**Qualification aims and objectives:**

* understand and apply the fundamental principles and concepts of Computer Science, including abstraction, decomposition, logic, algorithms, and data representation
* analyse problems in computational terms through practical experience of solving such problems, including designing, writing and debugging programs
* think creatively, innovatively, analytically, logically and critically
* understand the components that make up digital systems, and how they communicate with one another and with other systems
* understand the impacts of digital technology to the individual and to wider society
* apply mathematical skills relevant to Computer Science.

**Key features of OCR GCSE Computing course:**

* Consists of two papers, one focusing on computer systems and one with a focus on programming, computational thinking, and algorithms. Both papers have identical weighting and mark allocations
* The course has been designed to transition seamlessly into Computer Science at AS Level and/or A Level
* Developing valuable thinking and programming skills that are extremely attractive in the modern workplace
* Develop an understanding of computational thinking and how to apply it through a chosen programming language

**Content overview: J277/01: Computer systems.**

**Topics**:

* 1.1 Systems architecture
* 1.2 Memory and storage
* 1.3 Computer networks, connections and protocols
* 1.4 Network security
* 1.5 Systems software
* 1.6 Ethical, legal, cultural and environmental impacts of digital technology

**Assessment**:

* Written paper: 1 hour and 30 minutes, 50% of total GCSE, 80 marks. This is a non-calculator paper
* This paper consists of multiple-choice questions, short response questions and extended response questions.

**Content overview: J277/02: Computational thinking, algorithms and programming.**

**Topics**:

* 2.1 Algorithms
* 2.2 Programming fundamentals
* 2.3 Producing robust programs
* 2.4 Boolean logic
* 2.5 Programming languages and Integrated Development Environments

**Assessment**:

* Written paper: 1 hour and 30 minutes, 50% of total GCSE, 80 marks. This is a non-calculator paper
* This paper has two sections: Section A and Section B
* In Section B, questions assessing students’ ability to write or refine algorithms must be answered using either the OCR Exam Reference Language or the high-level programming language they are familiar with, such as Python.

**Practical Programming**

All students must be given the opportunity to undertake a programming task(s), either to a specification or to solve a problem (or problems), during their course of study. Students may draw on some of the content in both components when engaged in Practical Programming.

**Year 10 Computing**

|  |  |  |  |
| --- | --- | --- | --- |
| **Week** | **Term** | **Topic** | **Learning Objectives** |
| 1 | Autumn Term 1 | The purpose of the CPU:  The fetch-execute-cycle | * 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 |
| 2 | Common CPU components and their function:   * ALU (Arithmetic Logic Unit) * CU (Control Unit) * Cache * Registers |
| 3 | Von Neumann architecture:   * MAR (Memory Address Register) * MDR (Memory Data Register) * Program Counter Accumulator |
| 4 | How common characteristics of CPUs affect their performance:   * Clock speed * Cache size * Number of cores | * Understanding of each characteristic as listed * The effects of changing any of the common characteristics on system performance, either individually or in combination |
| 5 | Embedded systems:   * The purpose and characteristics of embedded systems * Examples of embedded systems | * What embedded systems are * Typical characteristics of embedded systems * Familiarity with a range of different embedded systems |
| 6 | 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 | * 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 |
| 7 | Secondary storage:   * The need for secondary storage * Common types of storage/   + Optical   + Magnetic   + 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:   + Capacity   + Speed   + Portability   + Durability   + Reliability   + Cost | * 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 |
| 8 | Autumn Term 1 assessment |  |
| 9 | Autumn Term 2 | The units of data storage   * Bit * Nibble (4 bits) * Byte (8 bits) * Kilobyte (1,000 bytes or 1 KB) * Megabyte (1,000 KB) * Gigabyte (1,000 MB) * Terabyte (1,000 GB) * 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 | * 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 |
| 10 | 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 including8 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 | * 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 bit 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) |
| 11 | Data storage: 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.: * ASCII * UNICODE | * 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 |
| 12 | Data storage: images   * How an image is represented as a series of pixels, represented in binary * Metadata * The effect of colour depth and resolution on: * The quality of the image * The size of an image file | * 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.) |
| 13 | Data storage: sound   * How sound can be sampled and stored in digital form ̈ The effect of sample rate, duration and bit depth on: * The playback quality * The size of a sound file | * 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) |
| 14 | Compression   * The need for compression * Types of compression: * Lossy * Lossless | * Common scenarios where compression may be needed * Advantages and disadvantages of each type of compression * Effects on the file for each type of compression |
| 15 | Autumn Term 2 assessment |  |
| 16 | Spring Term 1 | Networks and topologies  Types of network:   * LAN (Local Area Network) * 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 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 |
| 17 | Hardware needed to connect stand-alone computers into a Local Area Network:   * Wireless access points * Routers * Switches * NIC (Network Interface Controller/Card) * Transmission media * The different roles of computers in a client-server and a peer-to-peer network | * The tasks performed by each piece of hardware |
| 18 | * The Internet as a worldwide collection of computer networks: * DNS (Domain Name Server) * Hosting * The Cloud * Web servers and clients   Factors that affect the performance of networks and modes of connection:   * Wired * Ethernet * Wireless * Wi-Fi * Bluetooth * Encryption | * 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 * 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 * 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 |
| 19 | Network protocols / standards including:   * TCP/IP (Transmission Control Protocol/Internet Protocol) * HTTP (Hyper Text Transfer Protocol) * HTTPS (Hyper Text Transfer Protocol Secure) * FTP (File Transfer Protocol) * POP (Post Office Protocol) * IMAP (Internet Message Access Protocol) * SMTP (Simple Mail Transfer Protocol) * The concept of layers | * 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 |
| 20 | Threats to network security and computer systems   * Malware * Social engineering, e.g., phishing, people as the ‘weak point * Brute-force attacks * Denial of service attacks * Data interception and theft * The concept of SQL injection   Common prevention risks   * Penetration testing * Anti-malware software * Firewalls * User access levels * Passwords * Encryption   + Physical security | * Threats posed to devices/systems * Knowledge/principles of each form of attack including: * How the attack is used * The purpose of the attack * 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 |
| 21 |
| 21 |  | Spring Term 1 assessment |  |
|  |  |  |  |
| 22 | Spring Term 2 | The purpose and function of operating systems:   * + User interface   + Memory management and multi-tasking   + Peripheral management and drivers   + User management   + File management | * 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. |
| 23 | The purpose and functionality of utility software | * 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 |
| 24 | Utility system software: encryption |
| 25 | Utility software: defragmentation |
| 26 | Utility software: data compression |
| 27 | Spring Term 2 assessment |  |
| 28 | Summer Term 1 | Ethical issues | * Technology introduces ethical, legal, cultural, environmental and privacy issues * Knowledge of a variety of examples of digital technology and how this impacts 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 |
| 29 | Legal issues |
| 30 | Cultural issues |
| 31 | Environmental issues |
| 32 | Privacy issues |
| 33 | Summer Term 1 assessment |
| 34 | Summer Term 2 | The Data Protection Act 2018 |
| 35 | The Computer Misuse Act 1990 |
| 36 | Copyrights Designs and Patents Act 1988 |
| 37 | Software licences (i.e., open source and proprietary) |
| 38 | Misinformation |
| 39 | Summer Term 2 Assessment |

