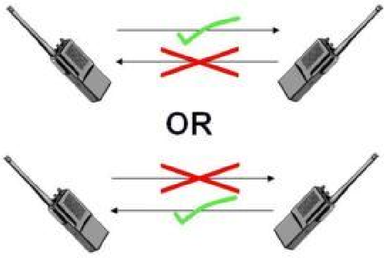


b) **Duplex mode.** When data can pass in both directions at the same time it is known as DUPLEX or FULL DUPLEX transmission.
An example of duplex transmission would be a telephone conversation as both users can speak and be heard at the same time.



c) **Half duplex mode**. When data can pass in both directions, but only in one direction at a time, the transmission is called HALF DUPLEX.
An example of half duplex is a CB radio system ("walkie-talkie") in which each handset can either be set to the receive mode or send mode.

A simple illustration of a half-duplex communication system:



Examples of different modes of data transmission

- ❖ The data transmission between the primary memory and the hard drive of the computer is half duplex.
- ❖ The data transmission between the computer and the printer is half duplex because the transmission of data files to the printer from the computer and the passing messages to the computer from the printer such as error messages happen at different times

Bit Rates

Information needs to be sent to devices in a computer system from other devices. For example, a picture stored in the memory of the computer needs to be sent down a telephone line to another computer. Remember that the more pixels that there are and the more colours that can be represented, the better the picture will be. However there is a limit to the amount of information that can be transmitted in a given time down the phone line, so a decision needs to be made. One choice is to have as much detail as possible and accept that it will take a long time to transmit it. The other is to limit the resolution of the picture, meaning that there is less information to send so that the message will be sent more quickly.

The number of bits that can be sent in one second is known as the BIT RATE. The units used to measure the bit rate are BAUD. **1 baud = 1 bit per second.**

Note that text can be sent much more quickly than other forms of information because it needs far fewer bits (1 byte per character) than other types of data. When data other than text is being transmitted, e.g. on the Internet, it is important to limit the amount of data that needs to be sent or the time it takes to download the data can be unreasonably long. The data can be limited by such simple things as reducing the size of pictures so that they only take up a small part of the screen, or that they are restricted to a few colours. Speeding up the transmission of the information by reducing the amount of data that is sent is known as compression. This will be studied in more detail in the second year of the course.

Error checking and correcting

When data, of whatever type, is being transmitted from one part of a computer system to another, it is transmitted as a series of binary digits. Any data that is transmitted is going to be made up of a very large number of bits. Consequently, there are bound to be occasions when the data is not transmitted correctly. Luckily there are only two possible mistakes that can occur, either a 1 is received as a 0, or a 0 is received as a 1. Mistakes occur rarely, but when they do occur they can be very serious, as the information is no longer correct. This makes it very important that there should be methods for checking the data when it has been transmitted.

a) Echoing Back. The simplest way of checking the transfer of the data is to send it back again. If the data that is sent back is the same as the data that was sent in the first place then the original data must have reached its destination correctly, if not then it needs to be sent again. This is known as ECHOING BACK. Echoing back is very effective, but suffers from having to send data twice, thus taking longer than necessary, and needing to be a duplex, or half duplex, system to allow data transfer in both directions.

b) Parity. All data is transmitted as bits (0s and 1s). The Number of 1s in a byte must always be either an odd number or an even number. If two devices that are communicating data decide that there will always be an odd number of 1s, then if a byte is received that has an even number of 1s, an error must have occurred. E.g. the byte 01011000 has 3 ones in it. 3 is an odd number, so it fits the rule that it must have an odd number of ones. When it is sent there is an error in transmission so that the first bit is received as a one. So, the byte received is 11011000. This has 4 ones in it, which is an even number, so there must be an error. The receiving device would ask for it to be sent again.

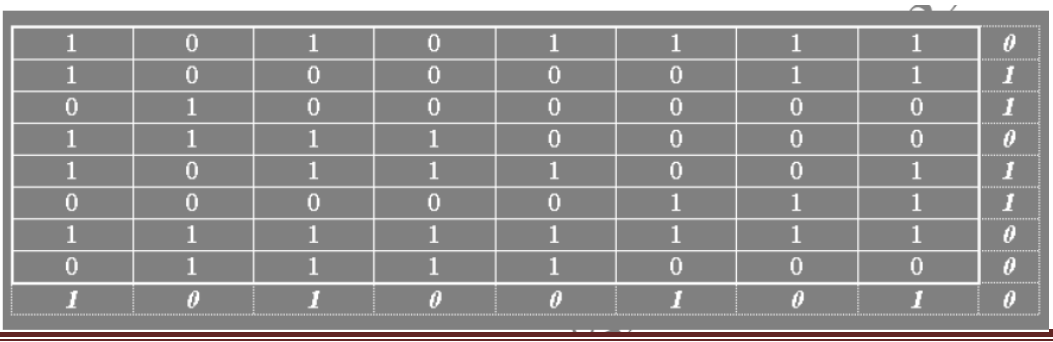
Notes:

- If two mistakes are made in the same byte they will cancel each other out and the faulty data will be accepted. This problem can be overcome, and in the same way, a clever way of correcting error mistakes can be implemented. This method is not part of this course.
- Earlier in this course it was said that a byte was the number of bits necessary to hold a character code. Specifically, an ASCII character uses 8 bits in a byte, giving 256 different characters. This is not true because one of the bits has to be reserved for a parity bit, the bit that can change to make sure that the number of 1's is always odd. This means that there are 128 different characters possible.
- The implication in all the above work is that odd parity is always used. Even parity can equally well be used, whichever has been agreed between the two devices.
- Parity is not only used during data transfer between devices, but also when data is transferred between different parts of the CPU.

c) Check Sum. Data will normally be sent from one place to another as a block of bytes rather than as individual bytes. The computer can add numbers together without any trouble, so another checking procedure is to add all the bytes together that are being sent in the block of data. The carry, out of the byte, is not taken into account, so the answer is an 8 bit number, just like the bytes. This answer is calculated before the data is sent, and then calculated again when it is received, and if there are no errors in the transmission, the two answers will match.

Parity Blocks and Parallel Parity

It is an error-checking technique involving the comparison of a transmitted block check character with one calculated by the receiving device. Parallel parity is based on regular parity. Parallel parity can detect whether or not there was an error, furthermore it can detect which bit has flipped. This method is implemented on a block of data which is made of sum words; the parity bit is then added to the columns and rows. Following an example:

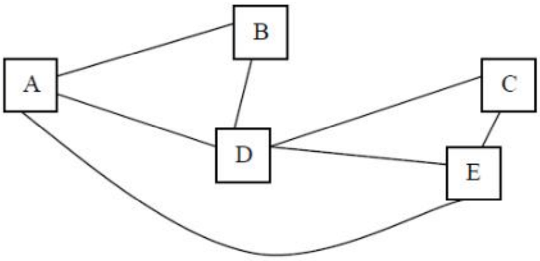


1	0	1	0	1	1	1	1	*0*
1	0	0	0	0	0	1	1	*1*
0	1	0	0	0	0	0	0	*1*
1	1	1	1	0	0	0	0	*0*
1	0	1	1	1	0	0	1	*1*
0	0	0	0	0	1	1	1	*1*
1	1	1	1	1	1	1	1	*0*
0	1	1	1	1	0	0	0	*0*
1	*0*	*1*	*0*	*0*	*1*	*0*	*1*	*0*

As we can see in the example above, every row has its parity bit, and every column has its parity bit. If one bit in the block has flipped, we get two parity errors, one in the row's parity and one in the column's parity, by intercepting the row and column we can detect the bit that has flipped.

Switching

When a message is being sent from one machine to another, particularly over a wide area network, the message may have to pass through other machines first. This may be forced on the system because there is no direct route from one machine to another.



In the network shown, it would be easy to send a message from A to D or from A to B because A is directly connected to both of them. However, sending a message from A to C is much more difficult because there is no direct route. There are two ways that the message can be sent

a) **Packet switching.** The message is split into a number of equal sized packets. Each packet has a label saying where it is meant to be going and what number packet it is. These packets are sent along communication lines towards the destination. Each time a packet reaches a node on the network the node decides which direction to send it on. So, one packet in the message from A reaches node D. The obvious route to take is the one directly to C, but it is already in use for another message, so D decides to send it to E instead. The next packet arrives at D and, this time, the line to C is free, so the packet is sent direct to C. When the message has all arrived at C it has to be reassembled in the correct order.

b) **Circuit switching.** Before the message is sent, the network reserves a route from A to C. The message can then be sent directly from A to C and will not need to be reordered when it gets there.
Packet switching allows optimum use of the connections around the network because as many routes are in use at one time as possible, whereas circuit switching means that the whole message is kept together so it does not need to be reassembled at the destination.

Protocols

When data is being transferred from one place to another in a computer system or from one computer/device to another computer/device there must be rules set up as to how the transfer is going to be done. Every single computer follows these agreed rules within the network.

This set of agreed rules is known as a protocol or network protocol.

There are a number of protocols that you may have heard of e.g. http, tcp(ip) but don't worry about them, you do not need to know specific examples of protocols, but you should be able to discuss some of the rules that go to make up a particular protocol. Some of these common rules are:

- Common error checks of the data packets when they arrive
- Speed across the network
- Agreed method to tell the data receiving machine that the sending machine has finished sending all of the data
- Method or signal to tell the other machine that the data has arrived correctly e.g. echo back
- Agreed error correction of the data packets on both ends
- If the data is compressed before sending it to other computer then does the protocol allow this to take place and if so then what method does it use?

Physical and logical protocols and interface layering

Computer nowadays are very popular gadgets with masses. Many manufacturers are coming up with different brands of the computers. With the passage of time these computers are getting cheaper and that is one more reason for their popularity.

Availability of computers and their proliferation gives increase to the idea to network them together for information and resource sharing. To make these computers talk to each other and share the resources and information many organizations have made standards. Without such standards computers cannot be networked to communicate with each other.

Interface

An interface is the connection between devices over which data can be sent. It is usually used interchangeably with 'protocol'. It includes hardware and software such as the cables connecting the devices, error correction and how messages are routed from place to place.

Interface design

There are two methods of designing interfaces:

1. Design it as one unit: This is how things were done in the very early days. Each program needed its own network interface.
2. Design it as a series of layers: This is how interfaces are designed nowadays. Each layer deals with a "rule" of the protocol, e.g. How error checking is carried out.

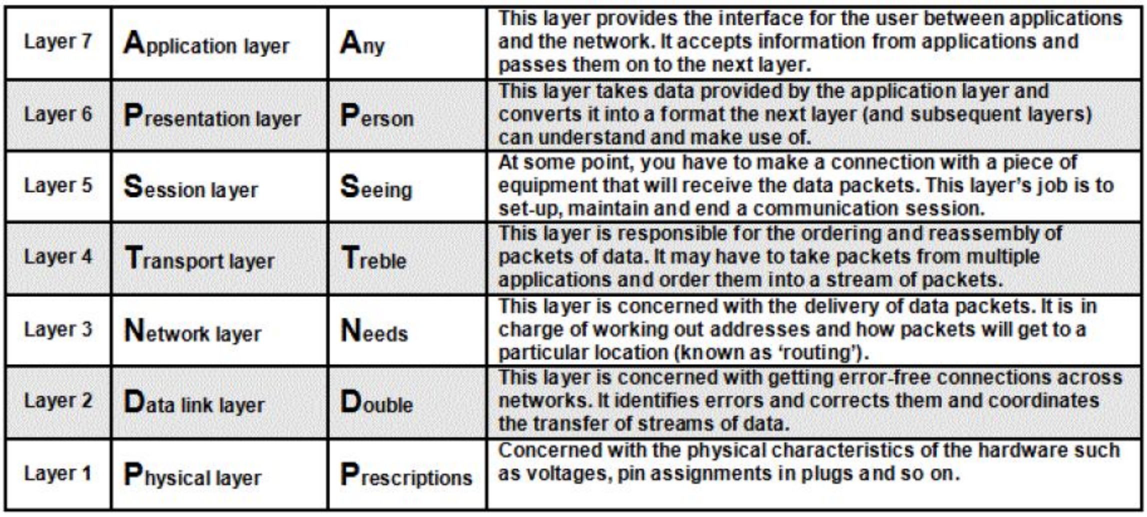
Why layering is used in a network system?

All layers work together to make networked computer share information with other computers that are also following the same layer model. Layering breaks the protocols down into different sections. Changes in one layer don't affect other layers. Layering allows each layer to be treated independently and enables security measures to be used for data transmission.

One of interface standards is called OSI model. OSI stands for Open Systems Interconnect. Protocols are combined with the OSI layers model. There are seven layers in the OSI model and each layer performs different function. These seven layers are described below and their individual names and details of working are not required to be remembered for an A level computing 9691 student.

Each layer knows how to communicate with the upper and lower layer as they can only talk to layers above or below them. You can remember the name of all the layers by the following sentence.

"Any person seeing treble needs double prescriptions."



Layer 7	Application layer	Any	This layer provides the interface for the user between applications and the network. It accepts information from applications and passes them on to the next layer.
Layer 6	Presentation layer	Person	This layer takes data provided by the application layer and converts it into a format the next layer (and subsequent layers) can understand and make use of.
Layer 5	Session layer	Seeing	At some point, you have to make a connection with a piece of equipment that will receive the data packets. This layer's job is to set-up, maintain and end a communication session.
Layer 4	Transport layer	Treble	This layer is responsible for the ordering and reassembly of packets of data. It may have to take packets from multiple applications and order them into a stream of packets.
Layer 3	Network layer	Needs	This layer is concerned with the delivery of data packets. It is in charge of working out addresses and how packets will get to a particular location (known as 'routing').
Layer 2	Data link layer	Double	This layer is concerned with getting error-free connections across networks. It identifies errors and corrects them and coordinates the transfer of streams of data.
Layer 1	Physical layer	Prescriptions	Concerned with the physical characteristics of the hardware such as voltages, pin assignments in plugs and so on.

Advantages of layering

The main advantage of a layered interface is that changes can be made to one layer without having to change them all. For example:

- If a manufacturer was to design a new network card, they would only need to design the way it communicates with the data link layer.
- If a software company was to design a new web browser, they would only need to design the way it communicates with the presentation layer.

If layered interfaces were not used, these manufacturers would have to design all the components of the interface.

Key points about layering

- If a layer needs to be changed, it only needs to communicate with the layer above and below
- Each layer does a specific job
- Data flow through the layers in order as described in the diagram above.

Protocols

A protocol is a set of rules that defines how data are sent over a connection. Without protocols, devices wouldn't know what format to expect data in.

Protocols include rules such as:

- Bit rate
- Transmission medium
- Error correction method
- Mode of transmission such as Serial/parallel, simplex/half duplex/duplex

Physical protocols

These are protocols concerned with how a device connects to a medium. For example how a printer connects with the cable. They ensure that devices can transmit through the medium. These are the protocols that make sure that the layout of pins on the connectors is the same and hubs and switches are correctly connected and configured. Few examples of physical protocols are 802.11a/b/g/n for WiFi connections and DSL for broadband.

Logical protocols

Logical protocols are concerned with data handling before it actually reaches the transmission medium and after it has been received from the transmission medium. Logical protocols ensure that data are in the right format for the application, the bit rates match at both ends, and both ends are using the same mode, e.g. simplex and the same error detection is used at both ends, etc. Few examples of logical protocols are HTTP, POP, TCP/IP, FTP etc.

Example Questions

1. Explain the difference between a wide area network (WAN) and a local area network (LAN). (2)

2. a) State three pieces of hardware that are needed to create a LAN from a set of stand alone computers. (3)
b) Explain why the communication over a WAN differs from that across a LAN and state how the hardware necessary for communication would differ from that used in part (a). (3)

3. Explain the difference between
(i) simplex
(ii) half duplex
(iii) duplex
transmission of data, giving an example of the use of each. (6)

4. Explain why the bit rate is more important when sending a colour picture from one device to another, than it is when sending a page of text. (3)

5. The following bytes of data are received by one device after being transmitted from another.

01001101
10001000
10101011
00011011

An automatic checking technique is used to check that the data has been transmitted without error.
a) State which byte has been received incorrectly, explaining how you arrived at your answer. (3)
b) Explain why it is possible that a byte of data could still be incorrect despite passing the test that you used in part (a). (1)

6. Explain the difference between packet switching and circuit switching. (2)

7. A computer is to use a printer to provide hardcopy output of jobs. In order for the data to be transmitted and received properly a protocol must be set up between the two pieces of hardware.

State two parts of the protocol which would be essential in this example, giving reasons why they are necessary. (4)

8. Define a protocol. (3)

9. Explain the need for protocols. (2)

10. What is meant by a layered interface? (2)

11. Describe the differences between logical and physical protocols. (4)

12. Describe the need for layered interfaces. (3)

13. What is meant by the OSI model? (2)

14. Why do we need the OSI model? (2)

15. Why are the protocols grouped into different layers - what are the advantages of layering? (3)

Systems Development Life Cycle

Commercial systems (payroll, accounts, stock control and so on) all share a common life cycle pattern. One method of doing things may work well for a period of time (maybe several years) and then, owing to expansion or changes in the nature of the business, the economic environment, the need to keep up with new technology or other factors, the system may start to deteriorate or seem inadequate. At this point investigations are made, requirements are analyzed, a new system is proposed and the cycle starts all over again.

THE SYSTEMS DEVELOPMENT

1. The systems life cycle
2. Feasibility study
3. Data Flow diagrams
4. Systems Flow charts

DESIGN TO EVALUATION

1. Systems design and development
2. Systems implementation and maintenance

The Systems Life Cycle

The Role of the Systems Analyst

The systems analyst has a key role to play in developing computer systems. It is the analyst's job to

- Analyse the data processing requirements of the organization
- Decide whether computerization should be introduced, or the current computer system modified or changed
- Specify how the new computer system should work, and what the hardware and software requirements will be
- Be responsible for implementing the new system and ensuring that it works efficiently.

The systems analyst therefore has to have a good understanding of the nature of the business or organization by whom he / she is employed, and a thorough understanding of how computers can be used. He or she needs to be an excellent communicator, capable of extracting the required information about the current system from people in the user departments without alienating them.

The Stages in Systems Development

1. **Analysis** : The problem must first be identified and defined, and written down in a document called the terms of reference. A preliminary investigation will then determine whether the request justifies further detailed investigation and analysis. Detailed systems analysis will then follow, involving a thorough study of the current system and a proposed solution to any problems found. Sometimes the systems analyst is asked to prepare a feasibility study and a cost / benefit analysis. The results of the analyst's work will be presented in a written report to both user and information systems management who consider the alternatives and the resources, such as time, people and money, of the organization. If a decision is made to proceed, the project enter the design phase.

2. **Design** : The design stage involves a number of tasks such as designing the output, input, files, database if applicable, system controls and test plan. Input forms must be designed, clerical procedures laid down and all aspects of the design must be documented.

3. **Development** : There are two aspects to development: program development and equipment acquisition. The systems analyst or sometimes a senior programmer will write program specifications to describe what each program in the system will do and how it will do it. Programmers will then code, test and document the programs.

4. **Implementation** : This is the phase of the systems development when the new system becomes operational. It is critical phase of the project, requiring careful timing and the coordination and training of all the user departments involved.

5. **Maintenance** : All systems need to be **maintained**; that is, performance monitored, modifications made if required, errors corrected, documentation kept up-to-date.

What Prompts A New System

Some manual systems have characteristics that make them obvious candidates for computerization. These characteristics include;

- The current system is no longer suitable for its purpose.
- Technological developments have left it outdated.
- It has become too inflexible or too expensive to maintain.
- Large volumes of data all requiring similar processing (e.g. payroll, invoicing, mail-order processing).
- Requirement for information to be available from several different locations (e.g. holiday booking, hospital patients database, customer accounts);
- Requirement for very accurate calculations (e.g. engineering projects or process control)

· Duplicated effort is involved (e.g. billing dental patients and then re-entering the same data to reclaim from the Dental Practitioner's Board)

· Manual methods are too slow (e.g. numerical weather forecasting, simulation problems).

Feasibility Study

The purpose of the feasibility study is to find out whether the system is suitable for computerisation before going any further and buying hardware etc. The analysts will make a brief study of the existing system and using their judgement and experience will decide whether it is suitable for computerisation. This may include a consideration of

· Technical feasibility. Is it technically possible to implement the system?

· Economic feasibility. Will it cost more to implement than it is worth?

· Legal feasibility. Will the system contravene any legislation e.g. the Data Protection Act?

· Operational feasibility. Will the system work in the context of the organization and its staff?

· Schedule feasibility. Can the system be implemented quickly enough to be of any use?

Finally the analysts will produce a feasibility report, explaining their ideas for the new system and giving estimates of the cost, and producing a time schedule for the development of the system. The management will look at this report before deciding whether to go ahead.

Systems Analysis (WHAT the system must do)

The analysis stage involves the analysts finding out in as much detail as possible WHAT the new system will have to do.

Much of the information that the analysts need to know will have to come from the people who work at the company and there are three basic methods for getting this information.

Interviewing

This is not quite the same as a cosy chat. The analysts will need to be prepared and have a list of questions that they want to ask before the interview starts. They will also need to record the answers in some way, as they could easily forget something if they left it until after the interview before writing anything down. A further problem is that the staff may feel nervous in an interview and so not be entirely honest e.g. telling the analysts what they think their job is supposed to be rather than what they actually do.

Questionnaires

These enable the analysts to gather a lot of information in a short time, but the information still has to be analysed and it may be that the results of the questionnaire still leave some questions unanswered.

Observation

Watching people at work can be a useful way of seeing what the actually do, and also seeing what problems they have or cause. It may well be the case that people behave in a different fashion when they are being watched.

It may be the case that the analysts employs a mixture of all three approaches, for example if an analyst is asked to look at a sales order system then they may use observation to follow an order through from when it is placed to when the goods are sent. They might then send out a questionnaire to staff asking them to identify problems that they find and then follow this up by interviewing the staff.

At the end of this fact finding session the analysts should have written descriptions of the outputs that the system produces or is required to produce, the inputs that come in to the system, and any files or other sources that the system uses.

Data Flow Diagrams

When a systems analyst looks at a system, a major task to be performed is to find out

· Where the data originates;

· What processing is performed on it and by whom;

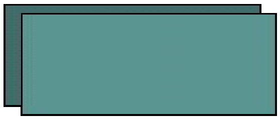
· Who uses the data;

· What data is stored and where;

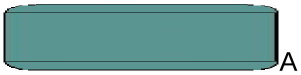
· What output is produced and who receives it.

One way of recording all this information is to use a **data flow diagram** (DFD). There are only four symbols used in dataflow diagrams, and they should not be confused with any other type of flowcharting symbols.

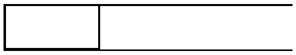
SYMBOLS FOR DATA FLOW DIAGRAMS



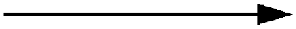
A data source or destination.



A process that is performed on the data



A data store or a file



A data flow which should be labeled

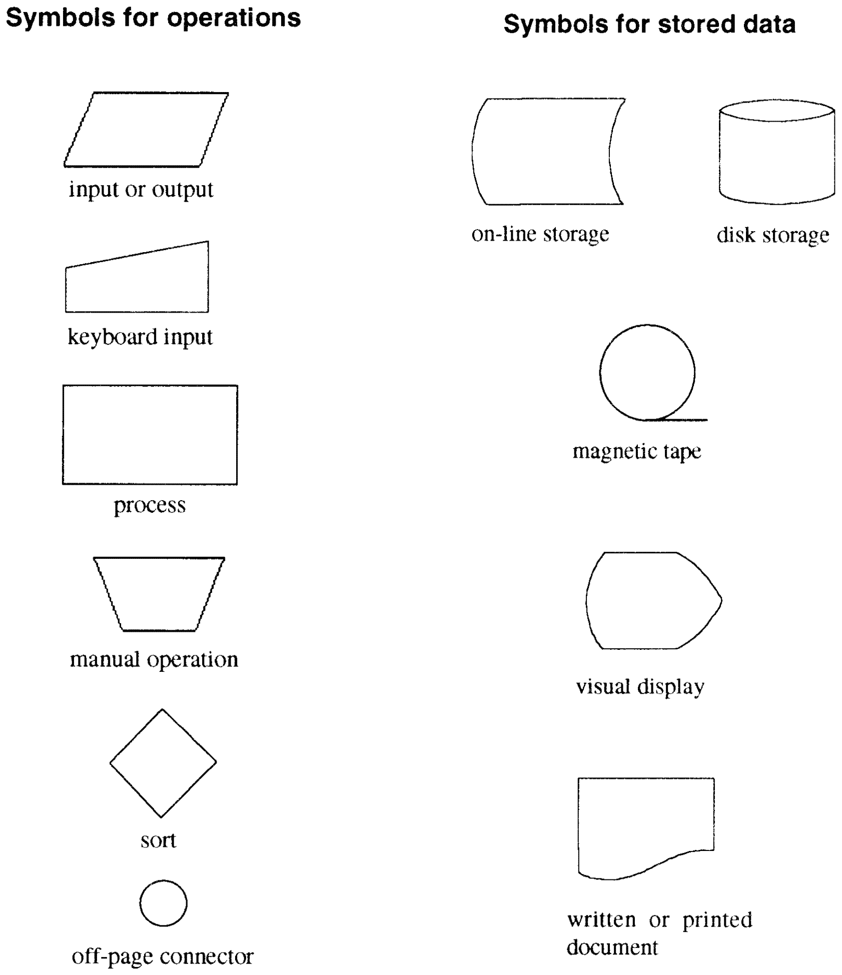
Systems Flowcharts

Systems Flowchart Symbols

When a system analyst is developing a new computer system, his ideas need to be written down. Frequently a pictorial representation of how the system will work is easier to understand and take in than a lengthy text. A systems flowchart is a diagram showing an overview of a complete system. It will show

- The tasks to be carried out in the new system, whether manual or by the computer
- The devices (disk drives, tape drives, terminals etc) that are to be used in the system
- The media used for input, storage and output
- The files used by the system

You should be familiar with the standard symbols used in systems flowcharts.flowcharts.



Drawing a System Flowchart

There are many factors to be taken into account when designing a new system. The analyst must establish the following facts:

- Is this a batch, on-line or real-time system?
- How is the data to be captured and input to the system? What manual procedures will be involved, and how will errors be prevented, or detected and corrected if they slip through?

- What is the hit rate on the master files, and what file organization is therefore appropriate?

- What storage media will be used; disk, tape, or some other medium?

- What will be the output from the system?

- What are the processing step to be carried out?

Example:

A customer file is held on tape and receipts held on a transaction file are sorted and then used to update the master file, creating a new master file. Draw a systems flowchart to illustrate this process.

Systems Design

Systems Design (HOW the systems will do it)
Systems Specification

The systems specification must describe how the new system will work. Screen layouts and report formats must be designed, file contents and organization specified, and each program in the system must be described by means of program specifications, structure charts, pseudo code or flowcharts.

The programmers must then code, test and debug all the programs in the system. In smaller organizations the roles of programmer and analyst may overlap, and in some cases the ‘analyst / programmer’ may design, code and test the programs.

The systems designer will consider

· Output: content, format, sequence, frequency, medium (e.g. Screen or hard copy) etc;

· Input: documents, screens and dialogues;

· Files: contents, record layout, organization and access methods;

· Processing: the programs and procedures needed and their detailed design;

· Security: how the data is to be kept secure from accidental corruption or deliberate tampering or hacking;

· Testing strategies: how the system is to be thoroughly tested before going ‘live’;

· Hardware: selection of an appropriate configuration.

The decisions made about the above will be written down and form the systems specification. The specification will describe the new system completely, but as it is going to be used to develop the new system it may be written in different parts, so that the appropriate parts can be used by different people. E.g.

· Hardware requirements, to be used for purchasing and by engineers for installing hardware.

· Software requirements, to be used for purchasing software.

· Input and Output design, including all paper based records and the user interface to the system

· File structures and organisations, with details of all the fields used.

· File creation procedures, to be used by the data processing department in creating the necessary files.

· Program specification processing tasks, to be used by programmers to write or adapt the software.

· Testing Strategy

Choosing Software

When choosing the software to be used for the new system the analysts will consider:

· Usability – will it serve users needs

· Performance – will it work with a full set of data

· Suitability – will it integrate with the existing systems

· Maintainability – will it be easy to update

There are three main choices are usually:

· Writing bespoke software. A team of programmers will be needed to write the software for the system. This means that the system should do precisely what the company wants, but it will be expensive.

· Using existing packages. If the system is a fairly standard one then it may be the case that there is an applications package already written for it, e.g. most companies accounting procedures will be similar so the company might be able to buy a standard accounting package. This will be much cheaper than having the software written specially for them, but it may mean that they have to make slight changes to the way they operate.

· Tailoring packaged software. This is a mixture of the two above, where an existing package is used but it has modifications made to it to suit the companies needs.

Choice of Hardware

The choice of hardware may depend on many factors, including :

· The volume of data

· The number of users

· The location of the users - are they all in one office or spread around the country?

· The type of user - is this a system to be used by the general public, or by technical specialists?

· The nature of the system - batch or on-line

· The hardware currently in use

· Security considerations

· The software. In particular, where a software package is to be used, this often dictates the choice of hardware, or at any rate narrows the choice.

Program Design

This involves drawing structure charts and writing detailed program specifications. Examples are top down design, bottom up design, modular design, hierarchy charts, structure charts and Jackson Structure diagrams.

PROTOTYPING

Prototyping is the creation of a model of the system which is not necessarily completely functional. It involves building a working model of a system in order to evaluate it, test it and have it approved before building the final product.

When applied to computer systems, this could involve, for example, using special software to quickly design input screens, and running a program to input data. The user can then experience the ‘look and feel’ of the input process and suggest alterations before going any further. The aim is to give the user a chance to experience what the system will look like when it is complete and to make it easier for the analyst and the user to discuss the details of the solution.

The prototype can be thrown away when making final system (throw-away prototyping). Alternatively it can be further developed into the final system (evolutionary prototyping).

Testing and Development

Testing

Choosing Test Data

Test data chosen to test every part of the system

- Typical data **-** then kind of data that the system will be expected to handle correctly on a daily basis.
- Extreme data **-** data that is still correct but falls right on the limits. An obvious example would be an exam mark of 100%, but a more subtle example would be an exam mark of 80% if this was the grade A/B borderline.
- Erroneous data **-** data that is incorrect and that should be identified and dealt with by validation checks in the program.

Testing Strategies

Obviously a system must be thoroughly tested before bring installed to make sure that all errors are discovered and corrected before going ‘live’. It is part of the designer’s job to come up with a test strategy which will ensure that all parts of the system are properly tested.

Different strategies that may be used for testing include:

- Dry run testing – follow the code manually and make a trace table
- Unit/Module testing – testing of individual modules or subroutines
- Integration testing – testing that it works when you put it together

Program Testing

There are two possible strategies:

Bottom-up testing

- Each individual module is tested as soon as it is written using pre-prepared test data.

- Each complete program in the system is tested.

Top-down testing

The skeleton of the complete system is tested, with individual modules being replaced by 'stubs' which may, for example, display a message to say that a certain procedure has been executed. As individual modules are completed they are included in subsequent tests.

System Testing

The system as a whole is tested using the following types of test:

- **Functional testing**: the system is tested to make sure all parts of it work correctly using test data.

- **Recovery testing**: tests are carried out to ensure that the system can recover from various types of failure including hardware failures or power failures.

- **Performance testing**: this is carried out to ensure that the system performs satisfactorily with a realistic volume of data, in conditions which will prevail in the user environment.

Development

This stage involves coding and testing programs according to the test plan. If a software package is being used it will probably involve tailoring the package, implementing screen designs and reports, writing macros, etc. once the system is complete, acceptance testing may take place.

Acceptance testing

Once the system is installed, acceptance testing may be carried out under the direction of the customer, who may provide the test plan. It has the following objectives:

To confirm that the system delivered meets the original customer specifications;

To find out whether any major changes in operating procedures will be added;

To test the system in the environment in which it will run, with realistic volumes of data.

Implementation

This is the stage in the systems life cycle when people actually begin to use a new system. There are several tasks to be faced before the changeover is complete.

- Hardware installation : before a new system can be put into operation, any new hardware will have to be installed. Even if it is only a matter of bringing in a couple of new PCs, this may mean changing office layouts, rewiring, acquiring new office furniture and moving personnel. In the case of a new mainframe, it will probably involve putting in a false floor in a specially designed computer room, laying cables and installing air-conditioning.

- File conversion : which can be a major part of the implementation. It often means converting old manual records into a medium usable by the computer, and this may involve hiring extra help to key in and verify data. Imagine keying in the details of 20,000 existing books for a new library system.

· Staff training : which may be done by sending staff on special courses, bringing in an expert to teach small groups of staff, or allowing staff time for computer-aided training at their own desks.

Methods of Conversion

Once the new system has been fully tested and accepted the changeover can be made. This can be done using any of the following strategies:

Direct Changeover.

The user stops using the system one day and starts using the new system the next - usually over a weekend or during a slack period. This may be unavoidable when the new system is quite different from the old system, where the new system is a real-time system or where extra staff is not available to cope with the extra workload arising from parallel conversion. The advantage of this system is that it is fast and efficient, with minimum duplication of work involved. The disadvantage is that normal operations could be seriously disrupted if the new system has errors in it or does not work quite as expected.

Parallel conversion.

The old system continues alongside the new system for a few weeks or months, processing current data. The advantage of this is that results from the new system can be checked against known results and if any difficulties occur, operations can continue under the old system while the errors or omissions are sorted out. The disadvantage of parallel running is the extra effort required to keep both systems running, which may put a strain on personnel.

Phased conversion.

This may be used with larger systems that can be broken down into individual modules that can be implemented separately at different times. For example in a supermarket the tills could be computerized first, being on-line to the stock master file to look up prices but not altering stock levels. When this aspect of the new system is known to be working well, the stock control system could be introduced so that items sold automatically reduce the quantity in stock. Phased conversion could be direct or parallel.

Pilot conversion.

The new system is introduced into one department or branch of an organization and run until it is proved to be working satisfactorily. This could be combined with parallel running in the relevant department

Documentation

The aims of system documentation

Both systems analysts and programmers are responsible for maintaining standards and documentation on a project. The aims of documentation are:

· To help in the design of the system by working to a set of standards and having available a clear description of the work done so far. The documentation needs to be kept up-to-date throughout the project.

· Good documentation ensures that everyone involved in the system (systems designers, programmers and users) fully understands how their aspect of the system will work; for example, what data will be input and how, and what information will be available from the system. This allows for any misunderstandings or disagreements to surface before they become deeply entrenched in the system.

· To ensure that the system can be maintained after completion of the system. All too often changes of staff within a company mean that no-one who was involved in the original design or programming of a system is still with the company. It is essential that proper documentation is kept to enable a newcomer to make necessary corrections, alterations or enhancements.

Documentation for a system includes user manuals, a computer operations manual, systems specifications and program documentation.

User Manual (User documentation)

This will typically contain

· An overview of the options available;

· Guidance on the sequence of operations to follow;

· Screen shots showing screen input ‘forms’;

· Instructions on how to enter data;

· Sample report layouts;

· Error messages that may be displayed and what action to take.

Computer Operations Manual (technical documentation)

This provides documentation of the procedures necessary to run the system. It may include

· System set-up procedures, including details for each application of the files required and stationery requirements;

· Security procedures;

· Recovery procedures in the event of system failure;

· A list of system messages that might appear on the operator’s console and what action to take.

Systems Specification (technical documentation)

This is a complete description of the whole system showing data flows, system flowcharts, inputs, files, outputs and processing.

Program Documentation (technical documentation)

Each program within the system should have documentation which will include structure charts or pseudo code, full annotated listings and details of test data and results.

Evaluating the system

Evaluation is the final stage of developing a new system after implementation.

This checks that the system has met the original requirements, is working properly and easy to use. Can it be improved further? Does it need modifying? It is really **re-analysing** - starting the same systems analysis process all over again.

PROGRAM MAINTENANCE

- All systems need to be maintained; that is performance monitored, modifications made if required, errors correct, documentation kept up-to-date.
- Alls programs require maintenance, and in fact the vast majority of programmers are employed to maintain existing programs rather than to write new ones.
- There are differing reasons for this, and different types of maintenance as below:

  (1) Perfective maintenance

  (2) Adaptive maintenance

  (3) Corrective maintenance

PERFECTIVE MAINTENANCE

- This maintenance implies that while the system runs satisfactorily, there is still room for improvement.
- For example, extra management information may be needed so that new report programs have to be written. Database queries may be very slow, and a change in a program may be able to improve response time.

ADAPTIVE MAINTENANCE

- All systems will need to adapt to changing needs within a company.
- As a business expands, for example, there may be a requirement to convert a standalone system to a multi-user system.
- New and better hardware may become available, and changes to the software may be necessary to take advantage of this.

- Competition from other firms may mean that systems have to be upgraded in order to maintain a competitive edge.

CORRECTIVE MAINTENANCE

- Problems frequently surface after a system has been in use for a short time, however thoroughly it was tested.
- Some part of the system may not function as expected, or a report might be wrong in some way; total missing at the bottom, incorrect sequence of data, wrong headings, etc.
- Frequently errors will be hard to trace, if for example a file appears to have been wrongly updated.

Choosing an application software

Choosing between an **off -the-shelf** solution and a **custom** designed solution is the principal consideration when acquiring newsoftware. Each choice has different consequences relating to cost and risk for each step of the project lifecycle.

At the initial acquisition stage, **off-the- shelf** solutions can meet primary requirements but demand greater flexibility from the business. **Custom** solutions offer greater flexibility and can therefore be designed to meet business needs.

At the deployment stage**, off-the-shelf** solutions generally have established training courses and material however place greater demands on businesses to change their existing processes. **custom** solutions can be designed to mirror existing business processes, however require training material to be developed from scratch and are prone to 'requirements creep.'

At the ongoing maintenance and evaluation stage: **off-the- shelf** solutions are generally quite limited in terms of catering for changes to business structure and processes, however they have well developed support infrastructure and regular updates. customised solutions can be modified to fit the changing business requirements; however businesses will need to establish their own support channels and policies to support the users of the new system.

An organisation should ideally pursue a third option that leverages the advantages of both options while minimizing the disadvantages. The third option is a hybrid platform like Information Leader that offers the flexibility of a customized system while at the same time offering the stability and maturity of an **off-the- shelf** solution.

When looking for a software solution to your current business problem, you need to carefully examine your options before deciding what will best meet your needs.

Generally, there are two options: off-the-shelf and **custom**. Each solution brings with it several considerations. The following are some of the questions that should be asked before deciding on a solution:

For an off-the-shelf Product:

- Does it meet all of your **essential requirements**?
- Is the product **easy to use**?
- If the product is complex, would your staff use the product **as** it is intended?
- Would you be paying for a lot of **features that you don't really need**?
- Does the cost of the product, including extra features you don't need or won't use plus training**, provide you with the returns you need** in order to justify the expense?

- In order for your staff to use the product effectively and efficiently, is additional **training** a necessity?

- Do you know the **cost of training and support** before you buy the product?

- Is the cost of training or support **reasonable** and within your budget?

- Are you required to pay **additional licensing fees**?
  When you buyoff-the-shelf software you are frequently subject to licensing fees. The initial purchase price for a product often limits your right of use to a limited number of current users. **You have to pay extra if you want the product to be accessible to more users.**

For a custom Solution:

In addition to the above questions, you also need to consider the following points:

- Your up-front investment cost may be higher than for an off-the-shelf package, so you need to factor in your **long-term return on investment** to determine the cost benefit of a custom solution.

- The time frame by which you require your application must be carefully determined. Sufficient **lead-time** is required in order to ensure that the product will be ready on time, whereas an **off-the-shelf package can usually be picked up whenever needed.**

- Sometimes a custom application can be **integrated** into an off-the-shelf product. An example of this is when you want to add a feature to an off-the-shelf product that it doesn’t currently have or that is very expensive to acquire as a ready-made solution. The latter situation would more likely apply to very large and expensive off-the-shelf products.

- Because custom software is designed to meet your specific needs, it will do **exactly what you want** it to.

- With a custom application you pay only for the **features you need** and have asked for.

- A well-designed custom application should be **easy to use** and intuitive.

- The cost of and amount of training required should be known up-front before you commit to the project. Additional support can often be negotiated on an **as needed** basis.

- You need to identify how much your problem is **costing** you if **left unresolved**.

- There are normally **no licencing fees** with custom solutions. Once you have paid for the *product, you can usually make as many copies as you need.*

Major differences

Price: One of the biggest differences is the price. An off-the-shelf product is usually reasonably priced whereas a customized product can be relatively expensive to build. The logic is quite simple - the cost of development of the off-the-shelf product is distributed over a number of buyers and thus the pricing is spread over the number of licenses that will be sold. The customized software product on the other hand is made just for one buyer and thus the entire cost of development has to be borne by one customer. One might think that if the option of an off-the-shelf software product is available, why would one go for a customized development?

Generic build: The answer to the above question is again quite simple. An off-the-shelf product is made for a wide range of audience. Thus the development team has to provide a set design as well as a pre-defined framework. Since the product is ready to use, it may **not cater to the exact requirement of all organizations**. Usually there are some adjustments that an existing organization will need to make in order to implement an off-the-shelf product. This disadvantage is the biggest advantage in the case of customized software. Customized software picks up current processes and builds an electronic framework to automate them. There is no requirement to change any processes and people can usually get used to the software within a few days. Customized software provides flexibility as well for the organization to tinker around with the software and fine tune it for better performance.

Updates: Off-the-shelf products have off-the-shelf updates. Each time there is a revision in software purchased off-the-shelf, you will need to shell out an update fee. On the other hand, some off-the-shelf products may not have any updates for a long time, thus making you use obsolete software which might have negative repercussions on your business. Updates in customized software are done based on the requirements and budget of the company.

Integrated software is software for personal computers that combines the most commonly used functions of many productivity software programs into one application.

The integrated software genre has been largely overshadowed by fully functional office suites, most notably Microsoft Office, but at one time was considered the "killer application" type responsible for the rise and dominance of the IBM PC in the desktop business computing world

Generic Software Ready-made software. Shrink-wrapped software. Contrast with "custom software."

Chapter 8 - CHOOSING APPLICATIONS SOFTWARE FOR APPLICATION AREAS

Custom written software is a computer program or software written specifically for a company, according to the company's way of doing business. Sometimes it is also called *bespoke software*. Because custom written software is designed to meet the company's specific needs, it will do exactly what the company want it to. With a custom application the company pays only for the features it needs and has asked for. A well-designed custom application should be easy to use and intuitive. There are normally no licensing fees with custom solutions. Once you have paid for the product, you can usually make as many copies as you need. Usually custom written software is expensive as in certain case the company have to hire programmer to write the program also it have to consider the staff's training and system maintenance's cost.

Off-the-shelf software, also known as pre-packaged or pre-written software is an application program developed for sale to the general public. Packaged software is generally designed to appeal to a large audience of users, and although the programs may be tailored to a user's taste by setting various preferences, it is not as individualized as custom-designed and custom-programmed software. With pre-written software, the company do not technically own the software i.e. the company are paying for the right to use the software (a license), and the amount of money you pay usually increases with the number of users. But it is cheaper than hiring a programmer to do it and the company does not have to sent its staff for training as off-the-shelf software usually come with easy-to-understand user manuals and system manuals.

Business, Commercial and Industrial Applications

Stock Control

All ordering is performed by computer. There are fast electronic communication lines between the shops, the distribution centres and the head office. There are also direct links to the major suppliers, which means that orders can go straight through to production lines. One advantage of this is that stock arrives just in time before sale so it is always fresh. Another advantage of this system is that money does not need to be tied up in stock and can be used for more productive purposes.

Features

- Stored information which includes item code, item description, product group, product code, etc.
- Stock day-book is recorded for all movements which include date, item code, quantity, reference/invoice/order number, etc.
- Sales information stored on each item includes, date of last movement, period turnover quantity, period turnover value, etc.
- Function for amending cost or selling prices for a range of items selected by item number or by product group.

Sales-based ordering (Order processing)

Sales-based ordering is the automatic re-ordering of goods from the warehouse using the sales information from the checkouts. If, for example, 200 tins of baked beans are sold from a certain store in one day, then 200 tins will be automatically re-ordered and delivered to the store the following day from one of the store distribution centres.

Features

- Edit Payments - apply and edit payments for orders by using the POS menu to capture credit card payments, check payments, over-the-phone payments, and debits for returns.
- Edit Shipments - orders can be split into multiple shipments. The merchant can easily select any items that are not available to ship and a new shipment is created. Any number of new shipments configurations can be created.
- Shipment Tracking - a shipping tracking number can be entered into the Shipment Tracking Number text box. The tracking number that you enter will be available for the customer if they view their account from your store.
- Order Notes - you can add notes to keep track of details or explanations about any order. The notes are recorded with a date and time, and username of the person entering the note. Notes are for internal administration use only.
- Order Files - used for electronic file delivery and digital gift certificates; these files can be attached, removed, validated, and fulfilled right from the order detail page.

Payroll

Nearly all companies use computers to work out the wages of their staff. Also, the money is often transferred to the bank accounts of their staff by computers. Behind any payroll application there will be a database.

Data about different things is kept in separate tables, so the data tables required could include:

1. Employee details

2. Hours worked

3. Tax tables

4. Holidays taken

Control Equipment (Process Control)

The data in the tables will be changed from time to time and the software has to provide easy ways to do this. For example, employee details will change when a new employee arrives or someone leaves or an employee's pay rate changes.

Features

- Multi user system

- Support multiples companies, multiples pay period

- Online check history with detail and year-to-date summary with detail.

- Payroll detail for reporting for all years, any date range, calendar or fiscal.

- Tax filing.

- Direct Deposit

Control Equipment (Process Control)

Some schools have sophisticated computer controlled heating systems. They can estimate when to switch on the heating before school begins, taking into account the outside temperature. On a mild winter's day, the heating may need to come on only half an hour before staff and pupils arrive. On cold days it may need to go on several hours before school starts. All the system needs to know is the temperature required and when it is needed. Old systems only remembered a time when the system would automatically start, regardless of the weather. This meant that sometimes the school was cold until midmorning and other times the heating would start too early, wasting energy and money. Many heating systems are able to control the temperature of individual rooms from a control panel. In this way school gyms can be kept cooler than classrooms.

