2b. Content of Computer systems (J277/01)

1.1 – Systems architecture

Sub topic	Guidance
1.1.1 Architecture of the CPU	
☐ The purpose of the CPU: o The fetch-execute cycle ☐ Common CPU components and their function: o ALU (Arithmetic Logic Unit) o CU (Control Unit) o Cache o Registers ☐ Von Neumann architecture: o MAR (Memory Address Register) o MDR (Memory Data Register) o Program Counter o Accumulator	**Required** ✓ What actions occur at each stage of the fetch-execute cycle ✓ The role/purpose of each component and what it manages, stores, or controls during the fetch-execute cycle ✓ The purpose of each register, what it stores (data or address) ✓ The difference between storing data and an address **Not required** × Knowledge of passing of data between registers in each stage
1.1.2 CPU performance	
☐ How common characteristics of CPUs affect their performance: o Clock speed o Cache size o Number of cores	**Required** ✓ Understanding of each characteristic as listed ✓ The effects of changing any of the common characteristics on system performance, either individually or in combination
1.1.3 Embedded systems	
☐ The purpose and characteristics of embedded systems ☐ Examples of embedded systems	**Required** ✓ What embedded systems are ✓ Typical characteristics of embedded systems ✓ Familiarity with a range of different embedded systems

1.2 – Memory and storage

Sub topic	Guidance
1.2.1 Primary storage (Memory)	
☐ The need for primary storage ☐ The difference between RAM and ROM ☐ The purpose of ROM in a computer system ☐ The purpose of RAM in a computer system ☐ Virtual memory	**Required** ✓ Why computers have primary storage ▪ How this usually consists of RAM and ROM ✓ Key characteristics of RAM and ROM ✓ Why virtual memory may be needed in a system ✓ How virtual memory works ▪ Transfer of data between RAM and HDD when RAM is filled
1.2.2 Secondary storage	
☐ The need for secondary storage ☐ Common types of storage: o Optical o Magnetic o Solid state ☐ Suitable storage devices and storage media for a given application ☐ The advantages and disadvantages of different storage devices and storage media relating to these characteristics: o Capacity o Speed o Portability o Durability o Reliability o Cost	**Required** ✓ Why computers have secondary storage ✓ Recognise a range of secondary storage devices/media ✓ Differences between each type of storage device/medium ✓ Compare advantages/disadvantages for each storage device ✓ Be able to apply their knowledge in context within scenarios **Not required** × Understanding of the component parts of these types of storage

Sub topic	Guidance
1.2.3 Units	
☐ The units of data storage: o Bit o Nibble (4 bits) o Byte (8 bits) o Kilobyte (1,000 bytes or 1 KB) o Megabyte (1,000 KB) o Gigabyte (1,000 MB) o Terabyte (1,000 GB) o Petabyte (1,000 TB) ☐ How data needs to be converted into a binary format to be processed by a computer ☐ Data capacity and calculation of data capacity requirements	**Required** ✓ Why data must be stored in binary format ✓ Familiarity with data units and moving between each ✓ Data storage devices have different fixed capacities ✓ Calculate required storage capacity for a given set of files ✓ Calculate file sizes of sound, images and text files ▪ sound file size = sample rate x duration (s) x bit depth ▪ image file size = colour depth x image height (px) x image width (px) ▪ text file size = bits per character x number of characters **Alternatives** • Use of 1,024 for conversions and calculations would be acceptable • Allowance for metadata in calculations may be used
1.2.4 Data storage	
Numbers ☐ How to convert positive denary whole numbers to binary numbers (up to and including 8 bits) and vice versa ☐ How to add two binary integers together (up to and including 8 bits) and explain overflow errors which may occur ☐ How to convert positive denary whole numbers into 2-digit hexadecimal numbers and vice versa ☐ How to convert binary integers to their hexadecimal equivalents and vice versa ☐ Binary shifts	**Required** ✓ Denary number range 0 – 255 ✓ Hexadecimal range 00 – FF ✓ Binary number range 00000000 – 11111111 ✓ Understanding of the terms 'most significant bit', and 'least significant bit' ✓ Conversion of any number in these ranges to another number base ✓ Ability to deal with binary numbers containing between 1 and 8 bits ▪ e.g. 11010 is the same as 00011010 ✓ Understand the effect of a binary shift (both left or right) on a number ✓ Carry out a binary shift (both left and right)

