Programming Fundamentals

Lecture 1

Aamina Batool

What is a computer?

- [Norton] *A Computer is an electronic device that processes data, converting it into information that is useful to people.*
- [Wikipedia] *A Computer is a programmable device, usually electronic in nature, that can store, retrieve and process data.*

Elements of a Computer

1. Hardware
   - Central Processing Unit (CPU)
   - Main Memory
   - Secondary Storage
   - Input/ Output Devices

2. Software

Central Processing Unit (CPU)

The main components of the CPU are:

1. Control unit (CU)
2. Arithmetic and logic unit (ALU).
3. Registers.

Central Processing Unit (CPU)

1. CU (Control Unit):
   - Fetches and decodes instructions
   - Controls flow of information in and out of MM
   - Controls operation of internal CPU components
2. ALU (arithmetic logic unit): carries out all arithmetic and logical operations

Central Processing Unit (CPU)

3. Registers:

    1. PC (program counter): points to next instruction to be executed

    2. IR (instruction register): holds instruction currently being executed

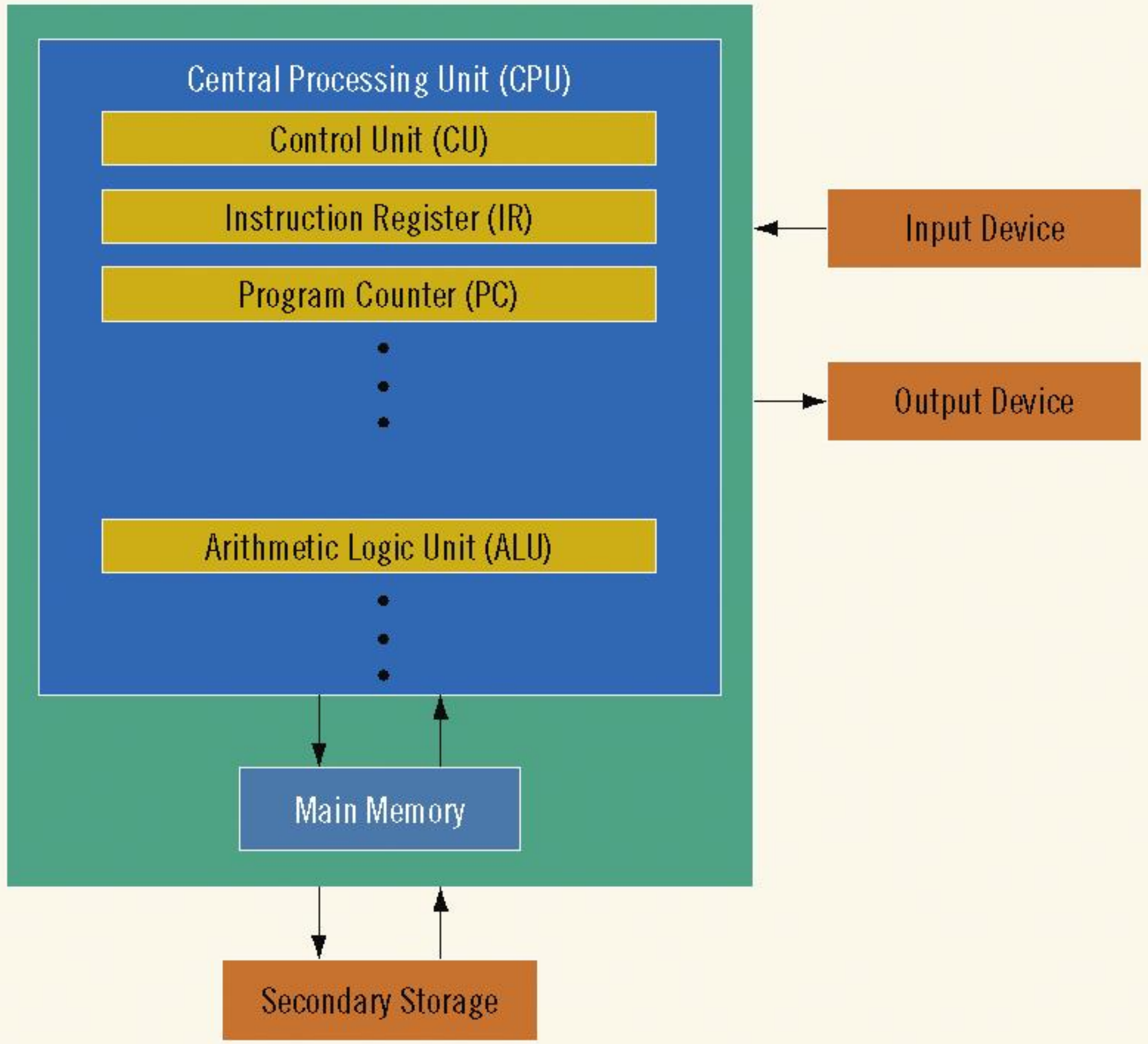


FIGURE 1-1 Hardware components of a computer

Main Memory

- Directly connected to the CPU
- All programs must be loaded into main memory before they can be executed
