AS & A'Level AQA Computer Science Revision check list

Topic 1 : Programming basics

Video revision link:

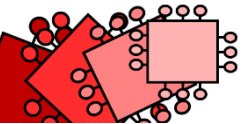
Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Understand the concept of a data type.	☐	☐	☐
Understand and use the following appropriately: Integer, real/float, Boolean, character, string, data/time, pointer/reference, records (or equivalent), arrays (or equivalent)	☐	☐	☐
Define and use user-defined data types based on language-defined (built-in) data types.	☐	☐	☐
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 and indefinite iteration, including indefinite iteration with the condition(s) at the start or the end of the iterative structure. A theoretical understanding of condition(s) at either end of an iterative structure is required, regardless of whether they are supported by the language being used.	☐	☐	☐
Use nested selection and nested iteration structures.	☐	☐	☐
Use meaningful identifier names and know why it is important to use them.	☐	☐	☐
Be familiar with and be able to use: Addition, subtraction, multiplication, real/float, integer division, including remainders, exponetation, rounding, truncation	☐	☐	☐
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	☐	☐	☐
Be familiar with and be able to use: NOT, AND, OR, XOR	☐	☐	☐
Be able to explain the differences between a variable and a constant.	☐	☐	☐
Be able to explain the advantages of using named constants.	☐	☐	☐
Be familiar with and be able to use: Length, position, substring, concatenation, character ==> character code, character code ==> character, string conversion operations	☐	☐	☐
Be familiar with, and be able to use, random number generation.	☐	☐	☐

Topic 2 : Programming next steps

Video revision link:

Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Be familiar with the concept of exception handling.	☐	☐	☐
Know how to use exception handling in a programming language with which students are familiar.	☐	☐	☐
Be familiar with subroutines and their uses.	☐	☐	☐
Know that a subroutine is a named 'out of line' block of code that may be executed (called) by simply writing its name in a program statement.	☐	☐	☐
Be able to explain the advantages of using subroutines in programs.	☐	☐	☐
Be able to describe the use of parameters to pass data within programs.	☐	☐	☐
Be able to use subroutines with interfaces.	☐	☐	☐
Be able to use subroutines that return values to the calling routine.	☐	☐	☐
Known that subroutines may declare their own variables, called local variables, and that local variables: Exist only while the subrountine is executing, are accessible only within the subroutine	☐	☐	☐
Be able to use local variables and explain why it is good practice to do so.	☐	☐	☐
Be able to contrast local variables with global variables.	☐	☐	☐

Specific knowledge required for A Level ONLY:	Need to Revise	Revised Once	Got it!
Be able to explain how a stack frame is used with subrountine calls to store: Return address, parameters, local variables	☐	☐	☐



Be familiar with the use of recursive techniques in programming languages (general and base cases and the mechanism for implementation).	☐	☐	☐
Be able to solve simple problems using recursion.	☐	☐	☐

Topic 3 : Programming paradigms

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Understand the structured approach to program design and construction.	☐	☐	☐
Be able to construct and use hierarchy charts when designing programs.	☐	☐	☐
Be able to explain the advantages of the structured approach.	☐	☐	☐

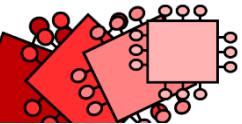
Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Understand the characteristics of the procedural- and object-oriented programming paradigms, and have experience of programming in each.	☐	☐	☐
Be familiar with the concepts of: Class, object, instantiation, encapsulation, inheritance, aggregation, composition, polymorphism, overriding	☐	☐	☐
Know why the object-oriented paradigm is used	☐	☐	☐
Be aware of the following object-oriented design principles: Encapsulate what varies, favour composition over inheritance, program to interface, not implementation	☐	☐	☐
Be able to write object-oriented programs.	☐	☐	☐
Be able to draw and interpret class diagrams	☐	☐	☐

Topic 4 : Data structures

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Be familiar with the concept of data structures.	☐	☐	☐
Use arrays (or equivalent) in the design of solutions to simple problems.	☐	☐	☐
Be able to read/write from/to a text file	☐	☐	☐
Be able to read/write data from/to a binary(nontext) file	☐	☐	☐

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Be familiar with the concept of a: Queue, stack, graph, tree, hash table, dictionary, vector	☐	☐	☐
Be able to distinguish between static and dynamic structures and compare their uses, as well as explaining the advantages and disadvantages of each.	☐	☐	☐
Describe the creation and maintenance of data within: Queues (linear, circular, priority), stacks, hash tables	☐	☐	☐
Be able to describe and apply the following to linear queues, circular queues and priority queues: Add an item, remove an item, test for an empty queue, test for a full queue	☐	☐	☐
Be able to describe and apply the following operations: Push, pop, peek or top, test for empty stack, test for stack full	☐	☐	☐
Be aware of a graph as a data structure used to represent more complex relationships.	☐	☐	☐
Be familiar with typical uses for graphs.	☐	☐	☐
Be able to explain: Graph, weighted graph, vertex/node, edge/arc, undirected graph, directed graph	☐	☐	☐
Know how an adjacency matrix and an adjacency list may be used to represent a graph.	☐	☐	☐
Be able to compare the use of adjacency matrices and adjacency lists.	☐	☐	☐
Know that a tree is a connected, undirected graph with no cycles.	☐	☐	☐



Know that a rooted tree is a tree in which one vertex has been designated as the root. A rooted tree has parent-child relationships between nodes. The root is the only node with no parent and all other nodes are descendants of the root.	☐	☐	☐
Know that a binary tree is a rooted tree in which each node has at most two children.	☐	☐	☐
Be familiar with typical uses for rooted trees.	☐	☐	☐
Be familiar with the concept of a hash table and its uses.	☐	☐	☐
Be able to apply simple hashing algorithms.	☐	☐	☐
Know what is meant by a collision and how collisions are handled using rehashing.	☐	☐	☐
Be familiar with the concept of a dictionary.	☐	☐	☐
Be familiar with simple applications of dictionaries, for example information retrieval, and have experience of using a dictionary data structure in a programming language.	☐	☐	☐
Be familiar with the concept of a vector and the following notations for specifying a vector: • [2.0, 3.14159, -1.0, 2.718281828] • 4-vector over $\mathbb{R}$ written as $\mathbb{R}4$ • function interpretation • 0 ↦ 2.0 • 1 ↦ 3.14159 • 2 ↦ -1.0 • 3 ↦ 2.718281828 • ↦ means maps to, That all the entries must be drawn from the same field, eg $\mathbb{R}$.			
Dictionary representation of a vector.			
List representation of a vector.			
1-D array representation of a vector.			
Visualising a vector as an arrow.			
Vector addition and scalar-vector multiplication.			
Convex combination of two vectors, u and v.			
Dot or scalar product of two vectors.			
Applications of dot product.			

Topic 5 : Algorithms

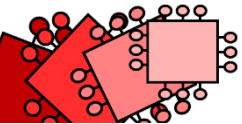
Video revision link:

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Be able to trace breadth-first and depthfirst search algorithms and describe typical applications of both.	☐	☐	☐
Be able to trace the tree-traversal algorithms: • pre-order • post-order • in-order.	☐	☐	☐
Be able to describe uses of tree-traversal algorithms.	☐	☐	☐
Be able to convert simple expressions in infix form to Reverse Polish notation (RPN) form and vice versa. Be aware of why and where it is used.	☐	☐	☐
Know and be able to trace and analyse the complexity of the linear search algorithm.	☐	☐	☐
Know and be able to trace and analyse the time complexity of the binary search algorithm.	☐	☐	☐
Be able to trace and analyse the time complexity of the binary tree search algorithm.	☐	☐	☐
Know and be able to trace and analyse the time complexity of the bubble sort algorithm.	☐	☐	☐
Be able to trace and analyse the time complexity of the merge sort algorithm.	☐	☐	☐
Understand and be able to trace Dijkstra's shortest path algorithm. Be aware of applications of shortest path algorithm.	☐	☐	☐

Topic 6 : Abstraction and automation

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Be able to develop solutions to simple logic problems.	☐	☐	☐
Be able to check solutions to simple logic problems.	☐	☐	☐
Understand the term algorithm.	☐	☐	☐



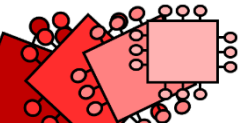
Be able to express the solution to a simple problem as an algorithm using pseudo-code, with the standard constructs: • sequence • assignment • selection • iteration.	☐	☐	☐
Be able to hand-trace algorithms.	☐	☐	☐
Be able to convert an algorithm from pseudocode into high level language program code.	☐	☐	☐
Be able to articulate how a program works, arguing for its correctness and its efficiency using logical reasoning, test data and user feedback.	☐	☐	☐
Be familiar with the concept of abstraction as used in computations and know that: • representational abstraction is a representation arrived at by removing unnecessary details • abstraction by generalisation or categorisation is a grouping by common characteristics to arrive at a hierarchical relationship of the 'is a kind of' type.	☐	☐	☐
Be familiar with the process of hiding all details of an object that do not contribute to its essential characteristics.	☐	☐	☐
Know that procedural abstraction represents a computational method.	☐	☐	☐
Know that for functional abstraction the particular computation method is hidden.	☐	☐	☐
Know that details of how data are actually represented are hidden, allowing new kinds of data objects to be constructed from previously defined types of data objects.	☐	☐	☐
Know that details are removed until the problem is represented in a way that is possible to solve, because the problem reduces to one that has already been solved.	☐	☐	☐
Know that procedural decomposition means breaking a problem into a number of sub-problems, so that each sub-problem accomplishes an identifiable task, which might itself be further subdivided.	☐	☐	☐
Know how to build a composition abstraction by combining procedures to form compound procedures.	☐	☐	☐
Know how to build data abstractions by combining data objects to form compound data, for example tree data structure.	☐	☐	☐
Understand that automation requires putting models (abstraction of real world objects/ phenomena) into action to solve problems. This is achieved by: • creating algorithms • implementing the algorithms in program code (instructions) • implementing the models in data structures • executing the code.	☐	☐	☐

Topic 7 : Regular & context free languages

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Be able to draw and interpret simple state transition diagrams and state transition tables for FSMs with no output and with output (Mealy machines only).	☐	☐	☐

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Be familiar with the concept of a set and the following notations for specifying a set: A = {1, 2, 3, 4, 5 } or set comprehension: A = {x | x ∈ ℕ ∧ x ≥ 1 } where A is the set consisting of those objects x such that x ∈ ℕ and x ≥ 1 is true. Know that the empty set, {}, is the set with no elements. Know that an alternative symbol for the empty set is Ø.	☐	☐	☐
Be familiar with the compact representation of a set, for example, the set {0n1n | n ≥ 1}. This set contains all strings with an equal number of 0 s and 1s.	☐	☐	☐
Be familiar with the concept of: • finite sets • infinite sets	☐	☐	☐



• countably infinite sets • cardinality of a finite set • Cartesian product of sets.			
Be familiar with the meaning of the term: • subset • proper subset • countable set.	☐	☐	☐
Be familiar with the set operations: • membership • union • intersection • difference.	☐	☐	☐
Know that a regular expression is simply a way of describing a set and that regular expressions allow particular types of languages to be described in a convenient shorthand notation.	☐	☐	☐
Be able to form and use simple regular expressions for string manipulation and matching.	☐	☐	☐
Be able to describe the relationship between regular expressions and FSMs.	☐	☐	☐
Be able to write a regular expression to recognise the same language as a given FSM and vice versa.	☐	☐	☐
Know that a language is called regular if it can be represented by a regular expression.	☐	☐	☐
Be able to check language syntax by referring to BNF or syntax diagrams and formulate simple production rules.	☐	☐	☐
Be able to explain why BNF can represent some languages that cannot be represented using regular expressions.	☐	☐	☐

Topic 8 : Classification of algorithms

Video revision link:

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Understand that algorithms can be compared by expressing their complexity as a function relative to the size of the problem. Understand that the size of the problem is the key issue.	☐	☐	☐
Understand that some algorithms are more efficient: • time-wise than other algorithms • space-wise than other algorithms.	☐	☐	☐
Be familiar with the mathematical concept of a function as a mapping from one set of values, the domain, to another set of values, drawn from the co-domain, for example $\mathbb{N} \rightarrow \mathbb{N}$.	☐	☐	☐
Be familiar with the concept of: • a linear function, for example y = 2x • a polynomial function, for example y = 2x2 • an exponential function, for example y = 2x • a logarithmic function, for example y = log10 x.	☐	☐	☐
Be familiar with the notion of permutation of a set of objects or values, for example, the letters of a word and that the number of permutations of n distinct objects is n factorial (n!).	☐	☐	☐
Be familiar with Big-O notation to express time complexity and be able to apply it to cases where the running time requirements of the algorithm grow in: • constant time • logarithmic time • linear time • polynomial time • exponential time.	☐	☐	☐
Be able to derive the time complexity of an algorithm.	☐	☐	☐
Be aware that algorithmic complexity and hardware impose limits on what can be computed.	☐	☐	☐
Know that algorithms may be classified as being either: • tractable – problems that have a polynomial (or less) time solution are called tractable problems. • intractable – problems that have no polynomial (or less) time solution are called intractable problems.	☐	☐	☐

Be aware that some problems cannot be solved algorithmically.	☐	☐	☐
Describe the Halting problem (but not prove it), that is the unsolvable problem of determining whether any program will eventually stop if given particular input.	☐	☐	☐
Understand the significance of the Halting problem for computation.	☐	☐	☐

Topic 9 : A model of computation

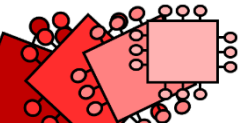
Video revision link:

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Be familiar with the structure and use of Turing machines that perform simple computations.	☐	☐	☐
Know that a Turing machine can be viewed as a computer with a single fixed program, expressed using: • a finite set of states in a state transition diagram • a finite alphabet of symbols • an infinite tape with marked-off squares • a sensing read-write head that can travel along the tape, one square at a time. One of the states is called a start state and states that have no outgoing transitions are called halting states.	☐	☐	☐
Understand the equivalence between a transition function and a state transition diagram.	☐	☐	☐
Be able to: • represent transition rules using a transition function • represent transition rules using a state transition diagram • hand-trace simple Turing machines.	☐	☐	☐
Be able to explain the importance of Turing machines and the Universal Turing machine to the subject of computation.	☐	☐	☐

