Universiti Malaysia PAHANG
Engineering • Technology • Creativity

BTE2313

Chapter 1: Introduction to Computers and Programming

by
Sulastri Abdul Manap
Faculty of Engineering Technology
sulastri@ump.edu.my

Learning Outcomes

- To understand computer sciences: basic concepts
- To be aware of different types of programming languages
- To apprehend the C++ program development environment
- To familiarize with the IDE (Integrated Development Environment) for computer programming

Introduction

- C++ : a computer programming language that's widely used by technically oriented people with or without programming experience, and for skilled programmers to use in developing information systems.
- Instructions in commanding computers are written/coded to perform tasks.
- *Software* is known as the instructions or commands you write to control *hardware*

Hardware and Software

- Computers can carry out calculations and make logical decisions faster than human being can.
- Computer process *data* by implementing set of control instructions called *computer programs*.
- These computer programs, guide the computer through orderly sets of actions indicated by *computer programmers*.

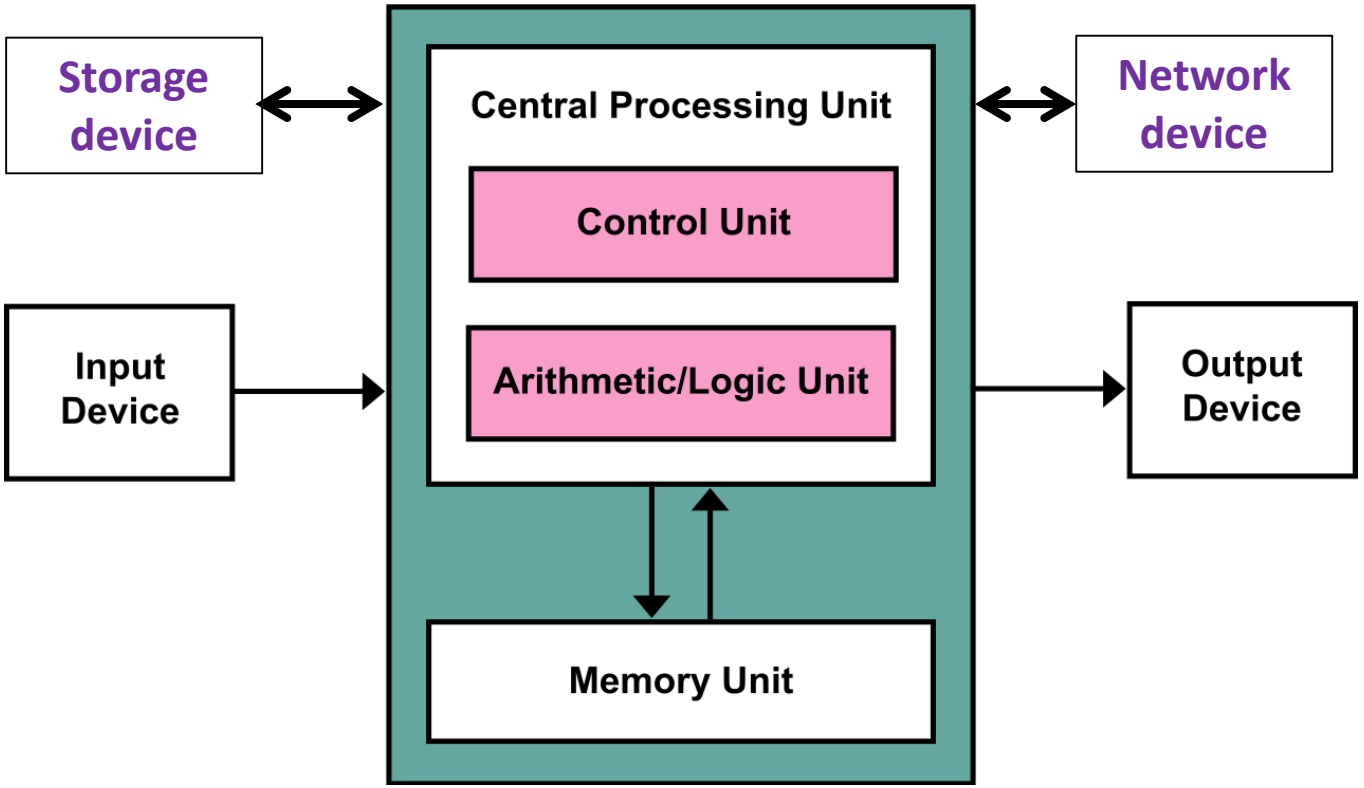
Images source: Wikipedia

Hardware and Software (cont.)

