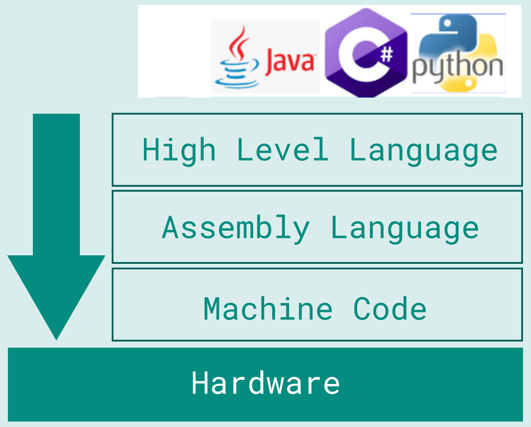
Component 1: Principles of programming

Key terms

Term	Definition
High Level Language	A programming language designed to simplify computer programming. It is "high-level" because it is several steps removed from the actual code run on a CPU
Low Level Language	A programming language that contains basic instructions recognised by a CPU. Two common types of low-level languages are assembly language and machine code.
Assembly Language	A low-level programming language designed for a specific type of processor that can be converted to machine code using an assembler.
Mnemonic	A short code used in assembly language; chosen to remind the programmer of the program instruction it represents.
Machine Code	A low-level language comprised of binary digits.
Program Translation	Translating code, using either an interpreter or compiler into executable machine code
Embedded Software	Software built into embedded systems written to control machines or devices that are not typically thought of as computers

Characteristics of High Level Languages

High-level languages are designed to be closer to humans than to computers.



The programs:

- Require translation into machine code.
- Are portable. The translated programs can be run on different computers running different operating systems without modification.
- Are written in code similar to English, or other recognisable language. This helps when reading writing and maintaining the programs.
- Allow access to module libraries.
- Use data types and data structures, selection statements and repetition/iteration constructs.
- Use logic operators and functions that are built into the language.

Examples of High Level Languages

Most contemporary programs are written using high-level languages, including GCSE and A level projects.

Examples include Java, C#, Python and many others.

Uses of High Level Languages

Used when execution speed is not the most critical factor, including, for writing:

- Office applications and database packages
- Operating systems
- e-commerce software and social media apps.

Characteristics of Low Level Languages

The programs:

- May be finely tuned so that the code is more efficient.
- May have more system-dependent features available.
- Are usually not portable.
- Are usually harder to program:
  - because the programmer has to pay more attention to fine details,
  - and because it takes more lines of code to achieve the same result.

Examples of High Level Languages

Assembly language

Code written using mnemonics that can make use of machine-dependent instructions.

Advantages include:

- The translated program requires less memory
- Code can be executed faster

Machine Code: The opposite of a high-level language made up entirely of bit patterns that can be executed directly by the CPU. All programs must be translated into machine code before they can run on a computer.

Uses of High Level Languages

Used when execution speed and efficient memory use are critical. Examples include: Operating systems, device drivers and embedded software.

Professional game developers often use console specific development software, which is likely to include low-level features for performance.