- All data must be brought into main memory before it can be manipulated
- When computer power is turned off, everything in main memory is lost

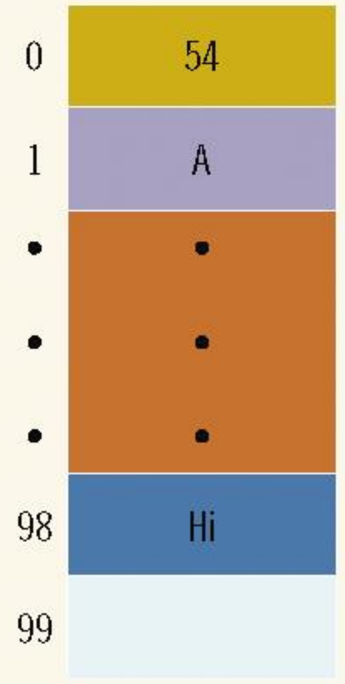


FIGURE 1-2 Main memory with 100 storage cells

Secondary Storage

- Secondary storage: Device that stores information permanently
- Examples of secondary storage:
  - Hard disks
  - Floppy disks
  - Zip disks
  - CD-ROMs
  - Tapes
  - Flash drives

Input/Output Devices

- Input devices feed data and programs into computers. They include:
  - Keyboard
  - Mouse
  - **Secondary storage**
- Output devices display results. They include:
  - Monitor
  - Printer
  - **Secondary storage**

Software

- Software: Programs that do specific tasks

1. **System programs** take control of the computer, such as an operating system

- Operating System monitors the overall activity of the computer and provides services.

2. **Application programs** perform a specific task
   - Word processors
   - Spreadsheets
   - Games

Language of a Computer

- Digital signals are sequences of 0s and 1s
- Machine language: language of a computer
- Binary digit (bit):
    - The digit 0 or 1
- Binary code:
    - A sequence of 0s and 1s
- Byte:
    - A sequence of eight bits