Topic 10 : Number system & bases

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Be familiar with the concept of a natural number and the set $\mathbb{N}$ of natural numbers (including zero).	☐	☐	☐
Be familiar with the concept of an integer and the set $\mathbb{Z}$ of integers.	☐	☐	☐
Be familiar with the concept of a rational number and the set $\mathbb{Q}$ of rational numbers, and that this set includes the integers.	☐	☐	☐
Be familiar with the concept of an irrational number.	☐	☐	☐
Be familiar with the concept of a real number and the set $\mathbb{R}$ of real numbers, which includes the natural numbers, the rational numbers and the irrational numbers.	☐	☐	☐
Be familiar with the concept of ordinal numbers and their use to describe the numerical positions of objects.	☐	☐	☐
Be familiar with the use of: • natural numbers for counting • real numbers for measurement.	☐	☐	☐
Be familiar with the concept of a number base, in particular: • decimal (base 10) • binary (base 2) • hexadecimal (base 16).	☐	☐	☐
Convert between decimal, binary and hexadecimal number bases.	☐	☐	☐
Be familiar with, and able to use, hexadecimal as a shorthand for binary and to understand why it is used in this way.	☐	☐	☐
Know that: • the bit is the fundamental unit of information	☐	☐	☐



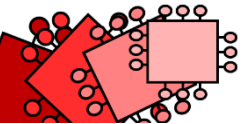
• a byte is a group of 8 bits. Know that the 2n different values can be represented with n bits.			
Know that the 2n different values can be represented with n bits.	☐	☐	☐
Know that quantities of bytes can be described using binary prefixes representing powers of 2 or using decimal prefixes representing powers of 10, eg one kibibyte is written as 1KiB = 210 B and one kilobyte is written as 1 kB = 103 B.	☐	☐	☐
Know the names, symbols and corresponding powers of 2 for the binary prefixes: • kibi, Ki - 210 • mebi, Mi - 220 • gibi, Gi - 230 • tebi, Ti - 240	☐	☐	☐
Know the names, symbols and corresponding powers of 10 for the decimal prefixes: • kilo, k - 103 • mega, M - 106 • giga, G - 109 • tera, T - 1012	☐	☐	☐

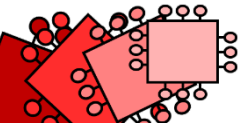
Topic 11 : Binary

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Know the difference between unsigned binary and signed binary.	☐	☐	☐
Know that in unsigned binary the minimum and maximum values for a given number of bits, n, are 0 and 2n -1 respectively.	☐	☐	☐
Be able to: • add two unsigned binary integers • multiply two unsigned binary integers.	☐	☐	☐
Know that signed binary can be used to represent negative integers and that one possible coding scheme is two's complement.	☐	☐	☐
Know how to: • represent negative and positive integers in two's complement • perform subtraction using two's complement • calculate the range of a given number of bits, n.	☐	☐	☐
Know how numbers with a fractional part can be represented in: • fixed point form in binary in a given number of bits • floating point form in binary in a given number of bits.	☐	☐	☐
Be able to convert for each representation from: • decimal to binary of a given number of bits • binary to decimal of a given number of bits.	☐	☐	☐

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Know and be able to explain why both fixed point and floating point representation of decimal numbers may be inaccurate.	☐	☐	☐
Be able to calculate the absolute error of numerical data stored and processed in computer systems.	☐	☐	☐
Be able to calculate the relative error of numerical data stored and processed in computer systems.	☐	☐	☐
Compare absolute and relative errors for large and small magnitude numbers, and numbers close to one.	☐	☐	☐
Compare the advantages and disadvantages of fixed point and floating point forms in terms of range, precision and speed of calculation.	☐	☐	☐
Know why floating point numbers are normalised and be able to normalise un-normalised floating point numbers with positive or negative mantissas.	☐	☐	☐
Explain underflow and overflow and describe the circumstances in which they occur.	☐	☐	☐





AS & A'Level AQA Computer Science Revision check list

Topic 12 : Coding text & graphics

Video revision link:

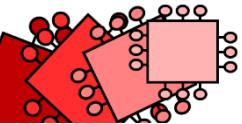
Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Differentiate between the character code representation of a decimal digit and its pure binary representation.	☐	☐	☐
Describe ASCII and Unicode coding systems for coding character data and explain why Unicode was introduced.	☐	☐	☐
Describe and explain the use of: • parity bits • majority voting • checksums • check digits.	☐	☐	☐
Describe how bit patterns may represent other forms of data, including graphics and sound.	☐	☐	☐
Understand the difference between analogue and digital: • data • signals.	☐	☐	☐
Describe the principles of operation of: • an analogue to digital converter (ADC) • a digital to analogue converter (DAC).	☐	☐	☐
Explain how bitmaps are represented.	☐	☐	☐
Explain the following for bitmaps: • resolution • colour depth • size in pixels.	☐	☐	☐
Calculate storage requirements for bitmapped images and be aware that bitmap image files may also contain metadata.	☐	☐	☐
Be familiar with typical metadata.	☐	☐	☐

