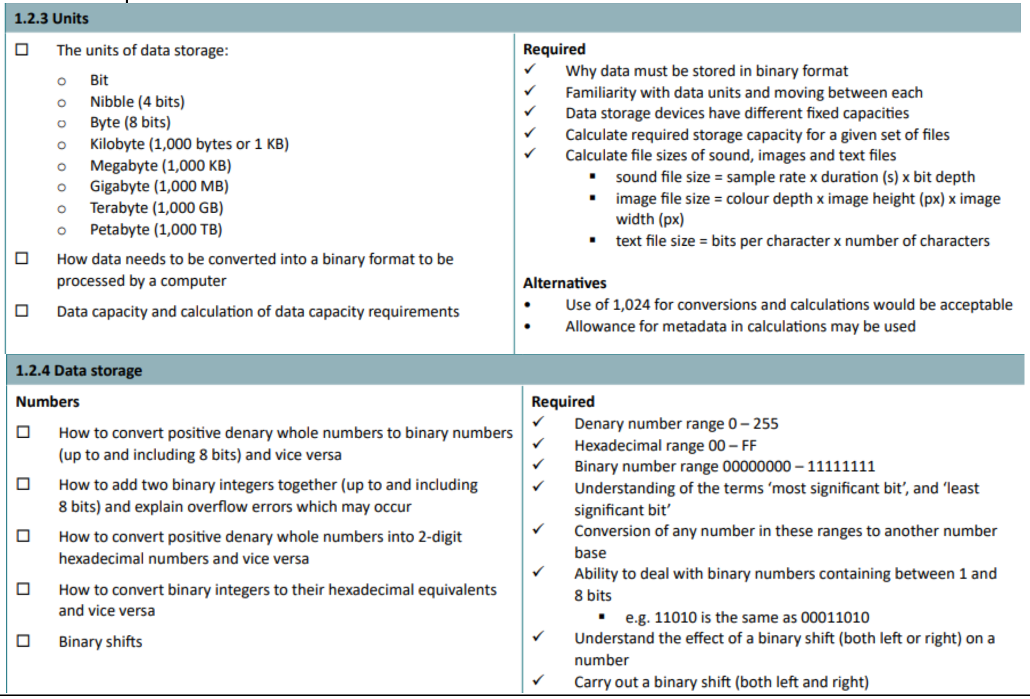
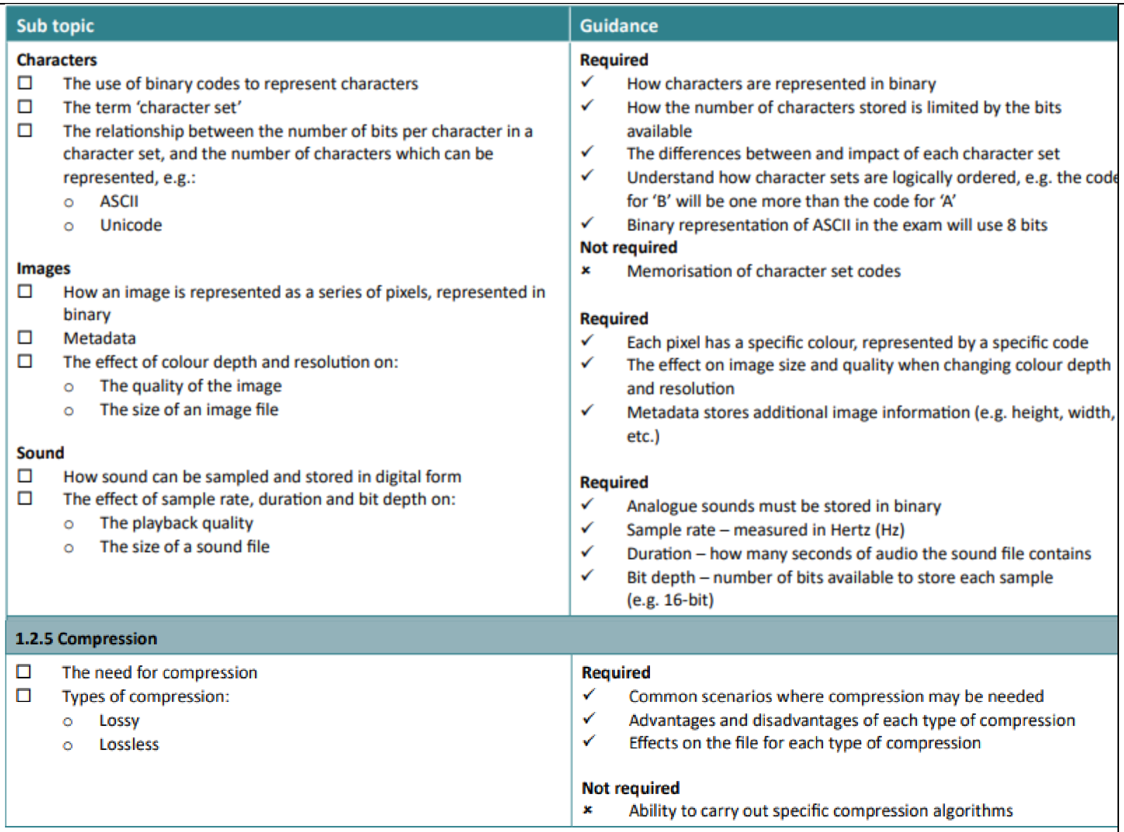
Plan 1

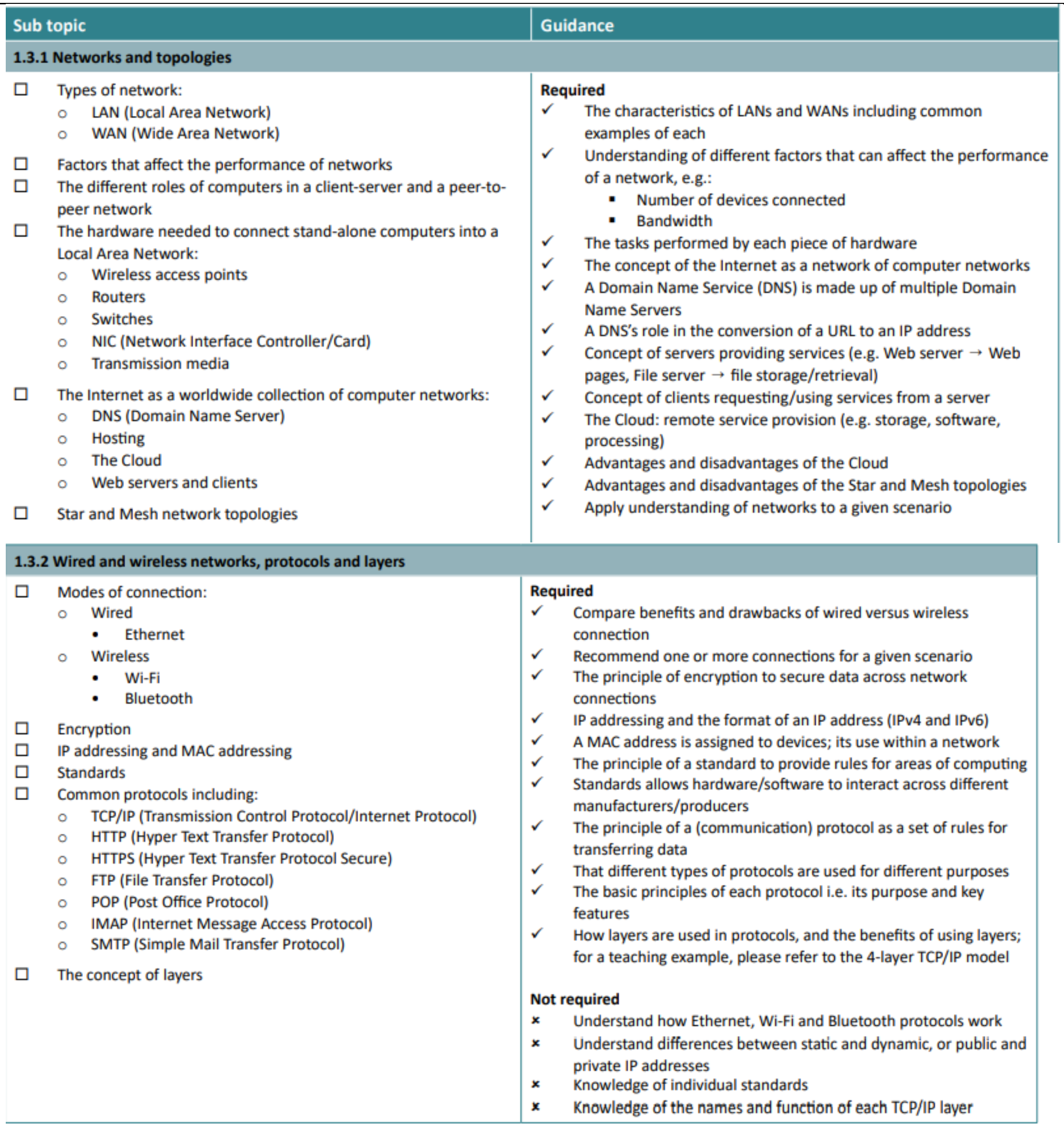
(Computer Science)