**Year 11 Computing: Computational Thinking, algorithms and programming (J277/02)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Week** | **Term** | **Topic** | **Learning Objectives** |
| 1 | Autumn Term 1 | Principles of computational thinking:   * Abstraction * Decomposition * Algorithmic thinking | * Understanding of these principles and how they are used to define and refine problems |
| 2 | Algorithms: Identify inputs, processes and outputs for a problem | * Produce simple diagrams to show: * The structure of a problem * Subsections and their links to other subsections |
| 3 | Algorithms: Structure diagrams |
| 4 | Algorithms: Flowcharts |
| 5 | Algorithms: Pseudocode   * Reference language/high-level programming language * Identify common errors | * Complete, write or refine an algorithm using the techniques listed * Identify syntax/logic errors in code and suggest fixes |
| 6 | Trace tables | * Create and use trace tables to follow an algorithm |
| 7 | * Standard searching algorithms * Binary search * Linear search * Standard sorting algorithms * Bubble sort * Merge sort * Insertion sort | * 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 |
| 8 | Autumn Term 1 assessment |  |
| 9 | Autumn Term 2 | Programming: The use of variables, constants, operators, inputs, outputs and assignments | * Practical use of the techniques in a high-level language within the classroom * Understanding of each technique * Recognise and use the following operators:   C:\Users\Jmorgan\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\77A5621.tmp |
| 10 | Programming: sequence and selection |
| 11 | Programming: iteration (count and controlled loops) |
| 12 | Programming: arithmetic operators and Boolean operators AND, OR and NOT. |
| 13 | Programming tasks using different data types:   * Integer * Real * Boolean * Character and string * Casting | * 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 |
| 14 | Programming task:   * How to use sub programs (functions and procedures) to produce structured code * Random number generation | * The use of functions * The use of procedures * Where to use functions and procedures effectively * Be able to create and use random numbers in a program |
| 15 | Autumn Term 2 assessment |  |
| 16 | Spring Term 1 | Arrays   * The use of arrays (or equivalent) when solving problems, including both one-dimensional (1D) and two-dimensional arrays (2D) | * Arrays as fixed length or static structures * Use of 2D arrays to emulate database tables of a collection of fields, and records |
| 17 | SQL:   * The use of basic string manipulation * The use of basic file handling operations: * Open * Read * Write * Close * The use of records to store data * The use of SQL to search for data | * SQL commands: * SELECT * FROM * WHERE |
| 18 | Defensive design considerations   * Anticipating misuse * Authentication * Input validation | * 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) |
| 19 | Maintainability:   * Use of sub programs * Naming conventions * Indentation * Commenting | * Understand why commenting is useful and apply this appropriately |
| 20 | Testing:   * The purpose of testing * Types of testing: * Iterative * Final / terminal * Identify syntax and logic errors * Selecting and using suitable test data: * Normal * Boundary * Invalid / erroneous | * 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 |
| 21 |  | Spring Term 1 assessment |  |
|  |  |  |  |
| 22 | Spring Term 2 | * Characteristics and purpose of different levels of programming language: * High-level languages * Low-level languages * The purpose of translators * The characteristics of a compiler and an interpreter | * 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 |
| 23 | The Integrated Development Environment   * Common tools and facilities available in an Integrated Development Environment (IDE): * Editors * Error diagnostics * Run-time environment * Translators | * 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 |
| 24 | Revision |
| 25 | Revision |
| 26 | Revision |
| 27 | Revision |  |
| 28 | Summer Term 1 | Ethical issues | * Technology introduces ethical, legal, cultural, environmental and privacy issues * Knowledge of a variety of examples of digital technology and how this impacts 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 |
| 29 | Legal issues |
| 30 | Cultural issues |
| 31 | Environmental issues |
| 32 | Privacy issues |
| 33 | Exam skills development |
| 34 | Summer Term 2 | J277/01 exam |
| 35 | J277/02 exam |
| 36 |  |
| 37 |  |
| 38 |  |
| 39 |  |

