Introduction to Programming

Program

- Consists of **set of instructions** that tells the computer to produce various kinds of outputs

- **Programming language** – formal language designed to communicate instructions to a computer.

There are two types of programming languages:

- Low-level Languages
- High-level Languages

Low-level Languages

- Are **machine oriented** and require extensive knowledge of computer hardware and its configuration.

Two categories of low-level languages:

- **Machine language**
- **Assembly language**

Machine Language

- Or **machine code**, is the language that is directly understood by the computer, and does not need to be translated.

- Consists of strings of **1’s** and **0’s**.

Assembly Language

- Made to make machine language **more readable**.
- Uses sets of **symbols** and **letters** or called as assembler to translate into machine language.
- Assembly language is still difficult to understand that’s why the high-level language is developed.

High-level Languages

- Are programming languages that uses words and mathematical symbols in its instructions.

- C++
- Java
- Python
- Fortran

- To learn a programming language, you need to learn **commands**, **syntax** and **logic**.

High-Level Languages

Advantage:

- It is much closer to the logic of the human language.

- **Most** of the programming language are **portable** – means that they can run on multiple devices with the same operating system.

High-level Languages

- Requires to be compiled or to be interpreted or both.

- **Compilers** – computer program that translates a program written in a high-level language to a machine language.
- **Interpreters** – computer program that simulates a computer that understands a high-level language. It is read line by line.

Compiler

Advantages:

- Source code privacy
- Fast

Disadvantages:

- Platform dependent
- Requires extra compilation step

Interpreter

Advantages:

- Cross platform
- No extra compilation delay

Disadvantages:

- No source code privacy
- Slow

Compiler and Interpreter

How compilers work:

- Source code -> Compiler -> Machine code -> Run program
- Input -> Run program -> Output

How interpreters work:

- Source code/Input -> Interpreter -> Output

Procedural Programming

- Describes algorithms that indicates processing procedures to solve problems.

- Has the “what to do” and “how to do” concept.

Non-procedural Programming

- Generated by providing input, output and processing conditions in order to solve a problem, and by selecting necessary processing routines.
- Has only the “what to do” concept. Database language such as SQL use this.
- Only one or a few statement is enough to perform the whole job.

Scripting Language

- Is a programming language designed for integrating and communicating with other programming languages. They are often found alongside HTML, Java or C++.
- Needs the support of writing scripts. It is interpreted rather than compiled.

- VBScript
- JavaScript
- PHP
- Python
- Ruby