Specific knowledge required for A Level ONLY:	Need to Revise	Revised Once	Got it!
Know that ADCs are used with analogue sensors	☐	☐	☐
Know that the most common use for a DAC is to convert a digital audio signal to an analogue signal.	☐	☐	☐
Explain how vector graphics represents images using lists of objects.	☐	☐	☐
Give examples of typical properties of objects.	☐	☐	☐
Use vector graphic primitives to create a simple vector graphic.	☐	☐	☐
Compare the vector graphics approach with the bitmapped graphics approach and understand the advantages and disadvantages of each.	☐	☐	☐
Be aware of appropriate uses of each approach.	☐	☐	☐

Topic 13 : Coding sound & music

Video revision link:

Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Describe the digital representation of sound in terms of: • sample resolution • sampling rate and the Nyquist theorem.	☐	☐	☐
Calculate sound sample sizes in bytes.	☐	☐	☐
Describe the purpose of MIDI and the use of event messages in MIDI.	☐	☐	☐
Describe the advantages of using MIDI files for representing music.	☐	☐	☐
Know why images and sound files are often compressed and that other files, such as text files, can also be compressed.	☐	☐	☐
Understand the difference between lossless and lossy compression and explain the advantages and disadvantages of each.	☐	☐	☐



Explain the principles behind the following techniques for lossless compression: • run length encoding (RLE) • dictionary-based methods.	☐	☐	☐
Understand what is meant by encryption and be able to define it.	☐	☐	☐
Be familiar with Caesar cipher and be able to apply it to encrypt a plaintext message and decrypt a ciphertext. Be able to explain why it is easily cracked.	☐	☐	☐
Be familiar with Vernam cipher or one-time pad and be able to apply it to encrypt a plaintext message and decrypt a ciphertext. Explain why Vernam cipher is considered as a cypher with perfect security.	☐	☐	☐
Compare Vernam cipher with ciphers that depend on computational security.	☐	☐	☐

Topic 14 : Hardware & software

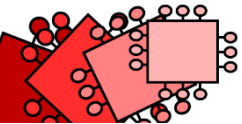
Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Understand the relationship between hardware and software and be able to define the terms: • hardware • software.	☐	☐	☐
Explain what is meant by: • system software • application software.	☐	☐	☐
Understand the need for, and attributes of, different types of software.	☐	☐	☐
Understand the need for, and functions of the folowing system software: • operating systems (OSs) • utility programs • libraries • translators (compiler, assembler, interpreter).	☐	☐	☐
Understand that a role of the operating system is to hide the complexities of the hardware.	☐	☐	☐
Know that the OS handles resource management, managing hardware to allocate processors, memories and I/O devices among competing processes.	☐	☐	☐

Topic 15 : Programming languages & translators

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Show awareness of the development of types of programming languages and their classification into low-and high-level languages.	☐	☐	☐
Know that low-level languages are considered to be: • machine-code • assembly language.	☐	☐	☐
Know that high-level languages include imperative high-level language.	☐	☐	☐
Describe machine-code language and assembly language.	☐	☐	☐
Understand the advantages and disadvantages of machine-code and assembly language programming compared with high-level language programming.	☐	☐	☐
Explain the term 'imperative high-level language' and its relationship to low-level languages.	☐	☐	☐
Understand the role of each of the following: • assembler • compiler • interpreter. Explain the differences between compilation and interpretation. Describe situations in which each would be appropriate.	☐	☐	☐



Explain why an intermediate language such as bytecode is produced as the final output by some compilers and how it is subsequently used.	☐	☐	☐
Understand the difference between source code and object (executable) code.	☐	☐	☐

Topic 16 : Logic gates & Boolean algebra

Video revision link:

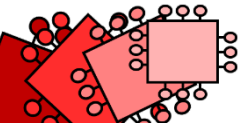
Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Construct truth tables for the following logic gates: • NOT • AND • OR • XOR • NAND • NOR.	☐	☐	☐
Be familiar with drawing and interpreting logic gate circuit diagrams involving one or more of the above gates.	☐	☐	☐
Complete a truth table for a given logic gate circuit.	☐	☐	☐
Write a Boolean expression for a given logic gate circuit.	☐	☐	☐
Draw an equivalent logic gate circuit for a given Boolean expression.	☐	☐	☐
Be familiar with the use of Boolean identities and De Morgan's laws to manipulate and simplify Boolean expressions.	☐	☐	☐

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Recognise and trace the logic of the circuits of a half-adder and a full-adder.	☐	☐	☐
Construct the circuit for a half-adder.	☐	☐	☐
Be familiar with the use of the edge-triggered D-type flip-flop as a memory unit.	☐	☐	☐

