GCSE Computer Science Personal Learning Checklist

Paper 1: Computer Systems

Topic	Red	Amber	Green
1.1.1 The purpose of the CPU			
I know the purpose of the CPU.			
I can describe the fetch-execute cycle.			
1.1.1 Common CPU components and their function			
I know the function of the ALU (Arithmetic Logic Unit).			
I know the function of the CU (Control Unit).			
I know the function of cache memory.			
I know the purpose of registers.			
1.1.1 Von Neumann architecture			
I know the role of the MAR (Memory Address Register).			
I know the role of the MDR (Memory Data Register).			
I know the role of the program counter.			
I know the role of the accumulator.			
1.1.2 CPU performance			
I can explain how clock speed affects CPU performance.			
I can explain how cache size affects CPU performance.			
I can explain how the number of cores affects CPU performance.			
1.1.3 Embedded systems			
I know the purpose and characteristics of embedded systems.			
I can give examples of embedded systems.			
1.2.1 Primary storage (Memory)			
I know the need for primary storage.			
I can explain the difference between RAM and ROM.			
I know the purpose of ROM in a computer system.			

I know the purpose of RAM in a computer system.			
I can explain why virtual memory may be needed in a system.			
I can explain how virtual memory works.			
1.2.2 Secondary storage			
I know the need for secondary storage.			
I can describe common types of storage: optical, magnetic, and solid state.			
I can recommend suitable storage devices and media for a given application.			
I can explain the advantages and disadvantages of different storage devices and media.			
1.2.3 Units			
I know the units of data storage (bit, nibble, byte, kilobyte, megabyte, gigabyte, terabyte, etc.).			
I can convert between data units.			
I know why data must be stored in binary format.			
I can explain how data is converted into binary format for processing.			
1.2.4 Data storage: Numbers			
I can convert positive denary numbers to binary and vice versa (up to 8 bits).			
I can add binary integers and explain overflow errors.			
I can convert denary numbers to hexadecimal and vice versa.			
I can convert binary integers to hexadecimal and vice versa.			
I can explain binary shifts.			
1.2.4 Data storage: Characters			
I know how binary codes represent characters.			
I know the term 'character set' and examples like ASCII and Unicode.			
I can calculate text file size using bits per character and the number of characters.			
1.2.4 Data storage: Images			
I know how an image is represented as a series of pixels in binary.			
I can explain metadata for images.			
I can describe the effect of colour depth and resolution on image quality and file size.			

1.2.4 Data storage: Sound			
I know how sound is sampled and stored in digital form.			
I can explain sample rate, bit depth, and their effects on quality and file size.			
1.2.5 Compression			
I know the need for compression and its types (lossy, lossless).			
I can describe the advantages and disadvantages of lossy and lossless compression.			
1.3.1 Networks and topologies			
I know the types of networks (LAN, WAN) and their characteristics.			
I can explain factors affecting network performance.			
I know the roles of hardware needed for LANs (e.g., switches, routers, NIC).			
I can describe star and mesh topologies and their advantages/disadvantages.			
1.3.2 Wired and wireless networks			
a) I understand the modes of connection:			
i. I can explain wired Ethernet connections and their characteristics.			
ii. I understand wireless connections using Wi-Fi and their uses.			
iii. I can describe Bluetooth connections and their typical applications.			
b) I can compare the benefits and drawbacks of wired versus wireless connections.			
c) I can recommend suitable connections for given scenarios.			
d) I know the importance of encryption in securing network communications.			
e) I understand IP addressing (both IPv4 and IPv6) and its purpose in networks.			
f) I can explain MAC addressing and its role in identifying devices on a network.			
g) I know the purpose of standards in computing and how they provide rules for interoperability.			
h) I can describe how standards ensure compatibility across hardware and software.			
j) I understand common network protocols, including:			
i. TCP/IP (Transmission Control Protocol/Internet Protocol)			
ii. HTTP (Hyper Text Transfer Protocol)			
iii. HTTPS (Hyper Text Transfer Protocol Secure)			

