Year 11 Mock 1
Revision List

Subject	**GCSE Computer Science**
Specification	OCR J277
Recommended Revision Guides	
Recommended websites and other resources	https://www.smartrevise.online/ Paper 1 Revision (Web view) Paper 2 Revision (Web view)

Topic 1.1 – Systems architecture	
Architecture of the CPU The purpose of the CPU: • The fetch-execute cycle 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	• 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
Performance of the CPU How common characteristics of CPUs affect their performance: • Clock speed • Cache size • Number of cores	• Understanding of each characteristic listed • The effects of changing any of the common characteristics on system performance, either individually or in combination
Topic 1.2 – Memory, Storage and Data Representation	
Primary 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 • Cache	• Why computers have primary storage (memory) • 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 VM when RAM is full
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	• 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

• Speed • Portability • Durability • Reliability • Cost	
Units of Data Storage 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
Data Representation: 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	• 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)
Data Representation: 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 character code for 'B' will be one more than the character code for 'A' • Binary representation of ASCII in the exam will use 8 bits
Data Representation: 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.)
Data Representation: Sound How sound can be sampled and stored in digital form	• Analogue sounds must be stored in binary • Sample rate – measured in Hertz (Hz)

Budehaven Community School

Year 11 Mock 1
Revision List

The effect of sample rate, duration and bit depth on: • The playback quality • The size of a sound file	• Duration – how many seconds of audio the sound file contains • Bit depth – number of bits available to store each sample (e.g. 16-bit)
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
Topic 1.3 - 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	• Compare benefits and drawbacks of wired versus wireless connection • Recommend one or more connections for a given scenario ü 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 ; its format • 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
Topic 1.4 – Network security	
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	• Threats posed to devices/systems • Knowledge/principles of each form of attack including: • How the attack is used • The purpose of the attack
Identifying and preventing vulnerabilities Common prevention methods: • Penetration testing • Anti-malware software • Firewalls	• 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

• User access levels • Passwords • Encryption • Physical security	
Topic 1.5 – Systems Software	

Operating systems
The purpose and functionality of operating systems:
• Memory management
• User interface
• Multitasking
• Peripheral management and drivers
• User management
• File management
	• What each function of an operating system does
	• Features of a user interface
	• Memory management:
	  • The transfer of data between memory
	  • The allocation of memory to applications
	  • How it 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

Utility Software The purpose and functionality of utility software Utility system software: • Encryption software • Defragmentation • Data compression	• Understand that computers often come with utility software, and how this performs housekeeping tasks • Purpose of encryption, defragmentation and data compression software and why it is required • Utility software is needed to perform additional tasks that may not be carried out by an operating system
Topic 1.6 – Ethical, legal, cultural and environmental impacts of digital technology	
Issues 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 • Computer Misuse Act 1990 • Copyright Designs and Patents Act 1988 • Software licences (i.e. open source and proprietary)	• 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 license • 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 license for a given scenario including benefits and drawbacks
Topic 2.2 – Programming fundamentals	

Budehaven Community School

Year 11 Mock 1
Revision List

<table>
<tr>
<td>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 condition-controlled loops)

The common arithmetic operators

The common Boolean operators AND, OR and NOT</td>
<td>Arithmetic Operators
+ Addition
– Subtraction
* Multiplication
/ Division
MOD Modulo
DIV Quotient
^ Exponentiation (to the power)

Comparison Operators
== Equal to
!= Not equal to
< Less than
<= Less than or equal to
> Greater than
>= Greater than or equal to</td>
</tr>
<tr>
<td>Data types
The use of data types:
• Integer
• Real
• Boolean
• Character and string
• Casting</td>
<td>• 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</td>
</tr>
<tr>
<td>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

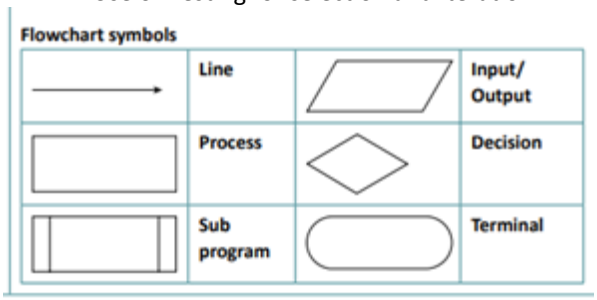
Random number generation</td>
<td>• Ability to manipulate strings, including:
• Concatenation
• Slicing
• Arrays as fixed length or static structures
• Use of 2D arrays to emulate database tables of a collection of fields, and records
• The use of functions
• The use of procedures
• Where to use functions 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
○ LIKE
• Be able to create and use random numbers in a program</td>
</tr>
<tr>
<td colspan="2">Topic 2.1 – Algorithms</td>
</tr>
<tr>
<td>Computational thinking:
• Abstraction
• Decomposition
• Algorithmic thinking</td>
<td>• Understanding of these principles and how they are used to define and refine problems</td>
</tr>
</table>

Budehaven Community School

Year 11 Mock 1
Revision List

<table>
<tr>
<td>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</td>
<td>• 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
• Use of nesting for selection and iteration

</td>
</tr>
<tr>
<td>Searching and sorting algorithms

Standard searching algorithms:
• Binary search
• Linear search

Standard sorting algorithms:
• Bubble sort
• Merge sort
• Insertion sort</td>
<td>• Understand the main steps of the algorithm and the segments of code in it
• Understand any pre-requisites of an algorithm
• Apply the algorithm to a data set
• Identify an algorithm if given the code, pseudocode or Exam Reference Language for it</td>
</tr>
</table>