Chapter 1 - The basic components of a computer system

1.1 Introduction

Computers today come in all shapes and sizes. There will be a whole range of 'input devices' and 'output devices' attached to every computer, a whole range of 'peripherals'. Some may come with a lot of 'memory' or a little memory, lots of 'cache' or a little cache and before you know it, it all becomes very confusing! You need to make sense of all of the components that make up a 'computer' to really understand it. You need to make sense of the 'hardware'.

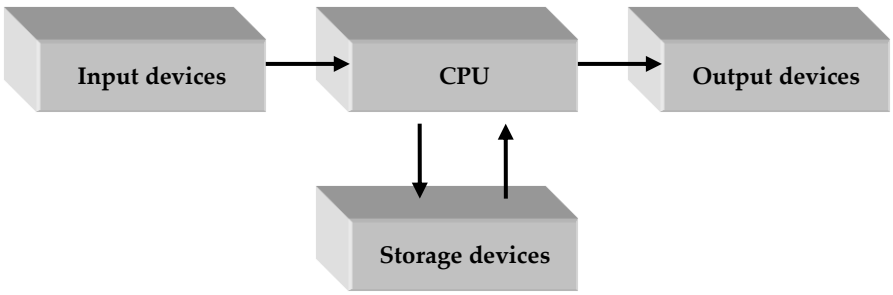
There are some basic definitions that you should get under your belt before you begin. Everyone will always want to argue over the choice of words in definitions! Opinions will vary, depending upon whose book you are using. The safest thing you can do as students is to read widely and then use the definitions as given in the British Computing Society's "A Glossary of Computing Terms". If you do not have this book, then you need to get it! it is an excellent reference guide for all Computing students!

1.2 Hardware definitions

1) **Hardware**. "Hardware is the physical part of a computer system - the processor(s), storage, input and output peripherals".
2) **Software**. "Software consists of programs, routines and procedures (together with their associated documentation) which can be run on a computer system".
3) **Input device**. An input device "is a peripheral unit that can accept data, presented in the appropriate machine-readable form, decode it and transmit it as electrical pulses to the central processing unit".
4) **Storage device**. A storage device is "a peripheral unit that allows the user to store data in an electronic form for a longer period of time and when the computer is switched off. The data can only be read by the computer and is not in human-readable form".
5) **Output device**. An output device is "a peripheral unit that translates signals from the computer into a human-readable form or into a form suitable for re-processing by the computer at a later stage".
6) **Peripheral**. This "is a piece of equipment (or hardware) which can be connected to the central processing unit. They are used to provide input, output and backing storage for the computer system".

1.3 The purposes of input, output and storage devices

All computers have the same four basic building blocks. These are the input devices, output devices, storage devices and the brain of the computer, the Central Processing Unit (CPU).



A diagrammatic representation of the four parts of a computer.

1.3.1 Input devices

Input devices take data from the 'outside world' (i.e. outside of the computer) and send it to the Central Processing Unit for processing. Data can be collected and entered into a computer in a range of ways. These include, for example, a keyboard, a mouse, a graphics tablet, a touch screen, a bar code reader, a magnetic stripe reader and so on. So data from the world outside of the computer is entered into the computer system using 'input devices'.

1.3.2 Output devices

Every computer system will need to have a way of displaying information or printing information out so that it is useful to humans. There needs to be, therefore, some 'output devices'. A very useful output device is a VDU (or Visual Display Unit).

Another handy one that allows you to produce ‘hardcopy’ (i.e. a printout on paper) is a printer. It is possible that you also need an audio output. Speakers or headphones would be useful in these circumstances!

1.3.3 Storage devices

Of course, the computer might not want to display or print out the results of some processing immediately. It might simply want to store the results for another time. It therefore needs some storage devices. Storage devices will store data, even when the power to the computer is switched off. When the power is switched on again, the data can be retrieved. For this reason, storage devices are known as ‘non-volatile’ devices. Examples include USB pen drives, hard disks, CDs, CD R/Ws, DVDs, SD and micro SD cards, Blu-ray disks and magnetic tape.

The hard drive is a very important storage device in your computer that deserves a special mention at this point in time. Not only does it hold all of your files, even when the power is switched off, it also holds your operating system and all of the programs you want to use, too.

1.3.4 The CPU

Once data has arrived inside the computer, it is 'processed' by a special piece of hardware known called the CPU, or Central Processing Unit. This special piece of hardware takes the data fed to it from the input devices and works on it, according to a set of instructions known as a ‘program’. A program is the ‘software’ part of the computer and we will deal with software later. Once the CPU has worked on the data, it has to do something with it.

1.4 Drivers

Hardware components in a computer system will not work unless they have a ‘software driver’. A driver is a program written for a specific piece of hardware that allows the operating system or an application program to use the hardware. Many drivers are installed automatically when you install an operating system, for example, you will get a ‘generic’ mouse driver and a ‘generic’ DVD player driver. Because they are ‘generic’ they will allow most if not all mice and DVD players to be used. However, if you buy a new type of mouse or a new type of DVD player, you will get with it a driver for that specific device, either on a CD or DVD or you will be asked to visit a website and told to download it from there. It is always worth using drivers written and designed for a specific piece of hardware because you will get more functions and better functionality.

1.5 General software definitions

In the same way that we used the British Computing Society's "A Glossary of Computing Terms" for hardware definitions, we will do the same for software definitions, and then elaborate further later. As already mentioned, different authors will put forward different definitions. To be safe, use the BCS Glossary for definitions but you also need to get into the habit of wide reading. You must not just use one or two textbooks exclusively on your course. You will get a very biased and narrow view if you do. Read widely and make sure you know how to use the Internet effectively. Another problem that you will come across all of the time (if you haven't already) is that there are many different words for the same thing in Computing and that authors' descriptions of the same thing can be very different! There's nothing you can do about this except to read widely and make your own mind up! The formal software-related definitions you should know are:

1.5.1 Operating system

An operating system is "a program or suite of programs that controls the entire operation of the computer". You have probably experienced some operating systems already, for example Windows XP, Windows Vista, RISC-OS, Linux or DOS. It has a number of important jobs to do such as making the hardware in a computer work together, reporting any errors it finds in the computer and providing a user interface between the user and the computer, to name but three.

1.5.2 User interface

A user interface is a method "for communication between the user and the computer". Users communicate using languages such as English, Spanish and so on. Computers do not understand these languages. They are digital devices - they understand only ones and zeros. So that communication in both directions can take place, an interface is provided. This effectively converts human-speak into computer-speak and vice versa.

1.5.3 Translator

A translator is "a computer program used to convert a program from one language to another, for example, from a low-level language to machine code". We have said a user interface is needed to allow humans and computers to work with each other because they don’t speak the same language. In much the same way, when a human writes a program, perhaps a computer game or a word processing application, it will not be in the language the computer understands. Once the program has been written, it first needs to be ‘translated’ and then the computer can run it.

1.5.4 Utility program

A utility program is a "systems program designed to perform a commonplace task, for example, the transfer of data from one storage device to another, sorting a set of data, or a disk editor for directly editing the contents of a disk". Operating systems come with ready-made pre-written programs that do very specific jobs a computer user will commonly need to do.

1.5.5 Programming language

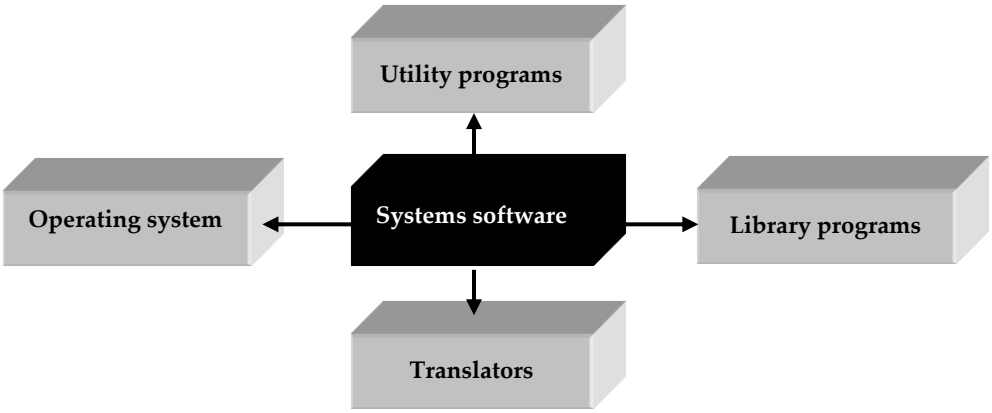
These are pieces of software used to write applications that can be useful to humans in some way. Once written, they are then translated by another piece of software called a 'translator' into the machine code that is understood by the CPU. The CPU then runs, or 'executes' the machine code. Applications such as PowerPoint and Word are written using a programming language.

1.5.6 Generic application

An applications program is "a computer information system designed to carry out a task (such as keeping accounts, editing text) which would need to be carried out even if computers did not exist". A 'generic application' is an application that can be used in many different circumstances. For example, the application Access can be used in many different database scenarios. The application Word can be used to produce many different kinds of word-processed documents.

1.6 What is 'systems software'?

When you use a computer you can use it without understanding all of the clever things that are going on in the background. You can take for granted operations such as saving work to memory stick, the ability to send work to a printer (FILE - PRINT) and typing in words and commands via a keyboard. You don't need to know *how* this happens - it just happens! The software that divorces a user who is actually using a computer from the way the computer actually does what the user asks it to is known as 'systems software'. Systems software is made up of the following four types of software: the operating system, utility programs, library programs and translators.



The four types of software that make up systems software.

1.6.1 The operating system

According to the BCS, an operating system is "a program or suite of programs that controls the entire operation of the computer". Imagine you went out to your computer shop and bought a new computer. When you took it home and switched it on, nothing happened!! It powers up okay but you can't actually do anything with it. The reason for this is that your computer needs an 'operating system' to make it work. You have probably used a computer that has DOS, RISC-OS, Windows or Linux running on them at some time. These are examples of operating systems. An operating system performs a number of very important jobs and we will go into some detail about these in future chapters. Just to give you an idea of *some* of the jobs that are done by operating systems, they: manage all of the components that make up a computer and ensure that they can work together. They provide users with a screen that can accept and display data and information (the user interface). They deal with and report any problems it finds with the computer. They make sure the CPU is working as hard as possible.

1.6.2 Utility programs

According to the BCS, a utility program is a "systems program designed to perform a commonplace task, for example, the transfer of data from one storage device to another, sorting a set of data, or a disk editor for directly editing the contents of a disk". The definition of utility programs was far clearer when operating systems were relatively basic compared to operating systems today. Some 'utilities' can now be considered full-blown applications, such as virus-protection software or compression software such as WinZip. These are best avoided as examples of utilities in exams. Stick to the ones about which there is no (or

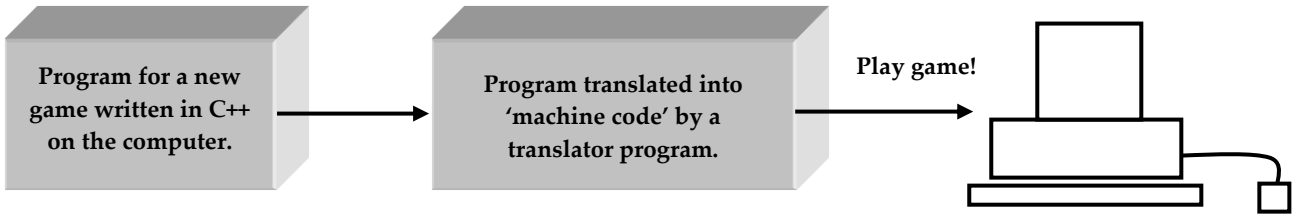
little) debate, such as text editors and formatting disk utilities. Utility programs, then, are programs that are available to the user via the operating system. Each utility program is designed to carry out a specific task, one that users regularly need.

1.6.3 Library programs

A program library is a collection of programs, each of which do a specific job. Library programs are available for users to call up and use in their own programs. On a network, you may have a library of programs on a server that are available to all users. On a PC, a user may have bought and installed a handy set of library programs for whichever programming language they are using or may use the library programs that came with whichever operating system they installed.

1.6.4 Translators

When a user writes a program in a programming language, it needs to be converted into 'machine code'. This is the only language that the computer understands. A program that converts a program written in assembler, Visual Basic, BASIC, Pascal, Fortran, Java and so on, into machine code is known as a 'translator'.



The role of translators.

1.7 What is 'applications software'?

An applications program is a program of instructions that someone has written that does something useful for a user. Examples include a word processor, a spreadsheet and a database application. If the application can be used in a range of circumstances, then it is known as a '**generic**' piece of software. For example, someone might write a spreadsheet program in Pascal for a company called BubbleTop Ltd. This piece of software will have been tailored to meet the company's exact needs using their existing hardware and software. The software couldn't then be given to another company to use because it wouldn't exactly match the needs of that company or their particular hardware and software configuration. It would have functions that aren't required and lack functions that are. On the other hand, both companies, whatever their individual needs, could use a generic piece of software such as Excel. Excel is an example of a generic piece of software.