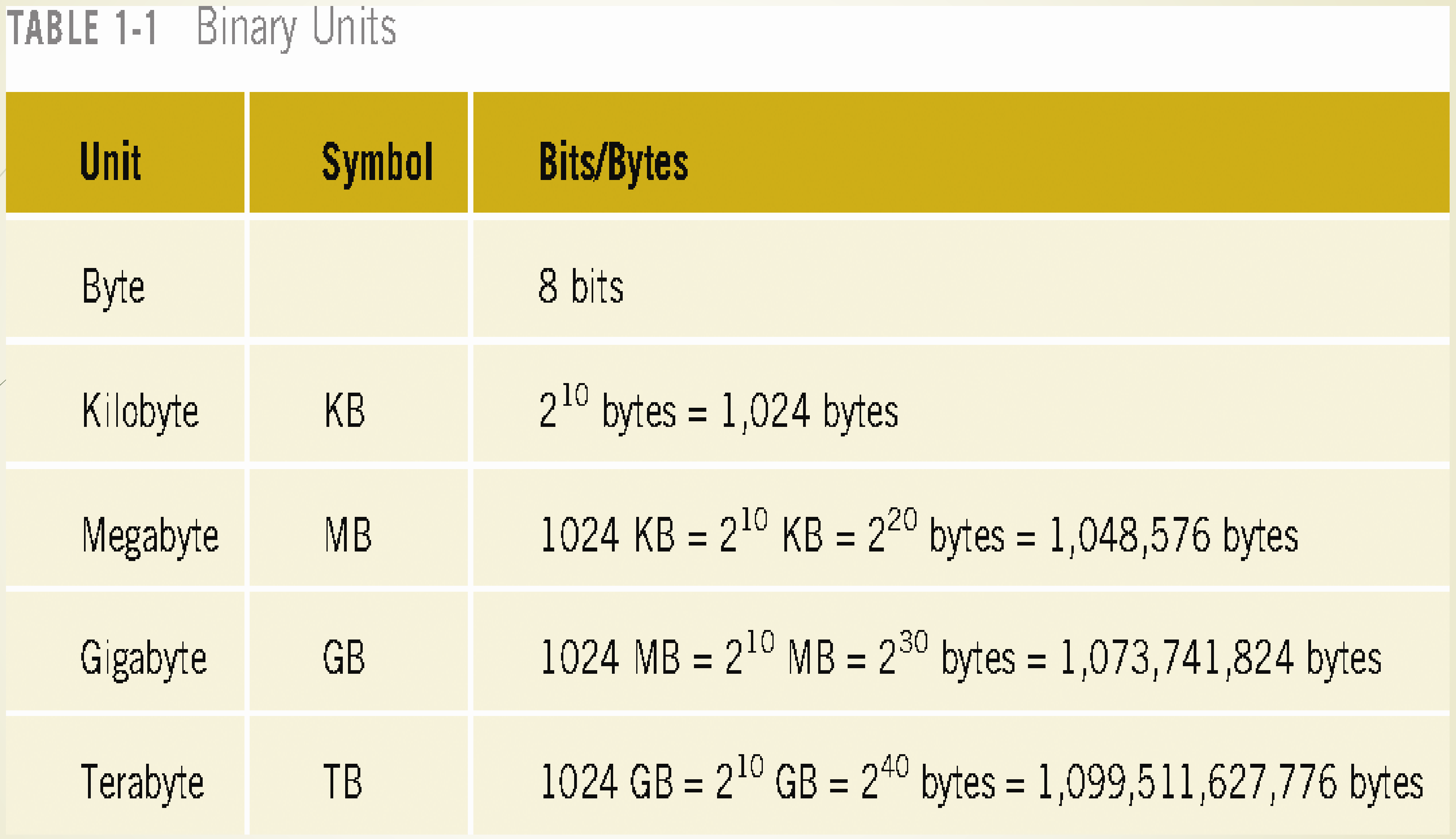


TABLE 1-1 Binary Units

Unit	Symbol	Bits/Bytes
Byte		8 bits
Kilobyte	KB	2^{10} bytes = 1,024 bytes
Megabyte	MB	1024 KB = 2^{10} KB = 2^{20} bytes = 1,048,576 bytes
Gigabyte	GB	1024 MB = 2^{10} MB = 2^{30} bytes = 1,073,741,824 bytes
Terabyte	TB	1024 GB = 2^{10} GB = 2^{40} bytes = 1,099,511,627,776 bytes

Programming Language Evolution

- Early computers were programmed in machine language
- To calculate `wages = rates * hours` in machine language:

```
100100 010001     //Load
100110 010010     //Multiply
100010 010011     //Store
```

Assembly Language

- Assembly language instructions are **mnemonic.**
- Mnemonic (easy-to-remember).
- Assembler: translates a program written in assembly language into machine language

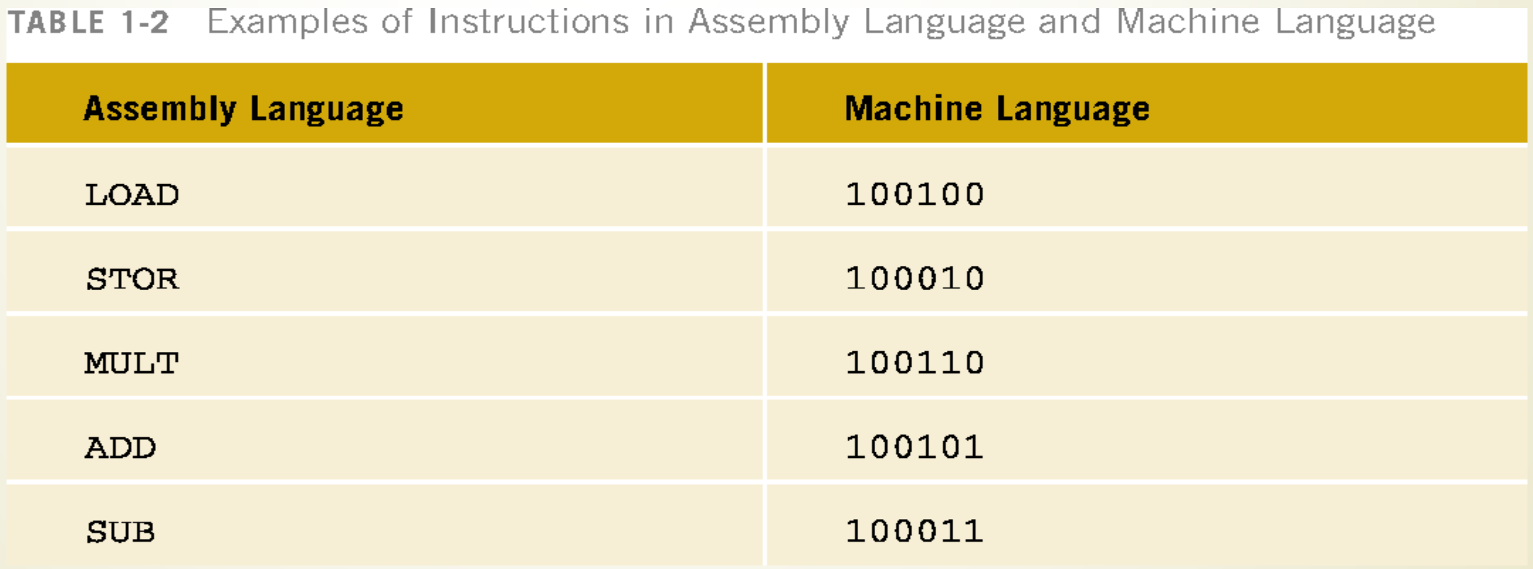


TABLE 1-2 Examples of Instructions in Assembly Language and Machine Language

Assembly Language	Machine Language
LOAD	100100
STOR	100010
MULT	100110
ADD	100101
SUB	100011

High-level Languages

- High-level languages include Basic, Pascal, C++, C, and Java
- The equation `wages = rate • hours` can be written in C++ as:

```
wages = rate * hours;
```

Limitations of Computers

- The computer can do nothing without being told what to do.
- A computer is not intelligent.
- It cannot analyze a problem and come up with a solution.
- A human must
  1. Analyze the problem
  2. Develop the instructions for solving the problem (the *program*),
  3. Then ask the computer to carry out these instructions

Advantage of Using a Computer

- Once a solution is written for the computer, it can repeat the solution
  - very quickly
  - consistently,
  - again and again, for different situations and data.

What is Computer Science

- Computer science is the study of
  1. Problems
  2. Problem-solving
  3. The solutions that come out of the problem-solving process.
- Given a problem, a computer scientist's goal is to develop an algorithm for solving a problem..
- An **algorithm** is a precise sequence of instructions for solving a problem.

Steps of Solving a Problem

1. Understand the Problem
2. Formulate a Model
3. Develop an Algorithm
4. Write the Program
5. Test the Program
6. Evaluate the Solution

Problem Solving Process

- Step 1 - Analyze the problem
  1. Outline the problem and its requirements
  2. Design steps (algorithm) to solve the problem
- Step 2 - Implement the algorithm
  1. Implement the algorithm in code
  2. Verify that the algorithm works
- Step 3 - Maintenance
  1. Use and modify the program if the problem domain changes

Example

Problem: Design algorithm to find the perimeter and area of a rectangle.

Analyze the problem:

- To find the perimeter and area of a rectangle, we need to know the rectangle's length and width.
- The perimeter and area of rectangle is given by the following formula:
  - **Perimeter = 2 . (length + width)**
  - **Area = length . width**

Example (continues)

Design algorithm:

1. Get length of Rectangle.
2. Get width of Rectangle.
3. Find the perimeter using the following equation:

$$\text{Perimeter} = 2 \,.\, (\text{length} + \text{width})$$

1. Find the area using the following equation:

$$\text{Area} = \text{length} \,.\, \text{width}$$

A C++ Program

```cpp
#include <iostream>
using namespace std;
int main()
{
    cout << "My first C++ program." << endl;
    cout << "The sum of 2 and 3 = " << 5 << endl;
    cout << "7 + 8 = " << 7 + 8 << endl;
    return 0;
}
```

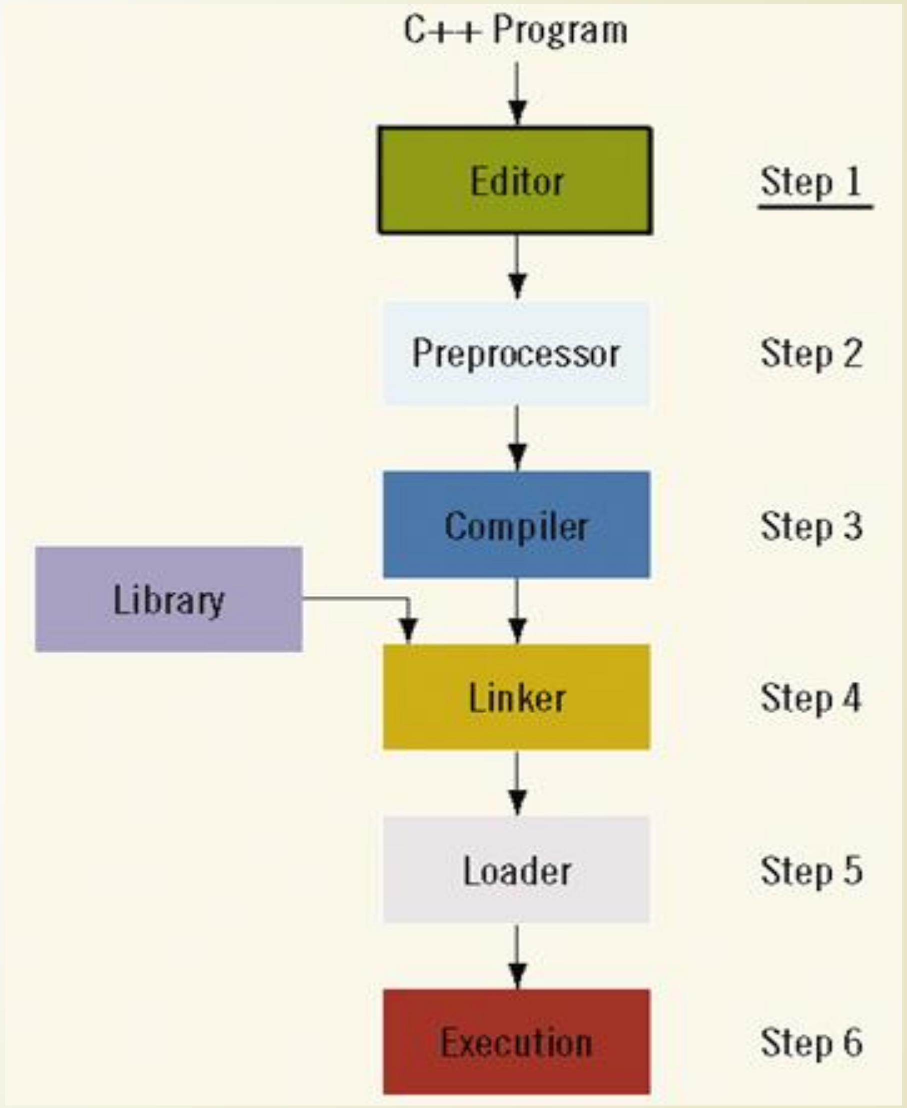
Sample Run:

```
My first C++ program.
The sum of 2 and 3 = 5
7 + 8 = 15
```