Week	Revision plan for half term	Homework	Resources
1 (13th February to 17th February)	• Revision topics for mock paper 1 (from the spec) Data representation  **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 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)	Target your SENECA revision for your mock topics	Seneca: Free Homework & Revision for A Level, GCSE, KS3 & KS2 (senecalearning.com) Outwood Learning: J277 (outwood.com) Craig n Dave: OCR GCSE (J277) 1.1 The purpose of the CPU - The fetch-execute cycle - YouTube



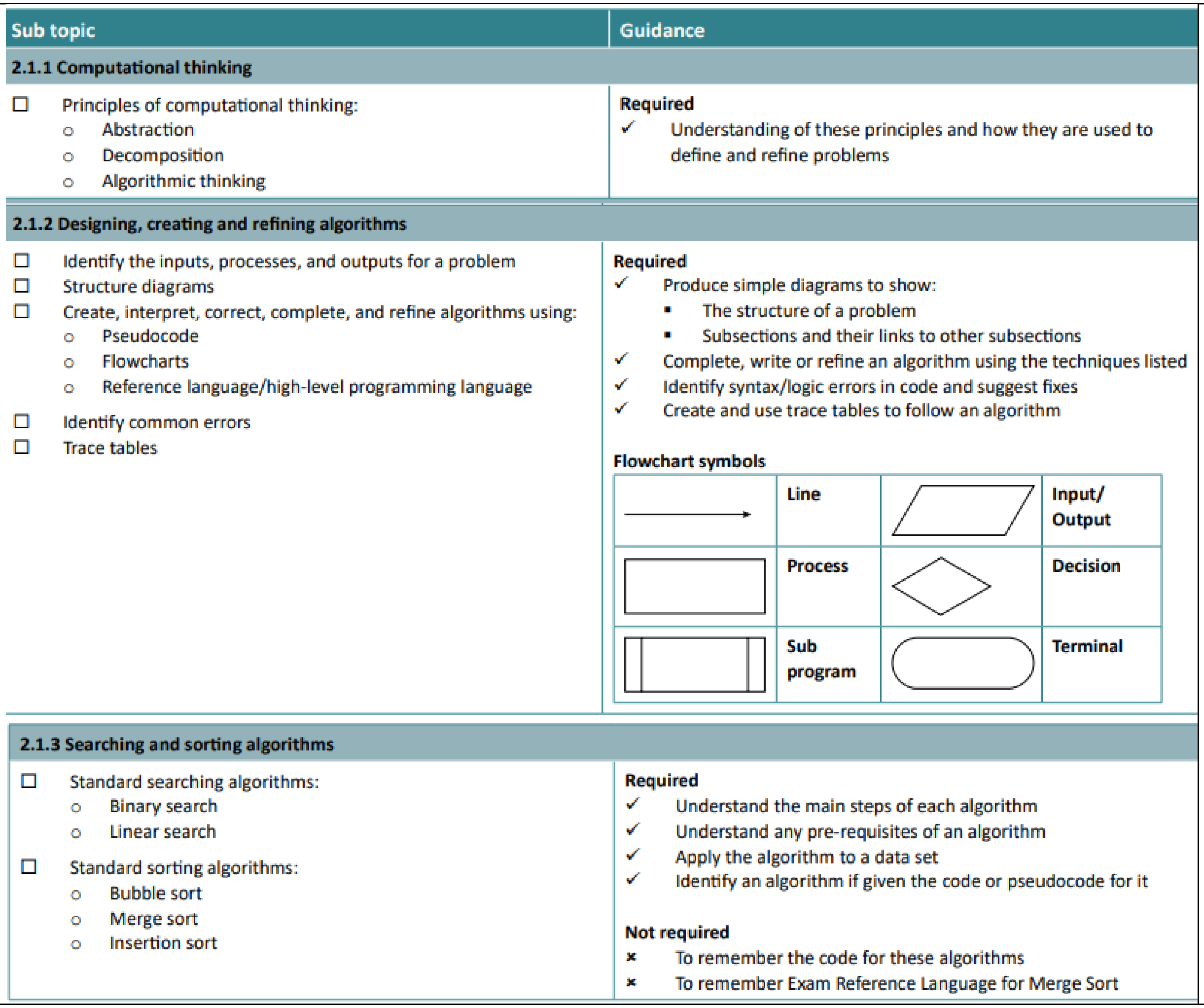
Sub topic	Guidance
Characters □ The use of binary codes to represent characters □ The term 'character set' □ The relationship between the number of bits per character in a character set, and the number of characters which can be represented, e.g.: ○ ASCII ○ Unicode	**Required** ✓ How characters are represented 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
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	**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.)
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** ✓ 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