Topic 17 : Internal computer architecture

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Have an understanding and knowledge of the basic internal components of a computer system.	☐	☐	☐
Understand the role of the following components and how they relate to each other: • processor • main memory • address bus • data bus • control bus • I/O controllers.	☐	☐	☐
Understand the need for, and means of, communication between components. In particular, understand the concept of a bus and how address, data and control buses are used.	☐	☐	☐
Be able to explain the difference between von Neumann and Harvard architectures and describe where each is typically used.	☐	☐	☐
Understand the concept of addressable memory.	☐	☐	☐
Be able to describe the stored program concept: machine code instructions stored in main memory are fetched and executed serially by a processor that performs arithmetic and logical operations.	☐	☐	☐
Explain the role and operation of a processor and its major components: • arithmetic logic unit • control unit	☐	☐	☐

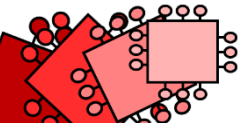


• clock • general-purpose registers • dedicated registers, including: • program counter • current instruction register • memory address register • memory buffer register • status register.			
Explain how the Fetch-Execute cycle is used to execute machine code programs including the stages in the cycle (fetch, decode, execute) and details of registers used.	☐	☐	☐
Understand the term 'processor instruction set' and know that an instruction set is processor specific.	☐	☐	☐
Know that instructions consist of an opcode and one or more operands (value, memory address or register).	☐	☐	☐
Understand and apply immediate and direct address modes.	☐	☐	☐
Understand and apply the basic machine-code operations of: • load • add • subtract • store • branching (conditional and unconditional) • compare • logical bitwise operators (AND, OR, NOT, XOR) • logical • shift right • shift left • halt. Use the basic machine-code operations above when machine-code instructions are expressed in mnemonic form- assembly language, using immediate and direct addressing.	☐	☐	☐
Explain the effect on processor performance of: • multiple cores • cache memory • clock speed • word length • address bus width • data bus width.	☐	☐	☐
Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Describe the role of interrupts and interrupt service routines (ISRs); their effect on the Fetch-Execute cycle; and the need to save the volatile environment while the interrupt is being serviced.	☐	☐	☐

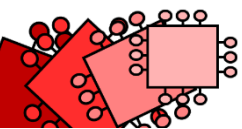
Topic 18 : Input & output devices

Video revision link:

Specific knowledge required for AS and A Level:	**Need to Revise**	**Revised Once**	**Got it!**
Know the main characteristics, purposes and suitability of the devices and understand their principles of operation.	☐	☐	☐
Explain the need for secondary storage within a computer system.	☐	☐	☐
Know the main characteristics, purposes, suitability and understand the principles of operation of the following devices: • hard disk • optical disk • solid-state disk (SSD).	☐	☐	☐



Compare the capacity and speed of access of various media and make a judgement about their suitability for different applications.	☐	☐	☐



AS & A'Level AQA Computer Science Revision check list

Topic 19 : Moral, social, ethical, legal & cultural issues and opportunities

Video revision link:

Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Show awareness of current individual (moral), social (ethical), legal and cultural opportunities and risks of computing. Understand that: • developments in computer science and the digital technologies have dramatically altered the shape of communications and information flows in societies, enabling massive transformations in the capacity to: • monitor behaviour • amass and analyse personal information • distribute, publish, communicate and disseminate personal information • computer scientists and software engineers therefore have power, as well as the responsibilities that go with it, in the algorithms that they devise and the code that they deploy • software and their algorithms embed moral and cultural values • the issue of scale, for software the whole world over, creates potential for individual computer scientists and software engineers to produce great good, but with it comes the ability to cause great harm. Be able to discuss the challenges facing legislators in the digital age.	☐	☐	☐

Additional notes:

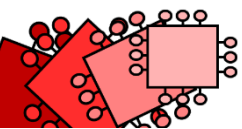
Please note, a question that requires an extended response can be asked from any area within the specification. These questions are assessed using a level of response framework, where the response requires specific areas to have been covered to allow it to reach that level. In this area of the specification, for example, a question may be asked on the social and ethical impacts of a specific technology in a specific scenario or context. To gain the highest level, candidates would need to discuss whichever moral or social elements are relevant in the question, and because there is a context, every point they make should be in the context given, or related after to the context. If the question requires a judgement, or conclusion, then this needs to be given and justified against the context given.

In order to prepare for this section we would recommend candidates regularly keep abreast of technological developments in the news.