1.8 What is an 'integrated package'?

Some applications come as an **integrated package**. For example, OpenOffice and Microsoft Office are integrated software packages. There are three features of integrated packages.

1) They contain a number of separate applications such as a spreadsheet, a word processor and a database.
2) Information can easily be exported from one application in the package and imported into another application in the package. You could create a chart showing the rise and fall of profits over the last year in the spreadsheet application, for example, and import it into the word processing application, so that it can be included in the company's annual report to shareholders.
3) There is a common user interface across the applications so that users can quickly learn and use each one.

Q1. What is the difference between hardware and software?
Q2. Define 'input device' and 'output device'.
Q3. Give three examples of a peripheral.
Q4. List as many different storage devices as you can.
Q5. What is the difference between systems software and applications software?
Q6. State two examples of utility programs.
Q7. What two advantages of an integrated software package?
Q8. Define what an operating system is.
Q9. What is the purpose of a software driver?
Q10. What does a translator do?

Chapter 2 - More about systems software

2.1 Introduction

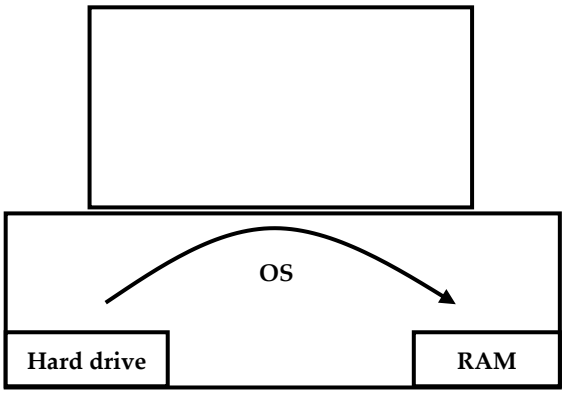
We have already introduced the idea of 'systems software'. We have said that to discuss systems software fully, we need to talk about four distinct areas of software. These are:

1) Operating systems (to run the computer).
2) Utility programs (to provide useful and frequently-used functions for the user).
3) Library functions (to provide pre-written functions that a user can call up and use in their own programs).
4) Translators (so that a user can turn their program into machine code and then run it).

2.2 Operating systems

There are different kinds of operating systems around but here we will simply describe further the purpose of a computer operating system - why every computer needs one! When you switch on a computer, the computer performs some basic checks on itself. These happen in the background and you won't know they are happening unless you are looking carefully or you receive an error message on the screen. For example, one message you might get sometimes is **keyboard not detected**.

Once the computer has completed these basic checks, you will want to use it - but you can't yet! All of the different components that actually make up a 'computer' (the input, output and storage devices along with the CPU) need some software to make them work together and then you also need an interface, so that you the human can 'talk' to the non-human devices and make them do things! So the next job after the basic checks have been carried out is to fetch the operating system from backing storage, usually the hard drive, and 'load' it into its memory, called RAM. RAM is short for Random Access Memory. Any program or file that the CPU wants to work with must be in RAM first.



When the computer is turned on (known as 'booting-up' a computer) it checks itself for problems. Then, before a user can get access to the hardware and use the computer, the operating system must be copied from the hard drive to the RAM.

ALL programs, including the operating system, MUST be copied to RAM before they are used.

The operating system can be thought of as a big program made up of lots of little programs, because it is in fact responsible for many things. A program that is made up of smaller self-contained units can be called 'modular' in design. We will discuss modular programs in more detail later in the book. Once an operating system is in RAM and running, the user can actually use their computer!

2.3 What does an operating system actually do?

The OS is responsible for many things, including providing a user interface, memory management, file management, CPU management, error detection and reporting and providing utility programs.

2.3.1 User interface

One part of the operating system is responsible for providing a user with an interface. A user doesn't need to know *how* the computer does what it does. They only need to be able to click on icons and select items from menus, for example, (assuming the interface is a 'Graphical User Interface') and then let the computer, or rather the operating system, work out what actually needs to be done with the hardware. The screens that are provided by the operating system, the methods by which the user enters data into the computer and what the user sees on the screen are all part of the user interface.

A computer is made up of hardware like a motherboard, a DVD drive and so on. A human needs to interact with them to use them but how? Humans don't understand the 'machine code' that the components use and the components don't speak the language of the human user. If they are to interact, if the user is to be able to give commands to the hardware and understand messages sent back by them, then they need to find a way. This is where the user interface comes in. The user interface is a screen which users can see (usually) and understand and it also includes the input devices like the mouse and keyboard as well as the outboard devices, such as the monitor. As commands are entered, as icons are clicked and menus selected, the user interface converts them into commands the hardware can understand. Messages sent by the hardware are also converted by the user interface and displayed in a form the user can understand, for example, by displaying a message on the screen. In this way, a human user can interact with something that does not speak or understand the language of the user.

2.3.2 Memory management

Users often want to be able to 'multi-task'. They want to be able to use different pieces of software at (apparently) the same time. When they select an application, it is the operating system that goes to the backing storage (usually but not always the hard drive), locates the application and then loads it into the computer's memory in just the same way as it loads the operating system into RAM. Exactly where in memory an application (or a file of data that an application needs to work on) is placed is the responsibility of the operating system. With a number of applications in memory (including the operating system itself) and a number of files of data, it needs to put them in places where they won't interfere with each other. How memory is managed, or 'memory management', is another responsibility of the operating system.

2.3.3 File management

When a user wants to 'store' a file, they first select a storage device, for example, the hard drive or a memory stick. Then they create a new folder or select an existing folder. Finally, they type in an appropriate file name and click on SAVE. The user is telling the computer to save a file. It is the operating system's job to intercept these commands and decide where **physically** on the storage device to actually store the file. The user only tells the computer what file to save and which storage device to use. It is the operating system that decides where exactly on the storage device to put it. The user doesn't need to know where it is saved **physically** on a storage device. They only need to know where it is saved **logically** (using the file name, folder and storage device) so that they can get it back again when they need it. The operating system keeps a record of every file's physical location on each storage device in a special table that is actually set up and maintained on each storage device itself. It can therefore retrieve any file that a user requests and then load it into RAM. File management is another responsibility of the operating system.

2.3.4 Error detection and reporting

An operating system is responsible for managing the components that make up a computer. Every component in the computer will eventually fail and sometimes the software in the computer will run into difficulties. Something needs to be constantly checking that the system is working well. If an error is detected with a piece of hardware or software, it is the job of the operating system to try and fix the problem and to report the problem to the user. Error detection and reporting is another responsibility of the operating system.

2.3.5 Provision of utility programs

The operating system may come with special 'utilities'. These are small programs that carry out specific commonly needed tasks for a user. Examples of these utilities in Windows include DEFRAG and SCANDISK. The operating system, then, provides handy utilities.

2.3.6 CPU management

Personal computers have only one CPU. However, you can (apparently) do many different things at the same time on a computer. For example, you could be using a chat room, printing a document and listening to some MP3 music, all apparently at the same time. In fact, they are not being done at the same time - it just seems that way. The CPU does one job for a **slice of time** and then it switches to the next job for a **slice of time** and then the next one and finally the CPU goes back to the first one again! It is switching between jobs so quickly that you think it is doing all of them at the same time. This is known as 'multi-tasking' and we will discuss this term in a lot more detail later in the book. It is the job of the operating system to allocate the **slices of time** that the CPU will give to each job so that it works as hard as possible and does as much as possible.

2.4 Utilities

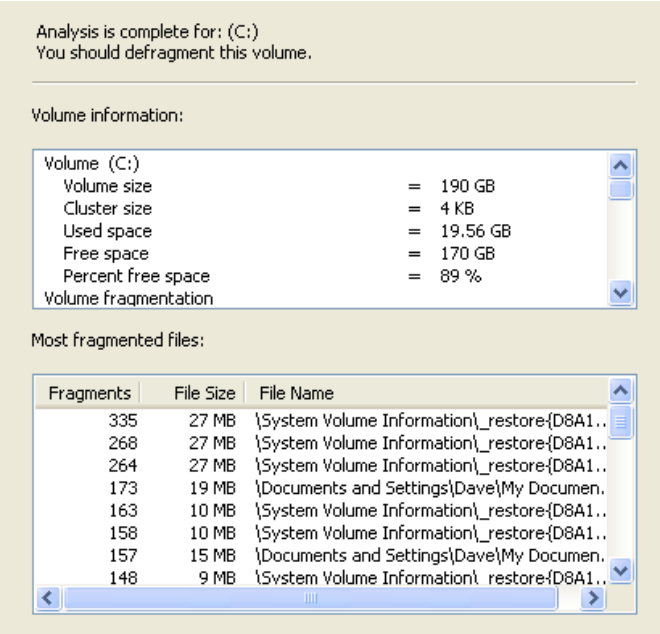
Another important part of systems software is the utility program. We have already said that, according to the British Computing Society, a utility program is a "systems program designed to perform a commonplace task". They provide handy extra functions for your computer. We will now describe in detail some examples of utilities.

2.4.1 Formatting a disk

A handy little utility that comes with most systems software is a 'format disk' utility. This allows a user to buy a standard storage device e.g. a USB pen drive and then set it up for their particular machine's operating system. When you 'format a disk' the utility checks that the disk can be written to and then divides up the disk into areas, giving each area an address. It then sets up a table that will keep track of what is stored in each area as well as areas that, for example, have become corrupted and cannot be used. You cannot use a storage device until it has been formatted although you can buy pre-formatted ones.

2.4.2 Disk maintenance

It is possible that parts of a hard disk, for example, become unusable. A utility is provided in operating systems to check to see if there are any problems areas on a disk and to try and fix them. In Windows, you may have used the Scandisk utility to do this. We have said that files are stored in areas on a disk. Sometimes, an area isn't big enough to hold the whole file so a number of areas have to be used. If lots of areas are used to hold one file and those areas are scattered all over the disk, then retrieving a file can be slow. You should run the defragmentation utility if you have one (Disk Defragmenter in Windows) every few weeks, to ensure that files are stored physically as close as possible on your disk. If your computer has become sluggish, one reason might well be that your hard disk has become fragmented - so defragment it! This is part of the defrag report that Windows generates.



A defragmentation report.

2.4.3 System use

It is often possible for a user to get some statistics from the operating system about how much RAM the system has got, how much hard disk space is available for use, what software drivers are installed and so on. This is possible because a 'system use' utility is available that can examine a computer system and display what it finds. See what 'System use' information you can find on your home PC. Use the Internet to find out what to do for your particular operating system.

2.4.4 Screen savers

Screen savers are moving pictures that are displayed on your monitor after a set period of time. The original purpose of them was to prevent a 'ghost' image of whatever was on a screen being burnt permanently into the screen of the VDU, if the VDU happened to be left on for a long time. This was true with the early monitors. Modern Visual Display Units are much less susceptible to this problem. However, screen savers nowadays have an extra feature. They can be password-protected. If you leave your computer alone for a period of time, a password-protected screen saver can be set to come on to prevent prying eyes from accessing your computer. Although it is not a robust form of defence, it can stop casual access.

2.4.5 File handlers

When you are using an application such as a word processor to write an essay, there comes a time when you need to save it onto a storage device. This requires a small program to do just that. If you then needed to get the essay back later so that you could carry on working on it, then you would need another small program to do that task. If you then wanted to print it out, or you wanted to delete it, you would need two more small programs. These small programs, which each performs a single, specific task, are together known as 'file handlers'. File handlers are another type of utility you can use.

2.4.6 Compression software

Sometimes, you need to compress one or more files or indeed a whole folder of files. This is sometimes known as zipping up a file or folder. File compression means that you take the file or folder and squash it. This makes it much smaller than the originals. Compressed files and folders take less time to transmit across the Internet than larger files. In addition, you can also store more on a storage device if you compress the files and folders you want to store first. Compression software is an example of a utility program.

2.4.7 Drivers

When you buy a new peripheral, you often get a 'driver' with it on a CD or you can download the latest one from the Internet. A driver is a piece of software that allows communication between a peripheral and the operating system on the computer it is connected to. They can therefore pass instructions to each other and each can obey any instructions received from the other. Sometimes, a peripheral uses a standard driver that comes with an operating system. Whilst this is fine for basic functions, any advanced functions that the peripheral may be able to carry out will not happen unless the software driver designed for that peripheral is installed and used. Drivers are another example of utility programs.

2.4.8 Virus checkers

These are another example of utility programs. They are designed to scan your computer for viruses and to intercept any that try to attack your computer, perhaps via a storage device or email. When a virus is detected, you are then given options to deal with it, such as quarantining it so it cannot harm your computer, deleting it if possible or reporting it, for example.

2.5 Library programs

Library programs are available on most modern operating systems such as Windows and Linux. They are programs that have been written by experienced programmers. The programs carry out specific jobs that may be frequently needed by the developers of new applications. For example, a user may be developing a new application that needs to be able to:

- write data to a storage device
- transfer data from one storage device to another
- open a Window with a menu on it
- send a file to a printer
- use a scanner.

Because these actions are so common, a library of programs are written and then made available to users. They may come as part of an operating system or they may be bought, loaded onto a computer or network and then be made available to users. All a developer has to do is 'CALL' a particular program when they need it from within their own program. Using library routines is a great idea! The code for any particular function in a library has already been written and therefore time and money doesn't have to be spent 'reinventing the wheel' by writing the code again.