- Networks



Sub topic	Guidance
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 □ 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: ○ 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 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

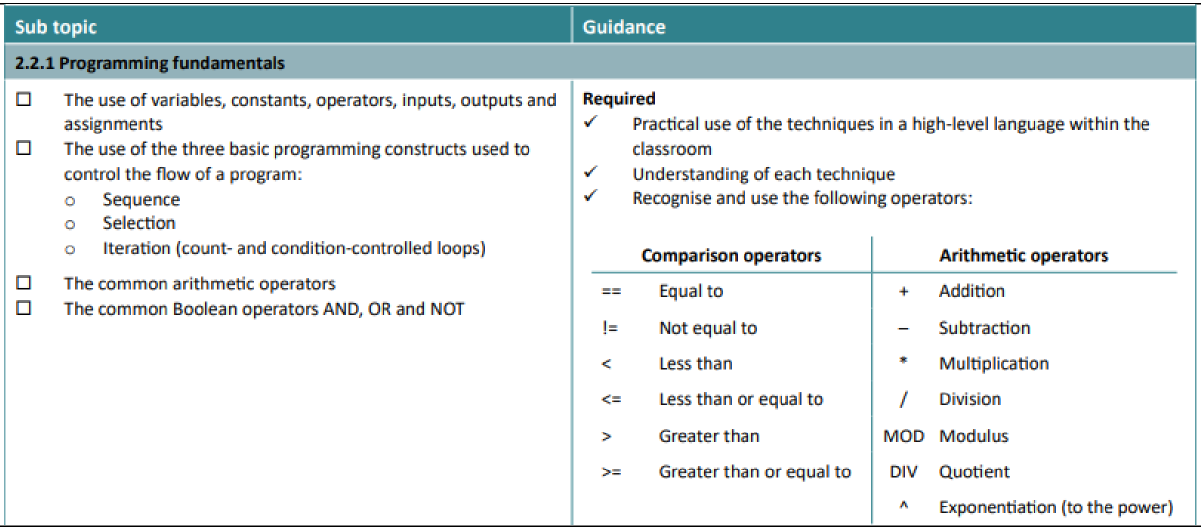
	Start with the topics below to revise for paper 1 • Understanding the different units of data and being able to convert between them • Converting between denary, binary and hexadecimal numbers • Performing binary additions and the problems that it can cause • Performing binary shifts and explaining their effects on the original data • Naming different character sets and understanding the difference between them • Understanding the properties of bitmaps (resolution, colour depth and how they link to file size) • Being able to explain metadata and give some examples • Understanding the properties of sound files (sampling frequency, duration and bit depth) • Being able to draw a network topology with different devices • Describing the role of different hardware that is being used in local and wide area networks • Being able to name and identify factors that could affect the performance of a network • Understanding protocols and giving examples for different situations • Understanding how cloud computing can affect a user • Revision topics for mock paper 2 (from the spec) • Every component 2 topic		



Sub topic	Guidance
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 **Flowchart symbols**
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

Flowchart symbols

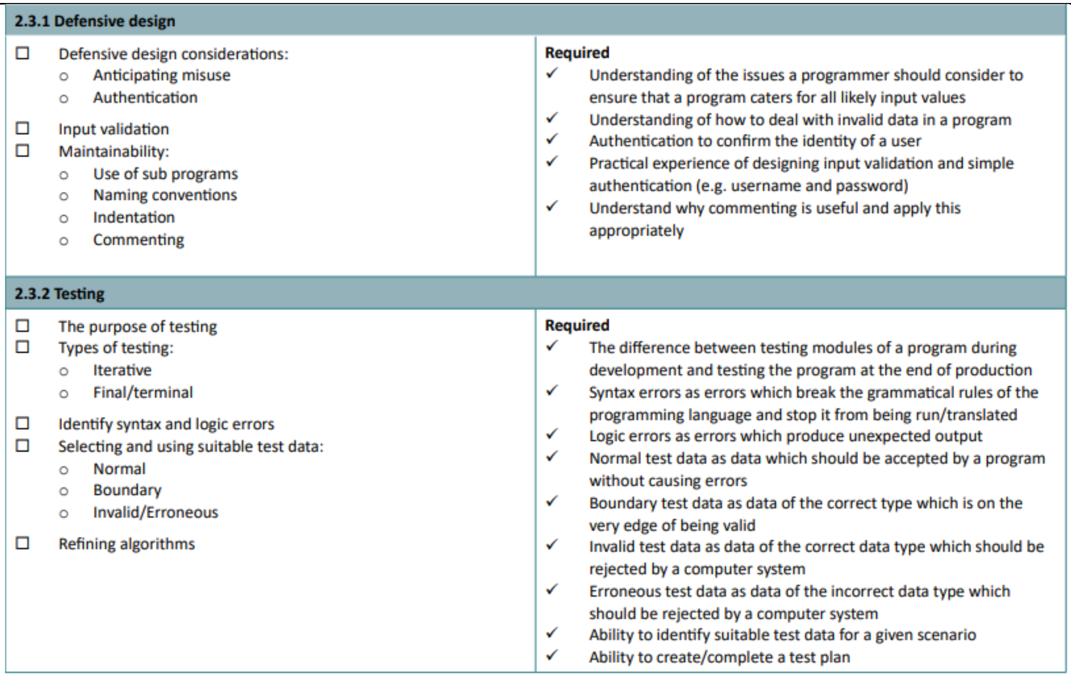
→	**Line**	▱	**Input/ Output**
▭	**Process**	◇	**Decision**
▭	**Sub program**	⬭	**Terminal**

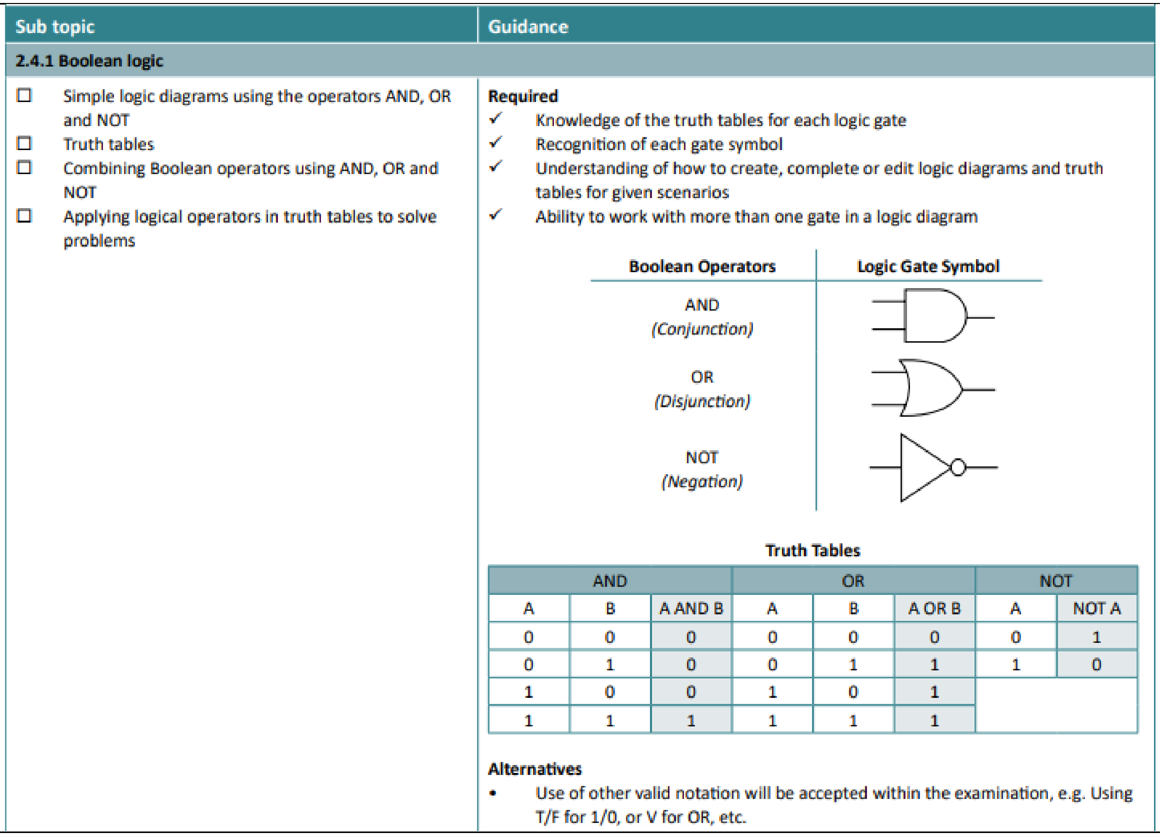


Sub topic	Guidance
2.2.1 Programming fundamentals	
□ The use of variables, constants, operators, inputs, outputs and assignments □ The use of the three basic programming constructs used to control the flow of a program: ○ Sequence ○ Selection ○ Iteration (count- and condition-controlled loops) □ The 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: ○ 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

<table>
<tr>
<td></td>
<td>

<table>
<tr><th colspan="2">2.3.1 Defensive design</th></tr>
<tr>
<td>□ Defensive design considerations:
○ Anticipating misuse
○ Authentication
□ Input validation
□ Maintainability:
○ Use of sub programs
○ Naming conventions
○ Indentation
○ Commenting</td>
<td><b>Required</b>
✓ 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</td>
</tr>
<tr><th colspan="2">2.3.2 Testing</th></tr>
<tr>
<td>□ 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
□ Refining algorithms</td>
<td><b>Required</b>
✓ 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</td>
</tr>
</table>
</td>
<td></td>
<td></td>
<td></td>
</tr>
</table>