Sub topic	Guidance
Characters ☐ The use of binary codes to represent characters ☐ The term 'character set' ☐ The relationship between the number of bits per character in a character set, and the number of characters which can be represented, e.g.: o ASCII o Unicode **Images** ☐ How an image is represented as a series of pixels, represented in binary ☐ Metadata ☐ The effect of colour depth and resolution on: o The quality of the image o The size of an image file **Sound** ☐ How sound can be sampled and stored in digital form ☐ The effect of sample rate, duration and bit depth on: o The playback quality o The size of a sound file	**Required** ✓ How characters are represented in binary ✓ How the number of characters stored is limited by the bits available ✓ The differences between and impact of each character set ✓ Understand how character sets are logically ordered, e.g. the code for 'B' will be one more than the code for 'A' ✓ Binary representation of ASCII in the exam will use 8 bits **Not required** × Memorisation of character set codes **Required** ✓ Each pixel has a specific colour, represented by a specific code ✓ The effect on image size and quality when changing colour depth and resolution ✓ Metadata stores additional image information (e.g. height, width, etc.) **Required** ✓ Analogue sounds must be stored in binary ✓ Sample rate – measured in Hertz (Hz) ✓ Duration – how many seconds of audio the sound file contains ✓ Bit depth – number of bits available to store each sample (e.g. 16-bit)
1.2.5 Compression	
☐ The need for compression ☐ Types of compression: o Lossy o Lossless	**Required** ✓ Common scenarios where compression may be needed ✓ Advantages and disadvantages of each type of compression ✓ Effects on the file for each type of compression **Not required** × Ability to carry out specific compression algorithms

1.3 – Computer networks, connections and protocols

Sub topic	Guidance
1.3.1 Networks and topologies	
☐ Types of network: o LAN (Local Area Network) o WAN (Wide Area Network) ☐ Factors that affect the performance of networks ☐ The different roles of computers in a client-server and a peer-to-peer network ☐ The hardware needed to connect stand-alone computers into a Local Area Network: o Wireless access points o Routers o Switches o NIC (Network Interface Controller/Card) o Transmission media ☐ The Internet as a worldwide collection of computer networks: o DNS (Domain Name Server) o Hosting o The Cloud o Web servers and clients ☐ Star and Mesh network topologies	**Required** ✓ The characteristics of LANs and WANs including common examples of each ✓ Understanding of different factors that can affect the performance of a network, e.g.: ▪ Number of devices connected ▪ Bandwidth ✓ The tasks performed by each piece of hardware ✓ The concept of the Internet as a network of computer networks ✓ A Domain Name Service (DNS) is made up of multiple Domain Name Servers ✓ A DNS's role in the conversion of a URL to an IP address ✓ Concept of servers providing services (e.g. Web server → Web pages, File server → file storage/retrieval) ✓ Concept of clients requesting/using services from a server ✓ The Cloud: remote service provision (e.g. storage, software, processing) ✓ Advantages and disadvantages of the Cloud ✓ Advantages and disadvantages of the Star and Mesh topologies ✓ Apply understanding of networks to a given scenario

1.3.2 Wired and wireless networks, protocols and layers

- ☐ Modes of connection:
  - o Wired
    - • Ethernet
  - o Wireless
    - • Wi-Fi
    - • Bluetooth
- ☐ Encryption
- ☐ IP addressing and MAC addressing
- ☐ Standards
- ☐ Common protocols including:
  - o TCP/IP (Transmission Control Protocol/Internet Protocol)
  - o HTTP (Hyper Text Transfer Protocol)
  - o HTTPS (Hyper Text Transfer Protocol Secure)
  - o FTP (File Transfer Protocol)
  - o POP (Post Office Protocol)
  - o IMAP (Internet Message Access Protocol)
  - o SMTP (Simple Mail Transfer Protocol)
- ☐ The concept of layers

