4 Software

In this chapter you will learn about:

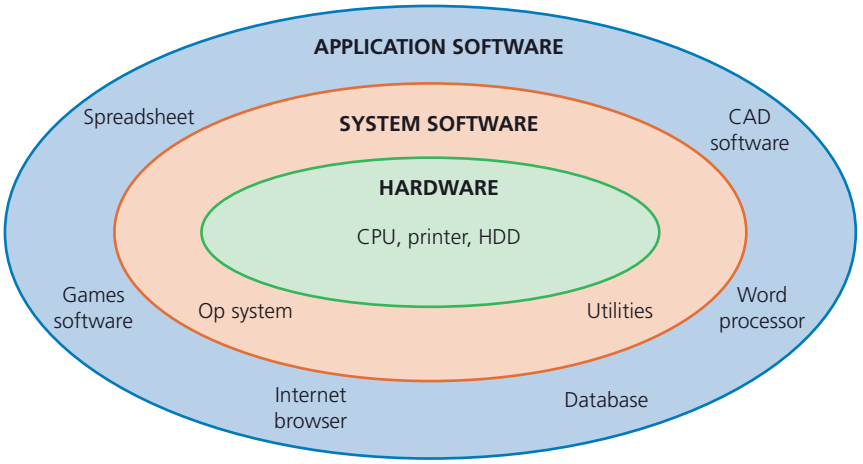
- ★ types of software and interrupts
  - the differences between systems software and applications software (including the operating system, utility programs and application software)
  - the role and basic functions of an operating system
  - the need for hardware, firmware and operating systems when running application software
  - role and operation of interrupts
- ★ types of programming language, translators and IDEs
  - advantages and disadvantages of high-level and low-level languages
  - assembly language is a low-level language that uses mnemonics and assemblers
  - the operation of compilers and interpreters for high-level languages
  - advantages and disadvantages of compilers and interpreters
  - the role and functions of integrated development environment (IDEs) when writing code.

In this chapter you will learn about some of the key software used in computer systems. The chapter will consider essential software (such as an operating system) all the way through to application software (such as word processors). The first part of the chapter will cover how the software is used, while the second part will cover how software is translated so that a computer can carry out the software's instructions.

4.1 Types of software and interrupts

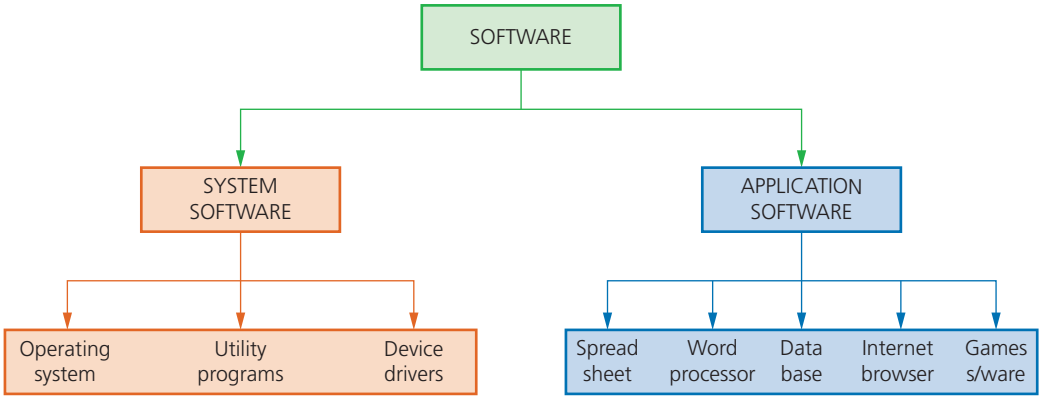
4.1.1 System software and application software

All computers begin life as a group of connected hardware items. Without software, the hardware items would be useless. This section considers the link between hardware and software. Figure 4.1 summarises the hierarchy of software and hardware.



▲ **Figure 4.1** Software and hardware hierarchy

You will notice from Figure 4.1 that there are two types of software: system software and application software:



▲ **Figure 4.2** Software types

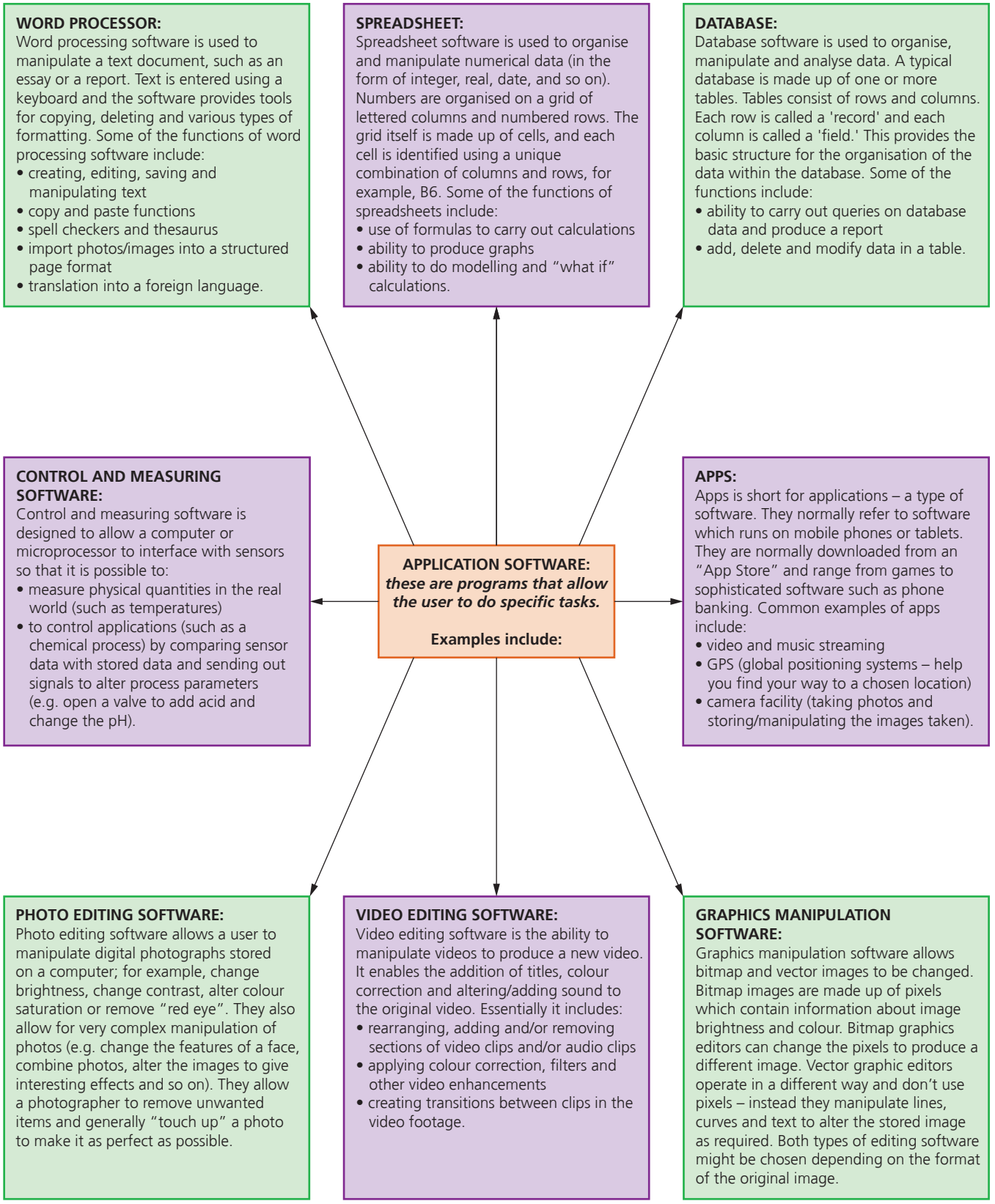
General features of system software

- set of programs to control and manage the operation of computer hardware
- provides a platform on which other software can run
- required to allow hardware and software to run without problems
- provides a human computer interface (HCI)
- controls the allocation and usage of hardware resources.

General features of application software

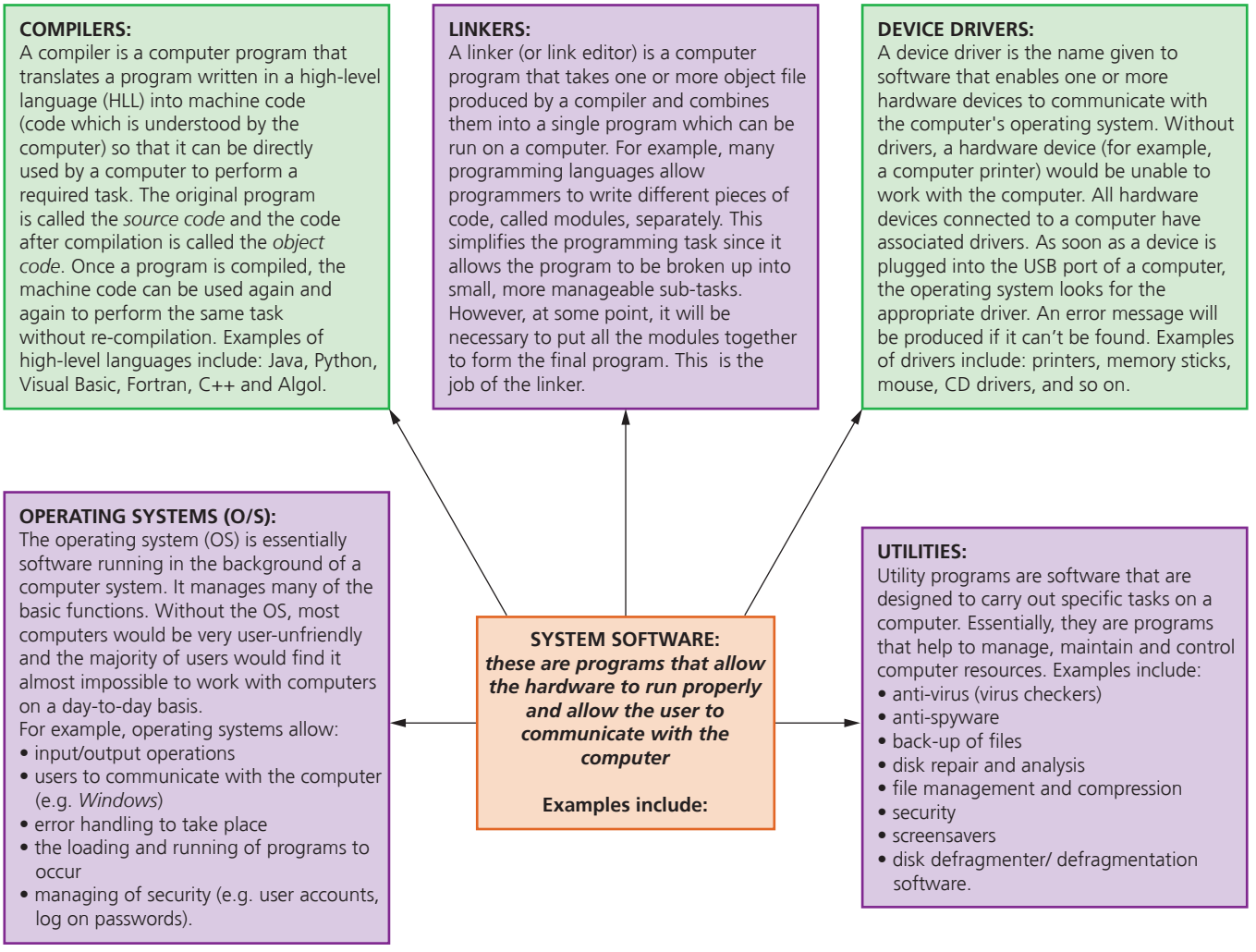
- used to perform various applications (apps) on a computer
- allows a user to perform specific tasks using the computer's resources
- may be a single program (for example, NotePad) or a suite of programs (for example, Microsoft Office)
- user can execute the software as and when they require.