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

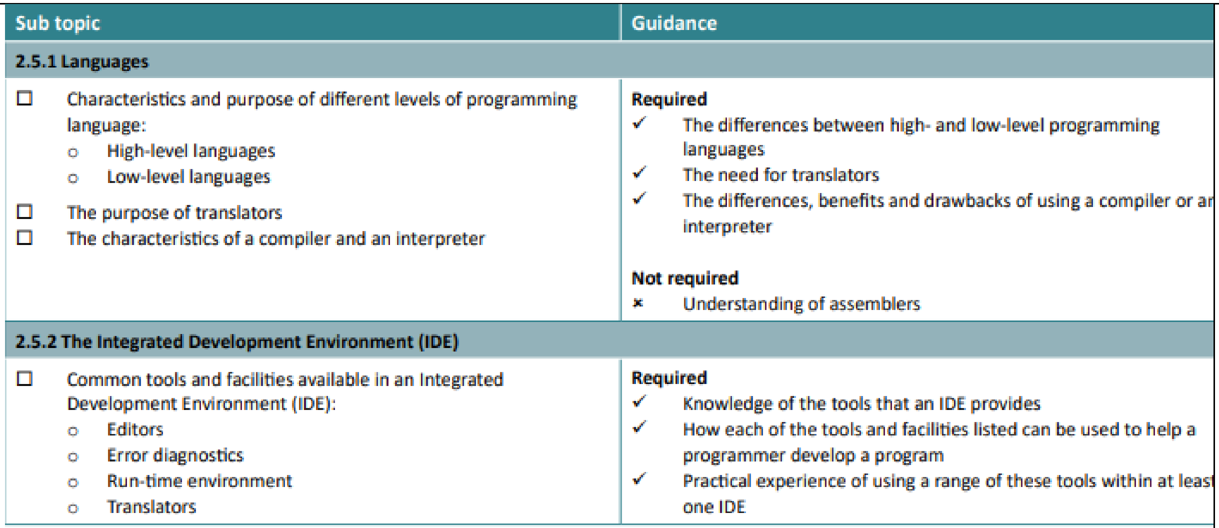
Boolean Operators	Logic Gate Symbol
AND *(Conjunction)*	
OR *(Disjunction)*	
NOT *(Negation)*	

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.



Sub topic	Guidance
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 ar 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): ○ 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 leas one IDE

Start with the topics below to revise for paper 2

- Understanding the differences between programming language
- Naming IDE tools and their usefulness
- Understanding how structure diagrams can be used to show decomposition
- Identifying different types of errors and correcting them (logic and syntax)
- Being able to apply a searching algorithm (linear and binary)
- Being able to understand and apply different computational thinking algorithms (abstraction, decomposition and logical thinking)
- Being able to apply and explain ether authentication or validation
- Understanding the difference between iterative and final testing
- Being able to get user input and storing them into different variables
- Being able to output variables with text
- Knowing how to apply SQL commands
- Being able to follow a flowchart along different paths
- Being able to write simple conditional statements

	• Identifying variables and completing array pseudocode				
1 (13th February to 17th February)					

Plan 2

(SUBJECT)

Week	Revision plan for half term	In Class Topics	Homework	Project Tasks	Resources
1	• Data representation conversions • networks	Binary, denary and hexadecimal conversions Binary addition Binary shifts Character sets Bitmaps, sound and compression Networks topologies Network hardware Cloud computing protocols	SENECA + past paper questions Exercise books		Knowledge organiser Exercise books Seneca: [Free Homework & Revision for A Level, GCSE, KS3 & KS2 (senecalearning.com)] Outwood Learning:

					J277 (outwood.com) Craig n Dave: OCR GCSE (J277) 1.1 The purpose of the CPU - The fetch-execute cycle - YouTube
W/C 20/2	Algorithms Programming Design, testing and IDE's	• programming languages • computational thinking • searching and sorting algorithms • programming errors • logic gates • SQL syntax • Subroutines • Defensive design and authentication • Test data • Writing algorithms 2D arrays			
W/C 27/2	•				
W/C 6/3	**Mock Exams**				

W/C 13/3	**Mock Exams**