Required

- ✓ Compare benefits and drawbacks of wired versus wireless connection
- ✓ Recommend one or more connections for a given scenario
- ✓ The principle of encryption to secure data across network connections
- ✓ IP addressing and the format of an IP address (IPv4 and IPv6)
- ✓ A MAC address is assigned to devices; its use within a network
- ✓ The principle of a standard to provide rules for areas of computing
- ✓ Standards allows hardware/software to interact across different manufacturers/producers
- ✓ The principle of a (communication) protocol as a set of rules for transferring data
- ✓ That different types of protocols are used for different purposes
- ✓ The basic principles of each protocol i.e. its purpose and key features
- ✓ How layers are used in protocols, and the benefits of using layers; for a teaching example, please refer to the 4-layer TCP/IP model

Not required

- × Understand how Ethernet, Wi-Fi and Bluetooth protocols work
- × Understand differences between static and dynamic, or public and private IP addresses
- × Knowledge of individual standards
- × Knowledge of the names and function of each TCP/IP layer

1.4 – Network security

Sub topic	Guidance
1.4.1 Threats to computer systems and networks	
☐ Forms of attack: o Malware o Social engineering, e.g. phishing, people as the 'weak point' o Brute-force attacks o Denial of service attacks o Data interception and theft o The concept of SQL injection	**Required** ✓ Threats posed to devices/systems ✓ Knowledge/principles of each form of attack including: ▪ How the attack is used ▪ The purpose of the attack
1.4.2 Identifying and preventing vulnerabilities	
☐ Common prevention methods: o Penetration testing o Anti-malware software o Firewalls o User access levels o Passwords o Encryption o Physical security	**Required** ✓ Understanding of how to limit the threats posed in 1.4.1 ✓ Understanding of methods to remove vulnerabilities ✓ Knowledge/principles of each prevention method: ▪ What each prevention method may limit/prevent ▪ How it limits the attack

1.5 – Systems software

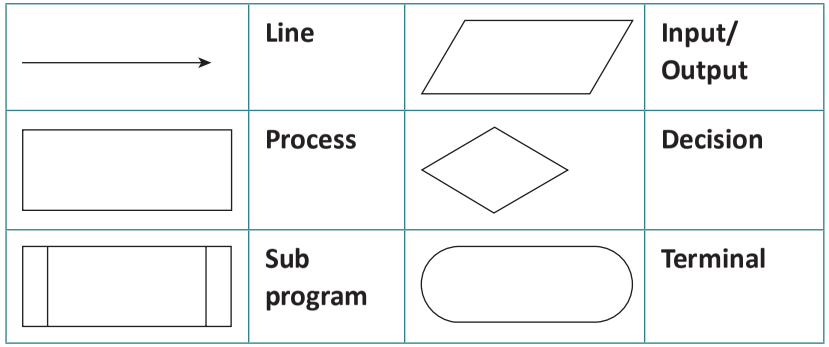
Sub topic	Guidance
1.5.1 Operating systems	
☐ The purpose and functionality of operating systems: o User interface o Memory management and multitasking o Peripheral management and drivers o User management o File management	**Required** ✓ What each function of an operating system does ✓ Features of a user interface ✓ Memory management, e.g. the transfer of data between memory, and how this allows for multitasking ✓ Understand that: ▪ Data is transferred between devices and the processor ▪ This process needs to be managed ✓ User management functions, e.g.: ▪ Allocation of an account ▪ Access rights ▪ Security, etc. ✓ File management, and the key features, e.g.: ▪ Naming ▪ Allocating to folders ▪ Moving files ▪ Saving, etc. **Not required** × Understanding of paging or segmentation
1.5.2 Utility software	
☐ The purpose and functionality of utility software ☐ Utility system software: o Encryption software o Defragmentation o Data compression	**Required** ✓ Understand that computers often come with utility software, and how this performs housekeeping tasks ✓ Purpose of the identified utility software and why it is required