iv. FTP (File Transfer Protocol)			
v. POP (Post Office Protocol)			
vi. IMAP (Internet Message Access Protocol)			
vii. SMTP (Simple Mail Transfer Protocol)			
k) I understand the concept of layers in network protocols and their benefits (e.g., the 4-layer TCP/IP model).			
1.4 Network security			
1.4.1 Common threats to systems and networks			
I can describe types of malware (e.g., viruses, worms, Trojan horses) and their effects on systems.			
I know what phishing is and how to identify phishing attempts.			
I can explain brute force attacks and their impact on system security.			
I know what Distributed Denial of Service (DDoS) attacks are and their implications.			
I can describe SQL injection attacks and how they exploit vulnerabilities in databases.			
I know what data interception is and how it compromises data confidentiality.			
I can identify insider threats and how they pose risks to organizations.			
1.4.2 Prevention methods			
I can describe the purpose of firewalls and how they filter network traffic.			
I know how anti-malware software detects and removes malicious software.			
I understand the importance of encryption in securing data during transmission.			
I can explain penetration testing and how it identifies system vulnerabilities.			
I know what user access levels are and how they prevent unauthorized access.			
I can describe secure password practices and their role in system security.			
I know the importance of network policies in defining security protocols and procedures.			
1.5 Systems software			
1.5.1 The purpose and functionality of operating systems:			
a) I can explain the purpose and functionality of an operating system.			
i. I understand how operating systems provide user interfaces, such as graphical (GUI) and command line (CLI).			
ii. I know how memory management enables multitasking by allocating and deallocating memory spaces.			

iii. I understand how operating systems manage peripherals and the role of drivers in communication.			
b) I can describe user management features.			
i. I understand how accounts are allocated to users.			
ii. I can explain how access rights are assigned to ensure security.			
iii. I can identify security features, such as password policies and biometrics, in user management.			
c) I can describe file management processes.			
i. I know how files are named and organized into folders.			
ii. I can explain how files are moved, copied, and saved.			
iii. I understand how file structures are organized to improve accessibility.			
d) I know how processes are managed, including the use of buffers for data transfer (e.g., to printers).			
1.5.2 Utility software:			
a) I can explain the purpose and functionality of utility software.			
i. I understand how encryption software secures data by encoding it.			
ii. I can describe defragmentation and its role in optimizing hard drive performance.			
iii. I know how data compression reduces file sizes and the differences between lossy and lossless compression.			
1.6 Ethical, legal, cultural, and environmental impacts of digital technology			
I know the ethical issues surrounding the development and use of digital technology.			
I can describe the legal implications of digital technology use and misuse			
I understand cultural considerations when developing and implementing digital technologies.			
I can explain the environmental impact of producing and disposing of digital devices.			
I know the privacy issues related to data collection, storage, and sharing.			
I can give examples of how digital technology affects wider society (e.g., automation, job markets).			
I understand the key features of relevant legislation, including the Data Protection Act 2018.			
I can explain the Computer Misuse Act 1990 and how it prevents unauthorised access to systems.			
I know the Copyright, Designs, and Patents Act 1988 and its relevance to software and media.			

I can compare open source and proprietary software licenses, including their benefits and drawbacks.			
I know why software licenses are necessary and what they aim to protect.			
I can identify scenarios where open source or proprietary software is more appropriate.			
I understand the ethical dilemmas faced by developers and users of digital technology.			
I can describe the importance of balancing innovation with ethical and legal responsibilities.			
I know how digital technology can influence cultural exchange and global interactions.			

Paper2: Computational Thinking

2.1 – Algorithms	Red	Amber	Green
2.1.1 a I know the principles of computational thinking.			
2.1.1 a i I know about abstraction.			
2.1.1 a ii I know about decomposition.			
2.1.1 a iii I know about algorithmic thinking.			
2.1.2 a I can identify inputs, processes, and outputs for a problem.			
2.1.2 b I can use structure diagrams to plan solutions.			
2.1.2 c i I can create, interpret, correct, complete, and refine algorithms using pseudocode.			
2.1.2 c ii I can create, interpret, correct, complete, and refine algorithms using flowcharts.			
2.1.2 c iii I can create, interpret, correct, complete, and refine algorithms using a reference/high-level programming language.			
2.1.2 d I can identify syntax errors and suggest fixes.			

2.1.2 e I can identify logic errors and suggest fixes.			
2.1.2 f I can create and use trace tables to follow an algorithm.			
2.1.2 g I can refine algorithms.			
2.1.3 a i I know how a binary search works.			
2.1.3 a ii I know how a linear search works.			
2.1.3 b i I know how a bubble sort works.			
2.1.3 b ii I know how a merge sort works.			
2.1.3 b iii I know how an insertion sort works.			
2.2 – Programming Fundamentals			
2.2.1 a I can use variables in programming.			
2.2.1 b I can use constants in programming.			
2.2.1 c I can use operators in programming.			
2.2.1 d I can use inputs in programming.			
2.2.1 e I can use outputs in programming.			
2.2.1 f I can use assignments in programming.			
2.2.1 g i I can use sequence as a basic programming construct.			
2.2.1 g ii I can use selection as a basic programming construct.			
2.2.1 g iii I can use iteration as a basic programming construct.			
2.2.1 g iv I can use count-controlled loops (e.g., for loops).			
2.2.1 g v I can use condition-controlled loops (e.g., while loops, repeat-until loops).			