Topic 20 : Communication

Video revision link:

Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Define serial and parallel transmission methods and discuss the advantages of serial over parallel transmission.	☐	☐	☐
Define and compare synchronous and asynchronous data transmission.	☐	☐	☐
Describe the purpose of start and stop bits in asynchronous data transmission.	☐	☐	☐
Define: • baud rate • bit rate • bandwidth • latency • protocol.	☐	☐	☐
Differentiate between baud rate and bit rate.	☐	☐	☐
Understand the relationship between bit rate and bandwidth.	☐	☐	☐



AS & A'Level AQA Computer Science Revision check list

Topic 21 : Networking and The Internet

Video revision link:

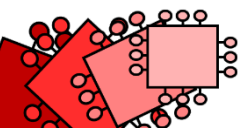
Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Understand: • physical star topology • logical bus network topology and: • differentiate between them • explain their operation.	☐	☐	☐
Explain the following and describe situations where they might be used: • peer-to-peer networking • client-server networking.	☐	☐	☐
Explain the purpose of WiFi.	☐	☐	☐
Be familiar with the components required for wireless networking.	☐	☐	☐
Be familiar with how wireless networks are secured.	☐	☐	☐
Explain the wireless protocol Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) with and without Request to Send/Clear to Send (RTS/CTS).	☐	☐	☐
Be familiar with the purpose of Service Set Identifier (SSID).	☐	☐	☐

Specific knowledge required for A Level ONLY:	Need to Revise	Revised Once	Got it!
Understand the structure of the Internet.	☐	☐	☐
Understand the role of packet switching and routers.	☐	☐	☐
Know the main components of a packet.	☐	☐	☐
Define: • router • gateway. Consider where and why they are used.	☐	☐	☐
Explain how routing is achieved across the Internet.	☐	☐	☐
Describe the term 'uniform resource locator' (URL) in the context of internetworking.	☐	☐	☐
Explain the terms 'fully qualified domain name' (FQDN), 'domain name' and 'IP address'.	☐	☐	☐
Describe how domain names are organised.	☐	☐	☐
Understand the purpose and function of the domain service and its reliance on the Domain Name Server (DNS) system.	☐	☐	☐
Explain the service provided by Internet registries and why they are needed.	☐	☐	☐
Understand how a firewall works (packet filtering, proxy server, stateful inspection).	☐	☐	☐
Explain symmetric and asymmetric (private/public key) encryption and key exchange.	☐	☐	☐
Explain how digital certificates and digital signatures are obtained and used.	☐	☐	☐
Discuss worms, trojans and viruses, and the vulnerabilities that they exploit.	☐	☐	☐
Discuss how improved code quality, monitoring and protection can be used to address worms, trojans and viruses.	☐	☐	☐

Topic 22 : TCP/IP & protocols

Video revision link:

Specific knowledge required for A Level ONLY:	Need to Revise	Revised Once	Got it!
Describe the role of the four layers of the TCP/IP stack (application, transport, network, link).	☐	☐	☐
Describe the role of sockets in the TCP/IP stack.	☐	☐	☐
Be familiar with the role of MAC (Media Access Control) addresses.	☐	☐	☐
Explain what the well-known ports and client ports are used for and the differences between them.	☐	☐	☐

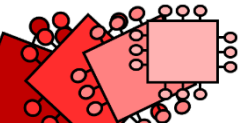


Be familiar with the following protocols: • FTP (File Transfer Protocol) • HTTP (Hypertext Transfer Protocol) • HTTPS (Hypertext Transfer Protocol Secure) • POP3 (Post Office Protocol (v3)) • SMTP (Simple Mail Transfer Protocol) • SSH (Secure Shell).	☐	☐	☐
Be familiar with FTP client software and an FTP server, with regard to transferring files using anonymous and non-anonymous access.	☐	☐	☐
Be familiar with how SSH is used for remote management.	☐	☐	☐
Know how an SSH client is used to make a TCP connection to a remote port for the purpose of sending commands to this port using application level protocols such as GET for HTTP, SMTP commands for sending email and POP3 for retrieving email.	☐	☐	☐
Be familiar with using SSH to log in securely to a remote computer and execute commands.	☐	☐	☐
Explain the role of an email server in retrieving and sending email.	☐	☐	☐
Explain the role of a web server in serving up web pages in text form.	☐	☐	☐
Understand the role of a web browser in retrieving web pages and web page resources and rendering these accordingly.	☐	☐	☐
Know that an IP address is split into a network identifier part and a host identifier part.	☐	☐	☐
Know that networks can be divided into subnets and know how a subnet mask is used to identify the network identifier part of the IP address.	☐	☐	☐
Know that there are currently two standards of IP address, v4 and v6.	☐	☐	☐
Know why v6 was introduced.	☐	☐	☐
Distinguish between routable and non-routable IP addresses.	☐	☐	☐
Understand the purpose and function of the DHCP system.	☐	☐	☐
Explain the basic concept of NAT and why it is used.	☐	☐	☐
Explain the basic concept of port forwarding and why it is used.	☐	☐	☐
Be familiar with the client server model.	☐	☐	☐
Be familiar with the Websocket protocol and know why it is used and where it is used.	☐	☐	☐
Be familiar with the principles of Web CRUD Applications and REST: • CRUD is an acronym for: • C – Create • R – Retrieve • U – Update • D – Delete. • REST enables CRUD to be mapped to database functions (SQL) as follows: • GET → SELECT • POST → INSERT • DELETE → DELETE • PUT → UPDATE.	☐	☐	☐
Compare JSON (Java script object notation) with XML.	☐	☐	☐
Compare and contrast thin-client computing with thick-client computing.	☐	☐	☐