1.6 – Ethical, legal, cultural and environmental impacts of digital technology

Sub topic	Guidance
1.6.1 Ethical, legal, cultural and environmental impact	
☐ Impacts of digital technology on wider society including: o Ethical issues o Legal issues o Cultural issues o Environmental issues o Privacy issues ☐ Legislation relevant to Computer Science: o The Data Protection Act 2018 o Computer Misuse Act 1990 o Copyright Designs and Patents Act 1988 o Software licences (i.e. open source and proprietary)	**Required** ✓ Technology introduces ethical, legal, cultural, environmental and privacy issues ✓ Knowledge of a variety of examples of digital technology and how this impacts on society ✓ An ability to discuss the impact of technology based around the issues listed ✓ The purpose of each piece of legislation and the specific actions it allows or prohibits ✓ The need to license software and the purpose of a software licence ✓ Features of open source (providing access to the source code and the ability to change the software) ✓ Features of proprietary (no access to the source code, purchased commonly as off-the-shelf) ✓ Recommend a type of licence for a given scenario including benefits and drawbacks

2c. Content of Computational thinking, algorithms and programming (J277/02)

2.1 – Algorithms

Sub topic	Guidance
2.1.1 Computational thinking	
☐ Principles of computational thinking: o Abstraction o Decomposition o Algorithmic thinking	**Required** ✓ Understanding of these principles and how they are used to define and refine problems
2.1.2 Designing, creating and refining algorithms	
☐ Identify the inputs, processes, and outputs for a problem ☐ Structure diagrams ☐ Create, interpret, correct, complete, and refine algorithms using: o Pseudocode o Flowcharts o Reference language/high-level programming language ☐ Identify common errors ☐ Trace tables	**Required** ✓ Produce simple diagrams to show: ▪ The structure of a problem ▪ Subsections and their links to other subsections ✓ Complete, write or refine an algorithm using the techniques listed ✓ Identify syntax/logic errors in code and suggest fixes ✓ Create and use trace tables to follow an algorithm **Flowchart symbols**  **Line** **Input/ Output** **Process** **Decision** **Sub program** **Terminal**

2.1.3 Searching and sorting algorithms	
☐ Standard searching algorithms: o Binary search o Linear search ☐ Standard sorting algorithms: o Bubble sort o Merge sort o Insertion sort	**Required** ✓ Understand the main steps of each algorithm ✓ Understand any pre-requisites of an algorithm ✓ Apply the algorithm to a data set ✓ Identify an algorithm if given the code or pseudocode for it **Not required** × To remember the code for these algorithms × To remember Exam Reference Language for Merge Sort

2.2 – Programming fundamentals

Sub topic	Guidance
2.2.1 Programming fundamentals	
☐ The use of variables, constants, operators, inputs, outputs and assignments ☐ The use of the three basic programming constructs used to control the flow of a program: o Sequence o Selection o Iteration (count- and condition-controlled loops) ☐ The common arithmetic operators ☐ The common Boolean operators AND, OR and NOT	**Required** ✓ Practical use of the techniques in a high-level language within the classroom ✓ Understanding of each technique ✓ Recognise and use the following operators:

Comparison operators		Arithmetic operators	
==	Equal to	+	Addition
!=	Not equal to	–	Subtraction
<	Less than	*	Multiplication
<=	Less than or equal to	/	Division
>	Greater than	MOD	Modulus
>=	Greater than or equal to	DIV	Quotient
		^	Exponentiation (to the power)

