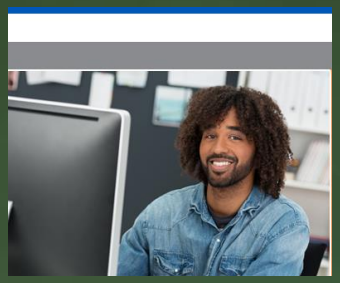
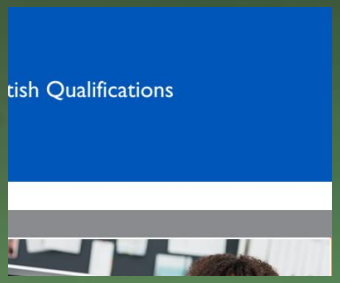
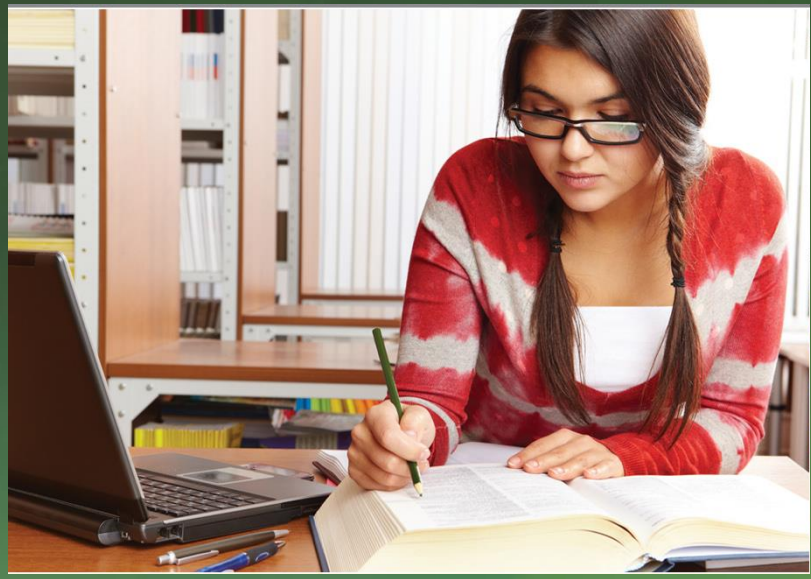
Awarding Great British Qualifications







Computer Systems

Topic 6:

Software, Installation and Configuration

Scope and Coverage

This topic will cover:

- **Systems** software
  - Utility software
  - Translation & compilation software
- **Applications** software
  - Standard packages
  - Customised packages
- Programming **Languages**
  - Types and features

Learning Outcomes

By the end of this topic, students will be able to:

- Explain the hardware, software and peripheral components of a computer system
- Build and configure a computer system to meet a design specification

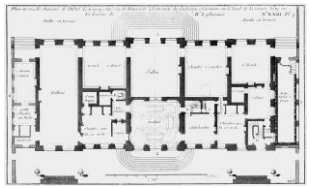
What is Software?

- Software is not a tangible ‘thing’ that you can see or touch - like computer hardware is
- Software is the *collection of instructions* telling the computer hardware what to do
- Software is written using a computer *language*
- There are different categories of software and many different types of languages
- This presentation covers software and languages

Software in Context

- Cooking
  - Hardware is the oven, the pots & pans etc.
  - Software is the **recipe** (specifies ingredients, temperatures etc.)

- Construction
  - Hardware is the bricks, concrete, windows etc.
  - Software is the **architect plan** (no. of rooms, location of doors etc)



- Music
  - Hardware is the instruments, musicians etc.
  - Software is the **composer's sheet music/score**

Think of software as the **logical** set of instructions that dictate how **physical** things behave.

Software Categories

- Software has the following classifications:
  - **System** software

    - Responsible for direct control of the computer (the ‘system’)
    - Looks ‘inwards’ towards controlling the computer
    - A ‘resource manager’ responsible for behaviour of the computer
    - Main piece of system software is the **operating system**
    - Includes **utility software** (anti-virus, firewalls, compression etc.)
  - **Application** software

    - Looks ‘outwards’ to the external world, not the computer
    - Designed to solve a ‘real-world’ problem, need or application
    - Examples include: word processors, databases, spreadsheets

Systems Software

It’s all about computers!

Systems Software

- As we have seen in previous presentations, there are many types of computer system
- They all need instructions in order to work
- These instructions focus on internal control and how that system will interface to the external world
- Such instructions are called the **operating system**
- So, for a typical PC or laptop, the operating system will be Microsoft Windows, Unix, Linux or Apple's macOS

Systems Software

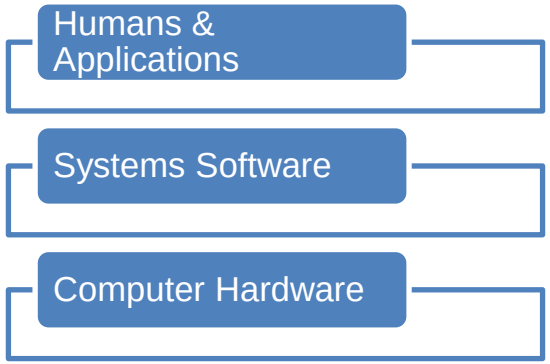
- Other types of system software include:
  - User Interfaces
  - Utility software
  - Security software
  - Communication software

Systems Software

- All systems software has one thing in common:
  - *The secure and efficient operation of the host computer*
- Operating systems are not interested in solving the user's needs or want's, they serve the computer
- Utility software is still primarily focussed on the host computer (like security tools) but does offer some features for the human user (like file compression)
- Neither the O/S or utility software is interested in solving 'real-world' problems (spreadsheets, email etc.)

Systems Software

- Systems software is all about the ‘system’
- Systems software is all about looking inwards
- Systems software is all about resource management
- Systems software provides a firm platform for the human user and applications (email, databases etc.)



Applications Software

It’s all about people!

Applications Software

- An ‘Application’ or ‘App’ is a specialised piece of software that *aims to satisfy or fulfil a real-world human need*
- An App is not interested in running the actual computer – that is the job of systems software
- Applications look ‘outwards’ to the world
- System software looks ‘inwards’ to the computer

Applications Software

- Examples of application software include:
  - Word processors
  - Spreadsheets
  - Databases
  - Email
  - Web browsers
  - Calendars and scheduling
  - Dating applications
  - E-commerce applications

Satisfy *human* needs

Application Software: 3 Options

- Three approaches to application software:
  - Option A: **Standard** package
    - Commercial-off-the-Shelf (COTS)
  - Option B: **Bespoke** software
    - Niche, specialised, unique, start from zero
  - Option C: **Customised** software
    - Extend existing software to your needs

Option A

- Commercial-off-The-Shelf (COTS) software
- Most PC applications are standard packages
- Do commonly required tasks needed by millions:
  - Word processing
  - Spreadsheets
  - Databases
  - Email
- Sold to customers as stand alone packages or ‘application suites’ – such as Microsoft Office

Microsoft® Office

Option A

- The same package may be used by thousands or even millions of people around the world
- Economies of scale result in a (relatively) low price for each individual copy
- The recent trend has been **cloud-based** 'utility' software where end-users download the application as needed from a remote 'cloud' platform
- The '*Software as a Service*' approach
- For example, Microsoft Office 365

Option B

- Another option is **Bespoke Software**
- Meets a very niche or specific requirement of a very small group of users - often just one company
- May be written in-house by the company that needs the software or by a specialist third-party software house
- Very narrow application areas with little or no potential to sell the package to other users
- Hence, can be very expensive
- But it may be the only way to get a suitable application

Option C

- Built on a standard package - often a database
- Then **customised** with added functionality for the specific application area
- May be minor modifications to fit the company's requirements or may be major alterations leaving little of the original package apparent
- Examples:
  - Financial application built on Oracle database
  - HR application built on IBM DB2 database

Open Source Software

- Most software (particularly COTS software) is written and then sold by companies as a commercial product – to make money!
- The source code is owned and controlled by them – it is **proprietary** or **closed-source** software. It is a privately-owned, commercial product.
- Examples include: MS Office, Oracle databases, Internet Explorer etc.
- The end-customer buys this product but is not allowed to modify it in any way or have access to the underlying source code – that is a secret
- It gives the software provider maximum control and, in large organisations with a single dominant software supplier, it can lead to the client being 'locked-in' to that software company.
- This could lead to unfair or disagreeable conditions being placed on the client – as they are so deep into this software, it is not feasible to back out of it

Open Source Software

- However, there are a large number of (very good) programmers who write software for their own entertainment - they just like writing code
- This led to some very useful software becoming available **for free**
- This in turn gave rise to the **open source** software movement
- This means that the source code is published and can be freely copied and modified as long as the same rule is applied to the new version
- Usually distributed via the Internet
- Most packages are supported and developed by teams of programmers across the world and the Internet
- Examples include:
  - Linux (https://opensource.com/resources/linux)
  - Apache Open Office (https://www.openoffice.org/)
  - LibreOffice (https://www.libreoffice.org/)
  - Open Stack (https://opensource.com/resources/what-is-openstack)

Open Source Software

See:

- https://opensource.com/resources/what-open-source
- https://opensource.org/
- https://www.gnu.org/software/software.en.html