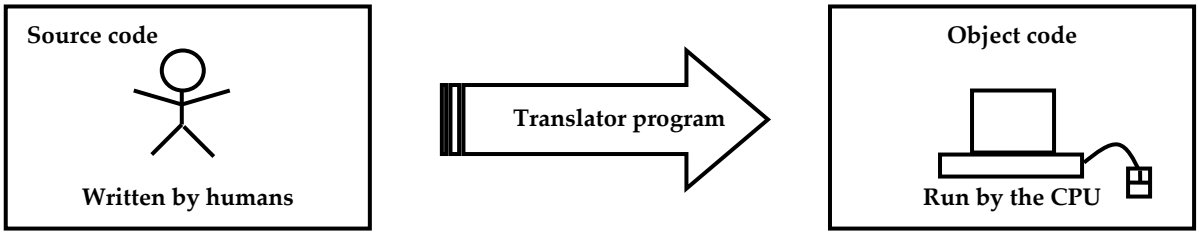
The code in a library program will have been written by experienced programmers and will have undergone thorough testing. It is therefore likely that the library program will not contain any programming errors, known as 'bugs'. It can be relied on to work. Because the library programs are called up only when they are needed and already present on the computer, a new application being developed will be *smaller* in size than if it had to have its own code to do particular functions. This means that it needs *less storage space* on the hard drive and *less RAM* when it is being run.

2.6 Translators

Translators are also part of systems software! A computer, or more specifically, a Central Processing Unit (CPU), ***only*** understands instructions called 'machine code'. Machine code is made up of a set of patterns of ones and zeros. Each pattern makes the CPU behave in a certain way; the pattern makes the CPU do things! The problem with this is that programmers and developers of new applications do not write in machine code - who wants to write programs that involve writing patterns of ones and zeros all day? Programmers use a 'programming language'. You will learn in later chapters that there are hundreds of different types of programming languages, each with their own special programming words (known as 'keywords'). When a program has been written in any programming language, it must then be converted into the equivalent patterns of ones and zeros so the CPU can understand the instructions in the program. It must be converted, or *translated*, into machine code.

When a programmer writes a program in a programming language, it is also referred to as '**source code**'. When it is in a form that a CPU can run (the machine code), it is known as '**object code**'. Getting code from being source code to object code is known as 'translation'.

We give a translator program some source code and it gives us back some object code, like this:



Source code is translated into object code.

2.7 Microprocessors

You may be familiar with operating systems like Windows and Linux but you shouldn't forget that all digital devices require an operating system in some form or other. Just because it doesn't have a screen and a keyboard, doesn't mean that a digital device hasn't got an operating system. It must have one, to control the program, to manage any input and output devices to deal with errors and so on. You probably have hundreds around your house!

• A digital watch. • Your calculator. • A computer-controlled microwave oven. • Your mobile phone has an operating system. • A computer controlled burglar alarm. • Your HD TV.	• Your DVD player. • Your MP3 player. • The SatNav in the car. • A digital radio. • A games console. • A digital answering machine.

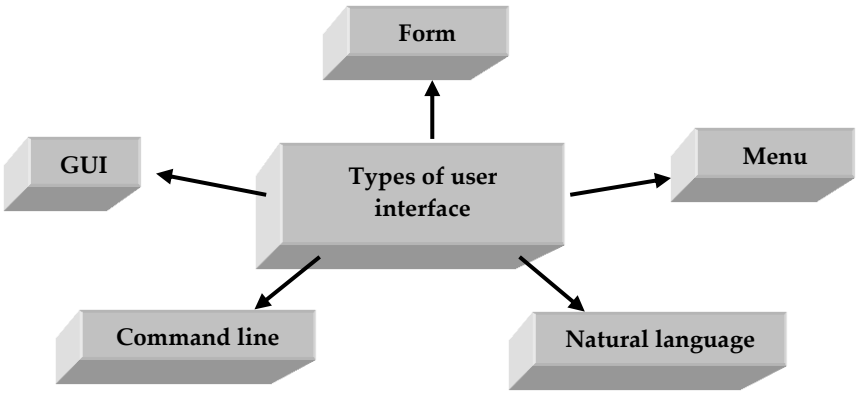
The list goes on. There are many different operating systems around. Some are generic and some are purpose-built. You should see how many different operating systems you can identify using the Internet. Have a look at some websites that introduce you to microprocessors. See what you can find out about the jobs they do.

Q1. State the jobs an operating system does.
Q2. Where are the files and programs you are not currently using usually stored in a computer?
Q3. Where are the files and programs you are currently using usually stored in a computer?
Q4. What does RAM stand for?
Q5. Describe two utility programs.
Q6. Why do computers need a user interface?
Q7. Why do programs have to be translated?
Q8. Define 'microprocessor'.
Q9. Do some research on the Internet. What is meant by an 'embedded microprocessor'?
Q10. Give two examples of common pieces of equipment that have an embedded microprocessor.

Chapter 3 - Different types of user interface

3.1 Introduction

Communication between a user and a computer is two-way. We know that one of the jobs of the operating system is to provide a 'user interface', so that a human can communicate with the hardware that makes up a computer. When you buy a piece of software, it too will have a user interface, so that you can access and use the software. A user will give data and instructions to a computer and a computer will give information back to a user. The way that a computer and a user communicate is known as the 'interface'. There are alternative terms to describe this. Another common term is the Human-Computer Interface, also known as the HCI. If you are going to describe the interface fully, you need to talk about the **input devices**, the **software interface** and the **output devices**. In this chapter, we will concentrate on the nature of the software interface. We will describe the five different types of software interface and identify their characteristics.



Five types of user interface.

3.2 Forms

The first kind of software interface we will look at is the form-based interface. Just for a moment consider a paper-based form that you are asked to fill in, perhaps for the membership of a club or an application for a driving licence. What you have to write down is highly directed. There are instructions to help you, boxes where you write or select information from some choices and boxes where you simply tick one of a selection. A form-based software interface on a computer is similar to a paper-based 'interface'. The input into the computer is predictable. If you used a range of form-based interfaces, you would start to see a number of common characteristics.

1) There are field names, names next to a place where information must be entered. The places where information should be entered in by the operator are known as 'response fields'.
2) Other types of response fields include radio buttons and drop-down selectors.
3) The cursor 'tabs' automatically from one response field to the next. This guides the user logically through the form, ensuring that all the information needed is gathered.
4) As data is entered, it is 'validated'. Validation attempts to ensure that only ***sensible*** data is entered into the system. Validation helps ensure that data entered into any system maintains its 'consistency'. This means that any data stored is only of the format expected in a particular field. Data can be validated using a range of methods. (These are discussed in more detail later in this chapter). The methods include:

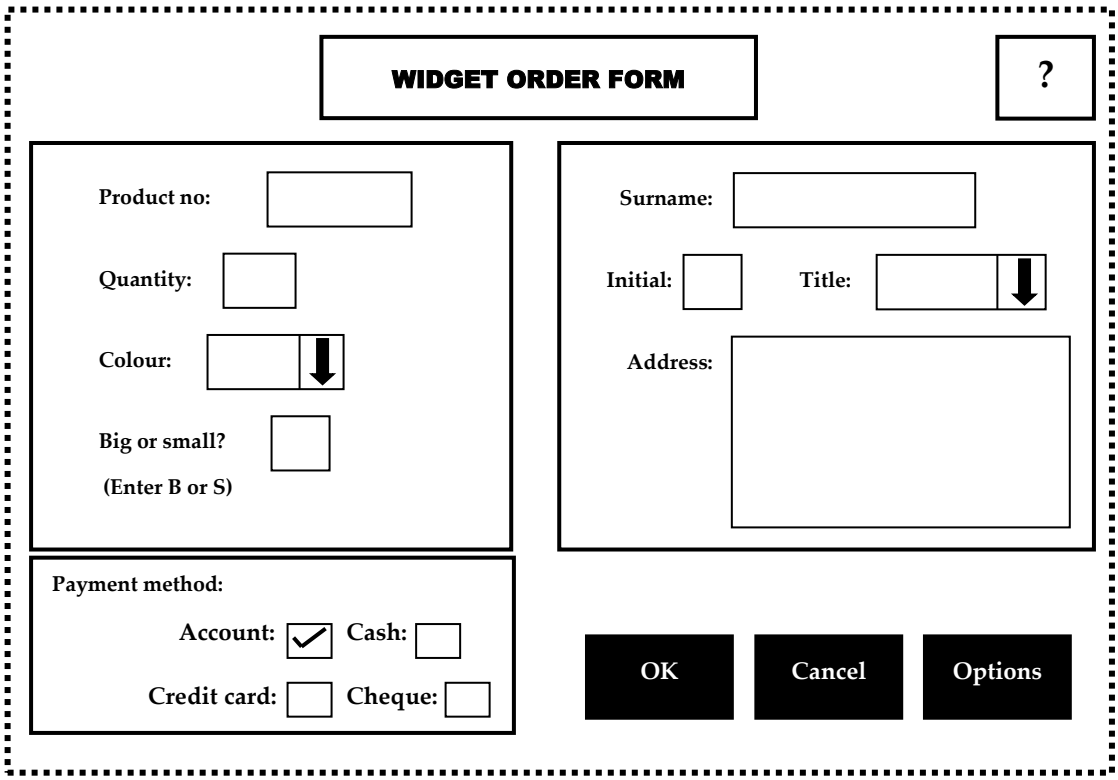
    i. A range check.
    ii. A character length check.
    iii. A data input mask.
    iv. A presence check.
    v. Getting the user to select from a list using combo boxes or look-up tables.
    vi. Using check digits.

5) Input can be changed or cancelled if necessary.
6) Data is finally entered into the system only when an 'OK' button, ENTER or something similar is pressed.
7) There is some kind of HELP facility.
8) Some options are not displayed on the main screen, to avoid cluttering up the form. Access to less commonly needed facilities is via a selection button that links to a separate screen.

Form-based interfaces are suitable for any application that involves entering predictable pieces of information into the computer:

- Someone taking telephone orders for a product such as a CD.
- Someone recording responses to questions in a telephone questionnaire.
- Someone entering in details of people who want to apply for a credit card.
- Someone applying to join a club or open a free email account on the Internet.
- Someone who is buying something online.

All of these activities might be done with the aid of a form-based interface. This is because the same, ***predictable*** information will be asked for by the operator or by the web-based organisation over and over again for each order or questionnaire or application. Here is an example of a form-based interface:



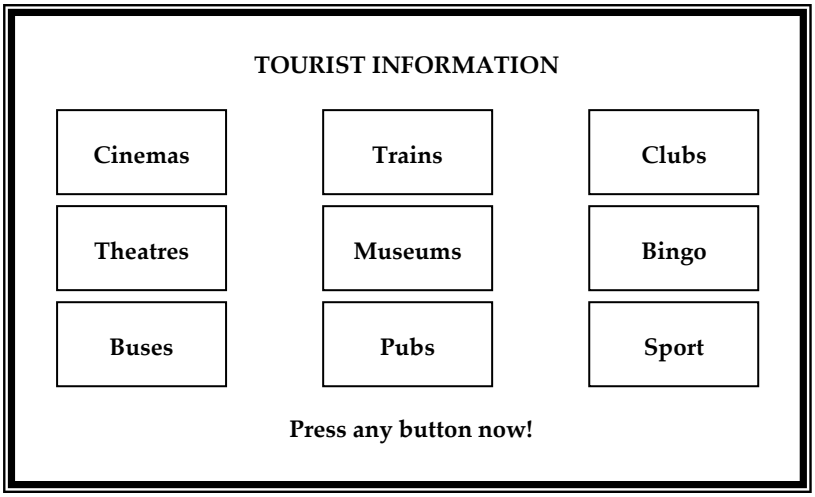
An example of a form-based interface.

3.3 Menus

Menu-based systems are ideal for situations where the user's IT skills cannot be guaranteed or in situations which require selections to be made from a very wide range of options or in situations which require very fast selection.

The user of a system that uses a menu-based interface will be presented with a limited number of options on the screen. Once a selection has been made, the user is presented with a sub-menu. This gives them further options. They make another selection and may be presented with a further sub-menu. This continues until the user is able to select exactly what they want from the choices finally displayed on the screen. Here is an example of a menu-based screen that might be found at a tourist office.

A tourist, who may not have any IT skills, could be presented with a screen with 9 buttons on it, perhaps including theatres, cinemas, pubs and trains, for example. They would touch the touch screen in the area of one of the buttons to make a selection. If they selected 'Cinemas', for example, they would then be presented with a sub-menu. This might look like another menu-based screen with six buttons on it, for example, one for each cinema in the area. If they then selected one of those, they would be presented with the films that are currently showing and the times they are on. This type of user interface is about as simple as you can get. You do not need any computer skills to access the wealth of information on a system like this.



A menu-based information system.

Consider a factory where workers are working in a noisy, dirty environment. Workers may not want to be fiddling around with keyboards, typing in commands. They could have a menu-based interface instead. This would quickly allow them to find the option they wanted and to select it, simply by touching a touch screen.

