Designing an algorithm

An algorithm is a plan, a logical step-by-step process for solving a problem. Algorithms are normally written as a flowchart or in pseudocode.

The key to any problem-solving task is to guide your thought process. The most useful thing to do is keep asking 'What if we did it this way?' Exploring **different** ways of solving a problem can help to find the **best** way to solve it.

When designing an algorithm, consider if there is more than one way of solving the problem.

When designing an algorithm there are two main areas to look at:

- the **big picture** - What is the final goal?
- the **individual stages** – What hurdles need to be overcome on the way to the goal?

Understanding the problem

Before an algorithm can be designed, it is important to check that the problem is completely understood. There are a number of basic things to know in order to really understand the problem:

- What are the inputs into the problem?
- What will be the outputs of the problem?
- In what order do instructions need to be carried out?
- What decisions need to be made in the problem?
- Are any areas of the problem repeated?

Once these basic things are understood, it is time to design the algorithm.

Pseudocode

Most programs are developed using programming languages. These languages have specific syntax that must be used so that the program will run properly. Pseudocode is not a programming language, it is a simple way of describing a set of instructions that does not have to use specific syntax.

Common pseudocode notation

There is no strict set of standard notations for pseudocode, but some of the most widely recognised are:

- **INPUT** – indicates a user will be inputting something
- **OUTPUT** – indicates that an output will appear on the screen
- **WHILE** – a loop (iteration that has a condition at the beginning)
- **FOR** – a counting loop (iteration)
- **REPEAT – UNTIL** – a loop (iteration) that has a condition at the end
- **IF – THEN – ELSE** – a decision (selection) in which a choice is made
- any instructions that occur inside a selection or iteration are usually indented

Using pseudocode

Pseudocode can be used to plan out programs. Planning a program that asks people what the best subject they take is, would look like this in pseudocode:

```
REPEAT
    OUTPUT 'What is the best subject you take?'
    INPUT user inputs the best subject they take
    STORE the user's input in the answer variable
    IF answer = 'Computer Science' THEN
        OUTPUT 'Of course it is!'
    ELSE
        OUTPUT 'Try again!'
UNTIL answer = 'Computer Science'
```

Flowcharts

A flowchart is a diagram that represents a set of instructions. Flowcharts normally use standard symbols to represent the different types of instructions. These symbols are used to construct the flowchart and show the step-by-step solution to the problem.

Common flowchart symbols

Name & Symbol	Usage
Start or Stop 	The beginning and end points in the sequence.
Process 	An instruction or a command.
Decision 	A decision, either yes or no.
Input or Output 	An input is data received by a computer. An output is a signal or data sent from a computer.
Connector 	A jump from one point in the sequence to another.
Direction of flow 	Connects the symbols. The arrow shows the direction of flow of instructions.

Name	Symbol	Usage
Start or Stop		The beginning and end points in the sequence.
Process		An instruction or a command.
Decision		A decision, either yes or no.
Input or Output		An input is data received by a computer. An output is a signal or data sent from a computer.
Connector		A jump from one point in the sequence to another.
Direction of flow		Connects the symbols. The arrow shows the direction of flow of instructions.

Using flowcharts

Flowcharts can be used to plan out programs. Planning a program that asks people what the best subject they take is, would look like this as a flowchart: