Appendix B: Glossary of core concepts

Abstraction

Abstractions are formed by identifying patterns and extracting common features from specific examples to create generalisations. Using generalised solutions and parts of solutions designed for broad reuse simplifies the development process and helps to manage complexity.

Data

Students learn how data about themselves and their world is collected and used. Data is collected and stored so that it can be analysed to better understand the world and make more accurate predictions.

Data is collected with both computational and non-computational tools and processes.

Computing systems

A computing system consists of hardware, software, computational processes and networks and users. Students will develop programming, analysis and design skills combined with the hardware knowledge needed to create network/Internet/cloud-based applications. They will learn how computing devices (such as smart devices, desktop computers and tablets) communicate with each other and the world around them and how to plan and design the infrastructure and systems that allow this to happen.

Algorithms

An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices. Students learn how to read, write, modify and test algorithms, as well as how to evaluate competing algorithms. The words programming, coding and programming language are defined as:

- Programming is the craft of analysing problems and designing, writing, testing and maintaining programs to solve them
- Coding is the act of writing computer programs in a programming language
- A programming language is the formal language used to give a computer instruction.

Software evaluation

Software evaluation is the process of determining if the program or combination of programs is the best possible solution to a given problem or task. The evaluation process should include factors such as feasibility, efficiency, and ethical use.

Software testing

Software testing is the process of finding and correcting errors (bugs) in a program or system and ensuring that the program produces the intended output. Debugging includes identifying errors, gaps, and missing requirements.

Heuristic

A heuristic is an approach to problem solving which aims to make an approximate solution to the problem. This can be used when time and resources are limited. The solution may not be feasible using classic or standard methods but should aim to approximate the optimal solution. This may involve the loss of precision, accuracy, optimal performance or completeness.