From exam board

Inferred and suggested by teachers

Link download the specification

https://www.bbc.co.uk/bitesize/subjects/z34k7ty

https://www.computerscience.gcse.guru/

Pap

Spec Ref	Name of topic	Content				Resources to support (the links are on your google classroom version)
3.1.1	**Representing algorithms**	Understand and explain the term algorithm. Understand and explain the term decomposition. Understand and explain the term abstraction. Use a systematic approach to problem solving and algorithm creation representing those algorithms using pseudo-code, program code and flowcharts. Explain simple algorithms in terms of their inputs, processing and outputs. Determine the purpose of simple algorithms				
3.1.2	**Effciency of algorithms**	Understand that more than one algorithm can be used to solve the same problem. Compare the efficiency of algorithms explaining how some algorithms are more efficient than others in solving the same problem.				
3.1.3	**Searching algorithms**	Understand and explain how the linear search algorithm works. Understand and explain how the binary search algorithm works. Compare and contrast linear and binary search algorithms.				

3.1.4	**Sorting algorithms**	Understand and explain how the merge sort algorithm works. Understand and explain how the bubble sort algorithm works. Compare and contrast merge sort and bubble sort algorithms.				
3.2.1	**Data Types**	Understand the concept of a data type Understand and use the following appropriately: • integer • real • Boolean • character • string.				
3.2.2	**Programming concepts**	Use, understand and know how the following statement types can be combined in programs: • variable declaration • constant declaration • assignment • iteration • selection • subroutine (procedure/function). Use definite (count controlled) and indefinite (condition controlled) iteration, including indefinite iteration with the condition(s) at the start or the end of the iterative structure. Use nested selection and nested iteration structures. Use meaningful identifier names and know why it is important to use them.				

3.2.3	**Arithmetic operations in a programming language**	Be familiar with and be able to use: • addition • subtraction • multiplication • real division • integer division, including remainders				
3.2.4	**Relational operations in a programming language**	Be familiar with and be able to use: • equal to • not equal to • less than • greater than • less than or equal to • greater than or equal to				
3.2.5	**Boolean operations in a programming language**	Be familiar with and be able to use: • NOT • AND • OR.				
3.2.6	**Data structures**	Understand the concept of data structures Use arrays (or equivalent) in the design of solutions to simple problems. Use records (or equivalent) in the design of solutions to simple problems.				
3.2.7	**Input/output**	Be able to obtain user input from the keyboard. Be able to output data and information from a program to the computer display.				
3.2.8	**String handling operations in a programming language**	Understand and be able to use: • length • position • substring • concatenation • convert character to character code • convert character code to character • string conversion operations.				

3.2.9	**Random number generation in a programming language**	Be able to use random number generation.				
3.2.10	**Structured programming and subroutines (procedures and functions)**	Understand the concept of subroutines. Explain the advantages of using subroutines in programs. Describe the use of parameters to pass data within programs. Use subroutines that return values to the calling routine. Know that subroutines may declare their own variables, called local variables, and that local variables usually: • only exist while the subroutine is executing • are only accessible within the subroutine. Use local variables and explain why it is good practice to do so. Describe the structured approach to programming. Students should be able to describe the structured approach including modularised programming, clear well-documented interfaces (local variables, parameters) and return values. Explain the advantages of the structured approach.				

3.2.11	**Robust and secure programming**	Be able to write simple data validation routines. Be able to write simple authentication routines. Understand what is meant by testing in the context of algorithms and programs. Be able to correct errors within algorithms and programs. Understand what test data is and describe the following types of test data: • normal (typical) • boundary (extreme) • erroneous data. Be able to select and justify the choice of suitable test data for a given problem. Understand that there are different types of error: • syntax error • logic error Be able to identify and categorise errors within algorithms and programs.				

Pap

Spec Ref	Name of topic	Content				
3.3.1	**Number bases**	Understand the following number bases: o decimal (base 10) o binary (base 2) o hexadecimal (base 16). Understand that computers use binary to represent all data and instructions. Explain why hexadecimal is often used in computer science.				

3.3.2	**Converting between number bases**	Understand how binary can be used to represent whole numbers. Understand how hexadecimal can be used to represent whole numbers. Be able to convert in both directions between: o binary and decimal o binary and hexadecimal o decimal and hexadecimal.				
3.3.3	**Units of information**	Know that: o a bit is the fundamental unit of information o a byte is a group of 8 bits. Know that quantities of bytes can be described using prefixes. Know the names, symbols and corresponding values for the decimal prefixes: o kilo, 1 kB is 1,000 bytes o mega, 1 MB is 1,000 kilobytes o giga, 1 GB is 1,000 Megabytes o tera, 1 TB is 1,000 Gigabytes. Be able to compare quantities of bytes using the prefixes above.				
3.3.4	**Binary arithmetic**	Be able to add together up to three binary numbers. Be able to apply a binary shift to a binary number. Describe situations where binary shifts can be used.				

3.3.5	**Character encoding**	Understand what a character set is and be able to describe the following character encoding methods: o 7-bit ASCII o Unicode. Understand that character codes are commonly grouped and run in sequence within encoding tables. Describe the purpose of Unicode and the advantages of Unicode over ASCII. Know that Unicode uses the same codes as ASCII upto 127				
3.3.6	**Representing images**	Understand what a pixel is and be able to describe how pixels relate to an image and the way images are displayed. Describe the following for bitmaps: • image size • colour depth. Know that the size of a bitmap image is measured in pixels (width × height). Describe how a bitmap represents an image using pixels and colour depth. Calculate bitmap image file sizes based on the number of pixels and colour depth.				

