# High- and low-level languages

## Key objectives

Objectives of this chapter are to revise:

- programming languages:
  - high-level languages
  - low-level languages
    - assembly language
    - machine code
- translators:
  - compilers
  - interpreters
  - assemblers

## Key definitions

| Term | Definition |
|---|---|
| Computer program | A list of instructions that enable a computer to perform a specific task |
| Translator | Translators convert a program into binary instructions that computers can understand |

# Computer programs

A computer program is a list of instructions that enable a computer to perform a specific task. Computer programs can be written in high-level languages and low-level languages depending on the task to be performed and the computer to be used. Most programmers write programs in high-level languages.

## High-level languages

High-level languages enable a programmer to focus on the problem to be solved and require no knowledge of the hardware and instruction set of the computer that will use the program. High-level programming languages are portable and can be used on different types of computer.

## Low-level languages

Low-level languages relate to the specific architecture and hardware of a particular type of computer. Low-level languages can refer to machine code, the binary instructions that a computer understands, or assembly language that needs to be translated into machine code.

## Assembly languages

Assembly language is a low-level programming language that needs to be translated into machine code by an assembler.

## Machine code

Machine code is the binary instructions that a computer understands. No translation is required. See the Student’s Book for examples of each type of programming language.

# Common errors

- Students often say that machine code needs translation.
- Students often reverse the definitions of high- and low-level languages.

# Translators

In order to be used by a computer, programs need to be translated into the binary instructions, machine code, that the computer understands.

## Compilers

A compiler is a computer program that translates a program written in a high-level language (HLL) into machine code so that it can be directly used by a computer to perform a required task.

## Interpreters

An interpreter is a computer program that reads a statement from a program written in a high-level language performs the action specified and then does the same with the next statement and so on.

## Assemblers

An assembler is a computer program that translates a program written in an assembly language into machine code so that it can be directly used by a computer to perform a required task.

# Summary

| Compiler | Interpreter | Assembler |
|---|---|---|
| Translates a high-level language program into machine code. | Executes a high-level language program one statement at a time. | Translates a low-level language program into machine code. |
| An executable file of machine code is produced. | No executable file of machine code is produced. | An executable file of machine code is produced. |
| One high-level language statement can be translated into several machine code instructions | One high-level language program statement may require several machine code instructions to be executed. | One low-level language statement is usually translated into one machine code instruction. |
| Compiled programs are used without the compiler. | Interpreted programs cannot be used without the interpreter. | Assembled programs can be used without the assembler. |
| A compiled program is usually distributed for general use. | An interpreter is often used when a program is being developed. | An assembled program is usually distributed for general use. |

# Common errors

- The statement: ‘ interpreters translate high-level program statements into a machine code program’ is incorrect since the machine code statements are just executed and not stored.
- Statements such as ‘low-level programs are faster’ or ‘high-level programs are slower’ are never worth any marks since they don’t say faster or slower at what; it is necessary to be clear and state, for example: ‘low-level programs complete the same task faster than …’.