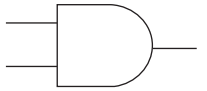
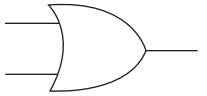
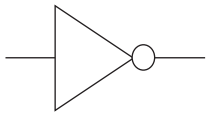
2.2.2 Data types	
☐ The use of data types: o Integer o Real o Boolean o Character and string o Casting	**Required** ✓ Practical use of the data types in a high-level language within the classroom ✓ Ability to choose suitable data types for data in a given scenario ✓ Understand that data types may be temporarily changed through casting, and where this may be useful
2.2.3 Additional programming techniques	
☐ The use of basic string manipulation ☐ The use of basic file handling operations: o Open o Read o Write o Close ☐ The use of records to store data ☐ The use of SQL to search for data ☐ The use of arrays (or equivalent) when solving problems, including both one-dimensional (1D) and two-dimensional arrays (2D) ☐ How to use sub programs (functions and procedures) to produce structured code ☐ Random number generation	**Required** ✓ Practical use of the additional programming techniques in a high-level language within the classroom ✓ Ability to manipulate strings, including: ▪ Concatenation ▪ Slicing ✓ Arrays as fixed length or static structures ✓ Use of 2D arrays to emulate database tables of a collection of fields, and records ✓ The use of functions ✓ The use of procedures ✓ Where to use functions and procedures effectively ✓ The use of the following within functions and procedures: ▪ local variables/constants ▪ global variables/constants ▪ arrays (passing and returning) ✓ SQL commands: ▪ SELECT ▪ FROM ▪ WHERE ✓ Be able to create and use random numbers in a program

2.3 – Producing robust programs

Sub topic	Guidance
2.3.1 Defensive design	
☐ Defensive design considerations: o Anticipating misuse o Authentication ☐ Input validation ☐ Maintainability: o Use of sub programs o Naming conventions o Indentation o Commenting	**Required** ✓ Understanding of the issues a programmer should consider to ensure that a program caters for all likely input values ✓ Understanding of how to deal with invalid data in a program ✓ Authentication to confirm the identity of a user ✓ Practical experience of designing input validation and simple authentication (e.g. username and password) ✓ Understand why commenting is useful and apply this appropriately
2.3.2 Testing	
☐ The purpose of testing ☐ Types of testing: o Iterative o Final/terminal ☐ Identify syntax and logic errors ☐ Selecting and using suitable test data: o Normal o Boundary o Invalid/Erroneous ☐ Refining algorithms	**Required** ✓ The difference between testing modules of a program during development and testing the program at the end of production ✓ Syntax errors as errors which break the grammatical rules of the programming language and stop it from being run/translated ✓ Logic errors as errors which produce unexpected output ✓ Normal test data as data which should be accepted by a program without causing errors ✓ Boundary test data as data of the correct type which is on the very edge of being valid ✓ Invalid test data as data of the correct data type which should be rejected by a computer system ✓ Erroneous test data as data of the incorrect data type which should be rejected by a computer system ✓ Ability to identify suitable test data for a given scenario ✓ Ability to create/complete a test plan

2.4 – Boolean logic

Sub topic	Guidance
2.4.1 Boolean logic	
☐ Simple logic diagrams using the operators AND, OR and NOT ☐ Truth tables ☐ Combining Boolean operators using AND, OR and NOT ☐ Applying logical operators in truth tables to solve problems	**Required** ✓ Knowledge of the truth tables for each logic gate ✓ Recognition of each gate symbol ✓ Understanding of how to create, complete or edit logic diagrams and truth tables for given scenarios ✓ Ability to work with more than one gate in a logic diagram

Boolean Operators	Logic Gate Symbol
AND *(Conjunction)*	
OR *(Disjunction)*	
NOT *(Negation)*	

Truth Tables

AND			OR			NOT	
A	B	A AND B	A	B	A OR B	A	NOT A
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

Alternatives

- Use of other valid notation will be accepted within the examination, e.g. Using T/F for 1/0, or V for OR, etc.

2.5 – Programming languages and Integrated Development Environments

Sub topic	Guidance
2.5.1 Languages	
☐ Characteristics and purpose of different levels of programming language: o High-level languages o Low-level languages ☐ The purpose of translators ☐ The characteristics of a compiler and an interpreter	**Required** ✓ The differences between high- and low-level programming languages ✓ The need for translators ✓ The differences, benefits and drawbacks of using a compiler or an interpreter **Not required** × Understanding of assemblers
2.5.2 The Integrated Development Environment (IDE)	
☐ Common tools and facilities available in an Integrated Development Environment (IDE): o Editors o Error diagnostics o Run-time environment o Translators	**Required** ✓ Knowledge of the tools that an IDE provides ✓ How each of the tools and facilities listed can be used to help a programmer develop a program ✓ Practical experience of using a range of these tools within at least one IDE