- The programs that run on a computer are known as software.
  - (e.g., Operating system, System utility software, Application software).

- A computer, physically consists of a number of devices referred to as hardware
  - (e.g., hard disks, mouse, LCD screen/monitor, keyboard, memory, optical drives and central processing units CPU).

Computer Architecture



Computer Organization

- Input Device: receives information (processed in 'receiving' section)
- Output Device: delivers the processed information to be used outside the computer (processed in 'shipping' section)
- Memory unit: Remembers information that has been entered into input unit ('warehouse' section of a computer)
- Central Processing Unit (CPU): brain of the computer ('administrator' of a computer).

Programming language

- Computer programmers may write commands or instructions in various programming languages
- Some of the languages are easily comprehensible by computers, while others require *translation steps* in between
- Three general types of programming languages:
  - Machine languages
  - Assembly languages
  - High-level languages

Programming language (cont.)

Machine Languages

- A computer can understand directly ONLY its own machine language, which is defined by its hardware architecture.
- Machine languages usually consist of 1s and 0s, which makes them difficult for human to understand.

Assembly Languages

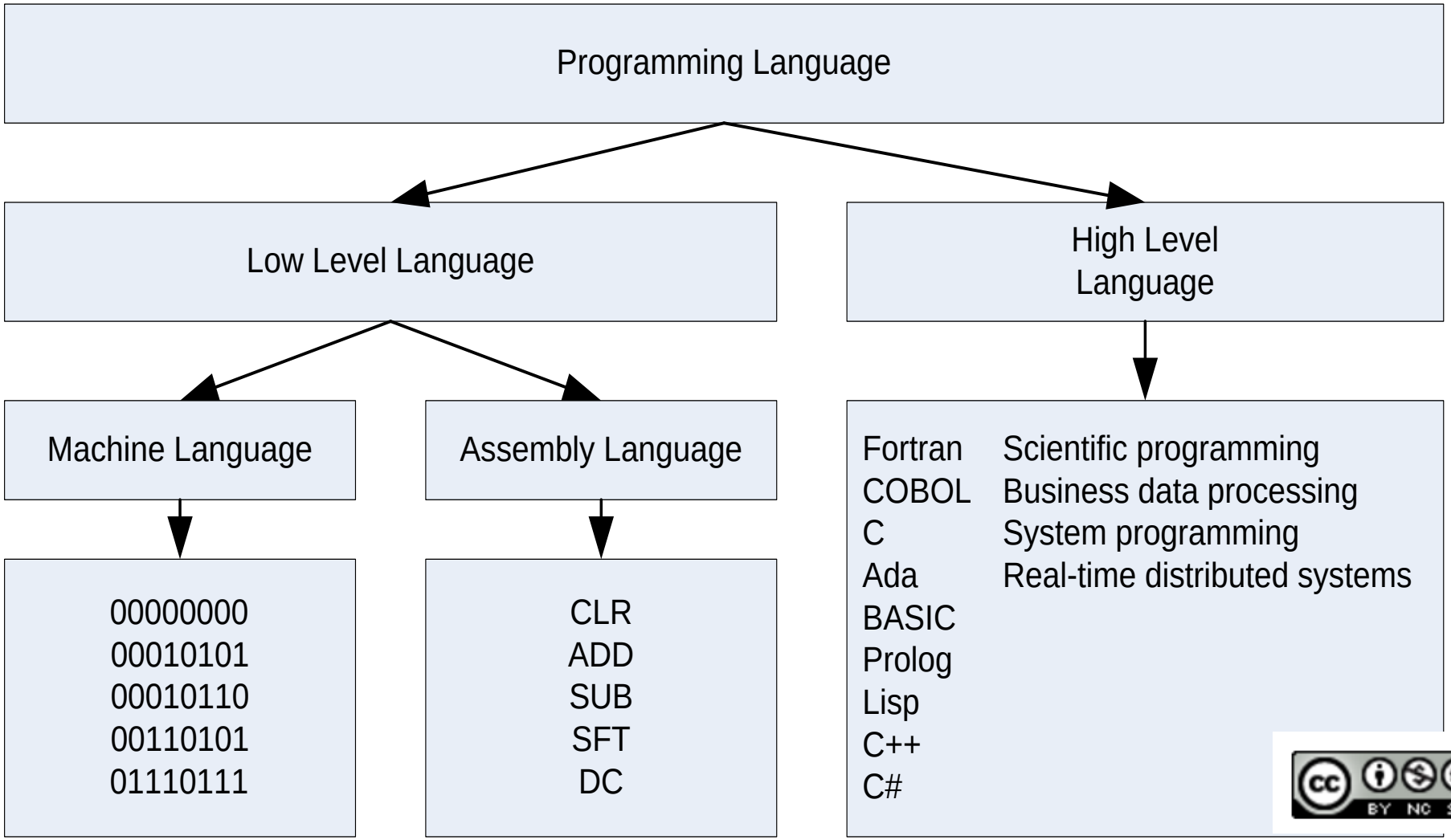
- Operations are represented by *abbreviations* in English, that form the basis of assembly languages.
- A translator is used to convert assembly language into machine language. It is called an assembler.

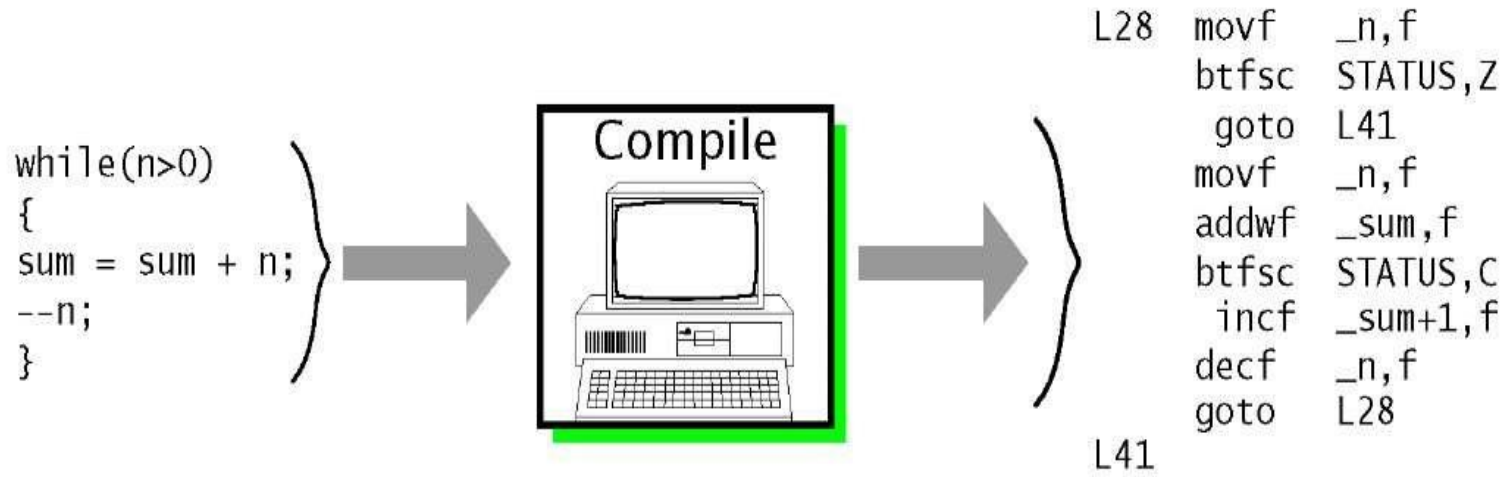
Programming language (cont.)