**Content of Computer systems (J277/01)**

|  |  |
| --- | --- |
| **1.1 - System Architecture** | |
| ***1.1.1 - Architecture of the CPU*** | |
| The purpose of the CPU   * The fetch-execute-cycle | 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 |
| Common CPU components and their function:   * ALU (Arithmetic Logic Unit) * CU (Control Unit) * Cache * Registers |
| Von Neumann architecture:   * MAR (Memory Address Register) * MDR (Memory Data Register) * Program Counter Accumulator |
| ***1.1.2 - CPU Performance*** | |
| How common characteristics of CPUs affect their performance:   * Clock speed * Cache size * 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** | |
| ***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/   + Optical   + Magnetic   + 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:   + Capacity   + Speed   + Portability   + Durability   + Reliability   + 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 |
| ***1.2.3 - Units*** | |
| * The units of data storage   + Bit   + Nibble (4 bits)   + Byte (8 bits)   + Kilobyte (1,000 bytes or 1 KB)   + Megabyte (1,000 KB)   + Gigabyte (1,000 MB)   + Terabyte (1,000 GB)   + 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 including8 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   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.:   + ASCII   + UNICODE   Images   * How an image is represented as a series of pixels, represented in binary * Metadata * The effect of colour depth and resolution on:   + The quality of the image   + 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:   + The playback quality   + The size of a sound file | 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 bit 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)   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:   + Lossy   + 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 and Protocols** | |
| ***1.3.1 - Networks and Topologies*** | |
| Types of network:   * LAN (Local Area Network) * 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:   + Wireless access points   + Routers   + Switches   + NIC (Network Interface Controller/Card)   + Transmission media * The Internet as a worldwide collection of computer networks:   + DNS (Domain Name Server)   + Hosting   + The Cloud   + Web servers and clients | 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:   + Wired     - Ethernet   + Wireless     - Wi-Fi     - Bluetooth * Encryption * IP addressing and MAC addressing * Standards * Common protocols including:   + TCP/IP (Transmission Control Protocol/Internet Protocol)   + HTTP (Hyper Text Transfer Protocol)   + HTTPS (Hyper Text Transfer Protocol Secure)   + FTP (File Transfer Protocol)   + POP (Post Office Protocol)   + IMAP (Internet Message Access Protocol)   + 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 hused 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** | |
| ***1.4.1 - Threats to computer systems and networks*** | |
| Forms of attack:   * + Malware   + Social engineering, e.g., phishing, people as the ‘weak point   + Brute-force attacks   + Denial of service attacks   + Data interception and theft   + 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:   + Penetration testing   + Anti-malware software   + Firewalls   + User access levels   + Passwords   + Encryption   + 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 - System Software** | |
| ***1.5.1 - Operating systems*** | |
| Forms of attack:   * + Malware   + Social engineering, e.g., phishing, people as the ‘weak point   + Brute-force attacks   + Denial of service attacks   + Data interception and theft   + The concept of SQL injection | 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. |
| ***1.5.2 - Utility software*** | |
| * The purpose and functionality of utility software * Utility system software:   + Encryption software   + Defragmentation   + 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** | |
| ***1.6.1 - Ethical, legal, cultural and environmental impact*** | |
| * Impacts of digital technology on wider society including:   + Ethical issues   + Legal issues   + Cultural issues   + Environmental issues   + Privacy issues * Legislation relevant to Computer Science:   + The Data Protection Act 2018   + The Computer Misuse Act 1990   + Copyrights Designs and Patents Act 1988   + 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 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 |

**Content of Computational Thinking, algorithms and programming (J277/02)**

|  |  |
| --- | --- |
| **2.1 - Algorithms** | |
| ***2.1.1 - Computational thinking*** | |
| * Principles of computational thinking:   + Abstraction   + Decomposition   + 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:   + Pseudocode   + Flowcharts   + 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 |
| ***2.1.3 - Searching and sorting algorithms*** | |
| * Standard searching algorithms   + Binary search   + Linear search * Standard sorting algorithms   + Bubble sort   + Merge sort   + 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** | |
| ***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:   + Sequence   + Selection   + Iteration (count and 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: |

|  |  |
| --- | --- |
| ***2.2.2 - Data types*** | |
| * The use of data types:   + Integer   + Real   + Boolean   + Character and string   + 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:   + Open   + Read   + Write   + 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** | |
| ***2.3.1 - Defensive design*** | |
| * Defensive design considerations   + Anticipating misuse   + Authentication * Input validation * Maintainability:   + Use of sub programs   + Naming conventions   + Indentation   + 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:   + Iterative   + Final / terminal * Identify syntax and logic errors * Selecting and using suitable test data:   + Normal   + Boundary   + Invalid / erroneous | 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** | |
| ***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     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** | |
| ***2.5.1 - Languages*** | |
| * Characteristics and purpose of different levels of programming language:   + High-level languages   + 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*** | |
| * Common tools and facilities available in an Integrated Development Environment (IDE):   + Editors   + Error diagnostics   + Run-time environment   + 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 |