

Introduction to Programming

Lecture Number:

What is Programming

- Programming is to instruct the computer on what it has to do in a language that the computer understands.
- Programming must be done in a programming language, most preferably it should be the binary language as computers work on 1's and 0's which is called machine or some times 'object' code.

Machine Code

- An example machine code could be:
1011001101010101 (Add two numbers)
1111001100110101 (Subtract two numbers)
1111000011110000 (Multiply two numbers)

It clearly is not memorisable or understandable by the human being so we need an alternative!

Assembly Language

- As an alternative the binary sequences of numbers were represented by simple english keywords:

1011001101010101 (ADD)

1111001100110101 (SUB)

1111000011110000 (MUL)

This is relatively easier as compared to the machine language, however modern applications need much higher levels of abstraction.

The need for High level languages

- ADD, SUB, MUL etc are usually limited to a specific maximum number of bits, we need generic coding schemes where we can concentrate on the logic instead of memory constraints.
- ADD might be SUM or ADDA in other machines, we needed a machine and platform independent solution so a single language could suffice for all machines.

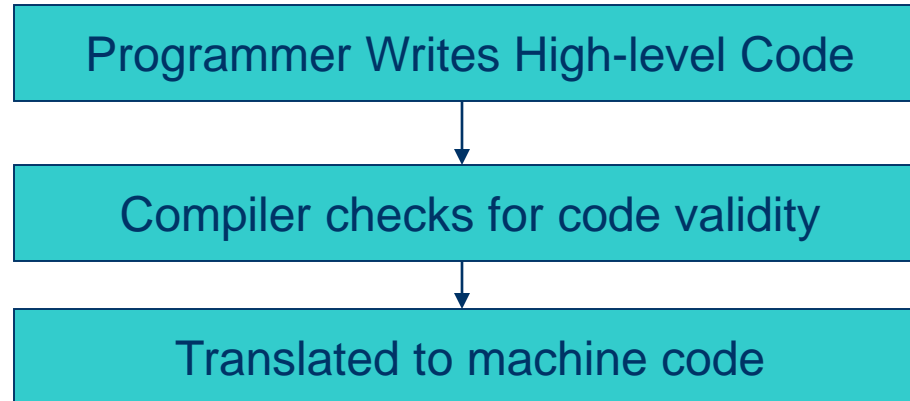
High-level Languages

- High level languages came into being solving various glitches (issues) that machine and assembly languages had.
- Examples of high-level languages are: C++, Java, Visual Basic, Visual C#, etc
- However, High level languages are not understandable by the machines so we need an interpreter, also called a compiler.

Compiler

- A compiler is a piece of software that translates high level language into machine or binary code.
- Every language has a specific style or way, called its syntax, just like there's grammar in human languages.
- Compiler also checks for any syntax errors in our 'code'.

The Compile process



Examples of Compilers

- Microsoft Visual Studio
- Borland C++
- Bloodshed Dev C++

What happens after compiling?

- Once the object-code/Machine-code is generated, another process called 'Linking' is done, which means to link any pre-available libraries with our code.
- A library is a collection of modules or functions made by other programmers for later reuse.
- Different languages have different ways of expressing libraries, like in C++ they're represented by .lib extension.

What happens after Linking?

- Loading!
- The program after linking is put into Main Memory (RAM).
- Execution is the next step
- The RAM acts like a conveyer belt for the processor which actually executes the code.

Building the Program Logic

- Before writing a program, we need to plan the flow of the program
- Pseudo-code and flow-charts are two general tools that help in planning for a program

Pseudo-code & Flow Charts

- **Pseudo-code** (As the name suggests, pseudo-code generally does not actually obey the syntax rules of any particular language; there is no systematic standard form. Usually natural language sentences could be used in the pseudo-code to explain the mechanism)
 - Informal language used to develop algorithms
 - Similar to a Natural Language
 - Not actually executed on computers
 - Allows us to “think out” a program before writing the code
 - Easy to convert into a program

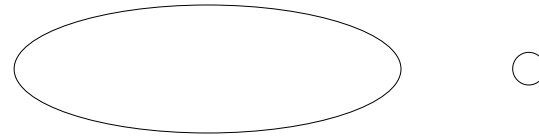
Pseudo-code & Flow Charts

- Flow Charts

- Flow Charts are a graphical representation, usually considered more understandable
- Drawn using certain special-purpose symbols, as described next

Pseudo-code & Flow Charts

- Oval symbol OR small circle
 - indicates beginning or end of a program, or a section of code



- Arrows called flow-lines
 - Indicate the flow of program



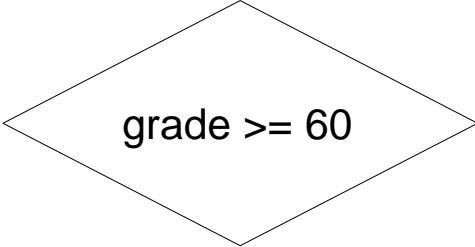
Pseudo-code & Flow Charts

- Rectangle symbol (action symbol)
 - Indicates any type of action.



add grade to total

- Diamond
 - Indicates Decision



grade \geq 60

Today we studied

- What is Computer Programming?
- Machine, Assembly and High-Level language
- Compilers
- The program planning process