Characteristics and purpose of different level languages:

What is meant by:

Machine code:

- Binary instructions that the CPU can decode and execute

High level code:

- Human oriented code written by programmers
- Needs to be translated before executed
- Closer to English
- One instruction = many instructions in machine code
- Portable to different systems

Low level code:

- Closer to machine code
- One instruction = one instruction in machine code
- Only works on one type of processor/machine

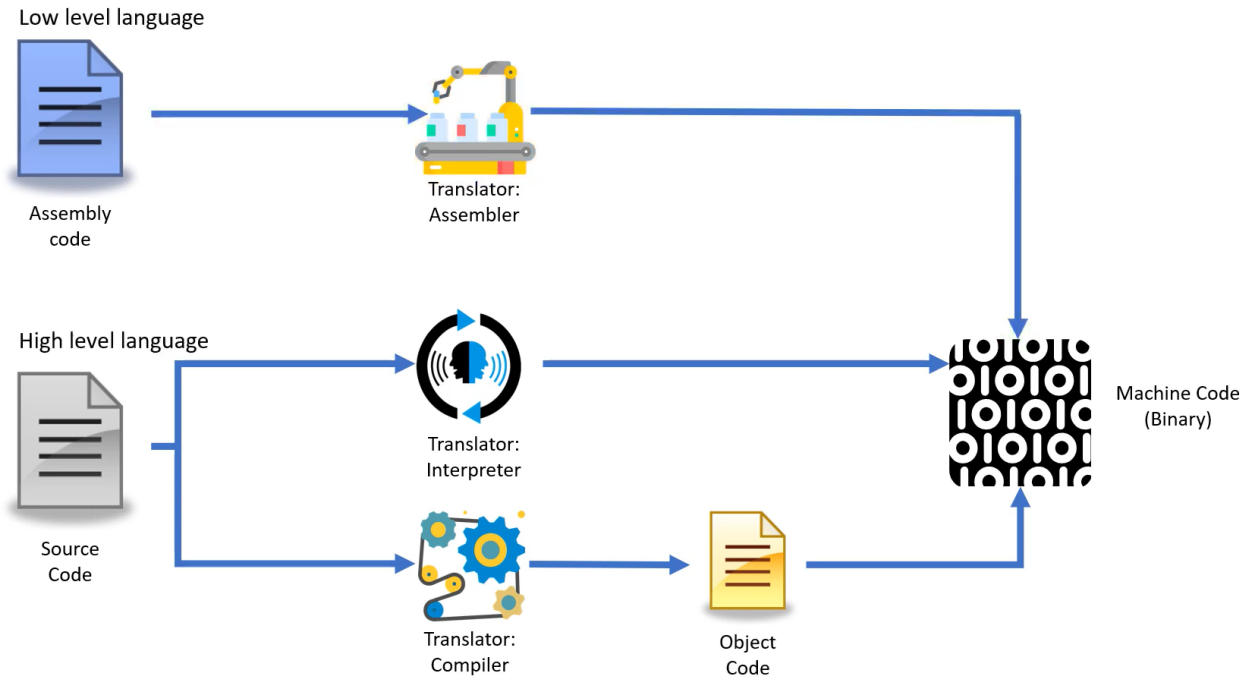
Low level language:

- Written in assembly language
- Translated by an assembler
- Used for embedded systems and device drivers
- One instruction is translated into one machine code instruction
- Code works on one type of processor only
- Programmer works with memory directly
- Code is harder to write and understand
- Memory efficient
- Fast to execute

High level languages:

- Source code is written in different languages
- Translated by a compiler/interpreter into machine code
- Easier to write
- One source code instruction translates to many machine code instructions
- Code will run on different type of processors
- Programmer has lot of data structures to use
- Code is quicker and easier to understand and write
- Less memory efficient
- Code is slower to execute if not optimised

Purpose of translator – Convert code written in both assembly and high level source code into machine code for execution



Assembly code translated into machine code by an assembler

High level source code is translated either using an interpreter or a compiler into machine code

Compiler:

- Translates source code from high level language into object code and then into machine code
- Whole program is translated into machine code at the same time before it's run and creates an exe file

- Once compiled, runs quickly but compiling may take a long time
- Returns a list of errors once compiling is done

Advantages:

- Don’t need translation software at run time
- Faster execution
- Code is optimised
- Source code is secret

Disadvantages:

- Program will not run with syntax errors and make it more difficult to write code
- Code needs to be recompiled when code is changed
- Designed for a specific processor

Interpreter:

- Translates source code into machine code
- Program is translated line by line as program is running and doesn’t create exe file

Advantages:

- Easy to write source code because it finds syntax errors
- Code doesn’t need to be recompiled
- Easy for beginners

Disadvantages:

- Translation software needed at run time
- Slower execution speed
- Code is not optimised
- Source code is needed

IDE:

IDEs provide:

- Code editor:
  - Auto line numbering
  - Auto-colour coding for things like variables, functions
  - Auto correct
  - Auto indentation
  - Auto complete
- Runtime environment:
  - Allows code to be run quickly within the IDE
- Output window to show the output of the program
- Error diagnostics and debugging tools to help find and fix errors
- Breakpoints stop at certain lines and allow you to gather information like variable values etc