The process of planning and regulating, with the objective of performing the process in an effective and efficient.

Features

- Detailed tracking capabilities
- Process trending
- Machine operations
- Fault monitoring and logging
- In-depth production data recording and management

Business, Commercial and Industrial Applications

Point of Sale Systems

POS terminals are the cash registers which also act as terminals to a main computer system. As well as providing customers with itemised bills, these systems also give useful management information.

POS terminals are the cash registers which also act as terminals to a main computer system. As well as providing customers with itemised bills, these systems also give useful management information.

Points of sale systems make a precise record of every item it is sold. These records are used for stock control and for accounting purposes. Most POS systems operate in real time: the stock levels are adjusted and the accounts are updated as the sale takes place. Other systems store the records for a day’s trading and process them as a batch overnight

Many shops are in groups (known as chains of shops), owned by a single company. The point of sale systems in these shops usually operates over a computer network. There is a microcomputer in every shop in the group, linked to the cash terminals. This sends data over a network to the main computer at the company’s headquarters. The computers in the warehouses are also connected to this central computer. There are often links to the computers of the companies which supply goods to the shops so that re-orders can be placed automatically.

Features

- Dynamic update of item sales and inventory data as sales occur
- New items can be added to inventory from point of sale inventory screen
- Bar Code Scanning and Printing for speed and accuracy
- Prints invoices and paper tape receipts (user definable)
- Payments on charge accounts with printed receipts
- Sales Order management

Marketing

It means to make a communication about a product or service a purpose of which is to encourage recipients of the communication to purchase or use the product or service.

Features

- Financial Spreadsheets that Validate Your Requirements and Potential
- Integrated Suite of Planning Tools
- Charts That Get Attention
- Customize Your Plan with a Powerful Outline Manager
- Milestones to Document Your Objectives
- Flexible Reporting Methods

Computer Aided Design

Computer Aided Design or CAD for short, is used by engineers, architects, etc. to produce high quality technical drawings drawn to scale. The sorts of diagram produced by these packages include plans for houses, maps, circuit diagrams, engineering drawings and 3-D plans of kitchens. Before computers were used, if a drawing needed a simple modification then it was often necessary to completely redraw it. Now, with the use of CAD software modifications can be made at the press of a button.

Architects can produce a plan view of a building and then immediately view it in three dimensions and even rotate it to see what it looks like from different angles.

Features

- Storage of a library of parts – this can speed up the design process
- Zooming in and out to make fine adjustments
- Rotation of objects so they can be seen from all angles
- Filling of designs to stimulate different manufacturing materials
- Production of a three-dimensional image from a two-dimensional drawing

Computer Aided Manufacturing

The process of using specialized computers to control, monitor, and adjust tools and machinery in manufacturing. Today a single computer can control banks of robotic milling machines, lathes, welding machines, and other tools, moving the product from machine to machine as each step in the manufacturing process is completed. Such systems allow easy, fast reprogramming from the computer, permitting quick implementation of design changes. The most advanced systems, which are often integrated with computer-aided design systems, can also manage such tasks as parts ordering, scheduling, and tool replacement.

Generic Application Software

Word processing

Word processing packages are the most popular type of computer software. A word processor started the electronic replacement of the typewriter. It has the advantage that work may be stored on disc and retrieved later and this has meant that its use has spread beyond that of the secretary or typing pool and is being used by managers, journalists, authors and just about anybody who might otherwise put pen to paper.

Most word processors will include the following features.

- a range of fonts and sizes.
- paragraph alignment eg. left align, centre, right align, justify
- cut and paste blocks of text

- automatic page / chapter numbering - page headers and footers
- spelling checkers
- thesaurus
- grammar checkers
- template documents e.g. memo format

Spreadsheets

Spreadsheets are one of he most popular programs in business. Uses range from financial applications such as working out a business cash flow, or forecasting sales to calculating the space taken up in a warehouse by packages, or analysing the results of an experiment. Any application which involves doing calculations with sets of figures can be tackled with a spreadsheet program.

One of the most powerful uses of a spreadsheet is the ability to ask 'What If..' questions. This means that it is possible to alter one figure in the spreadsheet and see the results instantly. For example a manager might alter a cell in the spreadsheet indicating fuel costs and instantly be able to see the effect on company profits.

Most spreadsheets include features such as

- a wide range of mathematical and scientific functions
- user defined functions
- sorting
- presentational tools e.g. grids, shading
- graphs and charts

Desk-Top-Publishing

Although many word processors will allow the user to control the page layout and to include graphics a Desk-Top-Publishing program will normally go further. They may be used by anyone for creating posters, notices, advertisements, or multi-page documents with complicated page layouts. DTP packages will have many of the features of word processors and include.

- page layout - the creation of master pages with frames in place
- a control of imported graphics e.g. cropping, rotating
- a wider variety of text fonts / sizes or support for rotated text
- simple graphics functions e.g. line drawing, shaded boxes etc.

Presentations

Pictures are able to communicate ideas and concepts to an audience much better than the written or spoken word alone. Presentations are used to sell products, services or just ideas to someone else. Good presentations use visual aids such as slides or transparencies on a projector. The visual aids are used to back up what the presenter is saying. Graphs are usually included in such presentations. If the work is displayed attractively, then the audience will be more able to digest the key points.

Graphics / Drawing Packages

For most of us, graphics packages can mean just a simple paint package where you can draw line diagrams, flood fill, and produce a variety of shading. You can also produce text in a variety in a variety of fonts and typefaces. These simple but very useful packages can be used as an alternative to specialist and more complicated desktop publishing packages. Producing diagrams is extremely difficult using a mouse so some professional artwork is hand drawn and then scanned into a graphic package.

Some graphic packages allow you to scan a photograph or diagram directly into the computer so that you can subsequently manipulate it in some way. Using special equipment and software it is possible to capture pictures from a television screen or from a video camera. The extra equipment needed to do is called a video grabber. Digital cameras are very popular because you do not need a film and there are no developing costs. If you need to use pictures then you can use clip art.

Expert systems

- These are computer systems that emulate the decision-making ability of a human expert.
- Contain as much data as possible about a restricted knowledge area.
- The first expert systems were created in the 1970s.
- Data is collected from many expert sources and stored in the system.
- Hence also called knowledge based system
- The use of expert systems relies on the belief that for any given area of knowledge there are rules that always apply to that knowledge and they can be applied according to a logical ordering.
- Oil exploration is a good example, for instance, if there is a rule that oil is always found between two particular types of rock strata (layer of sedimentary rock/soil), then if the system identifies the strata it can determine, according to the rules, where to drill for oil.

The '**knowledge base (k.b)**'

containing the facts, all the data the system has been told about. The knowledge base is created from information provided by human experts

- For a medical expert system, the k.b can be created using knowledge on diseases from all best doctors.

The '**rule base (r.b)**'

Contains a set of rules that apply to the situation. "if the patient has a headache, they should take aspirin".

The '**inference engine**'

This acts rather like a *search engine*, examining the knowledge base for information that matches the *user's query.*

This is what decides whether the patient should be given aspirin.. It's the inference engine that checks through everything to find out what will fit the rules.

The 'U**ser interface (HCI)'**

- This is the interactive platform with which the user communicates to the system.
- When the user is viewing or analyzing data from the expert system, it's through the HCI

The following questions may be used by the knowledge engineer to help understand what the expert does:

1. Exactly what decisions does the expert make?
2. What are the decision outcomes?
3. Which outcomes require greater reflection, exploration or interaction?

4. What resources or inputs are required to reach a decision?
5. What conditions are present when a particular outcome is decided?
6. How consistently do these conditions predict a given outcome?
7. At what point after exposure to influential inputs is a decision made?
8. Given the particulars of a specific case, will the outcome predictions of the knowledge engineering team be consistent with those of the expert?

Uses of expert systems

- Medical diagnosis
- Fault diagnosis of all kinds - *gas boilers, computers, power stations, engines*
- Geological surveys - *to find oil and mineral deposits*
- Financial services - *to predict stock market movement*
- Social services - *to calculate the benefits due to claimants*
- Industrial uses - *such as ELSIE in the construction industry*

Sample Qn

Explain how a knowledge-based (expert) system can be set up and used to help scientists at a national botanical centre to identify plants. (6)

- Limit volume of information because...
- small screen and...
- important to remove extraneous information...
- because driver can only glance at screen
- appropriate colour used to show route on map
- Sound to provide a commentary of directions...
- so that driver does not need to look away from road
- buttons/ touch screen bfor input / menubased …

1. (1 per –, max 5) [5]

- prompts question to ask
- Ensures all details are taken
- Allows for ease of validation routines/standard entry of data/reduces entry error
- All data is relevant
- Allows use of drop down lists and radio buttons

(1 per –, max 3) [3]

2.

- Colour used to show different types of information on screens
- different windows keeping data areas separate
- Some colours/reverse video/flashing reserved for important messages
- Sound to provide an alarm if something is wrong
- failsafe "are you sure" response to dangerous input
- so that attention of operators is immediately drawn to that information
- Content will remain constant for many outputs…
- retrieve history of past actions
- Peripherals for input, probably touch screen to isolate area for inspection…
- keyboards for changing parameters of processing

3. (1 per –, max 5) [5]

- Simple/easy to use, users may be computer illiterate
- Helps the user to get to information that they may not have been aware of
- Structure of the information system will not change very often/menus will remain unchanged
- Limits access by user to specific parts of system
- Fits use of an indexed sequential filing system
- Does not require any input from user except for choices from screen/allows for obvious use of touch screen

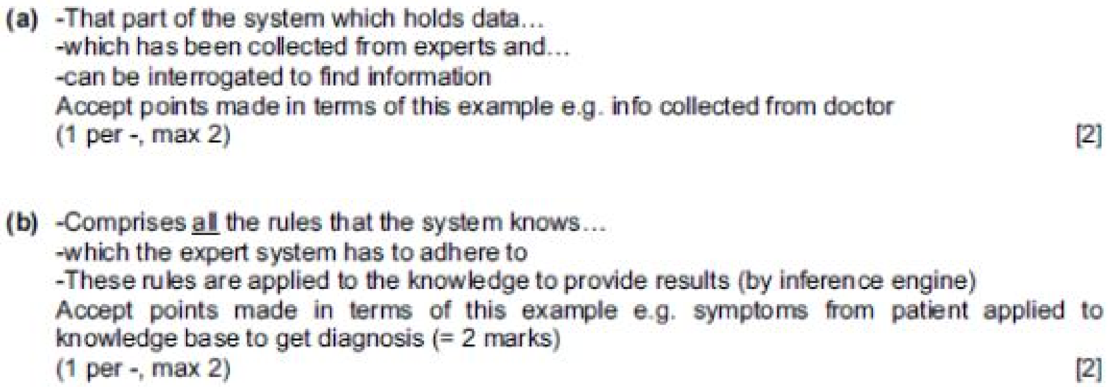
4.

Colour...
- Colours should provide suitable contrasts
- should be meaningful e.g. red for danger
- reference to colour blindness / epilepsy

Layout...
- should use whole screen...
- important information in top left hand corner/centre of screen
- big buttons for ease of navigation
- similar content grouped together
- consistent layout when moving from screen to screen

Content...
- must be relevant...
- must be understandable
- must be restricted so no information overload

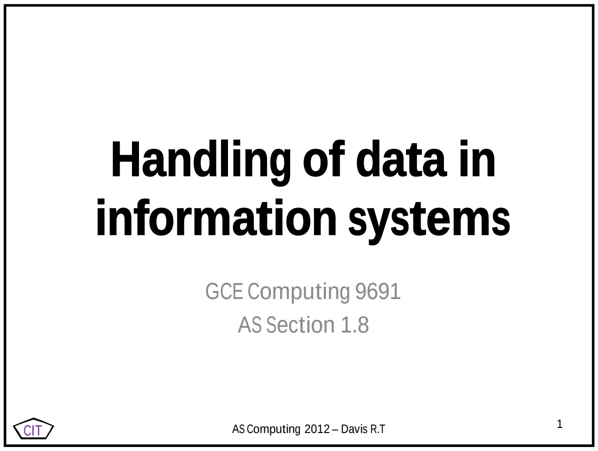
5. (1 per –, max 2 per section, max 6) [6]



(a) -That part of the system which holds data…
-which has been collected from experts and…
-can be interrogated to find information
Accept points made in terms of this example e.g. info collected from doctor
(1 per -, max 2) [2]

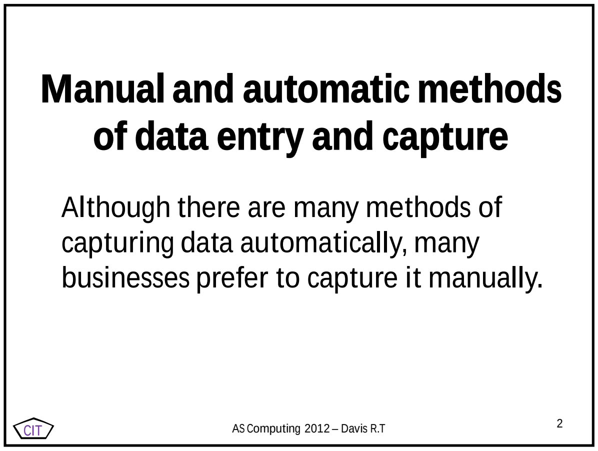
(b) -Comprises all the rules that the system knows…
-which the expert system has to adhere to
-These rules are applied to the knowledge to provide results (by inference engine)
Accept points made in terms of this example e.g. symptoms from patient applied to knowledge base to get diagnosis (= 2 marks)
(1 per -, max 2) [2]

6.



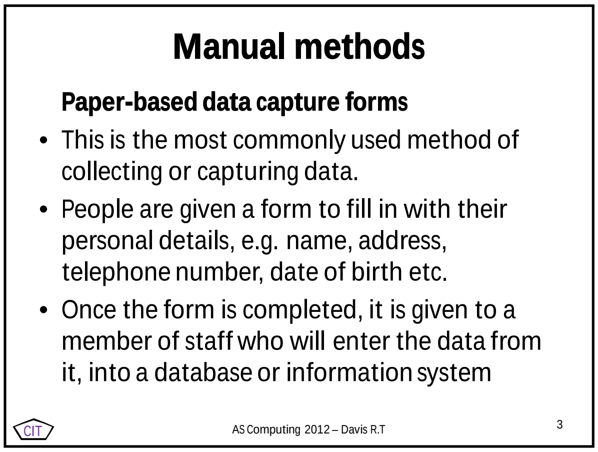
Handling of data in information systems

GCE Computing 9691
AS Section 1.8



Manual and automatic methods of data entry and capture

Although there are many methods of capturing data automatically, many businesses prefer to capture it manually.



Manual methods

Paper-based data capture forms

- This is the most commonly used method of collecting or capturing data.
- People are given a form to fill in with their personal details, e.g. name, address, telephone number, date of birth etc.
- Once the form is completed, it is given to a member of staff who will enter the data from it, into a database or information system

Manual methods (continued)

Computerised data entry forms

(usage of the keyboard and touch screen)

- Whilst the customer is present or on the telephone, a member of staff could ask the customer the questions on the data entry form and type their answers directly into it.
- This data will then be stored in the appropriate database table.
- If paper-based data capture forms are used (as above) then the member of staff can take the information from the data capture form and manually type it into the data entry form.

Manual methods (continued)

• If staff are doing this, it is important that the fields on both forms are laid out in the same order to speed up the process of entering the data.

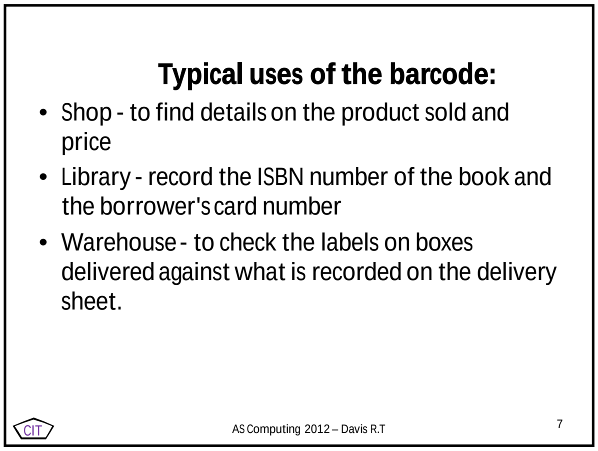
✓ Collecting/capturing data manually from sources/documents has for proved to be labour intensive and time consuming.

✓ Hence it's used where volumes of data are low and the cost of alternative methods cannot be justified.

Automatic data capture (Methods)

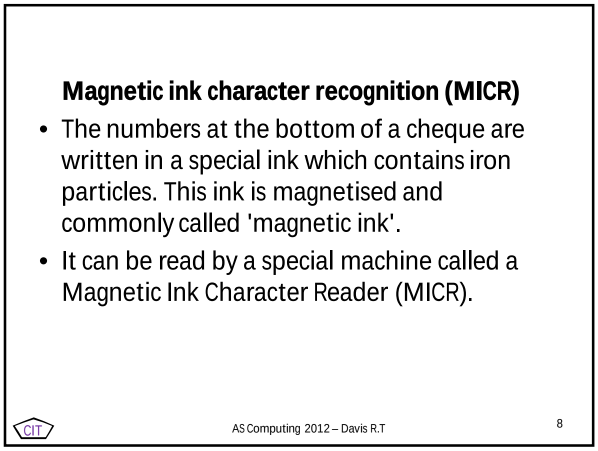
Barcode reader

- A bar code reader uses visible red light to scan and 'read' the barcode. As the red light shines across the light and dark bands of the barcode, so the reflected red light is also lighter and darker (do you see that on the picture opposite?)
- The Hand Scanner senses the reflected light and translates it into digital data. The digital data is then input into the computer. The computer may display the results on a screen and also input it into the correct fields in the database.



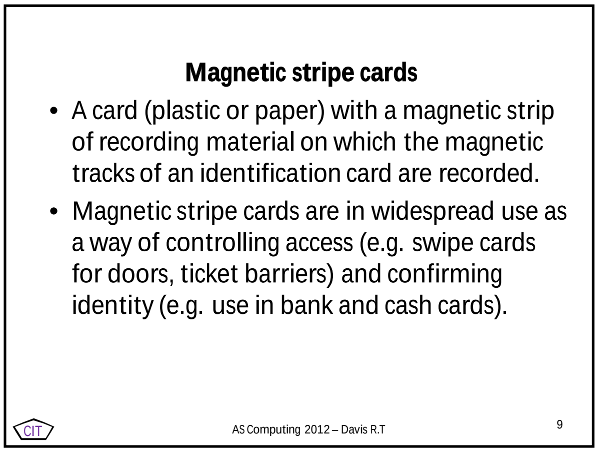
Typical uses of the barcode:

- Shop - to find details on the product sold and price
- Library - record the ISBN number of the book and the borrower's card number
- Warehouse - to check the labels on boxes delivered against what is recorded on the delivery sheet.



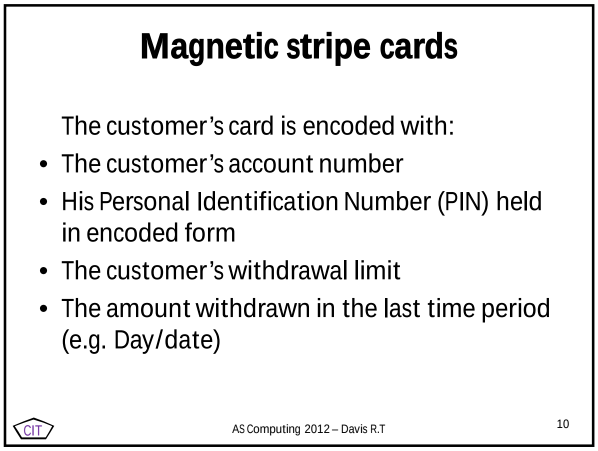
Magnetic ink character recognition (MICR)

- The numbers at the bottom of a cheque are written in a special ink which contains iron particles. This ink is magnetised and commonly called 'magnetic ink'.
- It can be read by a special machine called a Magnetic Ink Character Reader (MICR).



Magnetic stripe cards

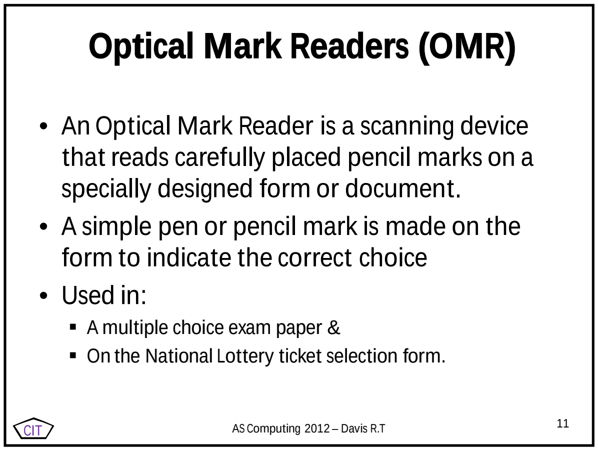
- A card (plastic or paper) with a magnetic strip of recording material on which the magnetic tracks of an identification card are recorded.
- Magnetic stripe cards are in widespread use as a way of controlling access (e.g. swipe cards for doors, ticket barriers) and confirming identity (e.g. use in bank and cash cards).



Magnetic stripe cards

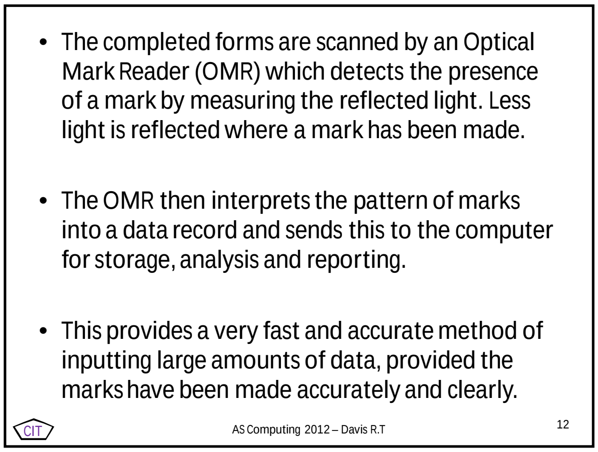
The customer's card is encoded with:

- The customer's account number
- His Personal Identification Number (PIN) held in encoded form
- The customer's withdrawal limit
- The amount withdrawn in the last time period (e.g. Day/date)

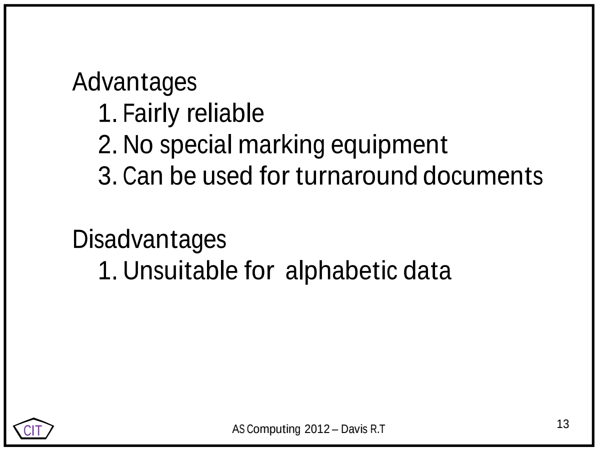


Optical Mark Readers (OMR)

- An Optical Mark Reader is a scanning device that reads carefully placed pencil marks on a specially designed form or document.
- A simple pen or pencil mark is made on the form to indicate the correct choice
- Used in:
  - A multiple choice exam paper &
  - On the National Lottery ticket selection form.



- The completed forms are scanned by an Optical Mark Reader (OMR) which detects the presence of a mark by measuring the reflected light. Less light is reflected where a mark has been made.
- The OMR then interprets the pattern of marks into a data record and sends this to the computer for storage, analysis and reporting.
- This provides a very fast and accurate method of inputting large amounts of data, provided the marks have been made accurately and clearly.

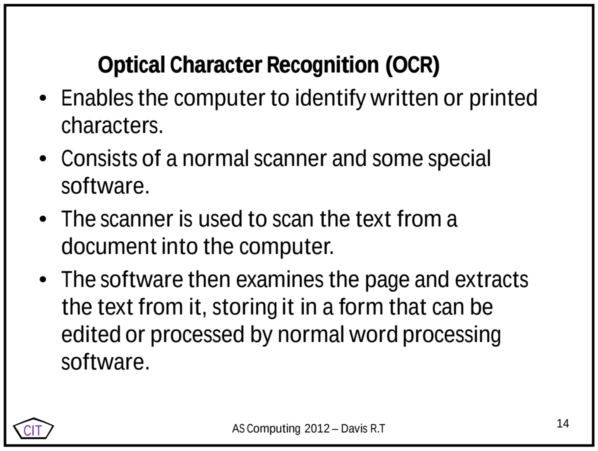


Advantages

1. Fairly reliable
2. No special marking equipment
3. Can be used for turnaround documents

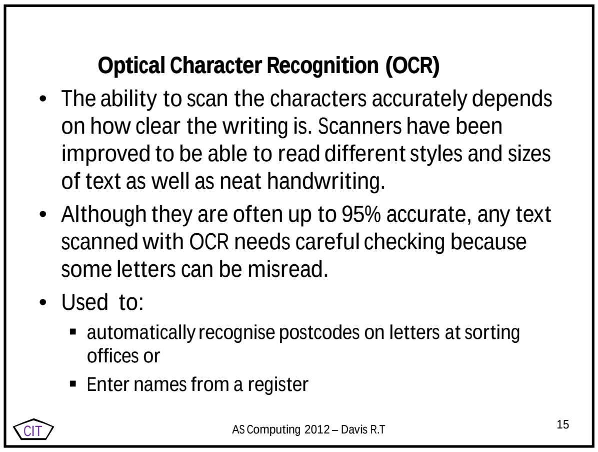
Disadvantages

1. Unsuitable for alphabetic data



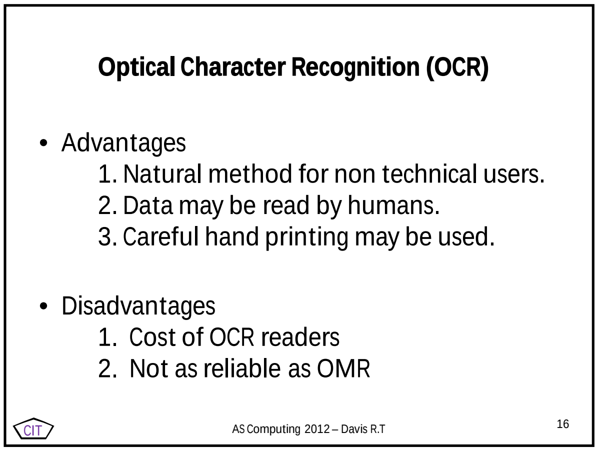
Optical Character Recognition (OCR)

- Enables the computer to identify written or printed characters.
- Consists of a normal scanner and some special software.
- The scanner is used to scan the text from a document into the computer.
- The software then examines the page and extracts the text from it, storing it in a form that can be edited or processed by normal word processing software.



Optical Character Recognition (OCR)

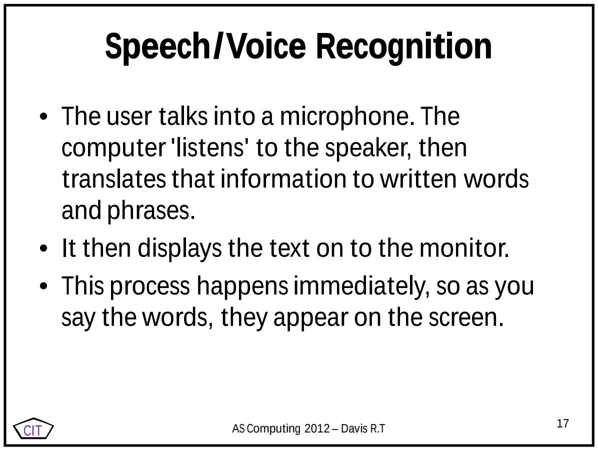
- The ability to scan the characters accurately depends on how clear the writing is. Scanners have been improved to be able to read different styles and sizes of text as well as neat handwriting.
- Although they are often up to 95% accurate, any text scanned with OCR needs careful checking because some letters can be misread.
- Used to:
  - automatically recognise postcodes on letters at sorting offices or
  - Enter names from a register



Optical Character Recognition (OCR)

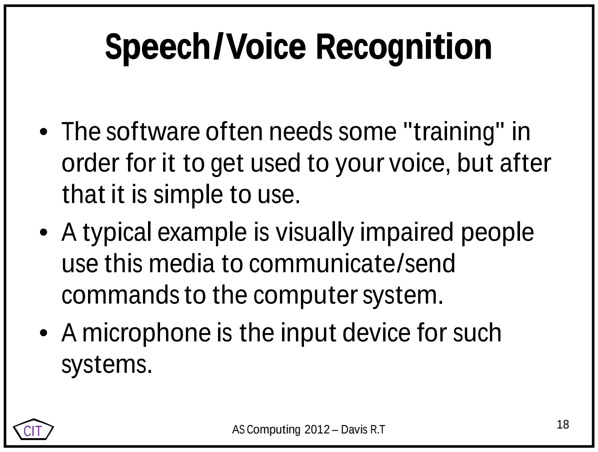
- Advantages
  1. Natural method for non technical users.
  2. Data may be read by humans.
  3. Careful hand printing may be used.

- Disadvantages
  1. Cost of OCR readers
  2. Not as reliable as OMR



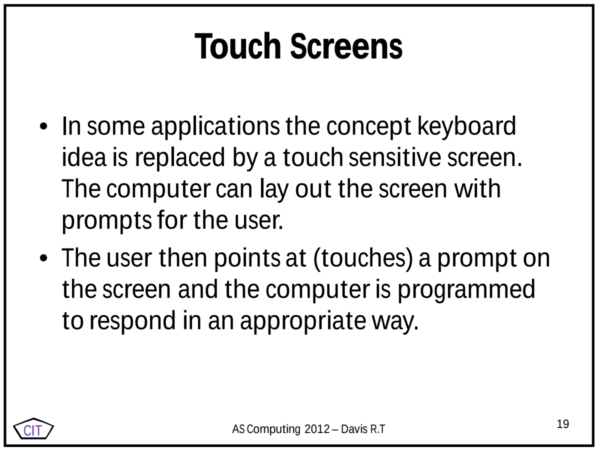
Speech/Voice Recognition

- The user talks into a microphone. The computer 'listens' to the speaker, then translates that information to written words and phrases.
- It then displays the text on to the monitor.
- This process happens immediately, so as you say the words, they appear on the screen.



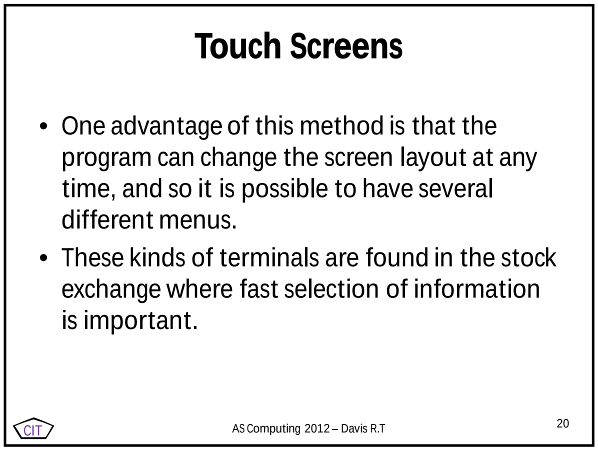
Speech/Voice Recognition

- The software often needs some "training" in order for it to get used to your voice, but after that it is simple to use.
- A typical example is visually impaired people use this media to communicate/send commands to the computer system.
- A microphone is the input device for such systems.



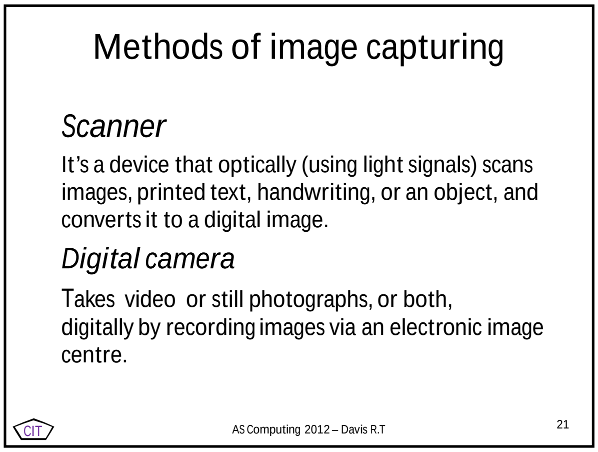
Touch Screens

- In some applications the concept keyboard idea is replaced by a touch sensitive screen. The computer can lay out the screen with prompts for the user.
- The user then points at (touches) a prompt on the screen and the computer is programmed to respond in an appropriate way.



Touch Screens

- One advantage of this method is that the program can change the screen layout at any time, and so it is possible to have several different menus.
- These kinds of terminals are found in the stock exchange where fast selection of information is important.



Methods of image capturing

Scanner

It's a device that optically (using light signals) scans images, printed text, handwriting, or an object, and converts it to a digital image.

Digital camera

Takes video or still photographs, or both, digitally by recording images via an electronic image centre.

Sensors

- Sensors are used to measure physical quantities such as temperature, light, pressure, sound, and humidity. They send signals to the processor.

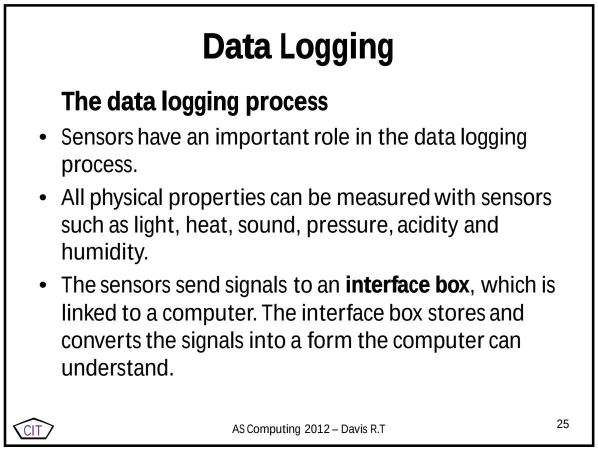
The role of sensors in control

- A security alarm system may have an **infra-red sensor** (Light sensor) which sends a signal when the beam is broken.

- A **heat sensitive sensor** in the corner of a room may detect the presence of a person.
- **Temperature sensors** could be used to control the heating in a large building.
- **Magnetic sensors** are inductive loops in the tarmac to detect metal above them and could detect traffic.
- **Motion sensor**

Data Logging

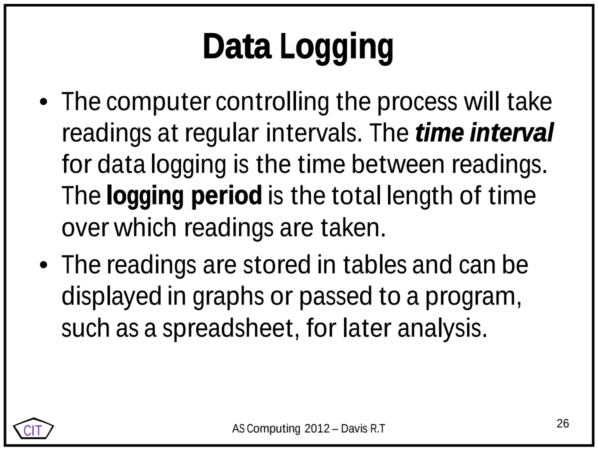
- This is the collection of *data* over a period of time, and is something often used in scientific experiments.
- Data logging systems typically monitor a process using sensors linked to a computer. Most data logging can be done automatically under computer control.



Data Logging

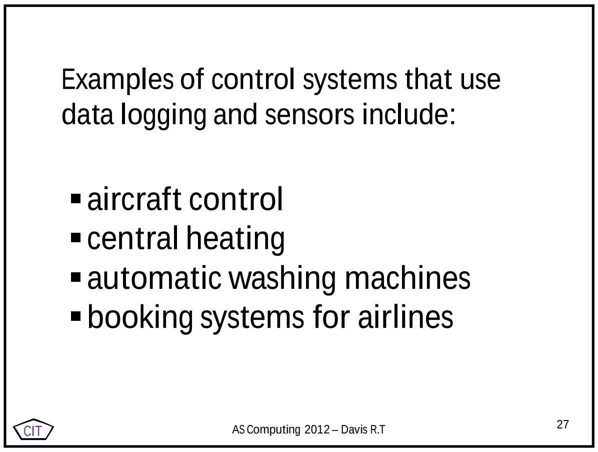
The data logging process

- Sensors have an important role in the data logging process.
- All physical properties can be measured with sensors such as light, heat, sound, pressure, acidity and humidity.
- The sensors send signals to an **interface box**, which is linked to a computer. The interface box stores and converts the signals into a form the computer can understand.



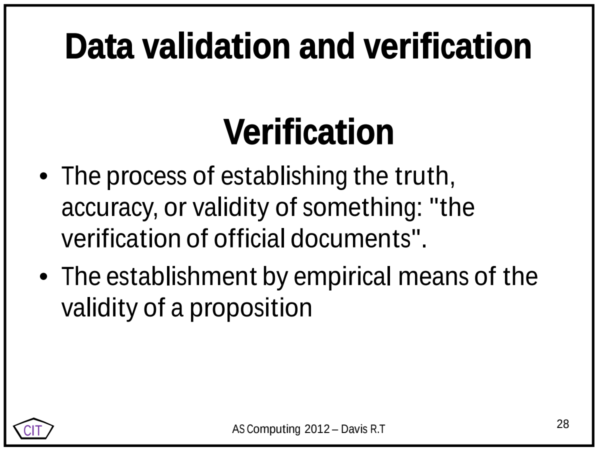
Data Logging

- The computer controlling the process will take readings at regular intervals. The ***time interval*** for data logging is the time between readings. The **logging period** is the total length of time over which readings are taken.
- The readings are stored in tables and can be displayed in graphs or passed to a program, such as a spreadsheet, for later analysis.



Examples of control systems that use data logging and sensors include:

- aircraft control
- central heating
- automatic washing machines
- booking systems for airlines

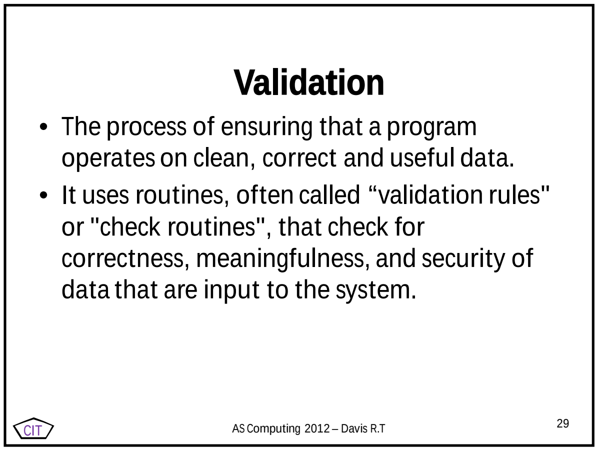


Data validation and verification

Verification

- The process of establishing the truth, accuracy, or validity of something: "the verification of official documents".
- The establishment by empirical means of the validity of a proposition

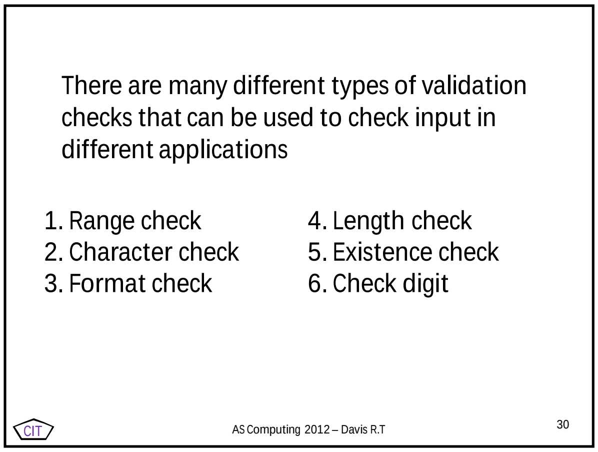
AS Computing 2012 – Davis R.T 28



Validation

- The process of ensuring that a program operates on clean, correct and useful data.
- It uses routines, often called “validation rules" or "check routines", that check for correctness, meaningfulness, and security of data that are input to the system.

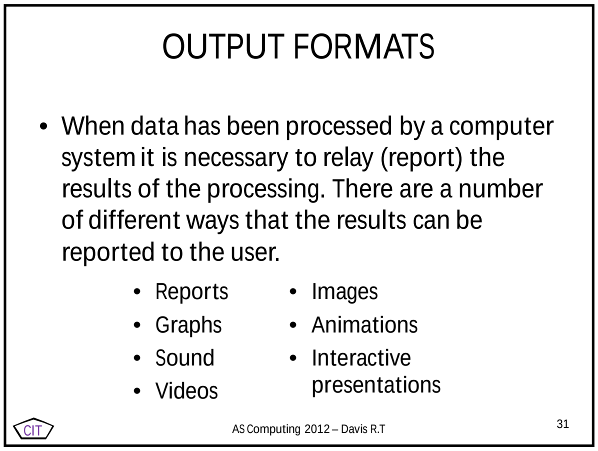
AS Computing 2012 – Davis R.T 29



There are many different types of validation checks that can be used to check input in different applications

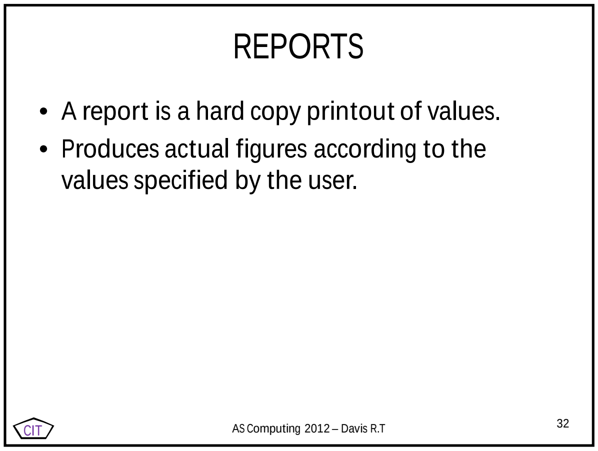
1. Range check
2. Character check
3. Format check
4. Length check
5. Existence check
6. Check digit

AS Computing 2012 – Davis R.T 30



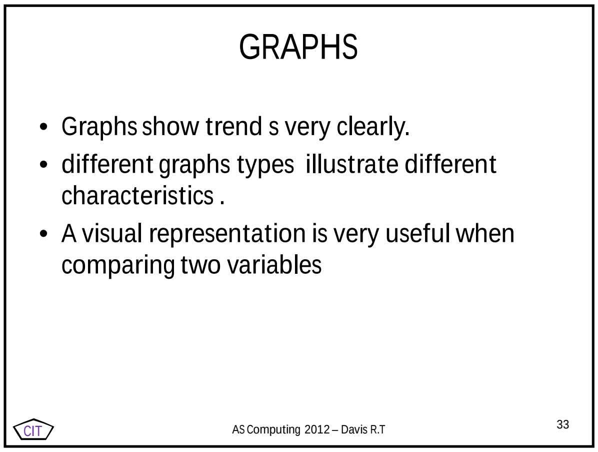
OUTPUT FORMATS

- When data has been processed by a computer system it is necessary to relay (report) the results of the processing. There are a number of different ways that the results can be reported to the user.
  - Reports
  - Graphs
  - Sound
  - Videos
  - Images
  - Animations
  - Interactive presentations



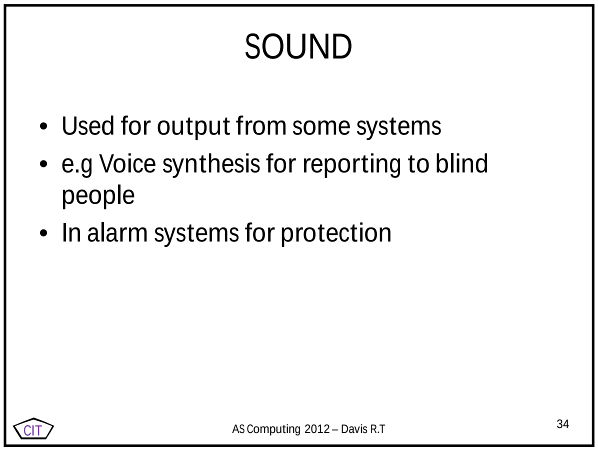
REPORTS

- A report is a hard copy printout of values.
- Produces actual figures according to the values specified by the user.



GRAPHS

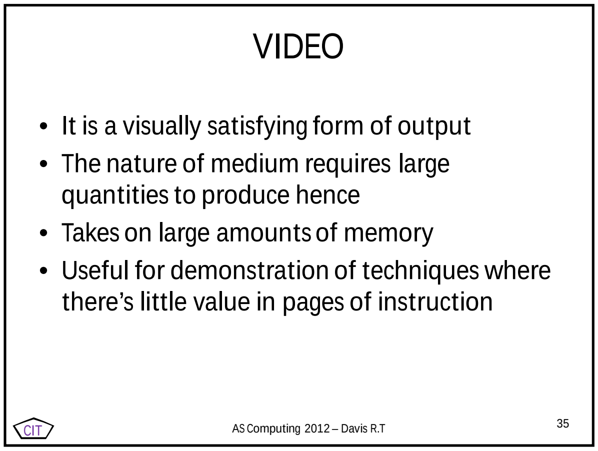
- Graphs show trend s very clearly.
- different graphs types illustrate different characteristics .
- A visual representation is very useful when comparing two variables



SOUND

- Used for output from some systems
- e.g Voice synthesis for reporting to blind people
- In alarm systems for protection

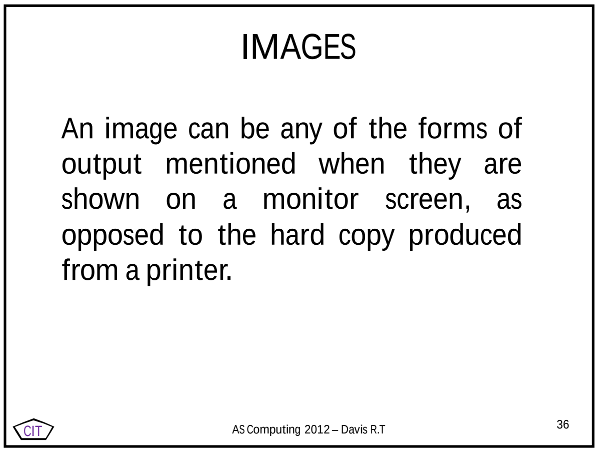
CIT AS Computing 2012 – Davis R.T 34



VIDEO

- It is a visually satisfying form of output
- The nature of medium requires large quantities to produce hence
- Takes on large amounts of memory
- Useful for demonstration of techniques where there's little value in pages of instruction

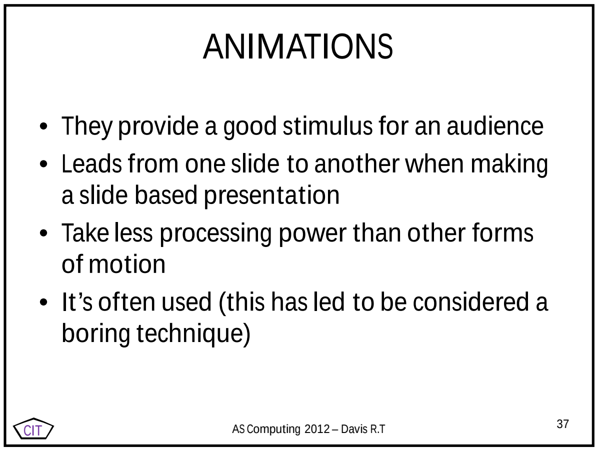
CIT AS Computing 2012 – Davis R.T 35



IMAGES

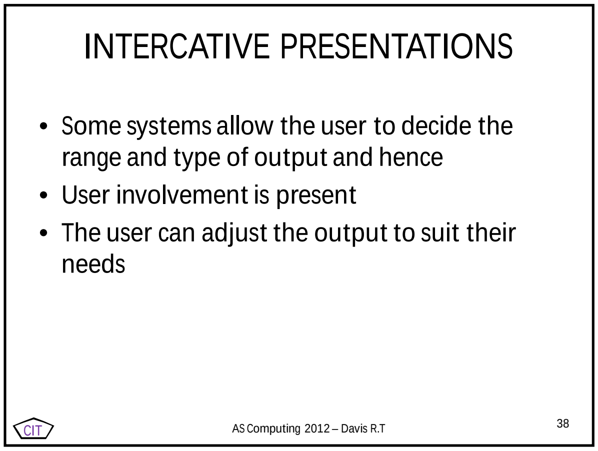
An image can be any of the forms of output mentioned when they are shown on a monitor screen, as opposed to the hard copy produced from a printer.

CIT AS Computing 2012 – Davis R.T 36



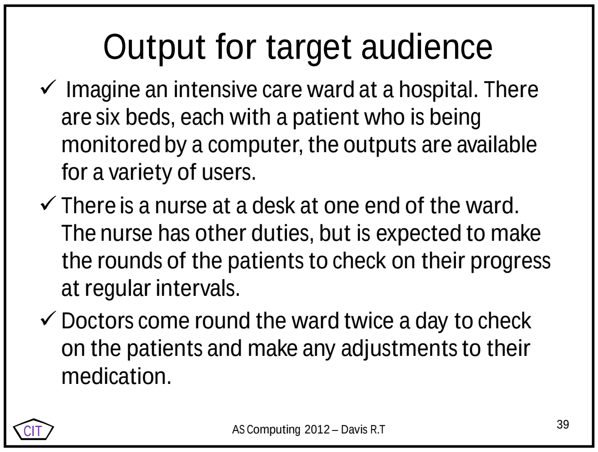
ANIMATIONS

- They provide a good stimulus for an audience
- Leads from one slide to another when making a slide based presentation
- Take less processing power than other forms of motion
- It's often used (this has led to be considered a boring technique)



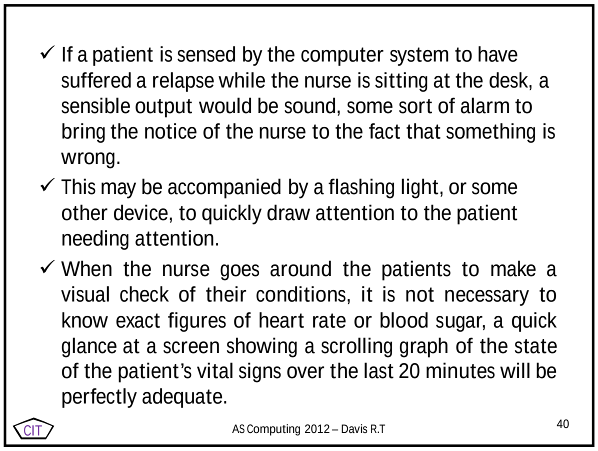
INTERCATIVE PRESENTATIONS

- Some systems allow the user to decide the range and type of output and hence
- User involvement is present
- The user can adjust the output to suit their needs

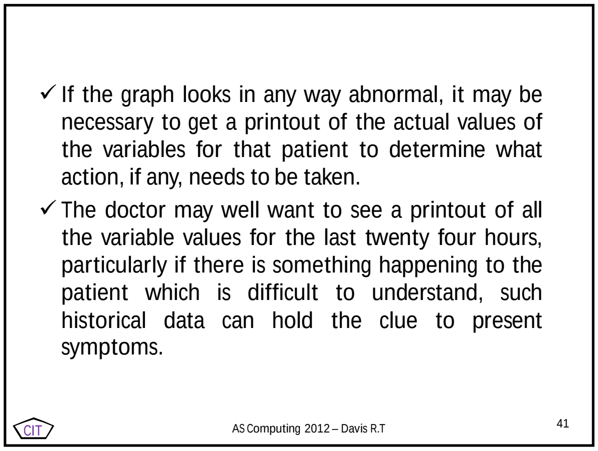


Output for target audience

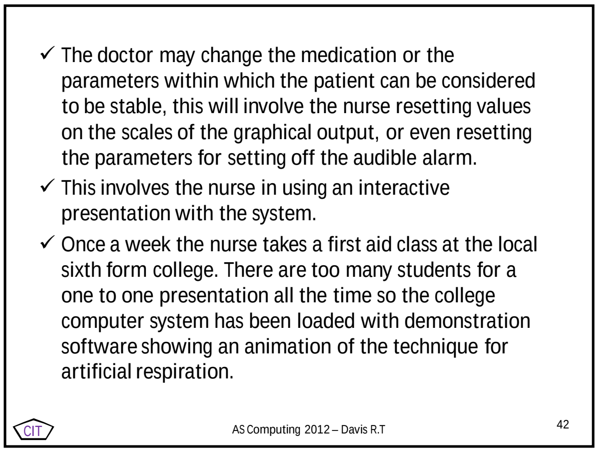
- ✓ Imagine an intensive care ward at a hospital. There are six beds, each with a patient who is being monitored by a computer, the outputs are available for a variety of users.
- ✓ There is a nurse at a desk at one end of the ward. The nurse has other duties, but is expected to make the rounds of the patients to check on their progress at regular intervals.
- ✓ Doctors come round the ward twice a day to check on the patients and make any adjustments to their medication.



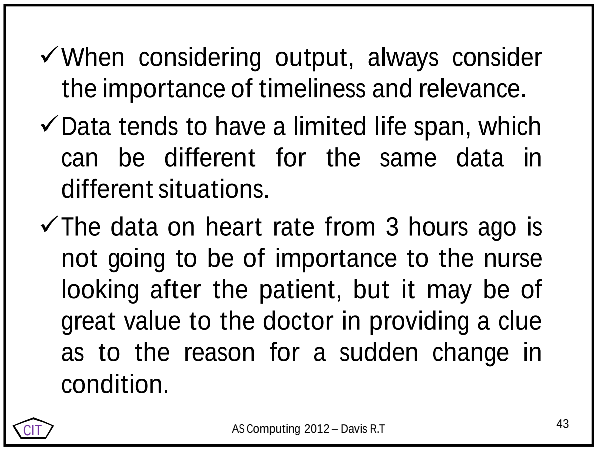
- ✓ If a patient is sensed by the computer system to have suffered a relapse while the nurse is sitting at the desk, a sensible output would be sound, some sort of alarm to bring the notice of the nurse to the fact that something is wrong.
- ✓ This may be accompanied by a flashing light, or some other device, to quickly draw attention to the patient needing attention.
- ✓ When the nurse goes around the patients to make a visual check of their conditions, it is not necessary to know exact figures of heart rate or blood sugar, a quick glance at a screen showing a scrolling graph of the state of the patient's vital signs over the last 20 minutes will be perfectly adequate.



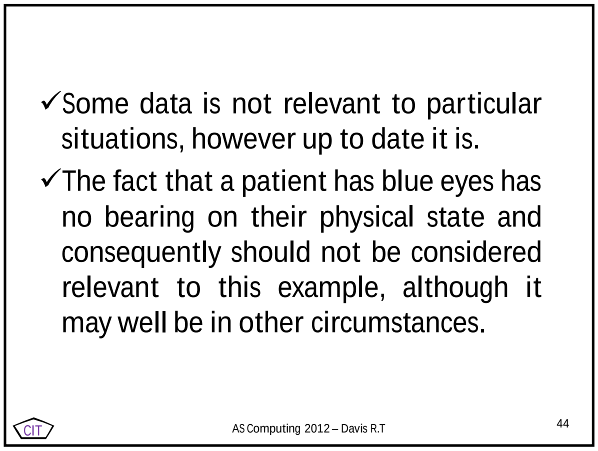
- ✓ If the graph looks in any way abnormal, it may be necessary to get a printout of the actual values of the variables for that patient to determine what action, if any, needs to be taken.
- ✓ The doctor may well want to see a printout of all the variable values for the last twenty four hours, particularly if there is something happening to the patient which is difficult to understand, such historical data can hold the clue to present symptoms.



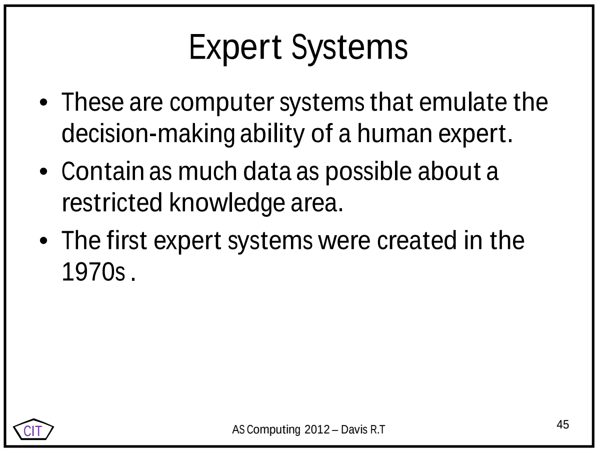
- ✓ The doctor may change the medication or the parameters within which the patient can be considered to be stable, this will involve the nurse resetting values on the scales of the graphical output, or even resetting the parameters for setting off the audible alarm.
- ✓ This involves the nurse in using an interactive presentation with the system.
- ✓ Once a week the nurse takes a first aid class at the local sixth form college. There are too many students for a one to one presentation all the time so the college computer system has been loaded with demonstration software showing an animation of the technique for artificial respiration.



- ✓When considering output, always consider the importance of timeliness and relevance.
- ✓Data tends to have a limited life span, which can be different for the same data in different situations.
- ✓The data on heart rate from 3 hours ago is not going to be of importance to the nurse looking after the patient, but it may be of great value to the doctor in providing a clue as to the reason for a sudden change in condition.



- ✓Some data is not relevant to particular situations, however up to date it is.
- ✓The fact that a patient has blue eyes has no bearing on their physical state and consequently should not be considered relevant to this example, although it may well be in other circumstances.



Expert Systems

- These are computer systems that emulate the decision-making ability of a human expert.
- Contain as much data as possible about a restricted knowledge area.
- The first expert systems were created in the 1970s .

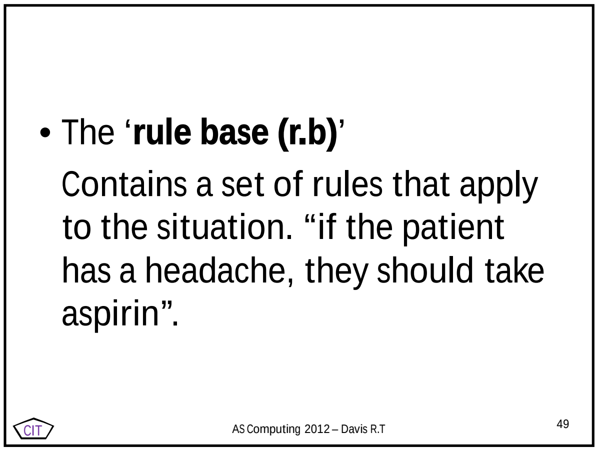
- Data is collected from many expert sources and stored in the system.
- Hence also called knowledge based systems
- The use of expert systems relies on the belief that for any given area of knowledge there are rules that always apply to that knowledge and they can be applied according to a logical ordering.

- Oil exploration is a good example, for instance, if there is a rule that oil is always found between two particular types of rock strata (layer of sedimentary rock/soil), then if the system identifies the strata it can determine, according to the rules, where to drill for oil.

How an expert system works

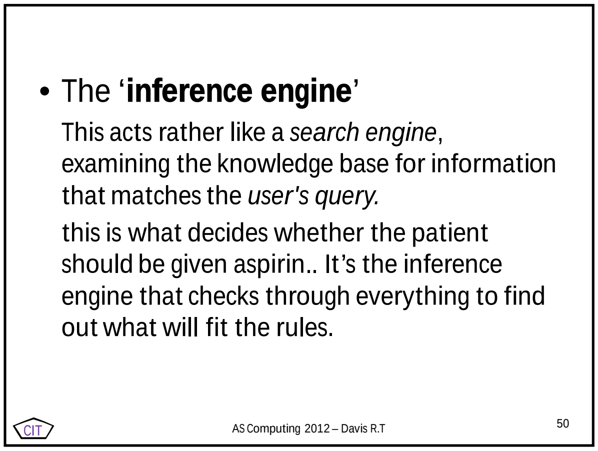
An expert system has the following constituents it uses to produce sensible results:

- The ‘**knowledge base (k.b)**’ containing the facts, all the data the system has been told about. The knowledge base is created from information provided by human experts
- For a medical expert system, the k.b can be created using knowledge on diseases from all best doctors.



- The ‘**rule base (r.b)**’

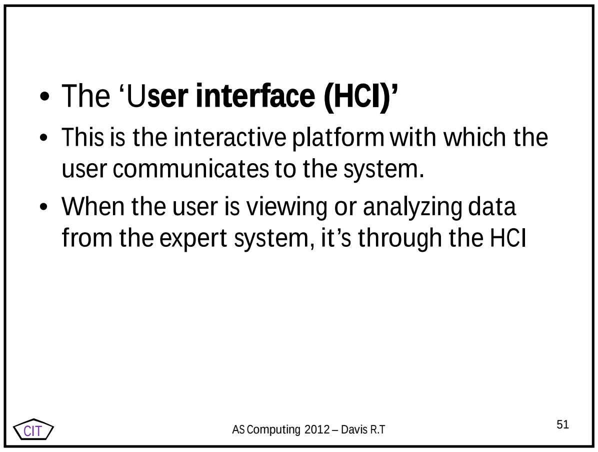
  Contains a set of rules that apply to the situation. “if the patient has a headache, they should take aspirin”.



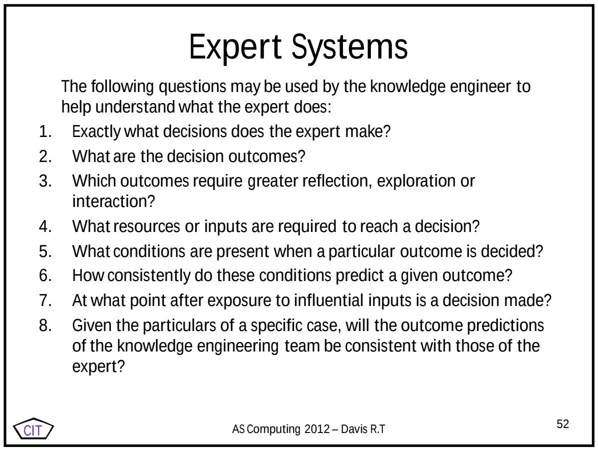
- The ‘**inference engine**’

  This acts rather like a *search engine*, examining the knowledge base for information that matches the *user's query.*

  this is what decides whether the patient should be given aspirin.. It’s the inference engine that checks through everything to find out what will fit the rules.



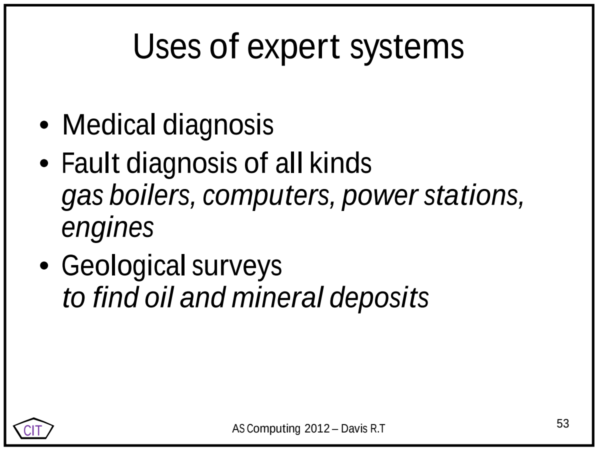
- The ‘U**ser interface (HCI)’**
- This is the interactive platform with which the user communicates to the system.
- When the user is viewing or analyzing data from the expert system, it’s through the HCI



Expert Systems

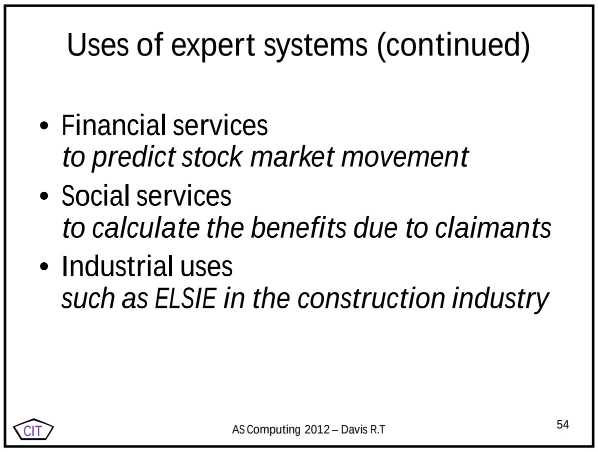
The following questions may be used by the knowledge engineer to help understand what the expert does:

1. Exactly what decisions does the expert make?
2. What are the decision outcomes?
3. Which outcomes require greater reflection, exploration or interaction?
4. What resources or inputs are required to reach a decision?
5. What conditions are present when a particular outcome is decided?
6. How consistently do these conditions predict a given outcome?
7. At what point after exposure to influential inputs is a decision made?
8. Given the particulars of a specific case, will the outcome predictions of the knowledge engineering team be consistent with those of the expert?



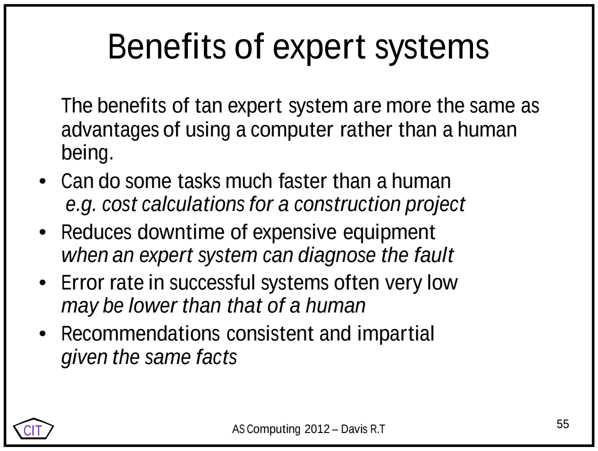
Uses of expert systems

- Medical diagnosis
- Fault diagnosis of all kinds
  gas boilers, computers, power stations, engines
- Geological surveys
  to find oil and mineral deposits



Uses of expert systems (continued)

- Financial services
  to predict stock market movement
- Social services
  to calculate the benefits due to claimants
- Industrial uses
  such as ELSIE in the construction industry

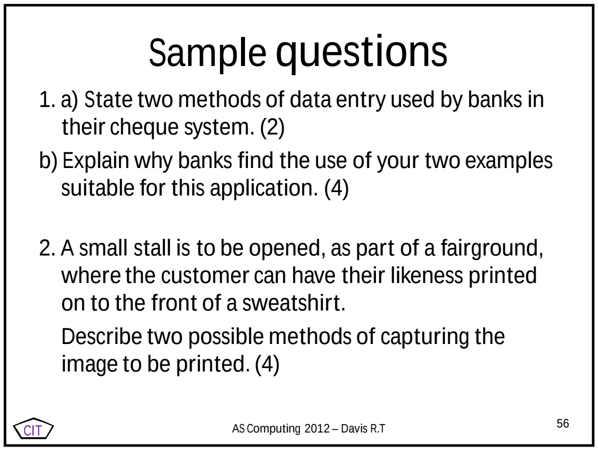


Benefits of expert systems

The benefits of tan expert system are more the same as advantages of using a computer rather than a human being.

- Can do some tasks much faster than a human *e.g. cost calculations for a construction project*
- Reduces downtime of expensive equipment *when an expert system can diagnose the fault*
- Error rate in successful systems often very low *may be lower than that of a human*
- Recommendations consistent and impartial *given the same facts*

AS Computing 2012 – Davis R.T 55



Sample questions

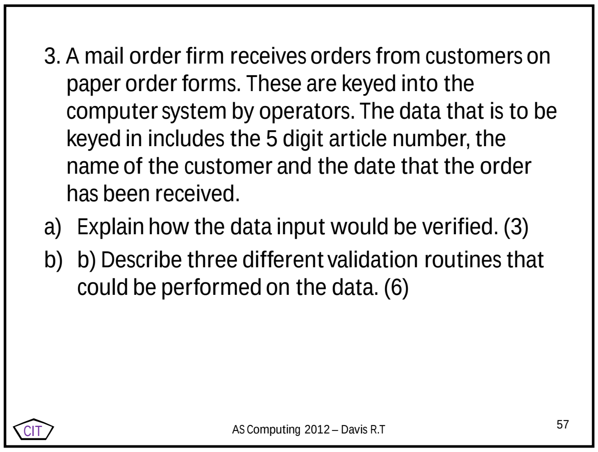
1. a) State two methods of data entry used by banks in their cheque system. (2)

b) Explain why banks find the use of your two examples suitable for this application. (4)

2. A small stall is to be opened, as part of a fairground, where the customer can have their likeness printed on to the front of a sweatshirt.

Describe two possible methods of capturing the image to be printed. (4)

AS Computing 2012 – Davis R.T 56

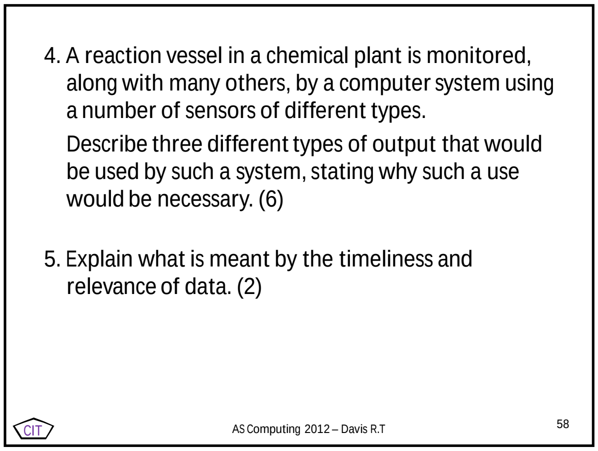


3. A mail order firm receives orders from customers on paper order forms. These are keyed into the computer system by operators. The data that is to be keyed in includes the 5 digit article number, the name of the customer and the date that the order has been received.

a) Explain how the data input would be verified. (3)

b) b) Describe three different validation routines that could be performed on the data. (6)

AS Computing 2012 – Davis R.T 57



4. A reaction vessel in a chemical plant is monitored, along with many others, by a computer system using a number of sensors of different types.

   Describe three different types of output that would be used by such a system, stating why such a use would be necessary. (6)

5. Explain what is meant by the timeliness and relevance of data. (2)

Validation and Verification

When data is input to a computer system it is only valuable data if it is correct. If the data is in error in any way then no amount of care in the programming will make up for the erroneous data and the results produced can be expected to be unreliable. There are three types of error that can occur with the data on entry. The first is that the data, while reasonable, is wrong. If your birthday is written down on a data capture form as 18th of November 1983, it will (except in very rare cases) be wrong. It can be typed into the computer with the utmost care as 181183, it can be checked by the computer to make sure that is a sensible date, and will then be accepted as your date of birth despite the fact that it is wrong. There is no reason for the computer to imagine that it may be wrong, quite simply when you filled out the original form you made a mistake. The second type of error is when the operator typing in the data hits the wrong key and types in 181193, or the equivalent. In this case an error has been made that should be able to be spotted if a suitable check is made on the input. This type of data checking is called a verification check. The third type of error is when something is typed in which simply is not sensible. If the computer knows that there are only 12 months in a year then it will know that 181383 must be wrong because it is not sensible to be born in the thirteenth month. Checks on the sensibility of the data are called validation checks.

Faulty data:

There is very little that can be done about faulty data except to let the owner of the data check it visually on a regular basis. The personal information kept on the school administration system about you and your family may well be printed off at regular intervals so that your parents can check to ensure that the stored information is still correct.

Verification:

Verification means checking the input data with the original data to make sure that there have been no transcription errors. The standard way to do this is to input the data twice to the computer system. The computer then checks the two sets of data (which should be the same) and if there is a difference between the two sets of data the computer knows that one of the inputs is wrong. It won’t know which one is wrong but it can now ask the operator to check that particular input.

Validation:

The first thing is to dispel a common misinterpretation of validation. In section 1.5.e (Networking) checking of data was mentioned. Specifically, the use of parity bits to check data. This is NOT validation. Parity bits and echoing back are techniques that are used to check that data has been transmitted properly within a computer system (e.g. from the disk drive to the processor), validation checks are used to check the input of data to the system in the first place.

Validation is a check on DATA INPUT to the system by comparing the data input with a set of rules that the computer has been told the data must follow. If the data does not match up with the rules then there must be an error. There are many different types of validation check that can be used to check input in different applications.

1. Range check. A mathematics exam is out of 100. A simple validation rule that the computer can apply to any data that is input is that the mark must be between 0 and 100 inclusive. Consequently, a mark of 101 would be rejected by this check as being outside the acceptable range.

2. Character check. A person's name will consist of letters of the alphabet and sometimes a hyphen or apostrophe. This rule can be applied to input of a person's name so that dav2d will immediately be rejected as unacceptable.

3. Format check. A particular application is set up to accept a national insurance number. Each person has a unique national insurance number, but they all have the same format of characters, 2 letters followed by 6 digits followed by a single letter. If the computer knows this rule then it knows what the format of a NI number is and would reject ABC12345Z because it is in the wrong format, it breaks the rule.

4. Length check. A NI number has 9 characters, if more or fewer than 9 characters are keyed in then the data cannot be accurate.

5. Existence check. A bar code is read at a supermarket checkout till. The code is sent to the main computer which will search for that code on the stock file. As the stock file contains details of all items held in stock, if it is not there then the item cannot exist, which it obviously does, therefore the code must have been wrongly read.

6. Check digit. When the code is read on the item at the supermarket, it consists of numbers. One number is special; it is called the check digit. If the other numbers have some arithmetic done to them using a simple algorithm the answer should be this special digit. When the code is read at the checkout till, if the arithmetic does not give the check digit it must have been read wrongly, it is at this point that the beeping sound would normally be heard if everything is alright.

LOGIC GATES

A logic gate performs a logical operation on one or more logic inputs and produces a single logic output.

A logic gate is an elementary building block of a digital circuit Every digital product, like computers, mobile, calculators even digital watches, contain logic gates. Most logic gates have two inputs and one output. At any given moment, every terminal is in one of the two binary conditions low (0) or high (1), represented by different voltage levels.

The three simple gates AND gate, OR gate and NOT gate which combine to perform complex decision making processes. The on or off state of a logic gate corresponds to the binary values.

The complex logic gates are XOR (exclusive-OR), NAND (NOT-AND), NOR (NOT-OR), and XNOR (exclusive-NOR).

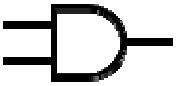
Truth Table

The Truth Table A truth table is a table that describes the behaviour of a logic gate or any combination of logic gates. A truth table shows how a logic circuit's output responds to various combinations of the inputs, using logic 1 for true and logic 0 for false

AND gate

The AND gate is so named because, if 0 is called "false" and 1 is called "true," the gate acts in the same way as the logical "and" (X) operator.
The output is "true" when both inputs are "true." Otherwise, the output is "false."

Symbol



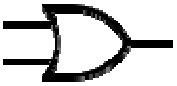
Truth table

INPUT		OUTPUT
A	**B**	**A AND B**
0	0	0
0	1	0
1	0	0
1	1	1

OR gate

The OR gate gets its name from the fact that it behaves after the fashion of the logical inclusive "or."(+) The output is "true" if either or both of the inputs are "true." If both inputs are "false," then the output is "false."

Symbol



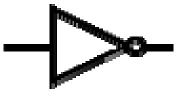
Truth table

INPUT		OUTPUT
A	**B**	**A OR B**
0	0	0
0	1	1
1	0	1
1	1	1

NOT gate

A logical inverter , sometimes called a NOT gate to differentiate it from other types of electronic inverter devices, has only one input. It reverses the logic state.

Symbol

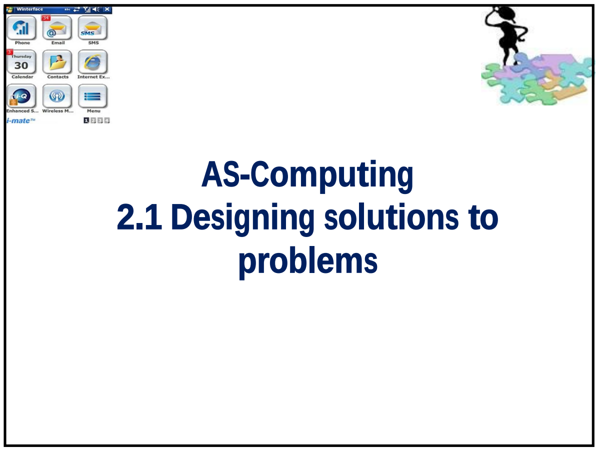


Truth table

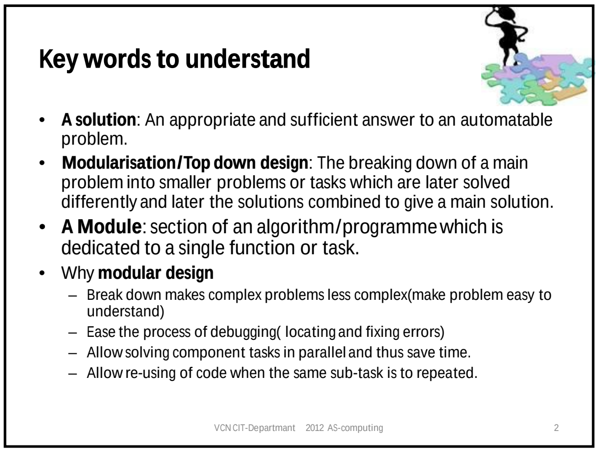
INPUT	OUTPUT
1	0
0	1

Applications

Half Adders

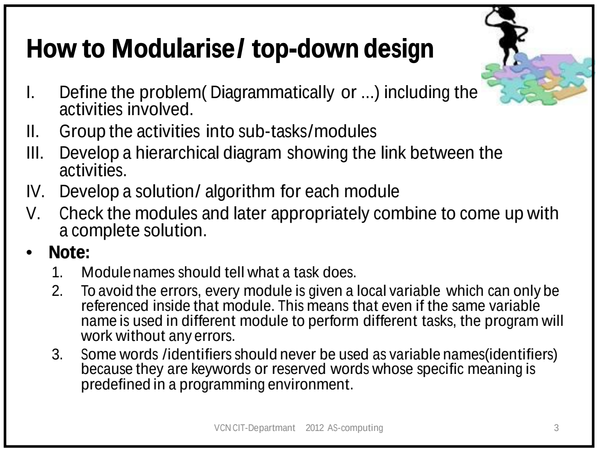


AS-Computing
2.1 Designing solutions to problems



Key words to understand

- **A solution**: An appropriate and sufficient answer to an automatable problem.
- **Modularisation/Top down design**: The breaking down of a main problem into smaller problems or tasks which are later solved differently and later the solutions combined to give a main solution.
- **A Module**: section of an algorithm/programme which is dedicated to a single function or task.
- Why **modular design**
  - Break down makes complex problems less complex(make problem easy to understand)
  - Ease the process of debugging(locating and fixing errors)
  - Allow solving component tasks in parallel and thus save time.
  - Allow re-using of code when the same sub-task is to repeated.



How to Modularise/ top-down design

I. Define the problem(Diagrammatically or …) including the activities involved.
II. Group the activities into sub-tasks/modules
III. Develop a hierarchical diagram showing the link between the activities.
IV. Develop a solution/ algorithm for each module
V. Check the modules and later appropriately combine to come up with a complete solution.

- **Note:**
  1. Module names should tell what a task does.
  2. To avoid the errors, every module is given a local variable which can only be referenced inside that module. This means that even if the same variable name is used in different module to perform different tasks, the program will work without any errors.
  3. Some words /identifiers should never be used as variable names(identifiers) because they are keywords or reserved words whose specific meaning is predefined in a programming environment.

Top-down design/ modularisation questi

1.Nathan is designing a software solution for stock control in a mobile phone shop. He has a colleague, called Andre, who will help him write the program. Nathan decides to modularise the solution.

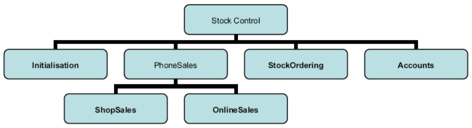
(a) As the first step in his design he splits the solution into the following main areas: Initialisation, PhoneSales, StockOrdering, Accounts. Use a structure/ tree diagram to show this.

(b) Nathan will write the ShopSales module and Andre will write the OnlineSales module. Nathan will use the identifier Sale for a sale in the shop, and Andre will use the identifier Sale for an online order. Explain how they can both use the same identifier and not cause a problem when the program is run.

(C) Both programmers need to choose other identifiers that they will use. Explain why there are some words that cannot be used as identifiers.

Answers to the question

(a).



(b) These will be used as local variables to only have effect in the module they are in, and therefore have no meaning outside that module

(c) Such identifiers are keywords/reserved words which already exist in a word in the vocabulary of the language and there fore can only have the meaning defined in that language

Interface design

The Golden Rules ("What to keep in Mind")

Definition: An interface is a medium that allows two parties to com with each other.

The Rules:

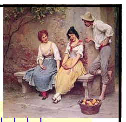
1. Strive for consistency
   - consistent sequences of actions should be required in similar situations
   - identical terminology should be used in prompts, menus, and help screens
   - **consistent color, layout, capitalization, fonts, and so on should be employed throughout.**
2. Enable frequent users to use shortcuts
   - to increase the pace of interaction use abbreviations, special keys, hidden commands, and macros
3. Offer informative feedback
   - for every user action, the system should respond in some way (in web design, this can be accomplished by DHTML - for example, a button will make a clicking sound or change color when clicked to show the user something has happened)
4. Design dialogs to yield closure
   - Sequences of actions should be organized into groups with a beginning, middle, and end. The informative feedback at the completion of a group of actions shows the user their activity has completed successfully

What do you say?????????
What are they up to??????????

Interface design
The Golden Rules (“What to keep in Mind”)

5. Offer error prevention and simple error handling
   - design the form so that users cannot make a serious error; for example, prefer menu selection to form fill-in and do not allow alphabetic characters in numeric entry fields
   - if users make an error, instructions should be written to detect the error and offer simple, constructive, and specific instructions for recovery
   - segment long forms and send sections separately so that the user is not penalized by having to fill the form in again - but make sure you inform the user that multiple sections are coming up
6. Permit easy reversal of actions
7. Support internal locus of control
   - Experienced users want to be in charge. Surprising system actions, tedious sequences of data entries, inability or difficulty in obtaining necessary information, and inability to produce the action desired all build anxiety and dissatisfaction
8. Reduce short-term memory load
   - A famous study suggests that humans can store only 7 (plus or minus 2) pieces of information in their short term memory. You can reduce short term memory load by designing screens where options are clearly visible, or using pull-down menus and icons

Data capture forms, Back then...



In a paper based environment, we were given papers with drawn boxes.

Limitations

•Unreadable and poor hand writings

•Limited space for descriptions or details

•Combing of data for processing was very hard and time consuming

•Subject to human errors

•Limited room for correcting human errors on the data capture form.

First Name |__|__|__|__|__|__|__|__|__|__|__|__|__|__|__|__|__|

Second Name |__|__|__|__|__|__|__|__|__|__|__|__|__|__|__|__|__|

Sex (*M or F*) |__|

Date of Birth (*Day / Month / Year*)

|__|__|__|__|__|__|

Have you seen the television advertisements (featuring Gary Lineker) for Walkers Crisps?
Please tick the appropriate box

Yes |__|

No |__|

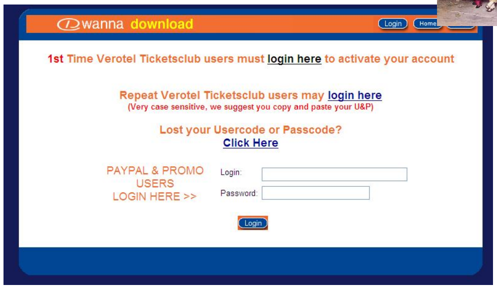
Have you bought Walkers Crisps in the last week?

Yes |__|

No |__|

Data capture/ input forms, Today…

Example



1st Time Verotel Ticketsclub users must login here to activate your account

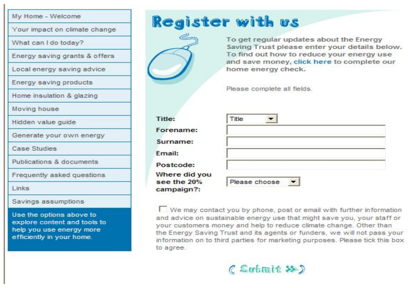
Repeat Verotel Ticketsclub users may login here
(Very case sensitive, we suggest you copy and paste your U&P)

Lost your Usercode or Passcode?
Click Here

PAYPAL & PROMO USERS LOGIN HERE >>

Login:
Password:

Data capture forms(DCF), Today…



Use the options above to explore content and tools to help you use energy more efficiently in your home.

Register with us

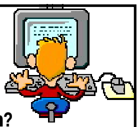
To get regular updates about the Energy Saving Trust please enter your details below. To find out how to reduce your energy use and save money, click here to complete our home energy check.

Please complete all fields.

Title: Title
Forename:
Surname:
Email:
Postcode:
Where did you see the 20% campaign?: Please choose

□ We may contact you by phone, post or email with further information and advice on sustainable energy use that might save you, your staff or your customers money and help to reduce climate change. Other than the Energy Saving Trust and its agents or funders, we will not pass your information on to third parties for marketing purposes. Please tick this box to agree.

Data capture forms(DCF), Today…

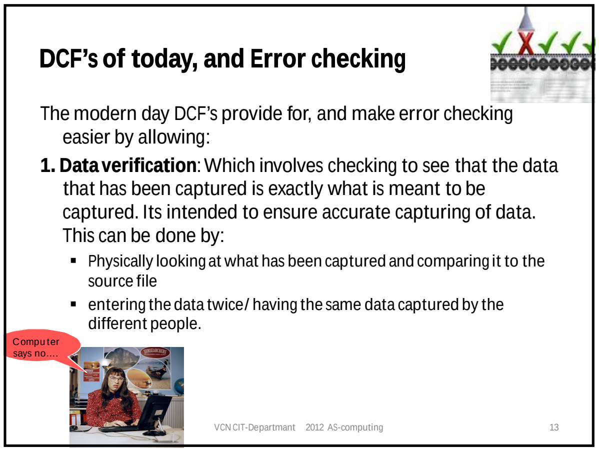


What are the main features of a Data Capture Form?

Features of a DCF

- Title
- Instructions to user (Cancel, clear, Back, Next, submit, etc.)
- Boxes to fill in
- Starred sections that must be filled in
- Combo boxes (for on line forms with drop down boxes)

Note: Also consider the positions of the buttons/instructions and the boxes on the screen. This is called screen Layout.

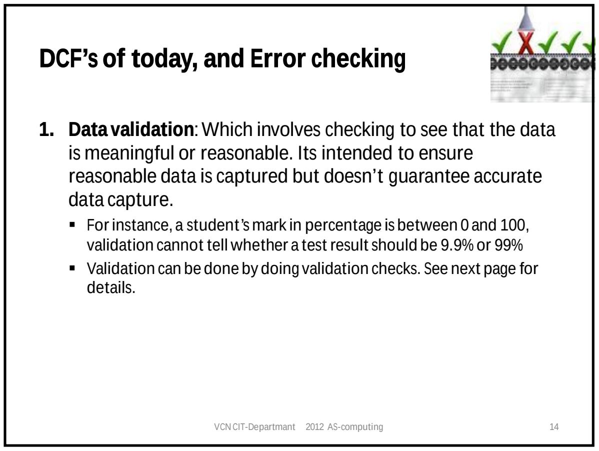


DCF's of today, and Error checking

The modern day DCF's provide for, and make error checking easier by allowing:

1. **Data verification**: Which involves checking to see that the data that has been captured is exactly what is meant to be captured. Its intended to ensure accurate capturing of data. This can be done by:
   - Physically looking at what has been captured and comparing it to the source file
   - entering the data twice/ having the same data captured by the different people.

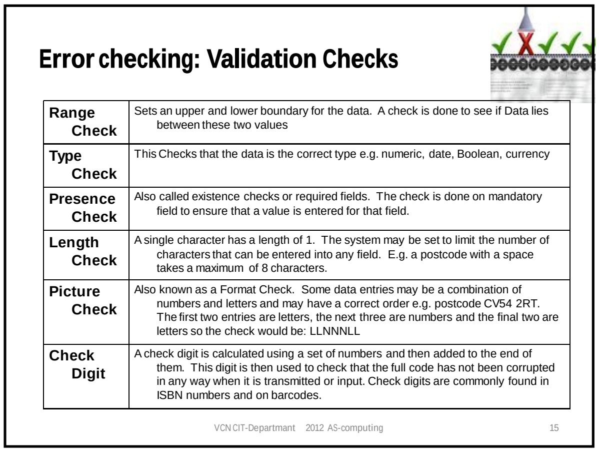
VCN CIT-Departmant 2012 AS-computing 13



DCF's of today, and Error checking

1. **Data validation**: Which involves checking to see that the data is meaningful or reasonable. Its intended to ensure reasonable data is captured but doesn't guarantee accurate data capture.
   - For instance, a student's mark in percentage is between 0 and 100, validation cannot tell whether a test result should be 9.9% or 99%
   - Validation can be done by doing validation checks. See next page for details.

VCN CIT-Departmant 2012 AS-computing 14



Error checking: Validation Checks

Range Check	Sets an upper and lower boundary for the data. A check is done to see if Data lies between these two values
Type Check	This Checks that the data is the correct type e.g. numeric, date, Boolean, currency
Presence Check	Also called existence checks or required fields. The check is done on mandatory field to ensure that a value is entered for that field.
Length Check	A single character has a length of 1. The system may be set to limit the number of characters that can be entered into any field. E.g. a postcode with a space takes a maximum of 8 characters.
Picture Check	Also known as a Format Check. Some data entries may be a combination of numbers and letters and may have a correct order e.g. postcode CV54 2RT. The first two entries are letters, the next three are numbers and the final two are letters so the check would be: LLNNNLL
Check Digit	A check digit is calculated using a set of numbers and then added to the end of them. This digit is then used to check that the full code has not been corrupted in any way when it is transmitted or input. Check digits are commonly found in ISBN numbers and on barcodes.

VCN CIT-Departmant 2012 AS-computing 15

AS Hand out 3 Section 2.2

2.2 THE STRUCTURE OF PROCEDURAL PROGRAMS

Procedural programs are ones in which instructions are executed in the order defined by the programmer. Procedural languages are often referred to as third generation languages and include FORTRAN, ALGOL, COBOL, BASIC, and PASCAL.

SOME TERMS IN PROCEDURAL LANGUAGE

Statement

A statement is a single instruction in a program, which can be converted into machine code and executed. In most languages a statement is written on a single line, but some languages allow multiple lines for single statements.

Examples of statements are:

```
DIM name As String
A=X*X
While x < 10
```

Subroutine

A subroutine is a self-contained section of program code that performs a specific task, as part of the main program.

Procedure

A procedure is a *subroutine* that performs a specific task without returning a value to the part of the program from which it was called.

Function

A function is a *subroutine* that performs a specific task and returns a value to the part of the program from which it was called.
Note: that a function is 'called' by writing it on the right hand side of an assignment statement.

Parameter (argument)

A parameter is a value that is 'passed' (given) to a subroutine (procedure or function).
The subroutine uses the value of the parameter within its execution. The action of the subroutine will be different depending upon the parameters that it is passed.

Parameters are placed in parenthesis after the subroutine name. For example:

```
Square(5) 'passes the parameter 5 – returns 25
Square(8) 'passes the parameter 8 – returns 64
Square(x) 'passes the value of the variable x
CalculateVolume(7,14,8) 'three parameters are passed
```

Control structures

Computer programs execute statements one line after the next, unless there is a command that instructs the program to 'jump' backwards or forwards to an alternative line. Some common Control Structures include:-

Sequence

Sequence is when the programming statements are executed one after the other, in the order in which they appear in the program.