Topic 23 : Databases

Video revision link:

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Produce a data model from given data requirements for a simple scenario involving multiple entities.	☐	☐	☐
Produce entity relationship diagrams representing a data model and entity descriptions in the form: Entity1 (Attribute1, Attribute2, ...).	☐	☐	☐
Explain the concept of a relational database.	☐	☐	☐
Be able to define the terms: • attribute • primary key	☐	☐	☐



• composite primary key • foreign key.			
Normalise relations to third normal form.	☐	☐	☐
Understand why databases are normalised.	☐	☐	☐
Be able to use SQL to retrieve, update, insert and delete data from multiple tables of a relational database.	☐	☐	☐
Be able to use SQL to define a database table.	☐	☐	☐
Know that a client server database system provides simultaneous access to the database for multiple clients. Know how concurrent access can be controlled to preserve the integrity of the database.	☐	☐	☐

Topic 24 : Big data

Video revision link:

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Know that 'Big Data' is a catch-all term for data that won't fit the usual containers. Big Data can be described in terms of: • volume – too big to fit into a single server • velocity – streaming data, milliseconds to seconds to respond • variety – data in many forms such as structured, unstructured, text, multimedia.	☐	☐	☐
Know that when data sizes are so big as not to fit on to a single server: • the processing must be distributed across more than one machine • functional programming is a solution, because it makes it easier to write correct and efficient distributed code. Know what features of functional programming make it easier to write: • correct code • code that can be distributed to run across more than one server.	☐	☐	☐
Be familiar with the: • fact-based model for representing data • graph schema for capturing the structure of the dataset • nodes, edges and properties in graph schema.	☐	☐	☐

Topic 25 : Functional programming paradigms

Video revision link:

Specific knowledge required for A Level ONLY:	**Need to Revise**	**Revised Once**	**Got it!**
Know that a function, f, has a function type f: A → B (where the type is A → B, A is the argument type, and B is the result type). Know that A is called the domain and B is called the co-domain. Know that the domain and co-domain are always subsets of objects in some data type.	☐	☐	☐
Know that a function is a first-class object in functional programming languages and in imperative programming languages that support such objects. This means that it can be an argument to another function as well as the result of a function call.	☐	☐	☐
Know that function application means a function applied to its arguments.	☐	☐	☐
Know what is meant by partial function application for one, two and three argument functions and be able to use the notations shown opposite.	☐	☐	☐
Know what is meant by composition of functions.	☐	☐	☐

AS & A'Level AQA Computer Science Revision check list

Topic 26 : Writing functional programs

Video revision link:

Specific knowledge required for A Level ONLY:	Need to Revise	Revised Once	Got it!
Show experience of constructing simple programs in a functional programming language.	☐	☐	☐
Higher-order functions.	☐	☐	☐
Have experience of using the following in a functional programming language: • map • filter • reduce or fold.	☐	☐	☐
Be familiar with representing a list as a concatenation of a head and a tail. Know that the head is an element of a list and the tail is a list. Know that a list can be empty. Describe and apply the following operations: • return head of list • return tail of list • test for empty list • return length of list • construct an empty list • prepend an item to a list • append an item to a list. Have experience writing programs for the list operations mentioned above in a functional programming language or in a language with support for the functional paradigm.	☐	☐	☐

Topic 27 : Aspects of software development

Video revision link:

Specific knowledge required for AS and A Level:	Need to Revise	Revised Once	Got it!
Be aware that before a problem can be solved, it must be defined, the requirements of the system that solves the problem must be established and a data model created. Requirements of system must be established by interaction with the intended users of the system. The process of clarifying requirements may involve prototyping/ agile approach.	☐	☐	☐
Be aware that before constructing a solution, the solution should be designed and specified, for example planning data structures for the data model, designing algorithms, designing an appropriate modular structure for the solution and designing the human user interface.	☐	☐	☐
Be aware that the models and algorithms need to be implemented in the form of data structures and code (instructions) that a computer can understand.	☐	☐	☐
Be aware that the implementation must be tested for the presence of errors, using selected test data covering normal (typical), boundary and erroneous data.	☐	☐	☐
Know the criteria for evaluating a computer system.	☐	☐	☐

Specific knowledge required for A Level ONLY:	Need to Revise	Revised Once	Got it!
Be aware that design can be an iterative process involving a prototyping/agile approach.	☐	☐	☐
Be aware that the final solution may be arrived at using an iterative process employing prototyping/ an agile approach with a focus on solving the critical path first.	☐	☐	☐
It should also undergo acceptance testing with the intended user(s) of the system to ensure that the intended solution meets its specification.	☐	☐	☐