Examples of typical application software



▲ **Figure 4.3** Application software

Examples of typical system software



▲ **Figure 4.4** System software

Link

Refer to Section 4.1.3 on the running of apps on a computer.

The remainder of this section considers the role of the operating system, utility programs and device drivers in much more depth. Compilers and linkers will be considered later on in this book.

Utility software (utilities)

Computer users are provided with a number of utility programs (often simply referred to as utilities) that are part of the system software.

Utility programs are often initiated by the user, but some, notably virus checkers, often just run in the background without the need for any user input. Utility programs offered by most computer system software include:

- virus checkers
- defragmentation software
- disk contents analysis and repair
- file compression and file management
- back-up software
- security
- screensavers.

Virus checkers (anti-virus software)

Any computer (including mobile phones and tablets) can be subject to a virus attack. Operating systems offer virus checkers, but these must be kept thoroughly up to date and should run in the background to maintain their ability to guard against being infected by such **malware**. There are many ways to help prevent viruses (such as being careful when downloading material from the internet, not opening files or websites given in emails from unknown senders or by not using non-original software). However, virus checkers still afford the best defence against such malware.

Link

See Section 5.3 for more on computer viruses.

Running **anti-virus software** in the background on a computer will constantly check for virus attacks. Although various types of anti-virus software work in different ways they all have the following common features:

- they check software or files before they are run or loaded on a computer
- anti-virus software compares a possible virus against a database of known viruses
- they carry out **heuristic checking** – this is the checking of software for types of behaviour that could indicate a possible virus; this is useful if software is infected by a virus not yet on the database
- any possible files or programs which are infected are put into **quarantine** which:
  - allows the virus to be automatically deleted, or
  - allows the user to make the decision about deletion (it is possible that the user knows that the file or program is not infected by a virus – this is known as a **false positive** and is one of the drawbacks of anti-virus software)
- anti-virus software needs to be kept up to date since new viruses are constantly being discovered
- full system checks need to be carried out once a week, for example, since some viruses lie dormant and would only be picked up by this full system scan.

Defragmentation software

As a HDD becomes full, blocks used for files will become scattered all over the disk surface (in potentially different sectors and tracks as well as different surfaces). This will happen because files will become deleted, partially-deleted, extended and so on over time. The consequence of this is slower data access time; the HDD read-write head will now require several movements just to find and retrieve the data making up the required file.

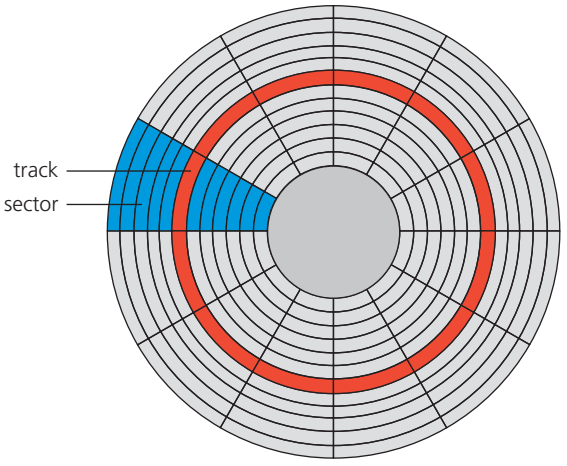
Link

Refer to Chapter 3 for more detail on how data is stored on a hard disk drive (HDD).

It would obviously be advantageous if files could be stored in **contiguous** sectors considerably reducing HDD head movements. (Note that due to the different operation of SSDs when accessing data, this is not a problem when using solid state devices.)

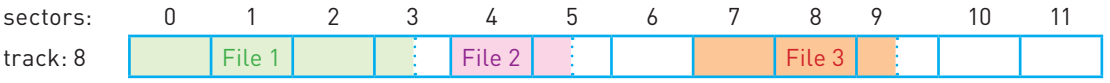
contiguous means 'next to each other'

Consider the following scenario using a disk with 12 (numbered 0 to 11) sectors per surface:

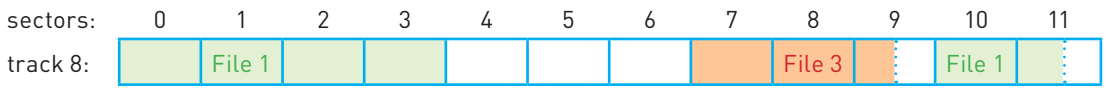


▲ **Figure 4.5** Hard disk drive tracks and sectors

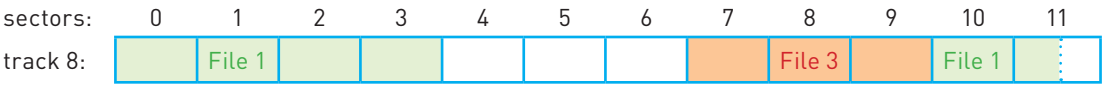
In this example we have three files (1, 2 and 3) stored on track 8 of a disk surface covering all 12 sectors:

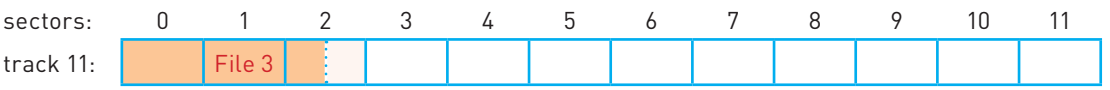


Now file 2 is deleted by the user and file 1 has data added to it; however, the file 2 sectors which become vacant are not filled up straight away by new file 1 data since this would require 'too much effort' for the HDD resources; we now get the following (file 1 is now stored in sectors 0, 1, 2, 3, 10 and 11):

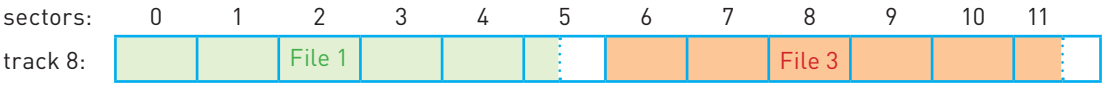


File 1 has now been extended to write data is sectors 10 and 11; now suppose file 3 is extended with the equivalent of 3 blocks of data; this now requires filling up sector 9 and then finding some empty sectors to write the remainder of the data – suppose the next free sectors are on track 11:





If this continues, the files just become more and more scattered throughout the disk surfaces. It is possible for sectors 4, 5 and 6 (on track 8) to eventually become used if the disk starts to fill up and it has to use up whatever space is available. A **disk defragmenter** will rearrange the blocks of data to store files in **contiguous** sectors wherever possible. After defragmentation Track 8 would now become:



This obviously allows for much faster data access and retrieval since the HDD will now require fewer read-write head movements to access and read files 1 and 3. Track 11 would be empty after the defragmentation process.

Back-up software

While it is sensible to take manual back-ups using, for example, a memory stick or portable HDD, it is also good practice to use the operating system **back-up utility**. This utility will:

- allow a schedule for backing up files to be made
- only carry out a back-up procedure if there have been any changes made to a file.

For total security there should be three versions of a file:

1. the current (working) version stored on the internal HDD or SSD
2. a locally backed up copy of the file (stored on a portable SSD, for example)
3. a remote back-up version stored well away from the computer (for example, using cloud storage).

The Microsoft Windows environment offers the following facilities using the back-up utility:

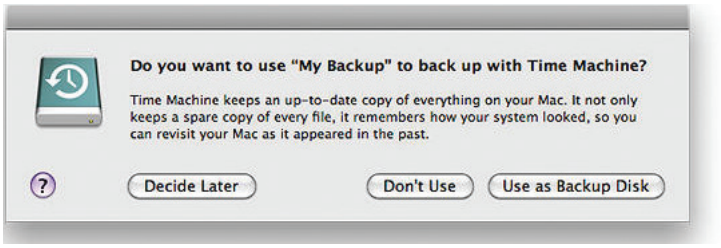
- restore data, files or the computer from the back-up (useful if there has been a problem and files have been lost and need to be recovered)
- create a restore point (this is basically a kind of 'time machine' where your computer can be restored to its state at this earlier point in time; this can be very useful if a very important file has been deleted and can't be recovered by any of the other utilities)
- options of where to save back-up files; this can be set up from the utility to ensure files are automatically backed up to a chosen device.

Windows uses **File History**, which takes snapshots of files and stores them on an external HDD at regular intervals. Over a period of time, **File History** builds up a vast library of past versions of files – this allows a user to choose which version of the file they want to use. **File History** defaults to backing up every hour and retains past versions of files for ever unless the user changes the settings.

Mac OS offers the **Time Machine** back-up utility. This erases the contents of a selected drive and replaces them with the contents from the back-up. To use this facility, it is necessary to have an external HDD or SSD (connected via USB port) and ensure that the Time Machine utility is installed and activated on the selected computer. Time machine will automatically:

- back-up every hour
- do daily back-ups for the past month, and
- weekly back-ups for all the previous months.

(Note: once the back-up HDD or SSD is almost full, the oldest back-ups are deleted and replaced with the newest back-up data.) The following screen shows the Time Machine message:



▲ **Figure 4.6** Time machine message on Mac OS

Security software

Security software is an over-arching utility that:

- manages access control and user accounts (using user IDs and passwords)
- links into other utility software, such as virus checkers and spyware checkers
- protects network interfaces (for example, through the use of firewalls)
- uses encryption and decryption to ensure any intercepted data is meaningless without a decryption key
- oversees the updating of software (does the update request come from a legitimate source, for example).

Link

Refer back to Section 2.3 for more information on encryption and decryption.

Screensavers

Screensavers are programs that supply moving and still images on the monitor screen after a period of inactivity by the computer. They were originally developed to protect older CRT (cathode ray tube) monitors which would suffer from 'phosphor burn' if the same screen image remained for any length of time. With modern LCD and OLED screens, this problem no longer exists; consequently, screensavers are now mostly just a way of customising a device. However, many screensavers are also used as part of the computer's security system. If a computer is unused for five minutes, for example, and hasn't been logged out, this will trigger the screensaver to be loaded. The computer user will then be automatically logged out and a screensaver will indicate that the computer is now locked. This gives an extra layer of security for computers used in an office environment, for example.

Some screensavers are often used to activate useful background tasks that can only go on when the computer is in an 'idle' state. For example:

- virus scans
- distributed computing applications – these allow apps to use the computer's resources only when it is idle (for example, an online gaming app).

Device drivers

Device drivers are software that communicate with the operating system and translate data into a format understood by a hardware peripheral device. Without device drivers, a hardware device would be unable to work with a computer – a message such as 'device not recognised' would appear on the screen. As soon as a device is plugged into a USB port (for example, a memory stick, printer or camera), the operating system looks for the appropriate device driver.