Selection

Selection is a control structure in which there is a test to decide if certain instructions are to be executed.

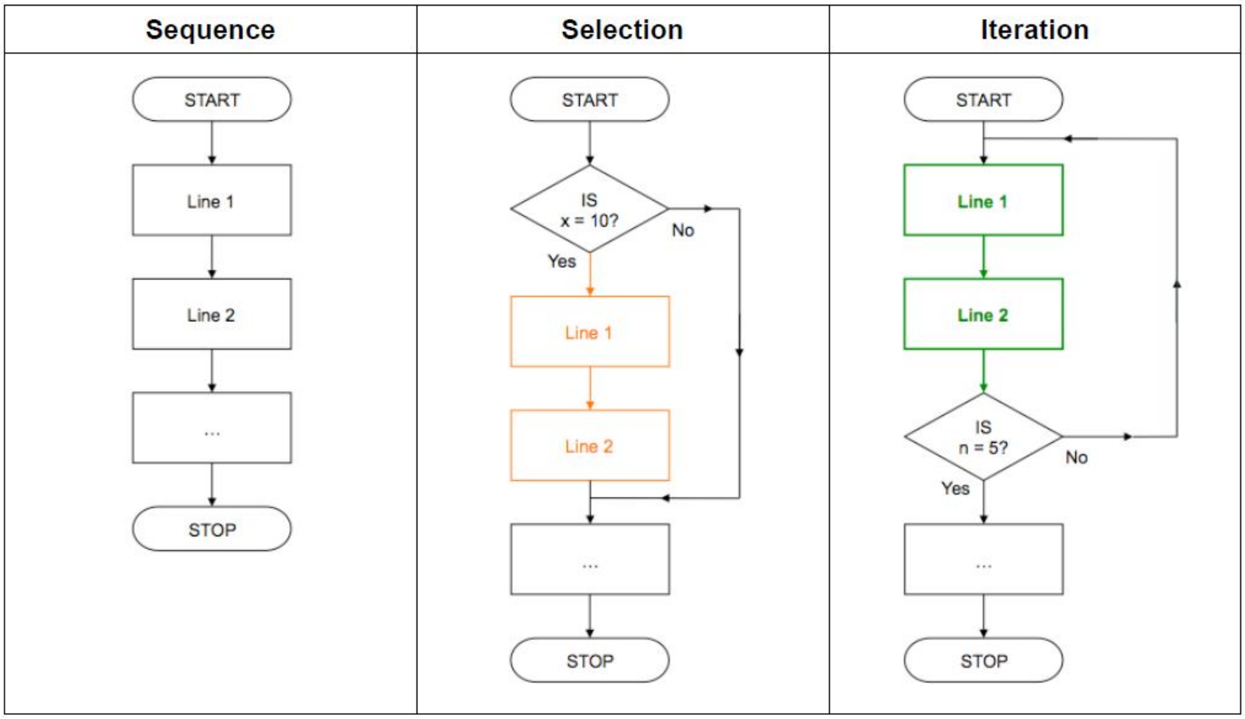
```
If x < 10 then
  Do this line
  And this line
Else
  Do this line
End if
```

Iteration

Iteration is a control structure in which a group of statements is executed repeatedly – either a set number of times or until a specific condition is met.

```
Repeat
  Do this line
  And this line
Until n=5
```

Flowcharts showing the control structures



Selection

Including the use of IF statements and CASE/SELECT statements.

IF-THEN-ELSE

This selection method is used if there are two possible outcomes to a test:

```
IF x < 0 THEN
  OUTPUT “Sorry, you can’t have negative values”
ELSE
  a = x*x
  OUTPUT a
END
```

SELECT-CASE

This selection method is used if there are more than two possible outcomes to a test:

```
SELECT CASE KeyPress
 CASE LeftArrow
   Move one character backwards
 CASE RightArrow
   Move one character forwards
 CASE UpArrow
   Move one character up
 CASE DownArrow
   Move one character down
END SELECT
```

Iteration

Including the use of count-controlled loops (FOR-NEXT loops) and condition-controlled loops (WHILE-ENDWHILE and REPEAT-UNTIL loops).

FOR-NEXT

This is an unconditional loop in which the number of repetitions is set at the beginning.

```
FOR X = 1 TO 5
  Answer = X*3
  OUTPUT X, Answer
NEXT
```

WHILE-ENDWHILE

This is a conditional loop, which has a test at the start and repeats until the condition is false:

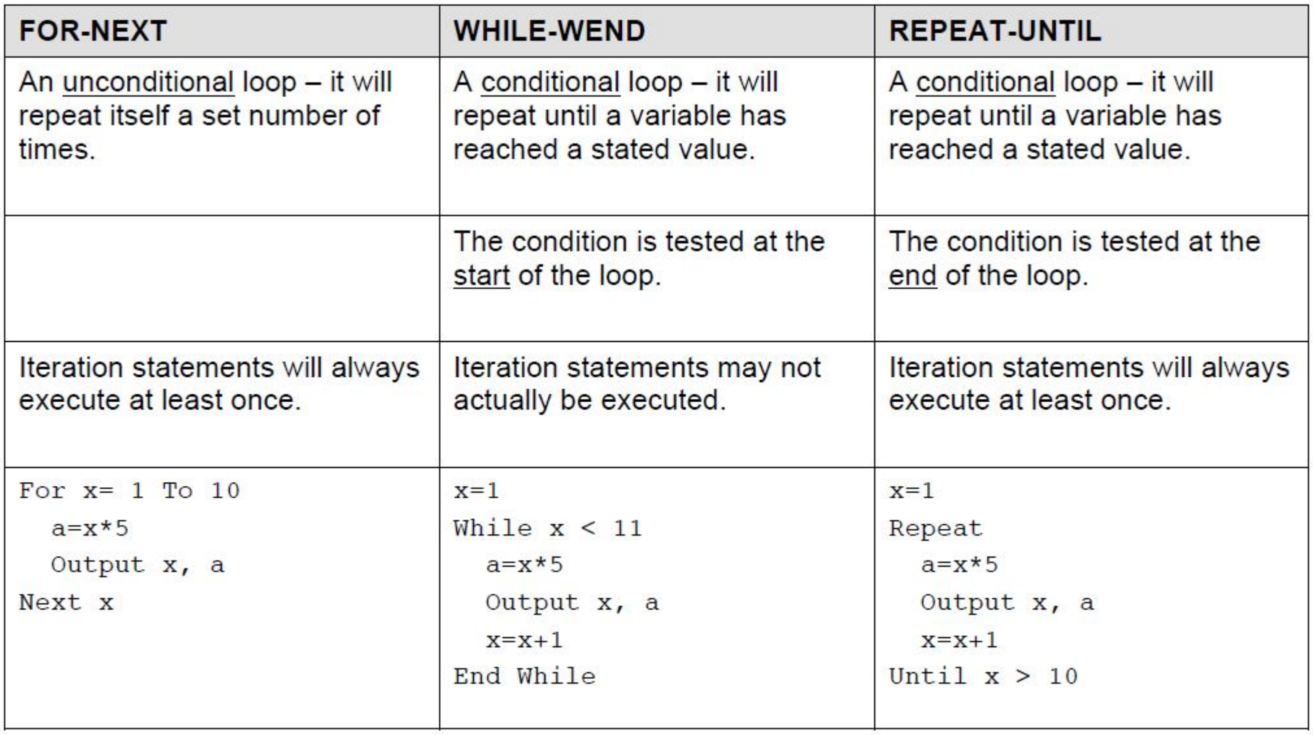
```
X = 0
WHILE X < 6 DO
  X = X + 1
  Answer = X*3
  OUTPUT X, Answer
END
```

REPEAT-UNTIL

This is a conditional loop, which has a test at the end and repeats until the condition is true:

```
X = 0
REPEAT
  X = X + 1
  Answer = X*3
  OUTPUT X, Answer
UNTIL X > 4
```

Comparison of the different iterations



FOR-NEXT	WHILE-WEND	REPEAT-UNTIL
An unconditional loop – it will repeat itself a set number of times.	A conditional loop – it will repeat until a variable has reached a stated value.	A conditional loop – it will repeat until a variable has reached a stated value.
	The condition is tested at the start of the loop.	The condition is tested at the end of the loop.
Iteration statements will always execute at least once.	Iteration statements may not actually be executed.	Iteration statements will always execute at least once.

For x= 1 To 10
  a=x*5
  Output x, a
Next x
	x=1
	While x < 11
	  a=x*5
	  Output x, a
	  x=x+1
	End While
		x=1
		Repeat
		  a=x*5
		  Output x, a
		  x=x+1
		Until x > 10

Nested selection and iteration

Nested selection

This is where there is an IF statement contained within an IF statement.

The following algorithm allows a maximum of four attempts to login to a computer system:

```
INPUT Password
IF NumberOfTries < 5 THEN
  IF Password is correct THEN
    OUTPUT “Successful Login”
  ELSE
   OUTPUT “Password was incorrect”
  END
ELSE
  OUTPUT “You have made too many attempts”
END
```

Nested iteration

This is where there is a loop within a loop.
A nested iteration is needed to initialise a two-dimensional array:

```
FOR row = 0 TO 7
  FOR column = 0 TO 5
    SET MyArray (row, column) = 0
  NEXT column
NEXT row
```

Subroutines

Subroutine/sub-program

A subroutine is a self-contained section of program code which performs a specific task and is referenced by a name.
A subroutine resembles a standard program in that it will contain its own local variables, data types, labels and constant declarations.

There are two types of subroutine. These are procedures and functions.

- procedures are subroutines that input, output or manipulate data in some way;
- functions are subroutines that return a value to the main program.

A subroutine is executed whenever its name is encountered in the executable part of the main program. The execution of a subroutine by referencing its name in the main program is termed ‘calling’ the subroutine.

Advantages of subroutines.

The advantage of using procedures and functions are that:

- the same lines of code are re-used whenever they are needed – they do not have to be repeated in different sections of the program.
- a procedure or function can be tested/improved/rewritten independently of other procedures or functions.
- it is easy to share procedures and functions with other programs – they can be incorporated into library files which are then ‘linked’ to the main program;
- a programmer can create their own routines that can be called in the same way as any built-in command.

Recursion

Recursion is when a function or a procedure contains a call to itself.
Recursion is used when calculating factorials:

```
FUNCTION Factorial(n)
  IF n = 1 THEN
    RETURN 1
  ELSE
    RETURN n * Factorial(n – 1)
  END IF
END
```

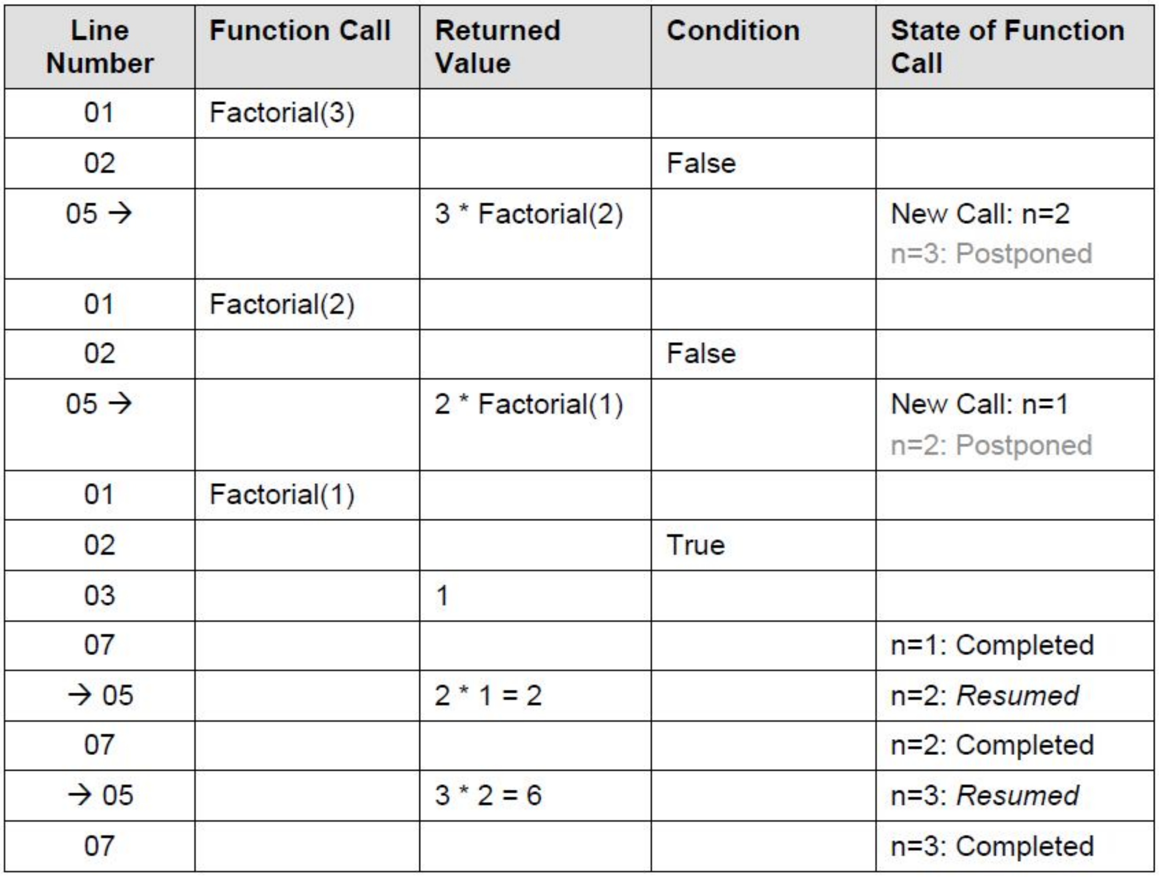
Note that a recursive function needs a ‘stopping condition’ – i.e. a statement that will stop it from repeating infinitely.

In the Factorial function above the stopping condition is IF n = 1.

Tracing the execution of x = Factorial(3)

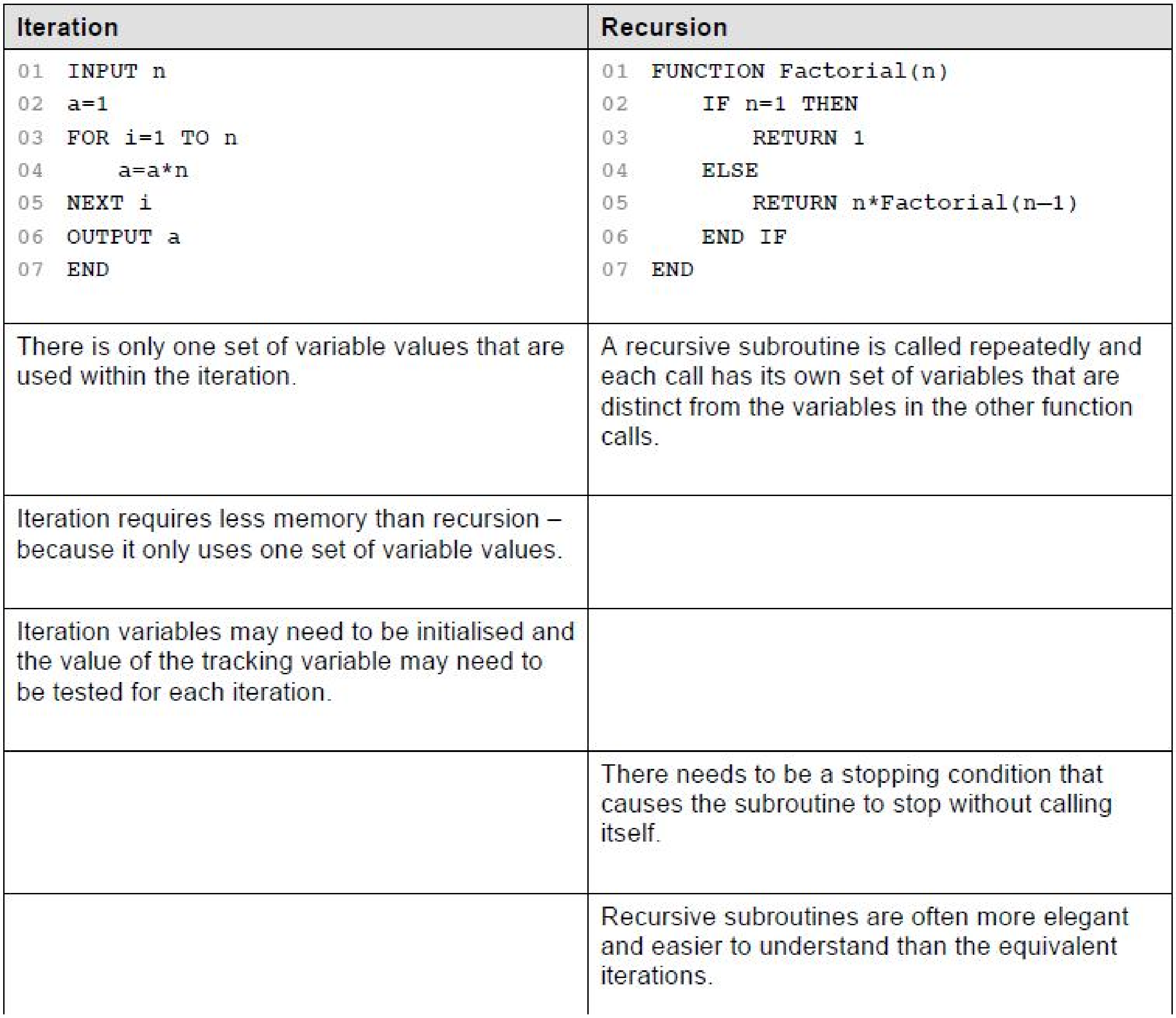
```
01 FUNCTION Factorial(n)
02  IF n = 1 THEN
03   RETURN 1
04  ELSE
05   RETURN n * Factorial(n – 1)
06  END IF
07 END
```

Note that in the above algorithm, the recursion ‘happens’ on line 05.



Line Number	Function Call	Returned Value	Condition	State of Function Call
01	Factorial(3)			
02			False	
05 →		3 * Factorial(2)		New Call: n=2 n=3: Postponed
01	Factorial(2)			
02			False	
05 →		2 * Factorial(1)		New Call: n=1 n=2: Postponed
01	Factorial(1)			
02			True	
03		1		
07				n=1: Completed
→ 05		2 * 1 = 2		n=2: *Resumed*
07				n=2: Completed
→ 05		3 * 2 = 6		n=3: *Resumed*
07				n=3: Completed

Iteration compared to recursion



Iteration	Recursion

`INPUT n`
`a=1`
`FOR i=1 TO n`
`    a=a*n`
`NEXT i`
`OUTPUT a`
`END`
	`FUNCTION Factorial(n)`
	`    IF n=1 THEN`
	`        RETURN 1`
	`    ELSE`
	`        RETURN n*Factorial(n–1)`
	`    END IF`
	`END`

There is only one set of variable values that are used within the iteration.	A recursive subroutine is called repeatedly and each call has its own set of variables that are distinct from the variables in the other function calls.
Iteration requires less memory than recursion – because it only uses one set of variable values.	
Iteration variables may need to be initialised and the value of the tracking variable may need to be tested for each iteration.	
	There needs to be a stopping condition that causes the subroutine to stop without calling itself.
	Recursive subroutines are often more elegant and easier to understand than the equivalent iterations.

Hand out 1 (2.3 Datatypes & Data Structures)

Data types and data structures

Definition of data type:- is a classification identifying one of various types of data, such as floating-point, integer, or Boolean, that determines the possible values for that type; the operations that can be done on that type; and the way the values of that type are stored.

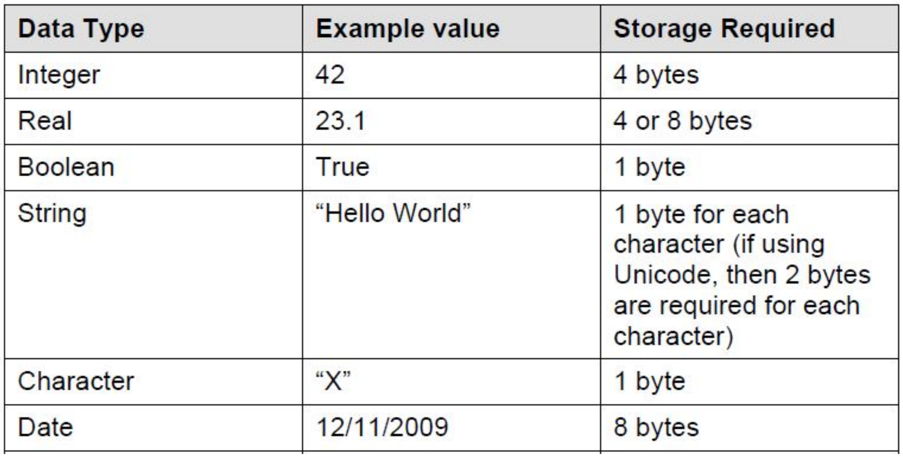
Data types

Boolean - A variable which can only hold only two values the value **true** or **false**
Integer/Numeric- Integers are the set of whole numbers and their opposites
Character/text - alphanumeric text, such as letters, numbers, spaces, symbols, and punctuation
String – Is a datatype derived from Text datatype and includes more than one character
Date/Time-
Currency - can carry money values like $,€,£ etc

Sizes of datataypes

Boolean – Has a size of 1 Byte
Integer/Numeric - each digit caries a size of 2 Bytes
Character/text - alphanumeric each character caries 1 Byte
String – A total number of all the bytes
Date/Time- size ranges from 5 to 10 Bytes
Currency – normally not smaller than 4 bytes, 2 bytes for the sign and other bytes for the digits

Example



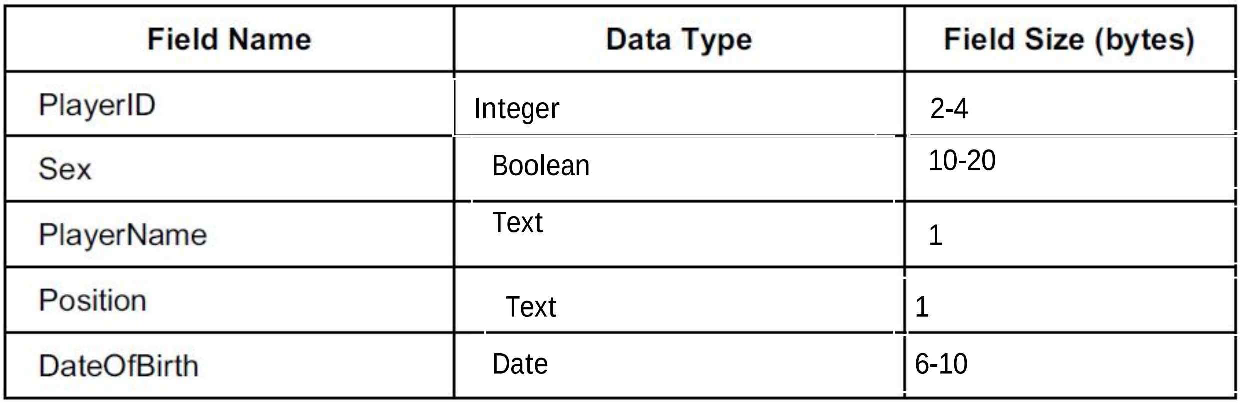
Data Type	Example value	Storage Required
Integer	42	4 bytes
Real	23.1	4 or 8 bytes
Boolean	True	1 byte
String	“Hello World”	1 byte for each character (if using Unicode, then 2 bytes are required for each character)
Character	“X”	1 byte
Date	12/11/2009	8 bytes

Estimating the size of a file

Step 1: Add up the ranges of a single record
Step 2: Multiply by the number of records
Step 3: add 10% for error
Step 4: Divide by 1024 consecutively to get Kb, Mb, Gb etc

Example:

Ahmed needs to store more information about the players. He creates a record structure that contains PlayerID (a whole number between 1 and 50), Sex (m or f), PlayerName, Position (f, d or g), and DateOfBirth. Assuming there are 1000 records



Field Name	Data Type	Field Size (bytes)
PlayerID	Integer	2-4
Sex	Boolean	10-20
PlayerName	Text	1
Position	Text	1
DateOfBirth	Date	6-10

Calculation:

1 record 2+1+10+1+6=20 , 4+1+20+1+10=36
1000 records =20 X 1000= 20000 to 36 X 1000= 36000
Error +10% = 22000 - 39000
Divide by 1024 = 22 – 39 Kb
Therefore the file will be approximately 39 Kb

User-defined data types

At times it is useful for programmers to be able to define their own data types. In such cases it is common for these user-defined data types to contain items from several of the different intrinsic types mentioned above.

