AQA Computer Science AS Level

3.4.1 Abstraction and automation

Concise Notes

Specification:

3.4.1.1 Problem-solving:

Be able to develop solutions to simple logic problems.

Be able to check solutions to simple logic problems

3.4.1.2 Following and writing algorithms:

Understand the term algorithm.

Be able to express the solution to a simple problem as an algorithm using pseudocode, 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.

3.4.1.3 Abstraction:

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

3.4.1.4 Information hiding:

Be familiar with the process of hiding all details of an object that do not contribute to its essential characteristics.

3.4.1.5 Procedural abstraction:

Know that procedural abstraction represents a computational method.

3.4.1.6 Functional abstraction :

Know that for functional abstraction the particular computation method is hidden.

3.4.1.7 Data abstraction:

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.

3.4.1.8 Problem abstraction/reduction:

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.

3.4.1.9 Decomposition:

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.

3.4.1.10 Composition:

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.

3.4.1.11 Automation:

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

Problem Solving

- The process of finding a solution to a difficult or complex issue
- In an exam, you might be given a series of statements from which you have to find the answer to a question

Algorithms

- Sequences of steps that can be followed to complete a task
- Always terminate rather than going on forever in a loop
- Can be written in pseudocode: a way of describing instructions that is independent of any particular programming language
- Pseudocode allows different programmers to communicate algorithms to one another

Assignment in pseudocode

- Assignment is the process of giving a value to a variable or constant
- In pseudocode, assignment is represented using an arrow pointing towards the variable or constant that is being given a value

```
counter ← 5
```

Sequence in pseudocode

- Sequence is the name given to instructions that follow on from one another
- Operations will be executed in the order that they appear

```
counter ← 18
counter ← counter + 1
remainingIterations ← 20 - counter
```

Selection in pseudocode

- Selection is the process of choosing an action to take based on the result of a comparison of values
- Different actions can be taken depending on the result of a comparison
- The statements IF, ELSE IF, ELSE and END IF can all be used

```
IF name = “Emma” THEN
      OUTPUT “Hello Emma”
ELSE If name = “George”
      OUTPUT “Hello George”
ELSE
      OUTPUT “Hello user”
END IF
```

Iteration in pseudocode

- Iteration is the process of repeating an operation
- Iteration structures include `FOR` and `WHILE` loops
- The code within an iteration structure is indented, allowing for easy identification of different loops

```
FOR number ← 6 to 12
     OUTPUT number / 2
END FOR

WHILE number < 18
     Number ← number + (number / 4)
END WHILE
```

Abstraction

- The name given to the process of omitting unnecessary details from a problem
- When solving a problem, abstraction can be used to simplify the problem which can in turn make finding a solution easier
- There are two distinct forms of abstraction:
    - representational abstraction
    - abstraction by generalisation / categorisation
- The definitions of these two forms of abstraction are often asked for in exams

Representational abstraction	Abstraction by generalisation / categorisation
A representation of a problem arrived at by removing unnecessary details from the problem.	A grouping by common characteristics to arrive at a hierarchical relationship of the "is a kind of" type.

Information hiding

- Defined as the process of hiding all details of an object that do not contribute to its essential characteristics
- Used in problem solving to simplify a problem without changing the essence of the issue

Procedural abstraction

- Involves breaking down a complex model into a series of reusable procedures
- The actual values used in a computation are abstracted away and a computational method is achieved

Functional abstraction

- Procedural abstraction results in a procedure
- Functional abstraction is the process of abstracting the result of procedural abstraction
- Abstracting a procedure further disregards the particular method of a procedure and results in just a function

Data abstraction

- Forms the basis of abstract data types
- Specific details of how data is actually represented are abstracted away
- This allows new kinds of data structures to be created from previously defined data structures

Problem abstraction / reduction

- Details are removed from a problem until it is represented in a way that is solvable
- A simplified problem is often similar to a problem that has already been solved, meaning that a solution for the problem can be found

Decomposition

- A problem is divided into a series of smaller sub-problems
- These smaller problems can be solved individually or further divided until all parts of the original problem have been solved

Composition

- Can be used to combine procedures to form a larger system
- Used in abstract data types, where a complex abstract data type is formed for smaller and simpler data types

Automation

- The process of putting abstractions of real world phenomena into action to solve problems
- These abstractions are referred to as models
- Achieved by creating algorithms which are later implemented in code, implementing models in data structures and finally executing the code on the data structures