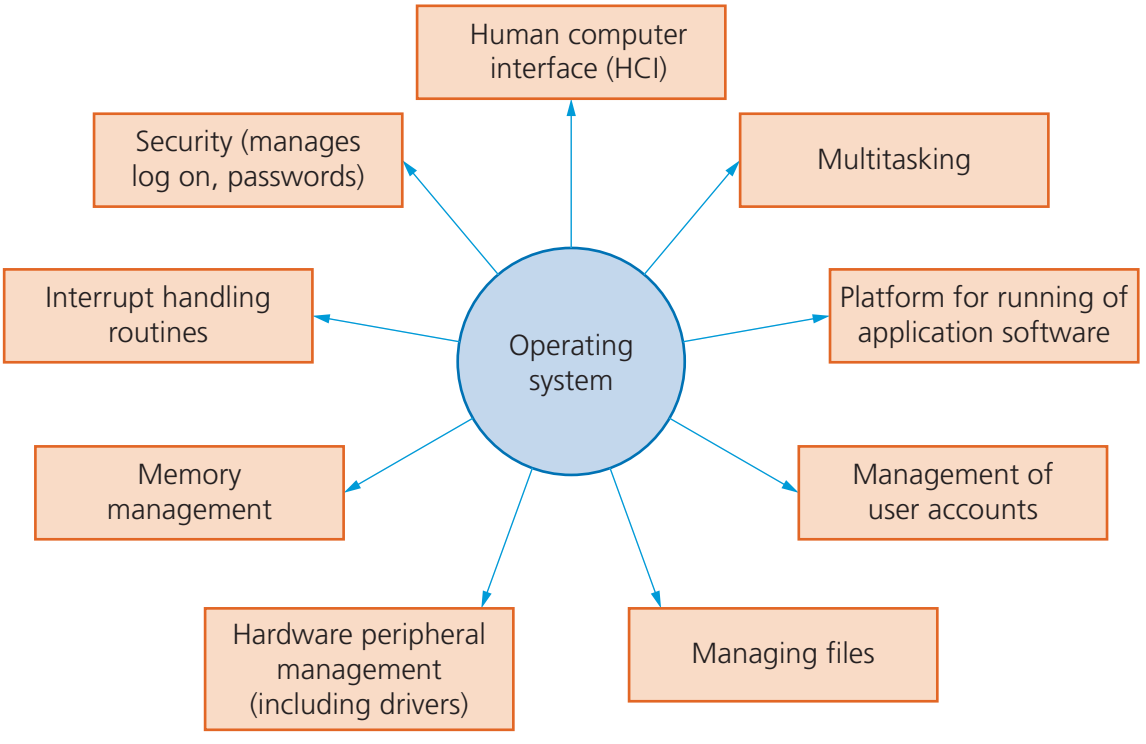
All USB device drivers contain a collection of information about devices called **descriptors**; this allows the USB bus to ask a newly connected device what it is. Descriptors include vendor id (VID), product id (PID) and unique serial numbers. If a device has no serial number associated with it, the operating system will treat the device as new every time it is plugged into a USB port. Serial numbers must be unique since this could prove rather interesting if two different devices with the same serial number were plugged into a computer at the same time.

4.1.2 Operating systems

To enable computer systems to function correctly and allow users to communicate with computer systems, software known as an **operating system** needs to be installed. An operating system provides both the environment in which applications can be run and a useable interface between humans and computer. An operating system also disguises the complexity of computer software and hardware. Common examples of operating systems include: Microsoft Windows, Apple Mac OS, Google Android and Apple IOS (the latter two being used primarily on tablets and smartphones).

Most computers store the operating system on a hard disk drive (HDD) or solid state drive (SSD) since they tend to be very large programs. Mobile phones and tablets store the operating system on a solid state device since they are too small to accommodate an HDD.

Figure 4.7 summarises some of the functions in a typical operating system.

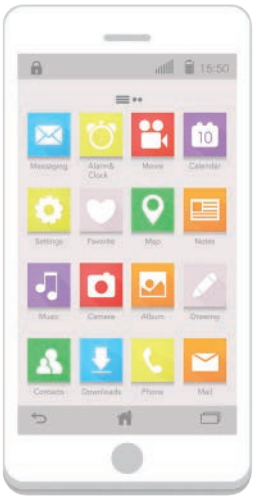


▲ **Figure 4.7** Operating system functions

The next section describes each of the nine functions shown in Figure 4.7.

Human computer interface (HCI)

The **human computer interface (HCI)** is in the form of a **Command Line Interface (CLI)** or a **Graphical User Interface (GUI)**.



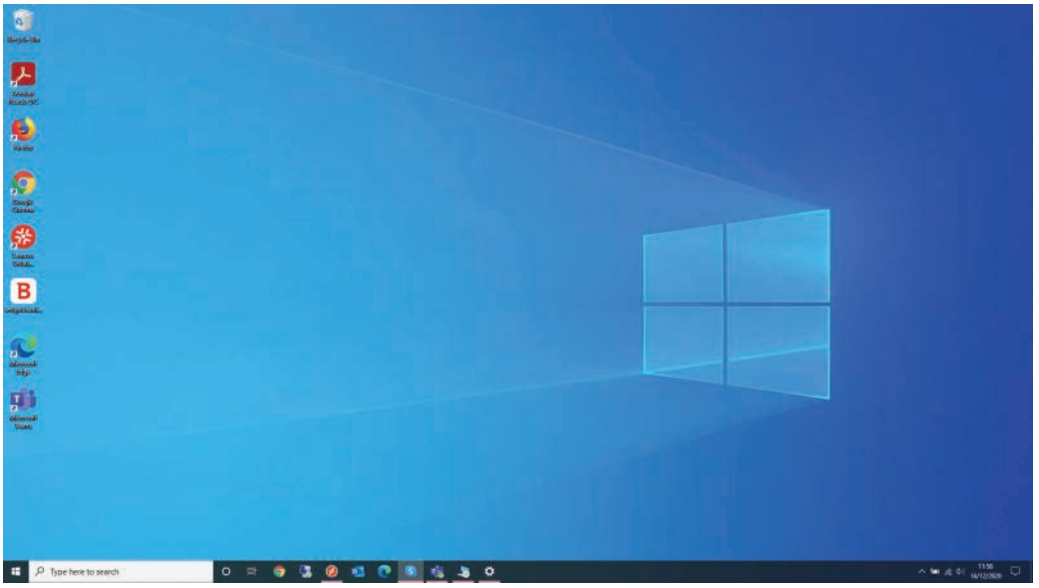
▲ **Figure 4.8** GUI icons on a mobile phone

A CLI requires a user to type in instructions in order to choose options from menus, open software, etc. There are often a number of commands that need to be typed in, for example, to save or load a file. The user has to therefore learn a number of commands just to carry out basic operations. It is also slow having to key in these commands every time an operation has to be carried out. However, the advantage of CLI is that the user is in direct communication with the computer and is not restricted to a number of pre-determined options.

A GUI allows the user to interact with a computer (or MP3 player, gaming device, mobile phone, etc.) using pictures or symbols (icons) rather than having to type in a number of commands. Figure 4.8 shows a mobile screen with a number of GUI icons.

Simply selecting any of the icons from the screen would automatically load the application into the phone ready to be used. There is no need to type in anything.

GUIs use various technologies and devices to provide the user interface. One of the most common is **WIMP (windows icons menu and pointing device)**, which was developed for use on personal computers (PC). Here a mouse is used to control a cursor and icons are selected to open/run windows. Each window contains an application and modern computer systems allow several windows to be open at the same time. An example is shown below (here a number of icons can be seen on the left-hand side and on the bottom):



▲ **Figure 4.9** Windows screen showing icons

A windows manager looks after the interaction between windows, the applications, the pointing devices and the cursor's position.

More recently, devices such as mobile phones and tablets increasingly use touch screens and use **post-WIMP** interactions. With this system, fingers are in contact with the screen allowing actions such as pinching and rotating, which would be difficult to do using a single pointer and a device such as a mouse.

▼ **Table 4.1** Differences between GUI and CLI interfaces

Interface	Advantages	Disadvantages
command line interface (CLI)	the user is in direct communication with the computer the user is not restricted to a number of pre-determined options it is possible to alter computer configuration settings uses a small amount of computer memory	the user needs to learn a number of commands to carry out basic operations all commands need to be typed in which takes time and can be error-prone each command must be typed in using the correct format, spelling, and so on
graphical user interface (GUI)	the user doesn't need to learn any commands it is more user-friendly; icons are used to represent applications a pointing device (such as a mouse) is used to click on an icon to launch the application – this is simpler than typing in commands or a touch screen can be used where applications are chosen by simply touching the icon on the screen	this type of interface uses up considerably more computer memory than a CLI interface the user is limited to the icons provided on the screen needs an operating system, such as Windows, to operate, which uses up considerable memory

Who would use each type of interface?

- CLI: a programmer, analyst or technician; basically somebody who needs to have a direct communication with a computer to develop new software, locate errors and remove them, initiate memory dumps (contents of the computer memory at some moment in time), and so on
- GUI: the end-user who doesn't have or doesn't need to have any great knowledge of how the computer works; a person who uses the computer to run software or play games or stores/manipulates photographs, for example.

Memory management

Memory management carries out the following functions:

- manages the primary storage (RAM) and allows data to be moved between RAM and HDD/SSD during the execution of programs
- keeps track of all the memory locations
- carries out memory protection to ensure that two competing applications cannot use the same memory locations at the same time. If this wasn't done the following might happen:
  - data would probably be lost
  - applications could produce incorrect results (based on the wrong data being in memory locations)
  - potential security issues (if data is placed in the wrong location, it might make it accessible to other software, which would be a major security issue)
  - in extreme cases, the computer could crash.

Security management

Security management is another part of a typical operating system; the function of security management is to ensure the integrity, confidentiality and availability of data. This can be achieved as follows (many of these features are covered in more depth elsewhere in this book):

Link

See Section 5.3 for more on cyber security.

- by carrying out operating system updates as and when they become available
- ensuring that anti virus software (and other security software) is always up to date, preserving the integrity, security and privacy of data
- by communicating with, for example, a firewall to check all traffic to and from the computer
- by making use of privileges to prevent users entering 'private areas' on a computer that permits multi-user activity (this is done by setting up user accounts and making use of passwords and user IDs); this helps to ensure the privacy of data
- by maintaining access rights for all users
- by offering the ability for the recovery of data (and system restore) when it has been lost or corrupted
- by helping to prevent illegal intrusion into the computer system (also ensuring the privacy of data).

Find out more

By checking out the remainder of this chapter and Chapter 5, find out the methods available to ensure the security, privacy and integrity of data and how these link into the operating system security management. It is important to distinguish between what constitutes security, privacy and integrity of data.

Hardware peripheral management

Hardware management involves all input and output peripheral devices. Hardware management:

Link

See Section 4.1.1 for more information on drivers.

- communicates with all input and output devices using device drivers
- uses a device driver to take data from a file (defined by the operating system) and translates it into a format that the input/output device can understand
- ensures each hardware resource has a priority so that they can be used and released as required
- manages input/output devices by controlling queues and **buffers**; consider the role of the printer management when printing out a document:
  - first of all, the printer driver is located and loaded into memory
  - then the data is sent to a printer buffer ready for printing
  - if the printer is busy (or the printing job has a low priority) then the data is sent to a printer queue before it can be sent to the printer buffer
  - it will send various control commands to the printer throughout the printing process
  - it receives and handles error messages and interrupts from the printer.

Find out more

Find out about the tasks carried out by a Keyboard Manager when a user is typing in the text to a word processor. Consider the use of buffers and queues in your answer.

You may need to do some research throughout this book to find out how the Keyboard Manager works.

File management

The main tasks of **file management** include:

file name | extension

- file naming conventions which can be used i.e. filename.docx (where the extension can be .bat, .htm, .dbf, .txt, .xls, etc.)
- performing specific tasks (for example, create, open, close, delete, rename, copy, and move)
- maintaining the directory structures
- ensuring access control mechanisms are maintained (for example, access rights to files, password protection, or making files available for editing or locking them)
- ensuring memory allocation for a file by reading it from the HDD/SSD and loading it into memory.

Interrupts

Please refer to Section 4.1.4 for a discussion on interrupts.

Platform for running of application software

Please refer to Section 4.1.3 for a discussion on the running of application software.

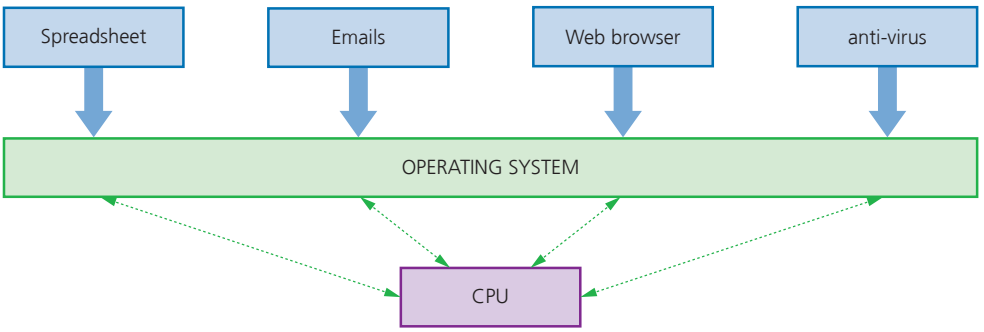
Multitasking

Multitasking allows computers to carry out more than one task (i.e. a process) at a time. Each of the processes will share the hardware resources under the control of the operating system software. To make sure that multitasking operates correctly (in other words, the processes don't clash with each other), the operating system needs to constantly monitor the status of each of the processes under its control:

- resources are allocated to a process for a specific time limit
- the process can be interrupted while it is running
- the process is given a priority so it can have resources according to its priority (the risk here is that a low priority process could be starved of resources).

these three bullet points are called pre-emptive multitasking

Using multitasking management, main memory, HDD/SSD and virtual memory are better managed making the most effective use of CPU time.