Arrays

An array is a data structure, which allows a set of items of identical data type to be stored together using the same identifier name.

Arrays are declared in a similar way to standard variables, except that *the array name, Datatype, size and dimensions* are included. For example, the following declares an array that reserves five locations in memory and labels these as ‘Names’:

```
DIM Names(4) As String
```

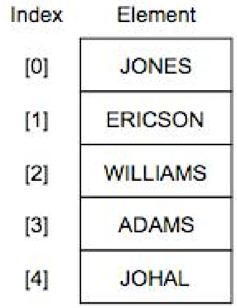
The five individual locations are Names(0), Names(1), Names(2), Names(3), Names(4).
Each data item is called an *element* of the array. To reference a particular element a programmer must use the appropriate *index*. For example, the following statement assigns data to the 5th element:

```
Names(4) = “Johal”
```

There are two types of arrays

Single Dimension array- A one-dimensional array is a data structure in which the array is declared using a single index and can be visually represented as a list.

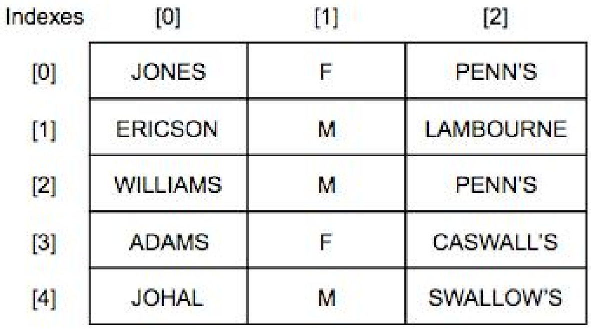
The following diagram shows the visual representation of the array Names(4):



Index	Element
[0]	JONES
[1]	ERICSON
[2]	WILLIAMS
[3]	ADAMS
[4]	JOHAL

Multi dimension Array - A Multi-dimensional array is a data structure in which the array is declared using two or more indices and can be visually represented as a table.

The following diagram shows the visual representation of an array Students(4,2):



Indexes	[0]	[1]	[2]
[0]	JONES	F	PENN'S
[1]	ERICSON	M	LAMBOURNE
[2]	WILLIAMS	M	PENN'S
[3]	ADAMS	F	CASWALL'S
[4]	JOHAL	M	SWALLOW'S

Each individual element can be referenced by its row and column indices. For example:

```
Students(0,0) is the data item “JONES”
Students(2,1) is the item “M”
Students(1,2) is the item “LAMBOURNE”
```

Initialization of an array

Size of array calculated
-Location of array decided according to data type/size and the locations are reserved.
-Array named in look up table and size of array stored in table
-Lower/Upper bound of array stored in table
-Data type stored in table
-Address of first element stored in table

How arrays are searched serially

-Index set to 0 (or other sensible value)
-Array(index) searched
-IF = Item then “Found”
-ELSE increment Index and repeat line 2
-Until found or produce “Error Report”

EXERCISES

Exercise: The following fields are to be stored.

- Customer name (to allow customer to be addressed properly when contacted)
- Customer telephone number (so that customer can be contacted when their order has been completed)
- Date of original commission (so that customers are not kept waiting too long)
- Whether or not the order has been paid for.

(a) State a suitable data type for each of the four fields. [4]
(b) It is assumed that there will never be more than 1000 records.

Estimate the total size of the file needed for these records. [4]

Exercise The data held about each car includes:

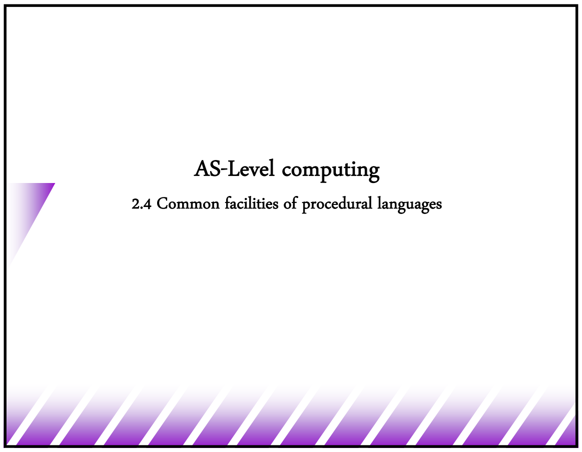
- Colour,
- Size of engine,
- Whether or not it has air conditioning,
- Price.

State a suitable data type for each of the items of data listed and estimate the size of the file if 10000 records are stored.

Example A stock file in a company has records of all the different items held in stock. The records each hold a number of fields:

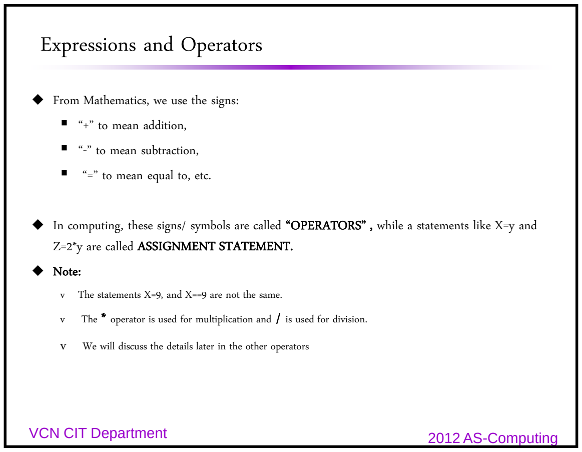
- the name of the item in stock
- description of the item
- cost
- whether or not in stock
- number in stock

(a) (i) State a suitable data type for each field.
(ii) If there are 1000 items in stock, estimate the size of the stock file. (Show how you worked out your answer.)



AS-Level computing

2.4 Common facilities of procedural languages

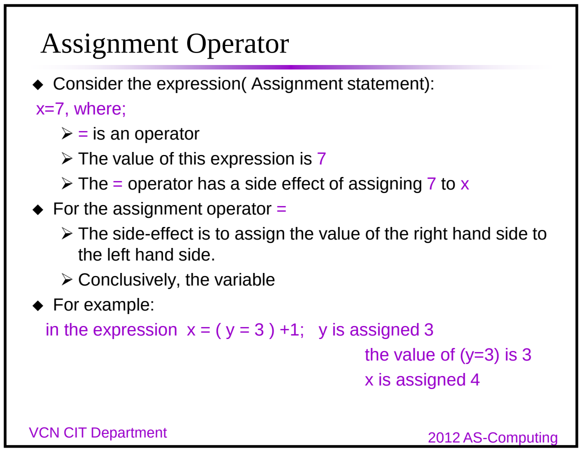


Expressions and Operators

- From Mathematics, we use the signs:
  - "+" to mean addition,
  - "-" to mean subtraction,
  - "=" to mean equal to, etc.
- In computing, these signs/ symbols are called **"OPERATORS" ,** while a statements like X=y and Z=2*y are called **ASSIGNMENT STATEMENT.**
- **Note:**
  - The statements X=9, and X==9 are not the same.
  - The * operator is used for multiplication and / is used for division.
  - We will discuss the details later in the other operators

VCN CIT Department

2012 AS-Computing



Assignment Operator

- Consider the expression(Assignment statement):

x=7, where;

  - = is an operator
  - The value of this expression is 7
  - The = operator has a side effect of assigning 7 to x
- For the assignment operator =
  - The side-effect is to assign the value of the right hand side to the left hand side.
  - Conclusively, the variable
- For example:

in the expression x = (y = 3) +1; y is assigned 3
the value of (y=3) is 3
x is assigned 4

VCN CIT Department

2012 AS-Computing

Arithmetic Operators

Operator	Symbol	Action	Example
Addition	+	Adds operands	x + y
Subtraction	-	Subs second from first	x - y
Negation	-	Negates operand	-x
Multiplication	*	Multiplies operands	x * y
Division	/	Divides first by second (integer quotient)	x / y
Modulus (MOD)	%	Divide and give the remainder as the answer	x % y

VCN CIT Department

2012 AS-Computing

Compound Assignment Operator

- The assignment operator can be used to update a variable. Eg.
  - x=x+1 means the new value of x will be the previous value plus 1, and
  - x=x*2, the new value of x is the previous value multiplied by 2...
- Sometimes they are written short form. for instance:

variable Op= expr   equivalent to variable = variable Op expr

Operator	Equivalent to:
x *= y	x = x * y
y -= z + 1	y = y - (z + 1)
a /= b	a = a / b
x += y / 8	x = x + (y / 8)
y %= 3	y = y % 3

– Update forms have value equal to the final value of expr
  - ❖ i.e.,   x=3; y= (x+=3);   /* x and y both get value 6 */

VCN CIT Department

2012 AS-Computing

Relational Operators

- Relational operators are used to compare variables.
  - They return a 1 value for true and a 0 for false.

Operator	Symbol	Example
Equals/ same as	==	x == y , NOT x = y
Greater than	>	x > y
Less than	<	x < y
Greater/equals	>=	x >= y
Less than/equals	<=	x <= y
Not equal	<>	x <> y

VCN CIT Department

2012 AS-Computing

Boolean Operators

- Boolean operators are applied on boolean variables(i.e. binary digits). A boolean variable can only take one of the two possible state. For instance, A person can either be Female or Male but not both.
- The operators are:
  - **AND**- this returns a true if both the right and left hand side of the operator are true.
  - **OR**- this is operator is used if we need to return a true for as long as any of the right and left hand side of the operator is true.
  - **NOT**- this is used to return the opposite of any variable or expression.
- Examples: consider x and y to be Boolean variable so that x=1and y=0, evaluate:
  - yANDx,
  - NOTy,
  - NOT((yORx)ANDy)
  - The output after using a Boolean operator on Boolean variable is always of Boolean type.

Operator precedence

Some expressions like **x+y/z*x** are complex to solve. Without specific principles, we are bound to get different answers and thus necessitating Operator precedence. This helps us to know where to start from when solving such expressions and thus get the same answer.

This is the precedence.

() []	Left Assoc.	(High)
* / MOD(%)	Left Assoc.	
+ -	Left Assoc.	
< <= > >=	Left Assoc.	
== <>	Left Assoc.	
^	Left Assoc.	
|	Left Assoc.	
AND	Left Assoc.	
OR	Left Assoc.	
? :	Left Assoc.	(Low)

The "Must-know" in String manipulation

- Every letter, number or symbol has a respective binary representation in a computer. Capital letters have a lower binary value than their respective lower-case letters.
- A character is an "allowable" letter or a symbol
- A **string** is a group of characters
- A **function** is a section of code within a program that performs a specific operation and returns a value. Eg. The LENGTH function that is used to determine the length of a string.
- **String Manipulation** involves being able to use operators and Computer in-built functions to be able to execute certain operations on strings.

Common In-Built functions

Function Name	Description	
LENGTH()	Find the length of the word in the brackets and returns the number of letters in the word.	
LOCATE()	Used to locate a letter or letters in a string. And give its position	
LEFT	Used to extract the left-hand most character in a string	
MID	Used to extract the middle character in a string	
RIGHT	Used to get the right-hand most character from a string	
COMPARE/COMP ()	Used to compare two strings	
CAT/CONCATNATE()	Used to combine two strings	

VCN CIT Department

2012 AS-Computing

Application of Operators and in-built functions

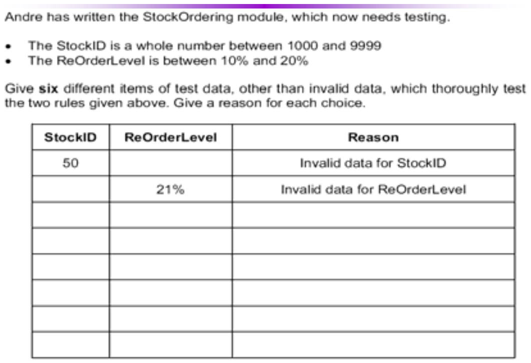
- Data validation: this involves if determining if data is complete, meaningful and follows rules.
- To evaluate expressions.

Now it's Practice questions!!!

VCN CIT Department

2012 AS-Computing

Revision questions



Andre has written the StockOrdering module, which now needs testing.

- The StockID is a whole number between 1000 and 9999
- The ReOrderLevel is between 10% and 20%

Give **six** different items of test data, other than invalid data, which thoroughly test the two rules given above. Give a reason for each choice.

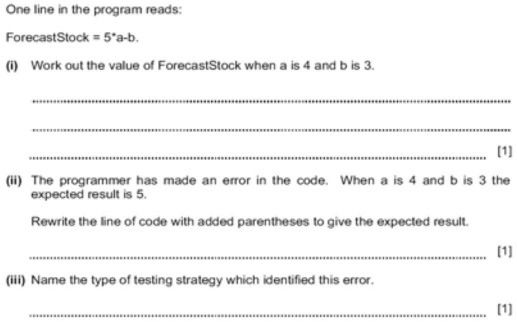
StockID	ReOrderLevel	Reason
50		Invalid data for StockID
	21%	Invalid data for ReOrderLevel

VCN CIT Department

2012 AS-Computing

Revision questions

- Andre writes a module which is part of a computer program.



One line in the program reads:

ForecastStock = 5*a-b.

(i) Work out the value of ForecastStock when a is 4 and b is 3.

....

....

.... [1]

(ii) The programmer has made an error in the code. When a is 4 and b is 3 the expected result is 5.

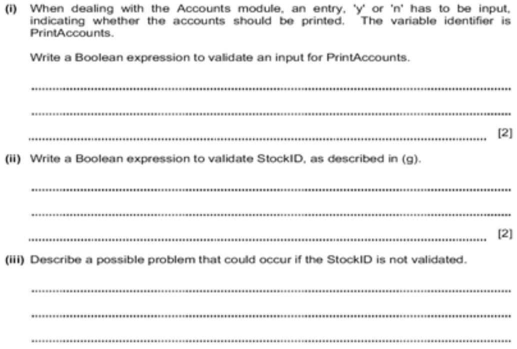
Rewrite the line of code with added parentheses to give the expected result.

.... [1]

(iii) Name the type of testing strategy which identified this error.

.... [1]

Revision questions



(i) When dealing with the Accounts module, an entry, 'y' or 'n' has to be input, indicating whether the accounts should be printed. The variable identifier is PrintAccounts.

Write a Boolean expression to validate an input for PrintAccounts.

....

....

.... [2]

(ii) Write a Boolean expression to validate StockID, as described in (g).

....

....

.... [2]

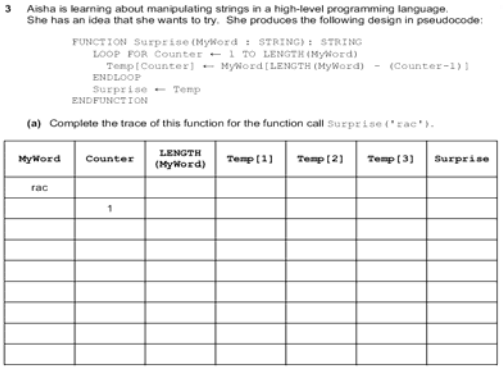
(iii) Describe a possible problem that could occur if the StockID is not validated.

....

....

....

Revision questions

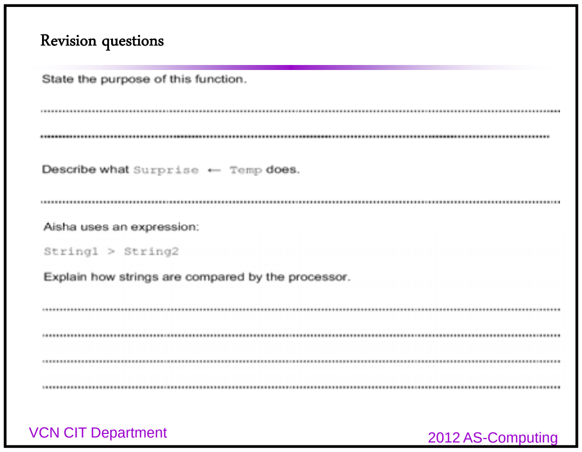


3 Aisha is learning about manipulating strings in a high-level programming language. She has an idea that she wants to try. She produces the following design in pseudocode:

```
FUNCTION Surprise(MyWord : STRING): STRING
   LOOP FOR Counter ← 1 TO LENGTH(MyWord)
     Temp[Counter] ← MyWord[LENGTH(MyWord) - (Counter-1)]
   ENDLOOP
   Surprise ← Temp
ENDFUNCTION
```

(a) Complete the trace of this function for the function call `Surprise('rac')`.

MyWord	Counter	LENGTH (MyWord)	Temp[1]	Temp[2]	Temp[3]	Surprise
rac						
	1					



Revision questions

State the purpose of this function.

....

....

Describe what `Surprise ← Temp` does.

....

Aisha uses an expression:

```
String1 > String2
```

Explain how strings are compared by the processor.

....

....

....

....

Hand out 2 (2.5 Writing Maintainable Programs)

Variable

A variable is a value that can change during the execution of a program.
There are two types of variables used by programmers and they are categorised according to their scope.

Scope

Scope indicates whether a variable can be used by all parts of a program or only within limited sections of the program – for example, within a subroutine.

Global variable

A global variable is one that is declared at the start of the main program and is visible (useable) everywhere in the program and exists for the lifetime of the program.
Note that if one procedure changes the value of a global variable, then the next procedure that uses the variable will be given this changed value – this is a common cause of errors.

Local variable

A local variable is one that is only visible inside the procedure or function in which it is declared. Note that a local variable cannot be referenced from outside the procedure. In fact a local variable does not exist until the procedure starts executing and it disappears when the procedure stops executing. Thus any value that is held by a local variable is only stored temporarily.
The lifetime of a local variable is the lifetime of the procedure in which the local variable is declared.

Constant

A constant is a value that is set when the program initialises and does not change during the program's execution.

Identifier

Identifier is the name given to a variable, constant or subroutine.
Assigning a variable or constant an identifier name means that it can be easily referenced from any part of the program within its scope and it helps to make the program more readable.

Reserved word/keyword

Good program writing techniques

This means that the programmer should write code that is self-documenting and split into small sections.
Specifically, the programmers should:

□ use meaningful identifier names for variables, constants and subroutines;
□ use declared constants;
□ annotate code with comments;
□ indent code within iteration and selection;
□ split the code into modules when possible.

The need for good program-writing techniques

It is important for a programmer to use good programming techniques when writing code so that the code:

□ can be easier to understand by other programmers (who are either part of the programming team, or who will be responsible for the maintenance of the program in the future);

□ can be understandable by the programmer themselves in the future;

□ is split into smaller blocks which are easier to debug individually and update;

□ is less prone to errors – because the meaningful variable names and comments make it selfdocumenting and easier to read.

Meaningful identifier names

Identifiers are used to give names to constants and variables. They are also used to name procedures, functions and the main program.

Naming conventions

Visual basic identifier names must conform to the following rules (other programming languages may have slightly different rules):

□ they must be unique;

□ spaces must not be used;

□ they must begin with a letter of the alphabet;

□ the rest of the identifier must not contain punctuation – it may only consist of a mixture of letters and digits (A–Z, a–z and 0–9) and the underscore character '_';

□ they must not be a 'reserved' word – eg Print, Repeat, For, Dim, Loop, etc.

Examples of good identifier names:
FirstName, Last_Name, PostCode,TelephoneNum, WeightAtBirth, Test_Score, AvgHeight

Using declared constants

Constants will be declared at the start of the main program. The following shows the declaration of constants for Pi and the VAT rate

```
CONST Pi = 3.142
CONST VatRate = 0.175
```

Declaring constants at the start of a program means that maintenance is made easier for two reasons:

□ if the value of a constant change, it only has to be changed in the one part of the program where it has been declared, rather than in each part of the program in which it is used;

□ the code is easier to interpret:

`Total=VatRate*CostPrice` allows a greater understanding than
`Total=0.175*CostPrice`

Annotating code

Comments/remarks

Comments (originally called remarks) are added to program code to provide useful, or essential, documentation for the programmer and other people who read the code.

All programs/subroutines should contain comments to indicate:

- the details of the person who wrote the code;
- the date that the code was written;
- the purpose of the code;
- how parts of the code work;
- the use of certain variables.

Comments can be used anywhere in a program – either as separate lines by themselves or on the same line as executable code provided they appear to the right of any code that will execute.

Comments are usually indicated by the use of an apostrophe (`‘`), or two forward slashes (`//`).

```
PROCEDURE Arrange(NumberOfScores)
‘Procedure to move the largest element in an array to the end
‘Coded by J Jones, 02/03/09

DIM Ptr, Temp As Integer ‘ptr is the value of the array index
FOR Ptr = 1 TO NumberOfScores ‘go through each element
‘test the relative values of the elements
IF Scores(Ptr) < Scores(Ptr + 1) THEN
‘use a temporary store to help swap the elements
Temp = Scores(Ptr)
Scores(Ptr) = Scores(Ptr + 1)
Scores(Ptr + 1) = Temp
END IF
NEXT Ptr ‘increment the index to examine the next element
END PROCEDURE
```

Indenting code

Indentation should be used within iteration and selection statements so that it is clear which instructions go together.

Examples:

Original code

```
INPUT Number
Total=0
WHILE Number > 0 THEN
Left=Number MOD 10
Right = Number DIV 10
Total=Total+Left
Number=Right
END WHILE
```

Indented code

```
INPUT Number
Total=0
     WHILE Number > 0 THEN
     Left=Number MOD 10
     Right = Number DIV 10
     Total=Total+Left
     Number=Right
END WHILE
```

Original code

```
FUNCTION TEST(X)
IF X=1 THEN
PRINT 1
RETURN 1
ELSE
Number=X*Test(X-1)
```

Indented code

```
FUNCTION TEST(X)
  IF X=1 THEN
   PRINT 1
   RETURN 1
  ELSE
   Number=X*Test(X-1)
   PRINT Number
   RETURN Number
  END IF
END TEST
```

```
PRINT Number
RETURN Number
END IF
```