Consider a stockbroker. Their job may involve sitting at a computer screen and watching how many different share prices are changing. Shares may be grouped and they can select a particular group or an individual share quickly by selecting a type of share from a menu. They might then select a subset of that type from a sub-menu and so on, until they get the group of shares or the individual share they want to monitor. It is far quicker to find what they want using menus and sub-menus than it is typing in commands.

3.4 Graphical User Interfaces (GUI)

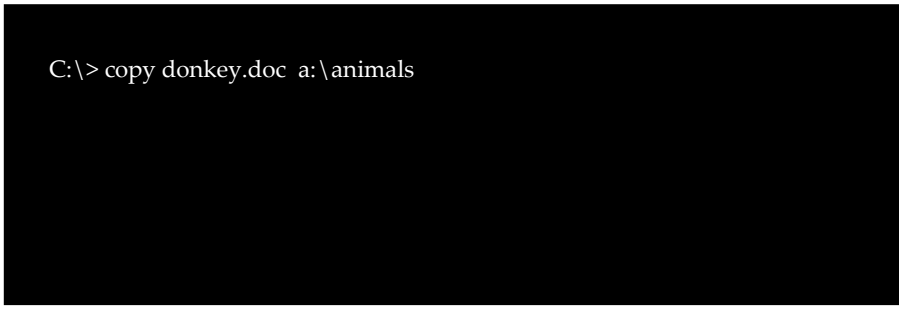
Interfaces that are graphical in nature are known either as Graphical User Interfaces (GUI) or WIMP interfaces (Windows, Icons, Menus and Pointer).

Typically, you would expect these types of interfaces to be available in multi-tasking environments (where you open and use more than one piece of software at a time) or in applications software that involve a considerable degree of complexity. You will all have used a GUI hundreds of times, when you used Windows, or Word, or a Star Office application, or Paint in primary school, or Explorer or Netscape to surf the web and so on. Each of these applications has its own ‘window’ that it opens up into, and you can open up more than one application (and therefore more than one window) at a time. Only one application is ‘active’ at any one time. In Windows, you know which one is active because the active window has a bright blue bar at the top of the window, as opposed to a dimmed blue bar. There are also icons you can click on for fast access to the tools in the application. There are drop down menus that ensure you don’t have hundreds of options constantly on display, taking up room on the screen. The pointer is usually a mouse. A mouse ensures that you can make selections quickly rather than having to use a keyboard, which is slower and prone to mistakes. To summarise, you would typically expect to find the following in a GUI or WIMP user interface:

- A 'window' for each open application. Many windows can be open at the same time but only one window can be active at any one time. There may be some way of indicating which one is active (perhaps by making the bar at the top of the active window bright blue).
- Menus and icons. Available functions can be selected in one of two ways, either by using pop-up menus or drop-down menus, or by clicking on 'icons'. An icon is simply a small picture that represents a specific function - clicking on it selects that function.
- A pointing device, to make selections. It is typically a mouse or a graphics tablet and pen. The use of a keyboard to navigate through the application is minimised because it is a relatively time-consuming way of working.

Companies who make different applications usually try to keep a common 'feel' to the interface in each application. This helps users who are familiar with one application to quickly pick up a new application designed by the same company. You probably have had some experience of this yourselves. You know how to save a file in a new application because it is done in the same way as in another application you used by the same manufacturer.

3.5 Command line interface



An example of a command used in a command line interface.

A command line interface requires a user to type in commands from a list of allowable commands. Suppose you want to backup a file called donkey.doc to a folder (directory) called animals on your pen drive. In a GUI, you would open your file manager, click on the file you want to save and drag it to the folder called animals on the pen drive. Anyone can do that! If you wanted to do the equivalent in DOS, for example, which has a command line interface, you would have to know how to construct the command to copy a file from one place to another. You would have to type: **C:\> copy donkey.doc a:\animals**

This type of interface can take a long time to learn and is not **intuitive**. For inexperienced users it can be a frustrating type of interface whilst for experienced users it can be very powerful. This is because command line interfaces provide commands that can get a user very close to the workings of the components of a computer system. There are commands that can manipulate the hardware and software in a computer system in a way that simply cannot be done using a GUI. Indeed, there are tasks where you have to use a command line interface to carry them out. UNIX and DOS are good examples of operating systems that use this kind of interface.

Typical users of command line interfaces are technicians and network managers. They need to perform many set-up tasks and system tasks. These tasks can only be done using this type of interface.

3.6 Natural language

This kind of interface requires the user to enter responses to questions asked by the computer. The questions are displayed on the VDU and the answers are entered via the keyboard. This kind of interface is called a 'natural language' interface because the computer and the user appear to be holding a conversation. For example, imagine the user has initiated a 'save file' request. The 'conversation' might go like this:

USER: Save file
COMP: What is the file name?
USER: chapter1.txt
COMP: What folder?
USER: UserGuide
COMP: File already exists. Overwrite?
USER: Yes
COMP: Done.

This kind of interface can be found on data entry terminals and other types of ‘dumb terminals’ connected to a network where non-expert users are guided by the computer through the complex tasks they need to perform.

3.7 Designing the user interface

We have seen that the way a user interacts with a computer is known as the 'user interface'. It is also commonly called the 'Human-Computer Interface', or HCI. Communication is two-way. The user must get information into the computer using appropriate methods and devices and the computer must send back data and information to the user. If you want to discuss fully the HCI, you need to discuss:

- The hardware needed to get information into the computer.
- The hardware needed to get information out of the computer.
- The methods provided by the software to aid data input.

We have already seen that there are a range of software interfaces that could be used; forms, menus, GUIs, command line and natural language. When an interface (the input and output hardware and the software methods) is being designed for a new situation, the designer must run through a checklist of things to consider. If this is not done properly, the interface may be cumbersome to use. It may not allow the user to get the right information into the computer easily, efficiently and effectively. If the wrong information is put into the computer, any processing will be invalid because it will be based on inappropriate data. A poorly designed interface may output confusing data and this will diminish a user's understanding of it. They may not be able to make sense of it and so it will be of limited value. What sort of things should be on the designer's checklist? The following list is not exhaustive but does attempt to give you some ideas of the questions that need to be asked.

3.7.1 Who will use the interface?

If a user has a disability, that will affect the selection of input devices needed, the style of the software interface as well as the nature of the output devices. Someone with restricted mobility may need a concept keyboard or a natural language input method. Someone with a visual impairment may appreciate audio output. Some consideration may need to be given to the screen. You may want to consider decreasing the resolution and increasing the size of the icons and menus, using a high-contrast set of colours and making a magnifier available. The age of the users is important. Clearly, the interface for a three year old will be different to an adult. For one thing, the child may not be able to read any but the most basic words and will therefore not be in a position to make selections based on written instructions. A QWERTY keyboard will be of limited use while a concept keyboard may be a much better choice.

The experience of a user will also be important. Many people have grown up with computers and will be comfortable with a GUI. Others with a little more experience may be happy to use a command line interface. Some users may have had little opportunity on computers and need a very basic menu system and touch screen, just to persuade them to even use the system!

3.7.2 What does the new system seek to do?

If it is trying to simulate flying a plane, then a joystick and pedals should be provided as input devices. If it is simulating driving a car then a steering wheel would be an appropriate input device. A process control system may need pushbuttons, keypads and selector switches so that a system can be told what situation to aim for and how to achieve it. If the system is seeking to entertain or be as realistic as possible then special effects in the shape of a vibrating steering wheel or sound effects may help, for example. If the system were a real-time simulation then a realistic visual depiction of the world would be appropriate on a suitable screen. If the application were to monitor and control the flow of trains in and out of a train station, for example, then a diagrammatic depiction of the current situation, regularly refreshed, would be an appropriate method of output, perhaps using indicator lights. If the application were to monitor and control the flow of water around some pipes, there would be a need for sensors to continuously monitor the flow and pressure of water and feed these back to the computer. There would also be a need for actuators (any computer-controlled devices that cause movement) to adjust valves and switch pumps on and off, thereby adjusting water flows. Again, a diagrammatic depiction on a large VDU or other large display unit would be necessary to show the operators the current situation.

3.7.3 What is the environment in which the system will be used like?

If it is a dirty, dusty place, a membrane keyboard or a touch screen may be necessary to prevent the keyboard quickly becoming unusable. If any systems are critical, for example, in nuclear reactors, then the use of klaxons may be necessary to signal the urgency of a situation to users. A hospital, which has patients on life support, may use a similar system or flashing indicator lights to alert nurses to a problem with a patient. If the system is to be used in an office to enter in information from paper forms then a QWERTY keyboard or natural language may be fine, in conjunction with forms-based input screens. Simple laser-jet or ink-jet printers may suffice for most office-type applications. However, if large, accurate engineering drawings are needed using colour, then a plotter may be selected as the output device. If the system was to be used under water then any input devices such as sensors should be waterproof and corrosion-resistant. If it was to be used in the open air, similar consideration should be given to any environmental factors that may affect system components.

3.7.4 Where will the information sent out from the system be used?

If the information is to be used at the computer system, it may only be necessary to provide a 'standard' VDU. Of course, engineering design applications may warrant the purchase of a 21-inch screen, for example. If the information is to be sent to someone or hardcopy is required for legal reasons then a printer will have to be selected. In a warehouse environment, a printer may be needed so that an order can be printed off and taken by the warehouseman around the warehouse while they collect together the order.

3.8 Designing the software interface

There are many common elements to the design of good interfaces, whether they are for data input forms, navigation interfaces, an interface for an application, web sites and web forms or dialogue boxes. In this section, we will document the features of these forms with a view to designing our own.

3.9 Dialogue boxes

A dialogue box is any window that appears because you have to make selections from choices. An application or utility wants to know how to proceed with an action and needs you to give it some details. It pops up a dialogue box for you to fill in. These are usually designed to be very quick and easy to fill in. For example, you might want to print out some work. You go FILE then PRINT and a dialogue box like this one might appear:

3.10 Possible features on any user interface

How many features have you got on your list, as a result of doing tasks 28, 29 and 30 above? Which ones did you miss out?

- Buttons to cancel the whole form, enter data and continue, get further help and get further options. Only the common options are shown. The rest are put on their own screen and linked from the main screen.
- A title for the main form and titles for each piece of information that has to be entered.
- Indications about which fields must be filled in and which ones are optional. This can be done by putting an asterisk next to compulsory fields, with a note near the fields saying that they are compulsory.
- General user instructions for the form and user instructions for certain pieces of data. E.g. Under an input requiring a date of birth, you might see DD/MM/YYYY showing you the format expected for the date.
- Examples for some data so the user knows what form to enter the data in e.g. 21/08/1986.
- A box next to the name of the data to be entered so that the user knows where to type.
- A scroll box can be used to display a block of writing. This is a box on the screen with a scroll bar. For example, if you were ordering something online, there might be a scroll box on the screen showing you the terms and conditions. This ensures that much of the screen is not taken up with huge amounts of text.
- Tick boxes. For example, a tick box next to "Tick if you need a VAT receipt".
- Radio buttons. These allow a user to see all the options available and to select one or more of them. Seeing all the options at once takes space on the screen so they are only suitable when there are a limited number.
- Combo boxes (drop down lists). These are suitable when there are a large number of options to choose from but you don't want them all on the screen at once because they would take up a lot of space. Tick boxes, radio buttons and combo boxes are faster ways of entering data than e.g. keyboards. They also remove the

possibility of incorrect data entry, for example, through spelling mistakes or entering an option that isn't allowed. They are a method of data validation. (Validation is making sure you get sensible data into a computer as opposed to verification, which is checking that the sensible data you have entered is actually the data you want, perhaps by double-checking it).

- Wherever possible, validation rules are used on data entry fields. These might include the use of data entry masks and range checks, for example.
- Data entry fields are logically grouped together. All the name and address fields are together, while the options in the 'Select your Interests' are grouped together, for example. Logical groupings are often reinforced by drawing boxes around groups or by selection of colours and shading to 'raise' or 'sink' an area from the rest of the form
- The use of pictures to reinforce a selection. For example, you could have used radio buttons to allow a user to select how to be contacted. If they select 'By telephone', a little icon of a telephone appears on the screen.
- Clear, readable fonts.
- Good selection of colours that are easy on the eye.
- Colour can be used to draw a user to a particular area of the screen. For example, you could put an 'Are you sure you want to proceed?' message in red.
- Tab order. You can't show this on a screen design but the order that you jump from one field to the next is important. This allows the designer to set an efficient and logical path for the user through the form.

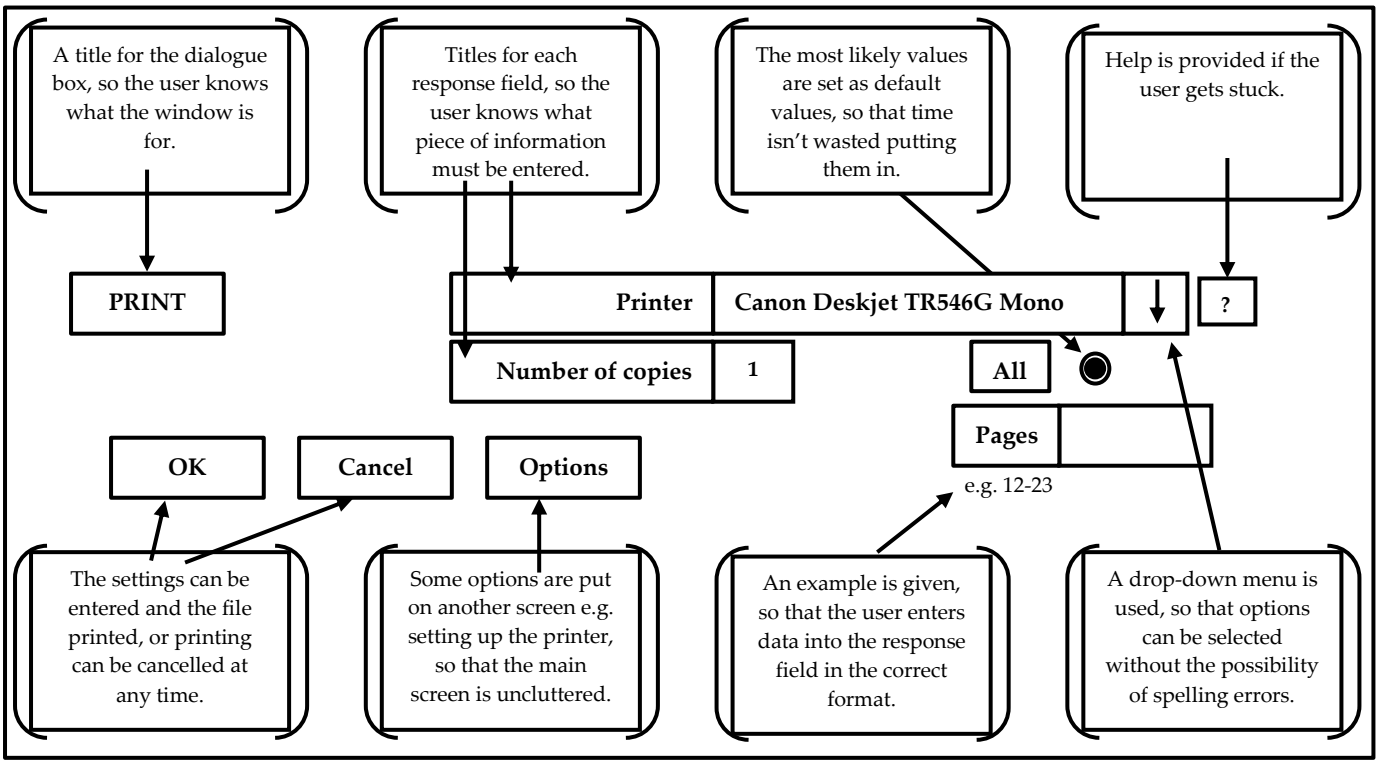
3.10.1 Reasons why good design is important

Good interface design is important for a number of reasons.

- Users need to be able to navigate quickly and easily around different screens.
- Users need to be able to get help when they get stuck.
- Users need to be able to enter data quickly and correctly.
- Users need to be able to change their minds and cancel what they are doing.

3.10.2 Justifying a form design

Whenever you are asked to design a form, you need to be able to justify why you have designed every part of it in the way you have. One method of doing this on paper and in an exam and for your coursework is to sketch out the design and then annotate it. You should circle features on the form, draw a line to the side of the form where there is some space to write and then say why you have done what you did! One key phrase to use in your annotated comments is 'so that'. Here's an example.



An example of annotating a form.

3.11 Data validation

Validation is the term used to ensure that only 'sensible' data is entered into a database. These rules are set up, for example, on a database on a computer and the computer then automatically checks the data entered against these rules. Sensible data means that it is of the correct data type and it follows the rules set up for that piece of data. (Don't use the phrase 'correct data' - some exam boards don't like it!)

There are a number of different ways of setting up validation rules. We will look at these in turn.

3.11.1 Range checks

You can set up rules to ensure that if someone enters a number it must follow some mathematical rules, for example, be greater than a number, or less than or equal to a number or be between two numbers and so on. We have already seen an example of setting up rules for an AGE field. If a value is entered outside of this range, then a helpful error message should be displayed. These are the maths symbols used in range checks:

<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
<>	Not equal to

If you have a problem remembering which maths symbol is which for less than and greater than, try remembering that the ***L*** in ***Less*** looks very similar and points in the same direction as the maths symbol **<**.

3.11.2 Format check

Sometimes, especially with membership and ID identifying codes, the code is made up of a mixture of numbers and letters. For example, a national chess club may have a membership ID that is made up of 2 letters, followed by 3 numbers, followed by a letter, for example, RG662P. You can set up rules that look for patterns such as:

LETTER-LETTER-NUMBER-NUMBER-NUMBER-LETTER.

If TYP23F is entered into a data input form, it will not get put in the database. It will be rejected and a helpful error message should pop up! This kind of check is known as a 'format check'.

3.11.3 Length check

In text fields, you can tell a database to accept an entry only if it is less than a certain number of characters. For example, you might set a field up called COLOUR to be less than or equal to 10 characters. If someone entered BLUE WITH A HINT OF MANGO, it would get rejected because it doesn't follow the validation rule.

3.11.4 Allowable values check

You can set up fields to only accept certain values. For example, you might have the Boolean field GENDER set up to only accept MALE or FEMALE. You might set up a text field called WHEN DO YOU WANT TO LEAVE? so that it can only accept MORNING, NOON or EVENING.

3.11.5 Check digit

Commonly used on bar codes, this is a technique used to see if a number has been entered correctly. A number, called a check digit, is placed at the end of a code. The bar code is read and the check digit calculated. If the result is the same as the check digit, then the code has been read correctly. If they are different, then the item will have to be read in again.

There are different methods of working out check digits. For example, the 13 digit ISBN code for this book is 978-0-954351472. The check digit is the number on the right hand side, 2. A barcode scanner would read the first 12 digits of the ISBN code and then see if it got a 2. If it did get a 2, it would know that the ISBN code had probably been scanned correctly. How does it work out the check digit?

First, it multiplies each digit in turn by a 1, then a 3, then a 1, then a 3 and so on, and adds up the result, like this:

(9 x 1) + (7 x 3) + (8 x 1) + (0 x 3) + (9 x 1) + (5 x 3) + (4 x 1) + (3 x 3) + (5 x 1) + (1 x 3) + (4 x 1) + (7 x 3) = 138

Next, it divides the 138 by 10 to get 13 remainder 8.

Finally, it subtracts the reminder from 10, so 10 – 8 = 2. (This system uses something called 'modulo 10', which just means the remainder after you divide a number by 10).

As you can see, the check digit is 2, which is also the number on the right hand side of the 13 digit ISBN code. Do note that 10 digit ISBN numbers use a different system to the one described here.

3.11.6 Presence or 'Required' check

Some fields are more important than others and must be filled in. For example, suppose you had a database that stored car adverts. A customer rings up to place an advert and the operator types the advert into the database using a data input form. If the customer doesn't know the colour of the car, it is not a disaster and so it doesn't need to be filled in. If the customer has forgotten his or her telephone number, however, this is a problem - how can anyone contact him to buy the car? This field cannot be left blank. If the operator presses ENTER to enter the whole advert and this field has been missed out, then a helpful error message should appear. This kind of rule is known as a 'presence check'.

3.12 Data verification

Once sensible data has been entered, verification techniques should be employed. Verification is the process of checking that the sensible data entered is actually the data you want! For example, suppose you have a field NAME. You enter JONNS. This is valid data. But when you double check it, you realise that it should be JONES not JONNS. You can then correct it. This is verification. There are a number of verification methods in common use.

3.12.1 Re-type a value

This is commonly used when you set up or change a password. You type in your password once, and then you type it in a second time. If they both match, then your password is accepted. If they don't match, even though both entries are valid, it is rejected and you are asked to try again.

3.12.2 Re-enter whole data files

This is used in data processing environments. Operator A types in a set of paper-based orders, for example. They are saved in a file, but not entered into the database yet. When operator A has finished typing in the orders, the paper-based orders are passed to operator B. He re-types them all in. As operator B types in an order, it is checked by the software to see if what he has typed in is the same as what is held on file, what operator A typed in. If they are the same, then the order is entered into the database. If they are not the same, then an error is flagged up and the order needs to be carefully checked. Flagging up can be done in a number of ways. The keyboard could 'beep'. An error message could be displayed, or all of the errors saved to file and an error report printed out at the end of data entry. Although this may seem a little long-winded, it is an important method of verification for critical data, such as for entering in data from passport applications.

3.12.3 Reading back

When you book a holiday at a travel agent, you give them all of your details and they type them into their computer. They can verify that your details are correct simply by reading them back to you and asking you if they're correct! Sometimes, organisations need you to confirm some details they have given you. In this case, they may print out what you told them on a form and then ask you to confirm the details, sign and return it. If you have ever ordered anything over the phone, then you will know that after you have placed your order, the operator will read it back and ask you to confirm that the order is correct.

3.12.4 Tick the box and click to proceed

If you buy something on the Internet, you will typically place an order by typing it into a form. The order will then be prepared, possibly rearranged and then displayed back to you on the screen. You will be asked to check it carefully, tick a CONFIRM button and then click a PROCEED button (or variations of this procedure). This procedure is another example of a verification method. Indeed, it may be followed up with a confirmation email, which is a further method of verification.

Data validation:
Checking that the data you entered is sensible and follows the rules you set up for the data.

Data verification:
Checking that the sensible data you entered is actually the data you meant to enter in the first place!

3.13 A database can still become corrupt

It should be recognised that even with clever validation and verification techniques in place, the data entered into a database could still be compromised. For example, it is possible that both operators during validation entered in the same, but wrong, data - perhaps they both misunderstood the handwriting on the paper-based forms. It is possible that both operators entered in the information on the form, but the information on the form was wrong! Even with the best methods of validation and verification in place, data can still be compromised.

3.14 Getting information out of the computer

When a piece of software has done some processing, the results of the processing need to be communicated to the users.

3.14.1 Reports

These are formal printed out documents. Reports organise information in a clear way, perhaps using tables. They are often used to present sets of numbers so that a skilled reader can look at them and then interpret them.

3.14.2 Graphs

These provide a pictorial representation of a set of numbers. They help a reader make sense of data.

- A pie chart highlights the contribution a particular item makes to the whole. A car dealer might, for example, show how many cars were sold in each of the last 12 months, as a percentage of the cars sold.
- A bar chart shows how many occurrences there are of a set of items. A car dealer could, for example, have a bar chart showing how many FORDs, AUDIs, VWs etc were sold in the last year.
- A scatter graph is used to plot one variable against another variable to see if there is a relationship. A car dealer might plot the price of each car sold against the age of a buyer, for example, to see if there is a relationship between the age of someone and how much money they typically spend on a car.

3.14.3 Sound

Sound can be used to output information. It can be used to highlight that a verification error has occurred during data input by 'beeping' the keyboard. It could be used to signal that an emergency situation has arisen in a chemical factory by sounding a klaxon. It could be used to read back text to visually impaired individuals through speakers, for example.

3.14.4 Multimedia output

The previous output formats can be used to provide immediate output from systems. Where it is required to present information in some way, video, sound, animations, special effects, text and pictures can all be combined using presentation software. Many packages are very user-friendly, providing a user with templates to put together a presentation and by providing on-screen help and tutorials. Even novices can create good presentations although it is an art form! It is easy to lose a message through poor design. Microsoft produces the widely used PowerPoint but there are other multimedia presentation packages available and some are free and arguably just as good as PowerPoint, for example, OpenOffice. Go to **http://www.openoffice.org** and have a look at what is in the latest version of OpenOffice. You will see an application called **Impress**. This is a free multimedia presentation package you can use as an alternative to PowerPoint.

3.15 Output formats and the audience

When considering what output format to use for a situation, it is important to think about how appropriate the output will be.

- Will the output communicate to the user what they need to know?
- Will the output communicate to the user data or information when they need to know it?
- Will the output communicate to the user data or information in a form that they can make sense of and use?

3.16 Examples of the selection of output formats

Selecting the right format for any situation gets easier the more experience you have! There are an infinite number of situations. We will look at a number of examples.

1) The output format for a burglar alarm should be a bell and a light. This is because attention needs to be drawn immediately an intruder is detected. A strobe light is also necessary for legal reasons - the bell must automatically switch off after 20 minutes and so to continue to signal that an intruder has been detected, a flashing light is used.
2) Sound is an excellent output to use for any situation where you need to attract somebody's attention quickly. For example, if any critical situation is reached in a nuclear power plant, or in a hospital where a patient is on life-support, an alarm should sound that says, "Immediate action needed now!" to the operators.