▲ **Figure 4.10** Multitasking diagram

Management of user accounts

Computers allow more than one user to log onto the system. It is therefore important that users' data is stored in separate parts of the memory for security reasons (also refer to security management earlier in this section). Each person logging onto the computer will be given a user account protected by a user name and password. The operating system is given the task of managing these different user accounts. This allows each user to:

- customise their screen layout and other settings
- use separate folders and files and to manage these themselves.

Very often an **administrator** oversees the management of these user accounts. The administrator can create accounts, delete user accounts and restrict user account activity. On large university or industrial computers, part of the operating system's tasks will be to oversee several users' accounts, since a complex multi-user system may be in place. The operating system has to maintain accounts for several users, managing data that may range from personal data and technical research work down to the ordering of stationery. Multi-access levels permit this control to take place. For example, a clerk in the office may have access to ordering stationery but can't have access to any personal data.

4.1.3 Running of applications

This section will bring together some of the topics covered in Section 4.1.2. As mentioned earlier, application software requires the operating system to provide a platform on which the software can run successfully.

When a computer starts up, part of the operating system needs to be loaded into RAM – this is known as **booting up** the computer (or a **bootstrap loader**). The start-up of the computer's motherboard is handled by the basic input/output system (BIOS). The BIOS tells the computer where the storage device that holds the operating system can be found; it then loads the part of the operating system that is needed and executes it.

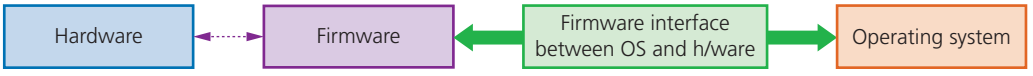
The BIOS is often referred to as **firmware**. Firmware is defined as a program that provides low level control for devices.

Advice

EEPROM is included to fully explain the BIOS, but details about EEPROM go beyond the requirements of the syllabus.

The BIOS program is stored in a special type of ROM, called an **EEPROM** (Electrically Erasable Programmable ROM). EEPROM is a flash memory chip, which means its contents remain even when the computer is powered down. However, it also means the BIOS can be rewritten, updated or even deleted by a user.

However, while the BIOS is stored on an EEPROM, the BIOS **settings** are stored on a CMOS chip (Complementary Metal Oxide Semi-conductor). The CMOS is powered up at all times via a rechargeable battery on the motherboard. Therefore, the BIOS settings would be reset if the battery was removed or disconnected for some reason. Once the CMOS is re-started, it will access the same BIOS program from EEPROM, but the settings will now be the default factory settings. Consequently, if a user has changed the BIOS settings (for example, the clock speed), the settings will revert to those settings made at the factory once power is restored to the CMOS.



▲ **Figure 4.11** Firmware interface between OS and hardware

The application software will be under the control of the operating system and will need to access system software such as the device drivers while it is running. Different parts of the operating system may need to be loaded in and out of RAM as the software runs.

4.1.4 Interrupts

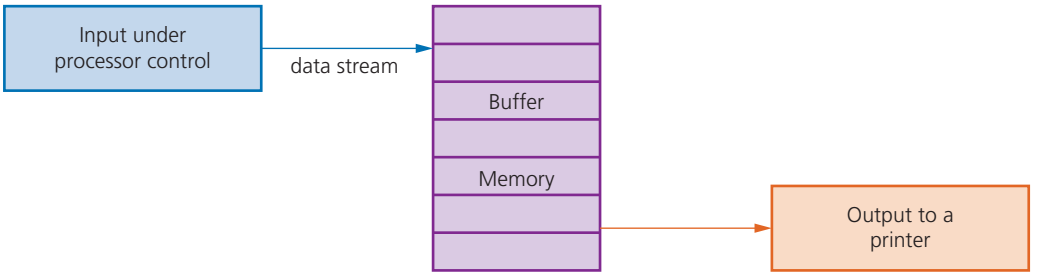
An **interrupt** is a signal sent from a device or from software to the microprocessor. This will cause the microprocessor to temporarily stop what it is doing so that it can service the interrupt. Interrupts can be caused by:

- a timing signal
- an input/output process (for example, a disk drive or printer requiring more data)
- a hardware fault (for example, a paper jam in the printer)
- user interaction (for example, the user presses a key (or keys) on a keyboard, such as <CTRL><ALT><BREAK>, which causes the system to be interrupted)
- software errors that cause a problem (for example, an .exe file that cannot be found to initiate the execution of a program, two processes trying to access the same memory location, or an attempt to divide by zero).

Once the interrupt signal is received, the microprocessor either carries on with what it was doing or stops to service the device or program that caused the interrupt. The computer needs to identify the interrupt type and also establish the level of **interrupt priority**.

Interrupts allow computers to carry out many tasks or to have several windows open at the same time. An example would be downloading a file from the internet at the same time as listening to some music from a library. Interrupts allow these two functions to co-exist and the user has the impression that both functions are being carried out simultaneously. In reality, data is being passed in and out of memory very rapidly allowing both functions to be serviced. This can all be achieved by using an area in memory known as a **buffer**. A buffer is a memory area that stores data temporarily (see Figure 4.12). For example, buffers

are used when downloading a movie from the internet to compensate for the difference between download speeds and the data requirements of the receiving device. The data transmission rate of the movie file from the web server to the buffer must be greater than the rate at which data is transmitted from buffer to media player. Without buffers, the movie would frequently 'freeze'.