High-Level Languages

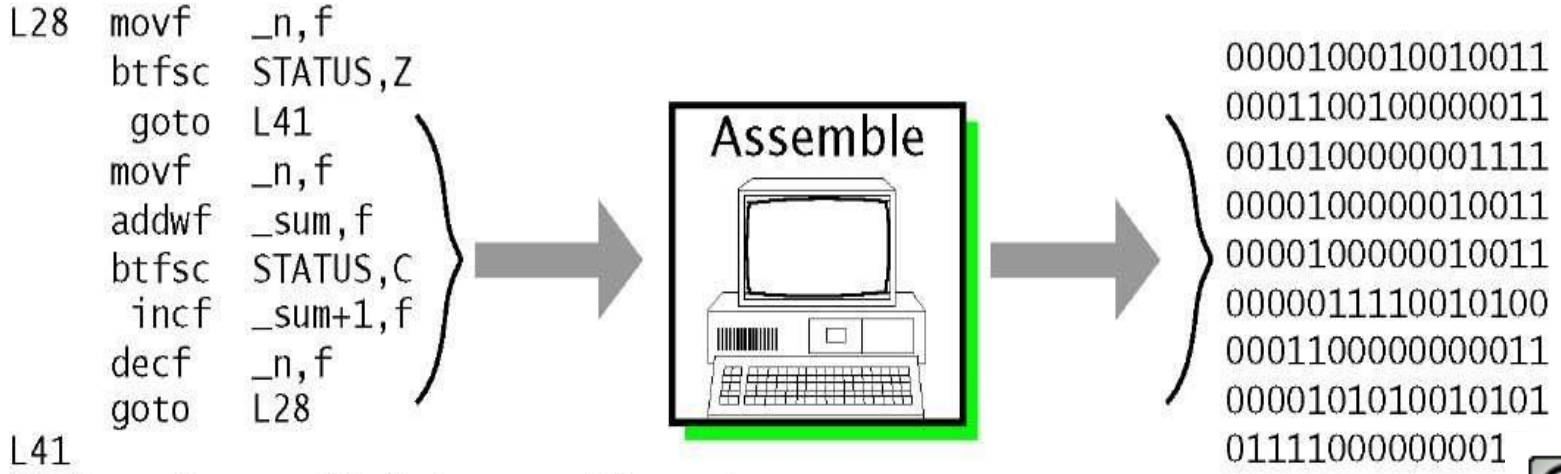
- To make the programming process faster, High-level languages were developed. This type of language will enable programmer to accomplish extensive tasks only by one statement/instruction.
- A translator is used to transform high-level language programs to machine code/language. It is called a compiler.
- This type of language allow programmers to write commands/codes that look like ordinary English and contain frequently used mathematical expressions.

Programming language (cont.)





(a) First, compile to assembly-level code.



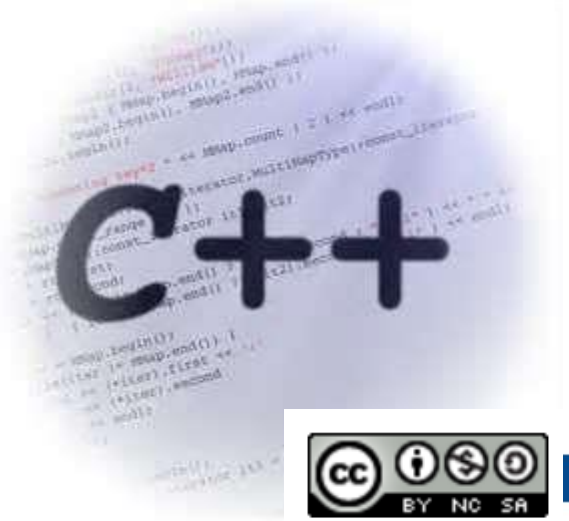
(b) Second, assemble-link to machine code.

History of C++

- Dennis Ritchie created C at Bell Telephone Laboratories in early 1970s.
- C exists from the difficulties of programming language (assembly language) for Unix OS.
- C++, is an extension of C that is developed at Bell Laboratories by Bjarne Stroustrup in the early 1980s.
- C++ exists from the enhancement of C with data abstraction and object-oriented.