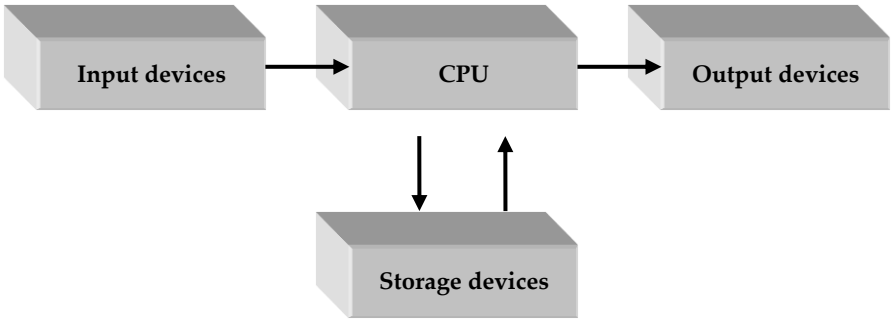
3) Sound is a useful output for visually-impaired individuals. Software can convert textual information into sound.
4) Hard copy is useful in many circumstances! For example, a sales assistant might take an order for some products that the customer can take away that day. The order is immediately sent to the warehouse. The warehouse staff may want to print off a copy of the order so that they can take it with them around the warehouse as they collect the products. Once collected, they can be taken to the counter and given to the customer. The customer can sign the printout to say that they have received the goods. Secretaries may want to print out letters to send them off.
5) VDUs strategically situated around a site may allow an operator to see textual orders. This kind of system would work well in a fast food restaurant. As orders are taken, they could be displayed on a screen for the cooks to see.
6) Operators and managers very often want to see how something has changed over a period of time, for example, how the water pressure in a pipe has changed over a day, or the pressure inside a chemical reaction tank has varied. Trends are best displayed on a graph. They allow an operator or manager to immediately see the changes over time and to interpret the information.
7) Sometimes, graphs and charts don't tell the whole story. It is possible that the accuracy of graphs is insufficient or an accident has occurred that needs investigating. This may require skilled people to look at the fundamental data rather than graphs and charts. To do this, a report could be generated with tables of data.
8) Multimedia presentations allow a wide range of media to be brought together in one package. This keeps a presentation lively and exciting. If just video, for example, were used, then it might be difficult to keep an audience's attention if the presentation were long.

Q1. Describe when form interfaces are typically used, giving an example.
Q2. State two places you might typically find a menu-driven interface.
Q3. What is a GUI interface and a WIMP interface?
Q4. Who might typically use a command line interface?
Q5. What is a dialogue box?
Q6. What is 'tab order' on a user interface?
Q7. Define validation.
Q8. Define verification.
Q9. Calculate the check digit for any 13 digit ISBN number. (Note that 10 digit ISBN numbers use a different system to the one described in this book for 13 digit ISBN numbers.)
Q10. Describe two ways that an order placed on a form can be verified.

Chapter 4 - Input and output devices

4.1 Introduction

We have identified that a computer can be broken down into four areas for discussion, as shown in this diagram:



A diagrammatic representation of the four parts of a computer.

Computers must have data to work on. Data must be moved from outside the computer and put into it. The methods of getting data into a computer have traditionally been discussed by talking about using manual methods of data input and automatic methods of data input. We will discuss these two methods in this chapter and will discuss the input devices that are used.

4.2 Manual data input methods

Each year at secondary school, pupils move up from various primary schools. Parents are sent a form to fill in, requesting such details as name, address, date of birth and contact numbers. This information is then typed in by the secretary. This is an example of a manual data input system. The data is collected first and then it is entered in via a keyboard by the secretary. It isn't entered into the computer by 'feeding' the data into the computer automatically using MICR, or OCR, or any other 'automatic' method (see later in the chapter). The data hasn't been prepared by coding it into a form that the computer can read automatically e.g. a barcode.

4.2.1 Keyboards

Practically all computers have QWERTY keyboards. These devices can be used to enter in data manually. They are very efficient in the right hands and not so efficient if a user has limited training. In fact, they can be very slow and mistakes are easy to make, although software tools can automatically fix many of them. A lot of work has been done to ensure that keyboards are designed **ergonomically**. This means that they are designed in a way that takes into account the 'design limitations' of a human being. There is a health and safety problem known as Repetitive Strain Injury (RSI) for people who use keyboards all the time, such as secretaries. The joints in their fingers can become 'worn'. An important consideration for this kind of manual data input when used to transfer paper-based data into the computer is the design of the user interface. Form-based interfaces should be used because they help the data input operator enter data as quickly and as efficiently as possible. The form on the computer and the paper-based data capture form should be laid out in a very similar if not identical way to each other.

4.2.2 Touch-sensitive keyboards and concept keyboards

An extension of manual data input into computer systems using keyboards is to use touch-sensitive keyboards and concept keyboards. These types of keyboard have 'keys' which are made up of touch-sensitive areas on a plastic cover. A user presses on the touch-sensitive area to input data into the computer. They have a wide range of applications. They can be used in fast food restaurants. Operators simply press on pictures on the keyboard. This speeds up data input and means that training needs are minimal. They can be used in areas where there is likely to be a lot of dirt such as in factories. Dirt can get into normal QWERTY keyboards and cause them to malfunction but the protective plastic cover on touch-sensitive keyboards stops this happening. They can be used to customise keyboards because some touch-sensitive keyboards (called 'concept keyboards') allow you to program what you want to happen when an area is pressed. You can imagine, for example, designing a concept keyboard for a two year old child, who may have limited co-ordination and cannot yet read. You could provide a keyboard with four big brightly coloured areas. When the child wants to make any selection at all, they just touch the right coloured area.

4.2.3 Touch screens

This is another manual data input method. A touch screen enables a user to touch their VDU screen to make selections. A plastic cover that has fine wires running through it can be placed over a VDU's screen. A user makes a selection by touching the screen with their finger. The exact position can be calculated from the signals sent back by the wires. Touch screens allow very fast selections from choices. They could be used in places where people need to find out information but may have zero computer skills, for example, an information system in a library or a museum. They would be of limited use if you had to type in a letter, for example.

4.2.4 Graphical tablets

Another manual data input method is the graphics tablet. These are touch-sensitive pads that allow you to 'draw' on them with a stylus. The pressure from the stylus on the pad is sent to the computer, which reproduces what was done on the pad in a drawing program or CAD program. These are more natural for designers to draw with than a keyboard and mouse.

4.2.5 Mice

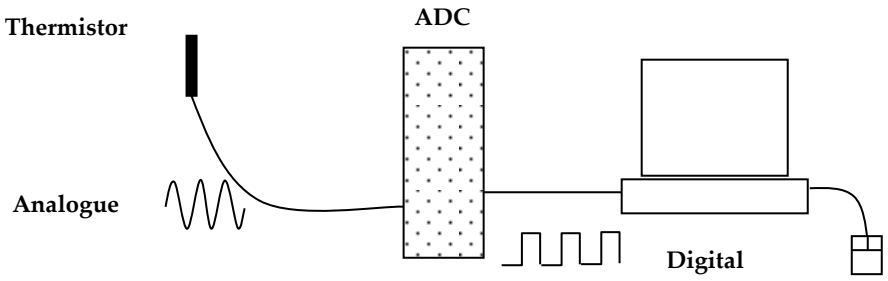
A mouse is a pointing and selecting device used with graphical user interfaces (GUI). There are different kinds of mice around, each with their own advantages and disadvantages although they all broadly do the same thing: point and select. A search on the Internet should enable you to quickly identify different sorts of mice. These include standard 2/3 button mice and mice with a scrolling wheel to allow you to better navigate applications and web pages, optical mice that aren't prone to collecting fluff and dirt and so don't need cleaning, mice that use radio waves to connect to the computer instead of wires, so that there is less clutter on the desk, ergonomically designed mice, for example, mice that are very small for the small hands of children and mice that can work with a serial port, a PS/2 port or a USB port.

4.3 Automatic data input methods

Automatic data input methods are methods where the data is collected and entered into the computer directly, without having to manually prepare the data first. We will see some examples of this to illustrate what we mean.

4.3.1 Data logging

A classic example of an automatic data entry method is data logging. Consider an experiment where a pupil wants to know how the temperature of a computer room varies over 24 hours. They would need to select a transducer to read the temperature. In this case, they select a **thermistor**. This is an electronic device with a special property. The output voltage from the thermistor constantly varies, depending upon the temperature. It is known as an 'analogue transducer' because there are an infinite number of outputs from it. The pupil knows that the computer is a digital device. They know that you cannot simply connect an analogue device to a digital device. You have to convert the analogue signal first using an 'interface'. In this case, the interface required is an **Analogue to Digital Converter** (ADC). This takes the analogue signal used by the thermistor and converts it into the digital signals needed by the computer. The thermistor is attached to the ADC and the ADC is attached to the computer.



An Analogue to Digital Converter in use.

We have said that an interface is often required in control and data logging applications. You certainly need one if you want to attach any analogue transducers to a computer. An interface may perform other functions:

- An interface may convert voltage signals from analogue into digital (ADC) or digital into analogue (DAC).
- An interface may be used like a switch, so that e.g. a small dc voltage from the computer can be used to switch on and off a large motor that uses 240Vac.
- An interface may provide compatible physical connections for a computer and the devices that need to be connected to it. Devices may use plugs and sockets that are physically different to a computer's and so they cannot be directly attached to it. They have to go via an interface.

4.3.1.1 Sample times

The pupil wants to know how the temperature varies in a 24-hour period. A suitable output for this would be a graph. If the pupil set up the computer to take a reading every 10 minutes, then the computer would take 6 readings an hour, or 144 readings in 24 hours. This sample would be more than sufficient to produce a good graph. Of course, the pupil could have set up the computer to take hundreds of readings every second! This would have been unnecessary in this case because it wouldn't have told the pupil any extra information. Taking readings at appropriate times is known as 'sampling' or 'taking a sample'. The trick is to be able to justify for any given problem what an ***appropriate*** time interval between sample readings is!

4.3.1.2 Data logging and satellite communication

Suppose we have a data logging system set up on the top of a mountain, to record the temperature. The data logger might take a reading every hour and store the value. How can the recorded values be sent back without having to send a person up to collect the readings? It can be achieved using satellite technology. A transmitter attached to the data logger sends a microwave signal to one of a small number of geostationary satellites that cover the planet. The message is amplified and retransmitted back to Earth. The signal is possibly bounced back up to another satellite and down again, so that the data is passed around the planet. The data is then collected and analysed by computer and possibly converted into graphs so trends can be examined.

We can log data in any place where people couldn't realistically go for long periods, perhaps because they are dangerous, such as in a nuclear reactor, in space or at the bottom of an ocean. Readings can potentially be taken 24/7 and are likely to be far more accurate than readings people take. We might also need data logging equipment where things happen so quickly that people couldn't take enough readings to analyse, for example, in scientific experiments. Of course, the equipment might breakdown so backup equipment may be necessary.

4.3.2 Optical Mark Readers (OMR)

This is another method of automatic data input. Data sheets are prepared and people put marks in set places to indicate a choice. They are used, for example, in multiple-choice tests because the answers can be scanned in and a pupil's mark calculated by the computer - less work for the teacher. They can be used to capture answers to questions on a questionnaire, or to select numbers on a lottery ticket. Some applications are not suitable for OMR. For example, you wouldn't collect names using OMR because you would have to provide 26 places for a mark to be made for each letter in the name! They are really only suitable when a small number of choices are available.

When OMR sheets are completed, they can be scanned in automatically and the results produced straight away. The data doesn't have to be typed in manually, which could introduce typing errors, takes time and can cost a lot of money if there is a lot of data to enter. If you are not experienced filling in OMR sheets, they can cause problems. It is easy to make a mistake. If this happens, you need to know how to correct it on the OMR sheet. There is usually an elaborate set of procedures to follow if this situation arises. Estimates of the number of OMR sheets rejected when used in questionnaires range from 10% up to 30%! In addition, if the OMR sheets themselves get torn or creased, then they may get rejected.

4.3.3 Optical Character Recognition (OCR)

This is a method suitable for getting data into a computer that has already been prepared. It can be used, for example, to transfer spreadsheet information directly into a spreadsheet or to get textual information from books into a word processor. You could easily use this method to scan the complete works of Shakespeare, for example, and then you could analyse each work, to check that it has really been done by Shakespeare! You could use this method to transfer human knowledge from books to computer to allow it to then be searched easily, transferred and distributed, for example.

OCR is used to read postcodes on letters by the post office to enable speedy and cost-effective sorting of mail. It can be used to convert written documents into a form that could then be output via speakers to people with problems with vision. A page of writing is first scanned on a scanner. A very bright light is shone on the text and the white and dark space caused by the letters on the page reflect different amounts of light back. This is measured and used to produce a bit map **picture** of the writing. It is **not** yet a word processing document. Some OCR software is then run on the picture. This scans the picture looking for patterns that represent characters on a keyboard. As it finds these patterns, it transfers them to a text file. When the whole picture has been scanned, a text file will have been produced that can then be opened in a word processor. Although software is becoming ever more powerful, OCR is still only reliable for typed work. It still struggles to recognise handwriting.

4.3.4 Bar codes

Bar codes are made up of black and white striped lines. The lines represent data in coded form. The data held in a bar code can be retrieved by using a laser scanner. The data obtained can then be used to look up further information held on computer. Bar codes can be attached to libraries to speed up taking out and returning books. Membership systems that require members' details to be retrieved could employ barcodes. Supermarkets could use them for stock control systems, to speed up the checkout

process and to produce itemised receipts. Bar code systems have built in validation techniques that greatly reduce errors. The data scanned can be integrated into management information systems so that, for example, managers can tell which books are never taken out of a library and should be removed, which members never attend an event or which products in a supermarket sell best on Sundays. In supermarkets, bar codes are an integral part of the stock control system. One problem with any system that is completely reliant on computerised systems, however, is what to do when the system breaks down. A bar code for a product in a supermarket will typically contain:

- the country of origin of the product
- the manufacturer's identity number
- the code for the actual product
- the check digit, used to check that a number has been scanned in correctly.

4.3.5 Magnetic stripe cards and smart cards

Data can be entered into computer systems by using cards that have a magnetic stripe on them. The magnetic stripe holds coded information. This can be retrieved by 'swiping' the card through a magnetic card reader. The information can then be used directly, or used to retrieve more information from a central computer.

This type of data input is typically used for credit cards, debit cards, loyalty cards, membership cards and security access cards. They are quick to use, can be used many times, are cheap to produce and the important information held on them cannot be read unless you have the right equipment. Magnetic stripe cards can be read but not written to. The data is placed on the card when it is made. For this reason, smart cards have become more widely used. These are cards that have an electronic chip on them. This can be used to record each time a transaction occurs, for example. In other words, smart cards can be written to as well as read from! Interestingly, cards with magnetic stripes typically have some details written on the card as well so that a person can read them. It might have the name of the owner of the card on it, in case the card gets lost. It might have the card number on it, in case the owner of the card wants to use it to buy something over the Internet or phone.

4.3.6 Voice recognition

Voice recognition software is getting better every year! Using a microphone and some appropriate software, it is possible to input data into a computer or directly into a tablet PC or phone. The software is not perfect. You often have to teach it to recognise your voice accurately. If you have a cold or a throat problem, or a very strong accent, it may reduce the accuracy of input. Nevertheless, it is still a fast way of inputting data.

4.3.7 Magnetic Ink Character Recognition (MICR)

When banks produce cheque books, they print on the bottom of the cheque the sort code of the bank, the account number and the cheque number. These numbers are printed in magnetic ink because the cheques can then be read automatically once a cheque has been written. It doesn't matter if the cheque gets creased or a little dirty because the data on them can still be read by the special magnetic ink readers.

When a cheque has been written, it will be paid in to the bank. It cannot yet be read automatically, because the date and amount have been written on by hand. When a cheque is paid into a bank, an operator prints on the cheque how much it is for and the date it was paid in. When all of the cheques for that day have been collected together, they are loaded up into a special machine that batch processes them - it reads each cheque and adjusts all the accounts as necessary. MICR cannot be considered completely automatic because some of the data must be prepared before it can be entered into a computer. It is an example of a cross between an automatic data input system and a manual one.

4.4 Image capture

We will frequently want to capture images and get them into a computer. Images can be captured in a number of ways. We will look at scanners, video capture cards and digital cameras.

4.4.1 Scanners

Images from magazines or photographs, for example, can be captured using scanners. The series of steps you need to follow to do this are as follows:

- Typically, the image is placed on a flat screen or 'bed'.
- A cover is placed over the image.
- The image is divided up into sections or 'pixels' by the software. The user can tell the software what resolution to use (how many pixels per square centimetre to split the picture up into). The higher the

resolution, the better the detail of the image but the bigger the file. Often, low resolution, smaller files will be perfectly adequate for most uses.

- A light is passed from one end of the flat bed to the other, so that it passes over the whole image.
- When the light hits each pixel, it gets reflected back. The intensity of the reflection depends on the colour at that pixel. Each pixel's information is stored.
- The information about the pixels is used by the software to reconstruct a bitmap image of the picture.
- Because bit maps are large files, they are often compressed. This can be done by telling the software to save the image as a different file type that uses compression, such as GIF files or JPG files.
- Software tools either within the scanning software or within a drawing package allow the user to manipulate an image in various ways. These typically include allowing the user to 'crop' an image (select just a portion of an image), allowing the user to improve the detail of the image, allowing the adjustment of colours and allowing the user to add special effects such as making a photo image look antique.

4.4.2 Video capture cards

A video is made up of a series of pictures, or frames, that are played quickly enough to appear moving. A video capture card is a piece of hardware that is plugged into an expansion slot inside the computer. A user then attaches a video camera or TV, for example, to the video card. When the video or TV is played through the card, the analogue signals that make up the moving picture from these devices are converted into digital images, frame-by-frame and stored. Once you have captured each frame, you can then use the software to do all kinds of clever things. For example, you can edit out frames, reorganise them, save individual frames and transfer them to word processing documents, create your own presentations using presentation software and some of the images or add your own soundtrack.

4.4.3 Digital cameras

These cameras do not store images on film. They store images digitally in memory. The images are then transferred to the computer. There are many points that could be made about digital cameras.

- The price of digital cameras has been steadily falling over recent years.
- The amount of memory is a very important consideration with these cameras, as is the ability of a camera to add memory. This is because storing images is memory-intensive.
- Cameras often allow the user to select between high resolution and low resolution modes. If you use a high resolution mode, you will be able to take fewer pictures than low resolution mode.
- You can immediately view photos and re-take or overwrite them if they are not what you want.
- You can often add extra information easily, such as the date or information about the photo.
- Many cameras allow you to add special effects as you take the picture.
- Images can easily be combined into digital photo albums and distributed or emailed to friends.
- Once the digital picture has been transferred to the computer, it can be opened in a drawing package and manipulated e.g. pictures could be cropped, colours changed and parts of the photo 'touched up'.

4.4.4 Camcorders

Video camera recorders, or camcorders, are cameras that store moving images. There is a wide range of types of camera. Some camcorders store the moving images as analogue signals. You may have heard of VHS, Super VHS or 8mm analogue camcorders. Analogue camcorder films lack really mint quality and lose quality if they are copied. Films made using digital camcorders, on the other hand, have much better quality and don't lose any quality if they are reproduced. This is because the films are stored digitally. Digital Video (DV) and Digital 8 are two of the common digital formats around. Typical features of digital camcorders include the ability to zoom in, record sound, view films through a viewfinder, some have night viewing capabilities, time-lapse photography, picture stability software and special effects.

4.5 Introduction to output devices

There are lots of commonly used output devices available for a user to select from.

4.5.1 Dot matrix printers, inkjet printers and laser printers

These are relatively slow and noisy and the quality of the hard copy is relatively poor compared to ink-jets and laser printers. They were very common a few years back in the early days of computing. Their uses are far more limited now but they do have one particular advantage. The hard copy is made by pins striking paper. That means that identical copies can be made of a printout by using carbon paper between sheets of paper. This system is used by credit card companies to produce **actual copies** of receipts when a customer makes a purchase. After a customer's credit card is swiped and authorised, two identical copies of a receipt are printed using small dot matrix printers. Both copies are then passed to the customer, who signs the top copy. This puts a carbon signature on the bottom copy. The customer keeps one copy and the shop keeps the other. You cannot make

actual carbon copies with ink-jets or laser printers although of course you can print out two copies of a document! Inkjet printers 'spray' ink onto paper. You can't produce carbon copies with inkjet printers but you can produce very high quality black and white as well as colour copies for a very low cost. An ink-jet printer is a good choice for low volume applications such as small businesses or homes. Laser printers produce very high quality black and white as well as colour hardcopy. A laser printer costs more to buy and run than ink-jets although costs have been steadily falling in recent years. The price of colour laser printers has been falling in recent years although refills are expensive compared to ink-jets.

4.5.4 Plotters

Plotters are widely used in some industries. They are used to plot very large drawings such as those needed by engineers and designers whereas standard printers commonly only print up to A4 (and sometimes A3). They can produce very high quality, very accurate, colour drawings but are relatively expensive compared to printers.

4.5.5 Visual Display Units

Monitors are ideal for displaying data and information to users. They come in a range of sizes. Larger ones such as 21-inch screens, for example, would be ideal for engineers using Computer Aided Design software applications. 15-inch screens are perfectly acceptable for users using a range of generic applications. CRT (Cathode Ray Tube) monitors (similar to televisions) do take up a lot of space on a desk. Flat panel liquid crystal display screens, often referred to as TFT screens, save a lot of space by comparison. They are not quite as sharp as CRT screens, however, and are more expensive. TFT screens produce less radiation than CRT monitors. Excessive exposure to radiation is seen as a potential risk to computer users. They also use about half the power a CRT screen uses. If you multiply up the savings in power use in an organisation with thousands of computers, the cost-savings do become significant.

4.5.6 Speakers

Some applications such as burglar alarms, factory warning systems and monitoring equipment make use of audio output. Some applications also require sound, such as video-conferencing, using your computer to make phone calls, listening to DVDs or CDs and playing games. There are different ways that audio output from a computer can be achieved. The cheapest option is simply a pair of speakers. They will plug into the computer in the computer's speaker output. The quality and level of sound will be perfectly adequate for many applications but they cannot produce a very loud output and cannot produce a very high quality sound. It is perfectly possible to connect the output from a computer into an amplifier and then pass the amplified signal to some speakers. This is a more expensive proposition but does produce hi-fi quality sound. In some noisy environments such as factories, **klaxons** (sometimes known as 'sirens') are common. These can be computer controlled and can produce a very loud sound that can be heard over noisy machinery.

4.5.7 Headphones

There are situations where a user wants to listen to sound in a public place but doesn't want to disturb others. For example, a user in a library might want to listen to CDs or a telesales operator might need to concentrate on what a customer is saying.

4.6 Target audience

Whenever deciding on the input and output devices, the starting point should always be the target audience. Who are they? What is their age group? What is their experience of computers, hardware and software? What is their educational background likely to be? Are there any disabilities that need to be taken into account? Do they speak English or other languages? What training and periodic retraining are they likely to need? What reports and screen displays do they need and why, for what purpose and in what format? What will happen to the information that any new system produces? Where will the users actually be based? What is the environment that they will be working in?

Q1. What is meant by an 'automatic' data entry method?
Q2. If a device is described as 'ergonomic', what does that mean?
Q3. Describe how the temperature on top of Mount Everest could be read over a six month period.
Q4. Using an example, explain what an 'appropriate sample time' is.
Q5. State two uses of OMR technology.
Q6. What data is coded into the barcode of a can of beans, on sale in a supermarket?
Q7. State some potential problems of using voice recognition.
Q8. How does a scanner work?
Q9. State a use for a dot matrix printer, that wouldn't be satisfied by a laser or inkjet printer.
Q10. What is a 'hardcopy'?

Chapter 5 - Bits, nibbles, bytes and codes

5.1 Introduction

Any switch can be in one of two positions, off or on. For example, the switch that controls a light in a room can be off or on. The power switch on your computer can be off or on. A mobile phone has a power switch that can be off or on. The basic building block of any computer is the switch. Computers, however, have millions and millions and millions of electronic switches in them, held in components such as RAM or the processor. When you group these switches together in a certain way, you can represent data, such as letters of the alphabet or numbers!

5.2 The denary numbering system

The numbering system that we commonly use every day of our lives is known as the **denary system**. This is because there are ten digits in use. These are 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9. We know that the symbol 7 is 'worth more' than the symbol 3. It is not only the size of the number that tells us something about the number, however. It is also the **position** of each digit relative to any other digits in a number.

For example, in the number 268, the 6 is worth a lot more than the 8 because it is to the left of the 8. Similarly, the 2 is worth a lot more than both the 8 and the 6 because it is on the very left of the number. When you learnt to count, you will have used headings to start with, to help you understand that the position of each digit is important to the worth of that digit. Any number e.g. 3892 would have been written down under the headings, as shown here.

Thousands	Hundreds	Tens	Units
3	8	9	2

This means the number is 'worth' (3 x 1000) + (8 x 100) + (9 x 10) + (2 x 1) which adds up to 3892. Can you use this method to break down the number 4390? Now break down the number 28642. What is the next position on the left after thousands worth? What's the next position on the *right* of the Units is worth? (HINT: It is used to represent the fractional parts of a number).

5.3 The binary numbering system

Computers use switches. A switch can be off or on and there are no other possible positions. We can represent switches using the binary numbering system. The off position will be represented by 0 and the on position will be represented by 1. Whilst the denary numbering system had 10 digits, the binary numbering system has only 2 digits, 0 and 1. However, like the denary system, the position of the **bi**nary digi**t** (or bit) is important. So for example, the binary number 10101101 means

One hundred and twenty eights	Sixty fours	Thirty twos	Sixteens	Eights	Fours	Twos	Units
1	0	1	0	1	1	0	1

This is worth (1 x 128) + (0 x 64) + (1 x 32) + (0 x 16) + (1 x 8) + (1 x 4) + (0 x 2) + (1 x 1) which adds up to the denary number 173. Can you use this method to find out what the denary equivalent of 01100100 is? Can you guess what the next position to the left of the 'One Hundred and twenty eights' position is worth? How about the position to the right of the Units position?