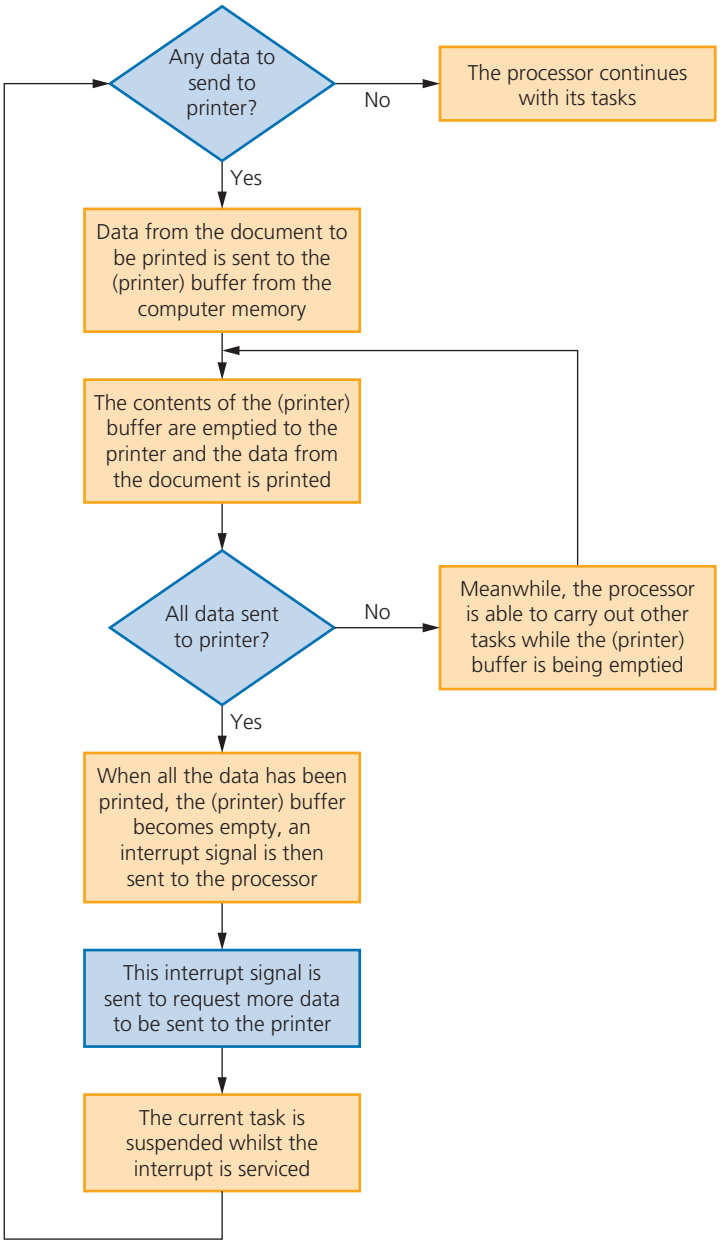
▲ **Figure 4.12** Use of a buffer when sending data to a printer (buffer used to store data temporarily since printer speed is much slower than microprocessor speed)

Find out more

Find out how buffers are used to stream movies from the internet to a device (such as a tablet).

Whenever an interrupt is received it needs to be **serviced**. The status of the current task being run first needs to be saved. The contents of the Program Counter (PC) and other registers are saved. Then the **interrupt service routine (ISR)** is executed by loading the start address into the Program Counter (PC). Once the interrupt has been fully serviced, the status of the interrupted task is reinstated (the contents of all the saved registers are then retrieved) and the process continues.

Buffers and interrupts are often used together to allow standard computer functions to be carried out. These functions are often taken for granted by users of modern computer systems. For example, the following diagram (Figure 4.13) shows how buffers and interrupts are used when a document is sent from memory to a printer. The important thing to remember here is the time taken to print out a document is ***much*** longer than the time it takes for the microprocessor to send data to the printer. Without buffers and interrupts, the microprocessor would remain idle waiting for a document to be printed. This would be an incredible waste of microprocessor time; the buffers and interrupts allow the microprocessor to carry on with other tasks while the document is being printed thus maximising its processing power and speed.



▲ **Figure 4.13** Use of interrupts and buffers when printing a document

Find out more

Try to produce a flow chart (similar to Figure 4.13) that shows the role of buffers and interrupts when the memory sends data to a disk drive for storage.

Remember that the time to write data to disk is much longer than the time it takes for the microprocessor to carry out its tasks.

Activity 4.1

1 Tick (✓) the appropriate column in the following table to indicate whether the named software is system software or application software.

Software	System software	Application software
screensaver		
anti-virus software		
control and measurement software		
printer driver		
video editing software		
compiler		
QR code reader		
on-screen calculator		
operating system software		

2 Mike is downloading a video from the internet to his laptop. The speed of data transfer from the internet is slower than the speed at which data is being sent to the media player.

- **a** What could be used to stop the video constantly freezing while Mike is watching it on his laptop?
- **b** While watching the video, Mike is meanwhile printing a 160-page document on his inkjet printer. Describe how interrupts could be used to allow him to watch his movie at the same time as the printing is being done. The printer memory can store up to 20 pages at a time.
- **c** Describe what happens if the inkjet printer runs out of black ink during the printing process.

3 Choose four features of an operating system and describe their function.

4 What is meant by a **descriptor** in a device driver? What role does the descriptor play when a new memory stick, for example, is plugged into a USB port of a computer for the first time?

5 **a** Describe what is meant by a ***BIOS*** and state its function. What is the task of a BIOS when a computer is first powered up?

b BIOS software and BIOS settings are different. Describe the different types of memory needs for both the software and its settings. In your explanation state why both types of memory are used.

6 Seven descriptions are shown on the left and seven computer terms are shown on the right. Draw lines to connect each description to the correct computer term.

when a computer starts up, OS is loaded into RAM	firmware
software that communicates with the OS and translates data into a format that can be understood by an I/O device	printer driver
computer carrying out many different processes at the same time	bootstrap loader
program that provides low level control for devices including embedded systems	interrupt
program that supplies static or moving images on a monitor when a computer has been idle for a period of time	screensaver
signal from a device or software sent to a microprocessor to temporarily halt the process currently being carried out	buffer
memory area used to hold data temporarily	multitasking