Pre-processing a Program

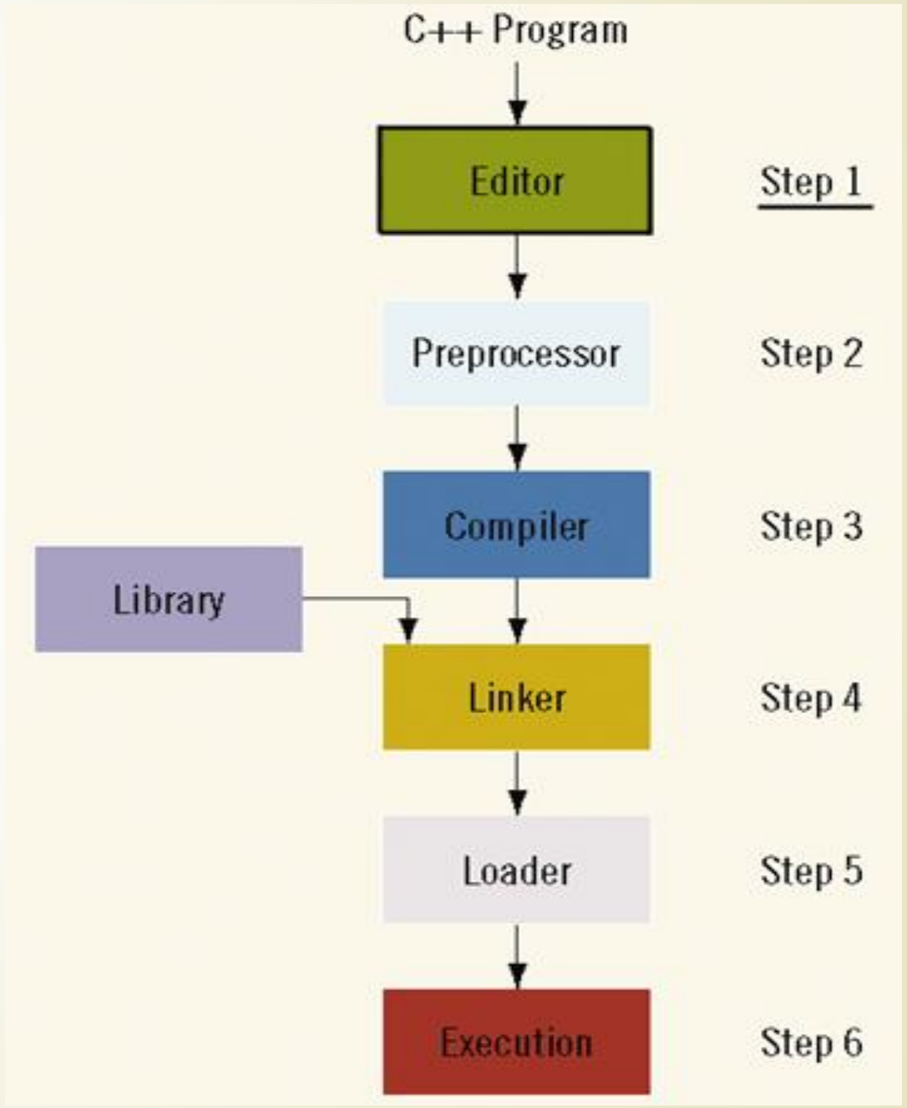
The following steps are necessary to process a program in C++:

Step 1: Use text editor to create a C++ program. This program is called **source code** or **source program.**



Pre-processing a Program

Step 2: In a C++ program, statements that begin with the symbol # are called preprocessor directives. These statements are processed by a program called **preprocessor**.

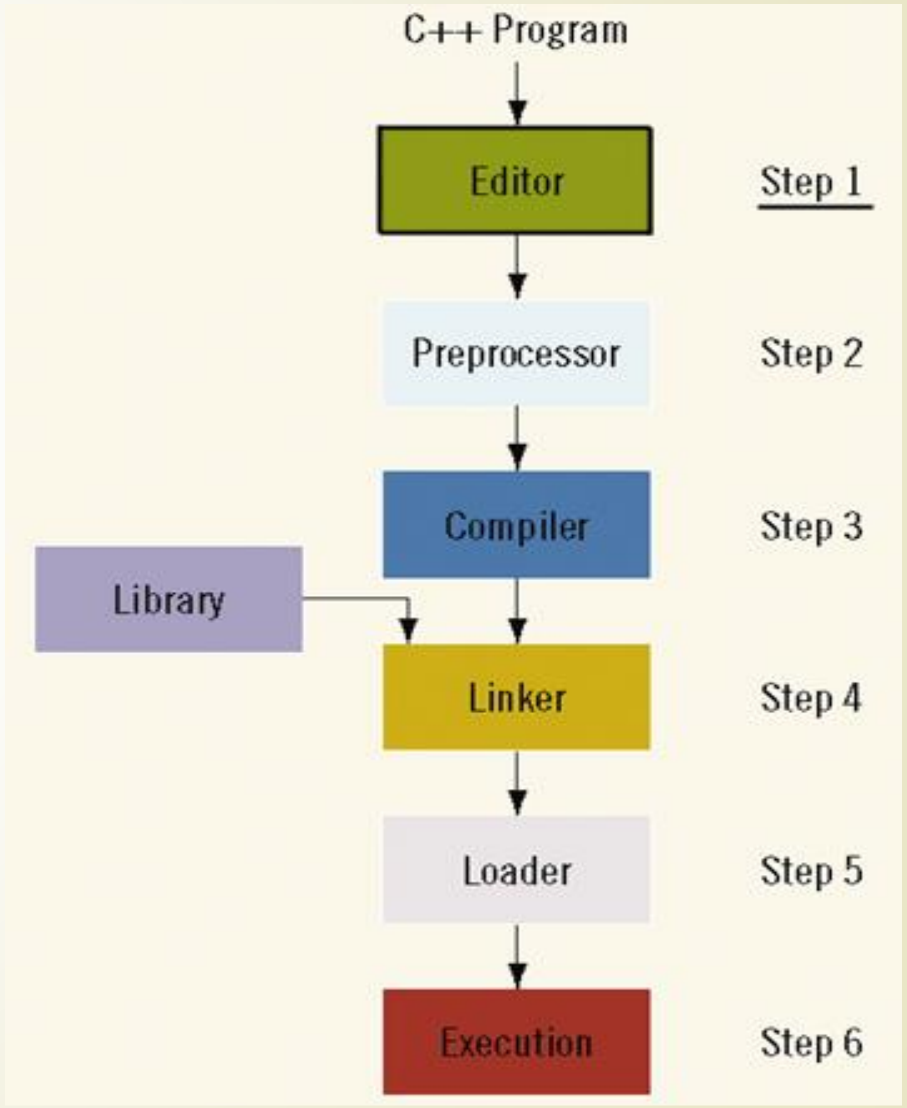


Pre-processing a Program

Step 3: Compiler is used to:

1. verifies that the program obeys the rules of the programming language and checks the source program for syntax errors.
2. Translate the program into equivalent machine language **(object program)**.