5.4 Using subscripts

How do you know whether a number is a binary number (sometimes called a 'base two number') or a denary number (sometimes called a base ten number)? For example, is 1000 the denary number 'one thousand' or the binary number equivalent of eight? It makes a big difference because the denary number will be so much larger than the binary one. To make it clear which numbering system is being used, we can add a **subscript** to a number. For example:

- 10001011_2 is a binary number whereas 10001011_{10} is a denary number.
- 3452_{10} is a denary number. 10000_{10} is a denary number. 011000_2 is a binary number.
- 320000_{10} is a denary number. 01100011_2 is a binary number.

You don't always see subscripts next to a number. For example, when we use base 10 in our normal lives, we don't write a 10 subscript next to the number. We don't because there is usually no confusion possible. Similarly, if we are only using binary numbers, we will omit the 2 subscript. There are times, however, when we will be using base 10, base 2 and other numbering

systems so we should make clear which numbering system we are using by using subscripts.

5.5 Nibbles and bytes

A single bit cannot hold a great range of numbers! It can hold either zero or one. You often read about nibbles. A **nibble** is a group of 4 bits. The smallest value a nibble can hold is 0000 in binary and the largest number is 1111 in binary. (0000 in binary is the same as 0 in denary. 1111 in binary is the same as (1 x 8) + (1 x 4) + (1 x 2) + (1 x 1) or 15 in denary. It is also very common to group bits together in groups of 8. A group of eight bits is known as a **byte.**

Bytes are extremely convenient units to work with, as you will find out on your course.

5.6 Kilobytes, Megabytes and Gigabytes

We have seen that a byte can be used to represent a number. We will see soon that the number can be thought of as a code that represents a character on a keyboard. Before we look at that, however, we should note that if one byte is going to represent one character on the keyboard then we are going to have to collect together lots of bytes to record a memo, for example. For that reason, we frequently talk about Kilobytes, Megabytes and Gigabytes.

- 1 Kilobyte (1 Kbyte) is 1024 bytes exactly, or 2^{10} bytes exactly, or about 1000 bytes, or about a thousand bytes.
- 1 Megabyte (1 Mbyte) is 1048576 bytes exactly, or 2^{20} bytes exactly, or about 1000000 bytes, or about a million bytes.
- 1 Gigabyte (1 Gbyte) is 1073741824 bytes exactly, 2^{30} bytes exactly, or about 1000000000 bytes, or about a thousand million bytes.

So 15 Kbytes is about 15 thousand bytes. 128 Mbytes is about 128 million bytes. 20 Gbytes is about 20 thousand million bytes. More often than not, you don't need to know the exact number of bytes, just an approximation!

5.7 Character sets

When you press a key on your computer's keyboard, a code is generated. Every lower case letter, every capital letter, every number, every 'special' character like the pound sign and @, for example, the space bar, the ENTER key and so on, has its own binary code.

When you write an email, for example, you press keys on the keyboard. Each key is converted into a binary code. Together, they are saved to form a message. You might then send this message to your friend. When your friend receives the message, they have to be using the same codes as you to 'decode' the binary codes back into letters. Imagine what would happen if the codes for ' H e l l o ' were different - according to your friend's codes, the message might be ' r % P } + '! Clearly, this is not good. The reason that computers can talk meaningfully to each other is that they all (more or less) use the **same codes**. There are three that you should know about. In addition, you should look up **ANSI** for yourself and find out how it differs from these three.

Three character sets.

5.7.1 Standard ASCII and Extended ASCII

The most important set of codes to represent all of the possible key presses on a computer keyboard is the American Standard Code for Information Interchange, or ASCII (pronounced 'ass-key'). It is the set of codes used by Personal Computers.

In Standard ASCII, each character on the keyboard is represented by a 7 bit code. There are 96 displayable characters and 32 codes that are used for controlling e.g. printing. In Standard ASCII, for example, the letter 'A' is represented by the 7 bit code 1000001 (65 in decimal), the letter 'a' is 1100001 (97 in decimal), the '?' is represented by 0111111 (63 in decimal), a space is 32 in decimal and Null is decimal code 0. All of the different possible codes together make up what is known as the ASCII character set. If you are using 7 bits to represent a code, you have a total of 2^7, or 128 possible combinations. That means you can represent 128 different characters!

0 NUL	16 DLE	32 SP	48 0	64 {@	80 P	96 `	112 p
1 SOH	17 DC1	33 !	49 1	65 A	81 Q	97 a	113 q
2 STX	18 DC2	34 “	50 2	66 B	82 R	98 b	114 r
3 ETX	19 DC3	35 #	51 3	67 C	83 S	99 c	115 s
4 EOT	20 DC4	36 $	52 4	68 D	84 T	100 d	116 t
5 ENQ	21 NAK	37 %	53 5	69 E	85 U	101 e	117 u
6 ACK	22 SYN	38 &	54 6	70 F	86 V	102 f	118 v
7 BEL	23 ETB	39 ‘	55 7	71 G	87 W	103 g	119 w
8 BS	24 CAN	40 (	56 8	72 H	88 X	104 h	120 x
9 HT	25 EM	41)	57 9	73 I	89 Y	105 i	121 y
10 LF	26 SUB	42 *	58 :	74 J	90 Z	106 j	122 z
11 VT	27 ESC	43 +	59 ;	75 K	91 [	107 k	123 {
12 FF	28 FS	44 ,	60 <	76 L	92 \	108 l	124 |
13 CR	29 GS	45 -	61 =	77 M	93]	109 m	125 }
14 SO	30 RS	46 .	62 >	78 N	94 ^	110 n	126 ~
15 SI	31 US	47 /	63 ?	79 O	95 _	111 o	127 DEL

The standard ASCII character set – decimal values and the character or control code they represent.

Standard ASCII uses 7 bits for each code. Programmers and computer people like to work in nice easy packets of 8 bits (called a byte). That means that we have an extra, 8th bit to play with! We can use this extra 8th bit in Standard ASCII for some error checking, looking for errors when bits are sent from one place to another. When bits are being transmitted, there is a real possibility that they get 'corrupted'. In other words, the bits and therefore the codes change and so the message changes, too! It is necessary to check for errors when data is transmitted. The error-checking that takes place using the 8th bit is known as ‘parity checking’. This is dealt with elsewhere in the book.

An alternative is to use all 8 bits for a code instead of 7 bits so you would have a total of 2^8, or 256 different combinations in your character set. In other words, you can represent 128 more characters than in 7 bit ASCII. You can have a code for the letters that appear in other languages but not in the English alphabet or for graphics symbols, for example. All of the 8-bit codes together are known as the Extended ASCII character set. Most computers today use Extended ASCII so extra characters can be represented. There is another character set, however, that is used in large commercial systems.

5.7.2 EBCDIC

The Extended Binary Coded Decimal Interchange Code is a character set used by older 'mainframes'. A mainframe computer is simply a computer that can be accessed from many terminals. It is often the preferred type of computer for businesses that process a lot of data. This character set uses different codes to ASCII so a PC couldn't directly 'talk' to one, although it could if a special program was written to allow them to talk.

It is unfortunate that there are two widely used character sets in use. This situation has come about because at one time, in the early days of computer development, manufacturers tried to make their own character sets the standard one to use and a lot of different ones were promoted. ASCII and EBCDIC emerged historically together and now we must live with it!

5.7.3 Unicode and UCS (Unified Character Set) – ISO 10646

When we are talking about the ASCII or EBCDIC codes we are viewing the world as a place where everyone speaks and uses English, using the characters and symbols we are all familiar with. There is a problem!

Many languages do not use the 26 letters of the English alphabet. There are literally thousands of symbols used, for example, to write Chinese. Then there are Japanese symbols, characters used in the Russian alphabet, Greek, Thai, Runic, Bengali, Tamil, Telugu, Arabic, Malay, Lao, Khmer, Tibetan, Ethiopian, Gujarati, Cherokee, Mongolian, Yi and the list goes on and on and on. It goes further than that, however. There are also many mathematics symbols in use all over the world and all kinds of other symbols, such as the scripts used by the writer Tolkien (of Lord of the Rings fame)! And of course there may be many new scripts and symbols added in the future. If there is to be a way for users of software to access the characters in any language, and if we all want to access the greatest possible range of symbols used in the world, then clearly we are going to have to think a little bit bigger than ASCII. This is especially true for companies who do business globally or who need to create multi-lingual documents.

We have seen that ASCII is simply a list of 256 numbers (8 bits), each number being allocated one of the characters or symbols that you can see on the keyboard in front of you. We have to do this because computers can only understand numbers not

characters. Unicode uses exactly the same process as ASCII. It is a list of numbers, each number being allocated a particular character or symbol. However, Unicode is a much bigger list than the 256 numbers available in ASCII. In fact, the standard that defines Unicode and UCS, called ISO 10646, uses 31 bits. This gives about 2000 million codes. Unicode only uses a subset of this however, using 16 bits to give about 65000 unique codes that have been allocated to symbols.

Don't be confused by the two common standards 'Unicode' and 'UCS'. They originally started out as two different standards but the two organisations saw the light and decided that one system would be better for all concerned. They are still separate standards but have become in practical terms interchangeable. Also note that Unicode has incorporated ASCII.

5.7.4 Unicode, HTML and web browsers

Have you ever written some HTML code for a website? Suppose you want to display a special symbol such as a trademark symbol on the website. You would use the &#8482 in your code. This is because the Unicode symbol for the trademark symbol, ™ is &#8482. &#916 will display the Greek letter delta. You can easily find a Unicode reference for your website by doing a search for 'HTML Unicode'.

Another point to make about Unicode is that if web browsers are to be used by many different peoples from the entire world then they need to understand more than just ASCII code. The latest web browsers make use of Unicode and can therefore be used universally. (You may need to set up your web browser properly or install the appropriate fonts if you are having problems displaying Unicode characters - there is plenty of help on the Internet).

Symbol	Decimal code
Trademark	**&#8482**
Bullet point	**&#8226**
Left double quote	**&#8220**
Right double quote	**&#8221**
Double dagger	**&#8225**
Florin	**&#402**

Some UNICODE examples for you to try out in a web page.

5.8 String manipulation and ASCII codes

In programming, you often need to compare words. For example, you might be given a list of words to put into an alphabetical order. How would the computer do this?

Suppose you had to compare the word 'dog' with 'cat'. The question we have to ask is, 'Is 'dog' greater than 'cat'? (In English, we would ask if 'dog' comes after 'cat' in a dictionary. We can represent this mathematically as 'dog' > 'cat' and the answer will either be TRUE (dog comes after cat) or FALSE (dog does not come after cat).

The computer looks at the first letter in each word and compares their ASCII codes. The ASCII code for 'd' is 100 and the ASCII code for 'c' is 99. 100 > 99 is TRUE, therefore 'dog' goes after 'cat'. We have to remember to look up small 'd' and not the capital letter as they have different ASCII codes.

How would the computer decide which word comes first between 'Cart' and 'Card'? Does 'Cart' go after 'Card'? Or to put it in a mathematical way, 'Cart' > 'Card'? Is this a TRUE statement or a FALSE statement?

The computer looks up the ASCII code for the first letter in both words, and it is the same (67). So it then gets the second letter in both words. They are the same, so it goes to the third letter and they are the same. It then gets the ASCII code for the fourth letters in each word, 116 for 't' and 100 for 'd'. 116 > 100 is TRUE, which means 'Cart' > 'Card' is TRUE so 'Cart' goes after 'Card'.

How about 'Ben' and 'ben'? The ASCII code for 'B' is 66 and for 'b' is 98 so Ben comes before 'ben'.

How about 'Ben' and 'Benny'? The first 3 characters have the same ASCII codes. When you get to the fourth character, there isn't one for 'Ben' so it has the Null ASCII code, 0. In the other word, we get the code for 'n', which is 110. 0 < 110 so 'Ben' comes before 'Benny'.

The computer can work out the correct order of any set of letters, numbers and symbols (more properly called 'strings' by programmers) using ASCII codes.

For example, the computer can decide if '45%3#' is before or after '45##67' by first comparing the codes for the first position 4 (they are the same), then the second position 5 (they are also the same) and then % and #. % has the ASCII code 37 and # has the ASCII code 35. So 45##67 comes before 45%3#.

Note that you are always comparing the same positions of each string with each other until they are different, until the ASCII code of one character is greater or less than the ASCII code of the other character. Then you can stop as you can now put them into the correct order.

Q1. Just as you can count in denary, you can count in binary too! Have a look at this table. Two of the entries are wrong! Find and correct the mistakes.

Binary	Denary	Binary	Denary
00000000	0	11110101	245
00000001	1	11110110	246
00000010	2	11110111	247
00000011	3	11111000	248
00000100	4	11111001	249
00000101	5	11111010	250
00000110	6	11111110	251
00000111	7	11111100	252
00001001	8	11111101	253
00001001	9	11111110	254
00001010	10	11111111	255

Q2. What is meant by a 'denary number' and a 'binary number'?
Q3. The maximum number a byte can hold is 1111 1111. What is this as a denary number?
Q4. What is a 'nibble'?
Q5. How many bytes approximately are there in 20 Kbytes?
Q6. How many bytes exactly are there in 20 Kbytes?
Q7. Explain what is meant by a 'character set'.
Q8. What is the name of the error checking done using the 8th bit in Standard ASCII?
Q9. Why do we need Extended ASCII?
Q10. Explain the need for Unicode.