3.3.7	**Representing sound**	Understand that sound is analogue and that it must be converted to a digital form for storage and processing in a computer. Understand that analogue signals are sampled to create the digital version of sound. Describe the digital representation of sound in terms of: o sampling rate o sample resolution. Calculate sound file sizes based on the sampling rate and the sample resolution.				
3.3.8	**Data Compression**	Explain what data compression is. Understand why data may be compressed and that there are different ways to compress data. Explain how data can be compressed using Huffman coding. Be able to interpret Huffman trees. Explain how data can be compressed using run length encoding (RLE). Represent data in RLE frequency/data pairs				

3.4.2	**Boolean logic**	Construct truth tables for the following logic gates: o NOT o AND o OR o XOR Construct truth tables for simple logic circuits using combinations of NOT, AND, OR and XOR. Interpret simple truth tables. Create, modify and interpret simple logic circuit diagrams. Students will only need to use NOT, AND, OR and XOR gates within logic circuits. Students will be expected to understand and use the logic circuit symbols (AND, OR, NOT, XOR) Create and interpret simple Boolean expressions made up of NOT (Overbar to represent not gate), AND (. to represent and gate), OR (+ to represent or gate) and XOR operations ($\oplus$ to represent xor gate). Create the Boolean expression for a simple logic circuit. Create a logic circuit from a simple Boolean expression.				

3.4.3	**Software classification**	Explain what is meant by: o system software o application software. o Give examples of both types of software. Understand the need for, and functions of, operating systems (OS) and utility programs. Understand that the OS handles management of the: o processor(s) o memory o input/output (I/O) devices o applications o security.				

3.4.4	**Classification of programming languages and translators**	Know that there are different levels of programming language: - low-level language - high-level language. Explain the main differences between low-level and high-level languages Know that machine code and assembly language are considered to be low-level languages and explain the differences between them. Understand that all programming code written in high-level or assembly languages must be translated. Understand that machine code is expressed in binary and is specific to a processor or family of processors. Understand the advantages and disadvantages of low-level language programming compared with high-level language programming. Understand that there are three common types of program translator: o interpreter o compiler o assembler. Explain the main differences between these three types of translator. Understand when it would be appropriate to use each type of translator.				

3.4.5	**Systems architecture**	Understand the different types of memory within a computer: • RAM • ROM • Cache • Register. Know what the different types of memory are used for and why they are required. Be aware of different types of secondary storage (solid state, optical and magnetic). Explain the operation of solid state, optical and magnetic storage. Discuss the advantages and disadvantages of solid state, optical and magnetic storage. Explain the term cloud storage. Explain the advantages and disadvantages of cloud storage when compared to local storage				

3.5	**Fundamentals of computer networks**	Describe the main types of computer network including: • Personal Area Network (PAN) • Local Area Network (LAN) • Wide Area Network (WAN). Understand that networks can be wired or wireless. Discuss the advantages and disadvantages of wireless networks as opposed to wired networks. Explain the purpose and use of common network protocols including: • Ethernet • Wi-Fi • TCP (Transmission Control Protocol) • UDP (User Datagram Protocol) • IP (Internet Protocol) • HTTP (Hypertext Transfer Protocol) • HTTPS (Hypertext Transfer Protocol Secure) • FTP (File Transfer Protocol) • email protocols: • SMTP (Simple Mail Transfer Protocol) • IMAP (Internet Message Access Protocol). Understand that the HTTP, HTTPS, SMTP, IMAP and FTP protocols operate at the application layer. Understand that the TCP and UDP protocols operate at the transport layer.				

3.6.2	**Cyber security threats**	Understand and be able to explain the following cyber security threats: o social engineering techniques o malicious code (malware) o pharming o weak and default passwords o misconfigured access rights o removable media o unpatched and/or outdated software. Explain what penetration testing is and what it is used for. Define the term social engineering. Describe what social engineering is and how it can be protected against. Explain the following forms of social engineering: o blagging (pretexting) o phishing o shouldering (or shoulder surfing). Define the term malware. Describe what malware is and how it can be protected against. Describe the following forms of malware: o computer virus o trojan o spyware.				
3.6.3	**Methods to detect and prevent cyber security threats**	Understand and be able to explain the following security measures: o biometric measures (particularly for mobile devices) o password systems o CAPTCHA (or similar) o using email confirmations to confirm a user's identity o automatic software updates.				

3.7	**Relational databases and structured query language (SQL)**	Explain the concept of a database. Explain the concept of a relational database. Understand the following database concepts: o table o record o field o primary key o foreign key. Understand that the use of a relational database facilitates the elimination of data inconsistency and data redundancy. Be able to use SQL to retrieve data from a relational database, using the commands: o SELECT o FROM o WHERE o ORDER BY…ASC | DESC Be able to use SQL to insert data into a relational database using the commands. INSERT INTO table_name (column1, column 2 …) VALUES (value1, value2 …) Be able to use SQL to edit and delete data in a database using the commands. UPDATE table_name SET column1 = value1, column2 = value2 ... WHERE condition				

3.8	**Ethical, legal and environmental impacts of digital technology on wider society, including issues of privacy**	Explain the current ethical impacts and risks of digital technology on society. Where data privacy issues arise these should be considered. This covers: - Computers in healthcare and if that is ethically sound - Computer based implants ie quadraplegic people being able to walk - Wearable technologies - Driverless vehicles - What is the correct ethical decision?! Explain the current environmental impacts and risks of digital technology on society. Where data privacy issues arise these should be considered. This covers: - e-waste - monitoring the planet/environment - Impacts of cloud storage on the environment - Wireless technology - good for us/planet?" Explain the current legal impacts and risks of digital technology on society. Where data privacy issues arise these should be considered. This covers: - General Data Protection Regulation (GDPR) - The Computer Misuse Act 1990 - Privacy issues Exam questions will be taken from the following areas: o cyber security o mobile technologies o wireless networking o cloud storage				