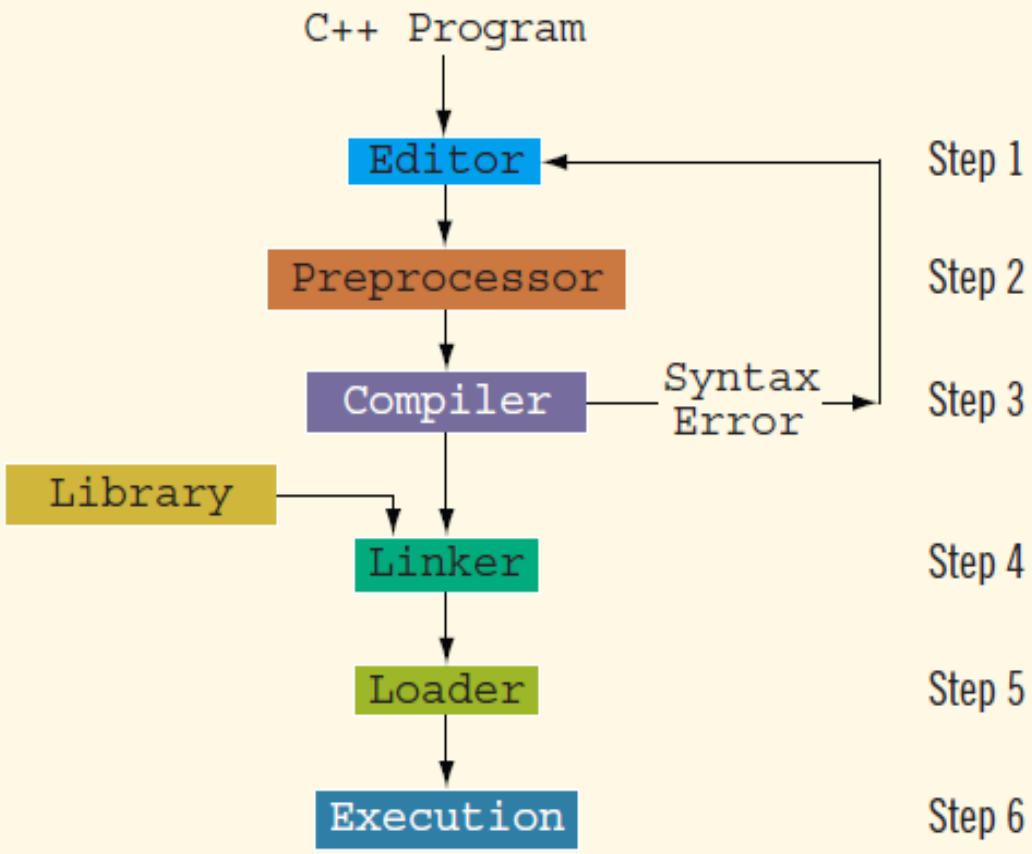
C++ Language

- C++ language facilitates computer-program design in a structured and disciplined approach.
- C++ is an extension of the C programming language (as a superset of C)
- Some C++ is not C, but ALL C code is C++ code!



A typical C++ Environment

- A typical six stages of C++ environments: editing source code, preprocessor, compiling, linking, loading and executing.



A typical C++ Environment (cont.)

- **Step 1**: edit source code/program file with an *editor* program.
  - File names for a C++ source code end with the `.cpp`, `.cxx`, `.cc` or `.C` extensions
  - What type of *editor*??
- integrated development environments (IDEs)
  - support the software-development process, from editors for writing/editing codes and compilers/debuggers to locate logic error (errors that cause programs to perform incorrectly)
  - Popular IDEs:

  Code Blocks, Dev C++, CodeLite, Microsoft® Visual Studio, Express Edition, NetBeans etc.

A typical C++ Environment (cont.)

- **Step 2 and 3**: compile the program.
  - Before the compiler's translation process begins, **preprocessor** program executes automatically (so we call preprocessing Step 2 and compiling Step 3).
  - Certain manipulations are to be performed on the program before compilation. This will be done in Step 2 where the C++ preprocessor comply with preprocessing directives.
  - Usually the manipulations consists of performing various text replacements, and including other text files to be compiled
  - In Step 3, the compiler translates the C++ codes into machine-language code (also known as object code where new file with extension *.obj* is automatically created)

A typical C++ Environment (cont.)

- **Step 4**: linking
  - A linker links the produced object code with other object codes and libraries specified, to produce an executable program ()
  - A new file will be produced with file extension of *.exe* if the program compiles and links correctly.
- **Step 5:** loading.
  - Memory will be used to store the program before it can be executed
  - A loader will take the executable image from disk and transfers it to memory.
- **Step 6**: Execution
  - The program will be executed by its CPU, one instruction at a time.