2.2.1 h I know the common arithmetic operators.			
2.2.1 j I know the common Boolean operators (AND, OR, NOT).			
2.2.1 k i I can use comparison operators (e.g., == Equal to) and arithmetic operators (e.g., + Addition).			
2.2.1 k ii I can use comparison operators (e.g., != Not equal to) and arithmetic operators (e.g., – Subtraction).			
2.2.1 k iii I can use comparison operators (e.g., < Less than) and arithmetic operators (e.g., * Multiplication).			
2.2.1 k iv I can use comparison operators (e.g., <= Less than or equal to) and arithmetic operators (e.g., / Division).			
2.2.1 k v I can use comparison operators (e.g., > Greater than) and arithmetic operators (e.g., MOD Modulus).			
2.2.1 k vi I can use comparison operators (e.g., >= Greater than or equal to) and arithmetic operators (e.g., DIV Quotient).			
2.2.1 k vii I can use the exponentiation operator (e.g., ^ To the power).			
2.2.2 a i I can use integer data types.			
2.2.2 a ii I can use real data types.			
2.2.2 a iii I can use Boolean data types.			
2.2.2 a iv I can use character and string data types.			
2.2.2 a v I can perform data type casting.			
2.2.3 a i I can use string concatenation.			
2.2.3 a ii I can use string slicing.			
2.2.3 b i I can use basic file handling operations (e.g., open a file).			
2.2.3 b ii I can use basic file handling operations (e.g., read a file).			
2.2.3 b iii I can use basic file handling operations (e.g., write to a file).			
2.2.3 b iv I can use basic file handling operations (e.g., close a file).			
2.2.3 c I can use records to store data.			
2.2.3 d I can use SQL to search for data.			
2.2.3 e I can use arrays as fixed-length static structures when solving problems.			
2.2.3 f I can use 2D arrays as fixed-length static structures when solving problems.			
2.2.3 g I know how to use subprograms (procedures) to produce structured code.			
2.2.3 h I know how to use subprograms (functions) to produce structured code.			

2.2.3 j I can use random number generation in my programs.			
2.2.3 k i I know how to use the SQL SELECT command.			
2.2.3 k ii I know how to use the SQL FROM command.			
2.2.3 k iii I know how to use the SQL WHERE command.			
2.3 – Producing Robust Programs			
2.3.1 a i I can anticipate misuse and invalid data in my programs.			
2.3.1 a ii I can implement authentication to confirm a user's identity.			
2.3.1 b i I can design input validation (e.g., length checks).			
2.3.1 b ii I can design input validation (e.g., range checks).			
2.3.1 b iii I can design input validation (e.g., presence checks).			
2.3.1 c I can design input validation and authentication systems (e.g., username/password).			
2.3.1 d i I can use subprograms to make code maintainable.			
2.3.1 d ii I can use proper naming conventions to make code maintainable.			
2.3.1 d iii I can use indentation to make code maintainable.			
2.3.1 d iv I can use comments to make code maintainable.			
2.3.2 a I know the purpose of testing.			
2.3.2 b i I know how to carry out iterative (module/unit) testing.			
2.3.2 b ii I know how to carry out final/terminal testing.			
2.3.2 c I can identify and fix syntax errors.			
2.3.2 d I can identify and fix logic errors.			
2.3.2 e i I can design test plans with normal test data.			
2.3.2 e ii I can design test plans with boundary test data.			
2.3.2 e iii I can design test plans with invalid test data.			
2.3.2 e iv I can design test plans with erroneous test data.			
2.4.1 a I can create simple logic diagrams using the AND operator.			
2.4.1 b I can create simple logic diagrams using the OR operator.			
2.4.1 c I can create simple logic diagrams using the NOT operator.			

2.4.1 d I can construct truth tables.			
2.4.1 e I can combine Boolean operators using AND, OR, and NOT.			
2.4.1 f I can apply logical operators in truth tables to solve problems.			
2.5 – Programming Languages and IDEs			
2.5.1 a i I know the characteristics and purpose of high-level programming languages.			
2.5.1 a ii I know the characteristics and purpose of low-level programming languages.			
2.5.1 b i I know the characteristics of a compiler.			
2.5.1 b ii I know the characteristics of an interpreter.			
2.5.2 a i I know how to use editors in an IDE.			
2.5.2 a ii I know how to use error diagnostics in an IDE.			
2.5.2 a iii I know how to use a runtime environment in an IDE.			