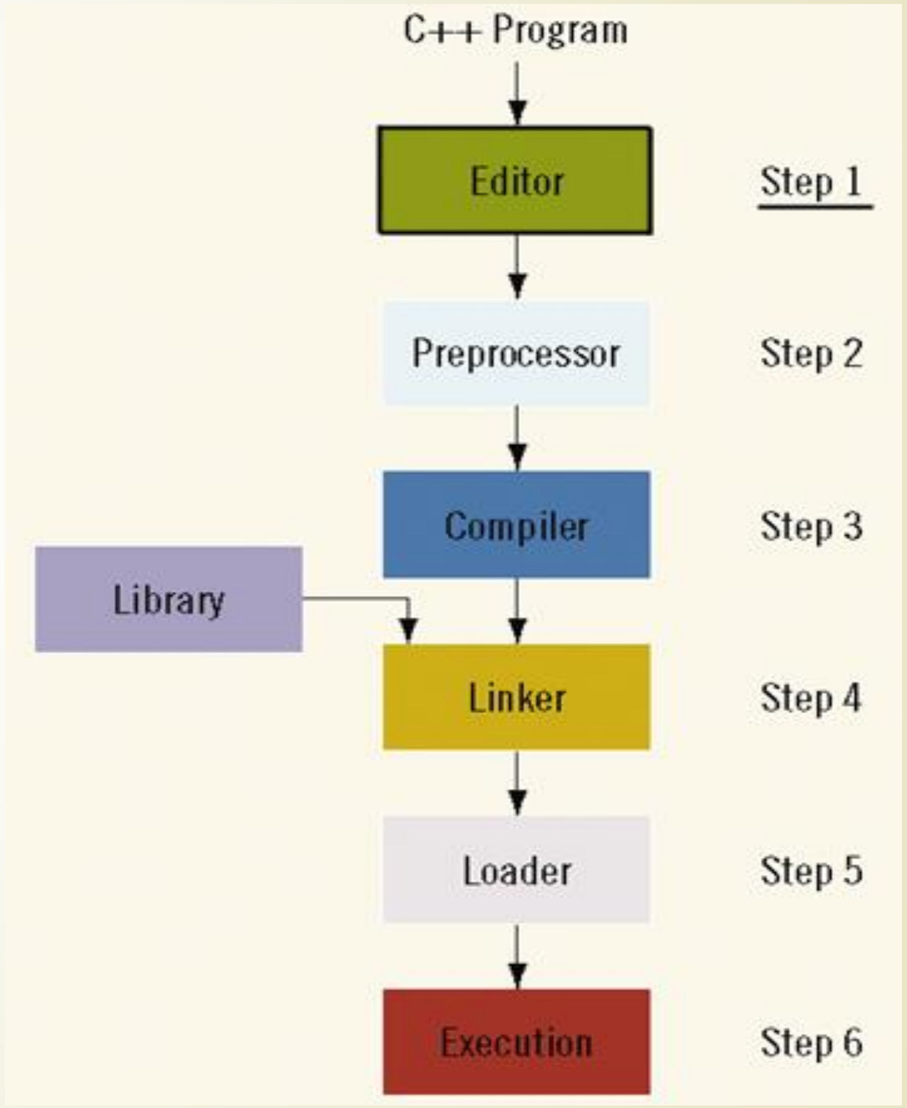
Compiler: translates a program written in a high-level language to machine language



Pre-processing a Program

Step 4: Programs in high level languages are developed using a software development kit (SDK).

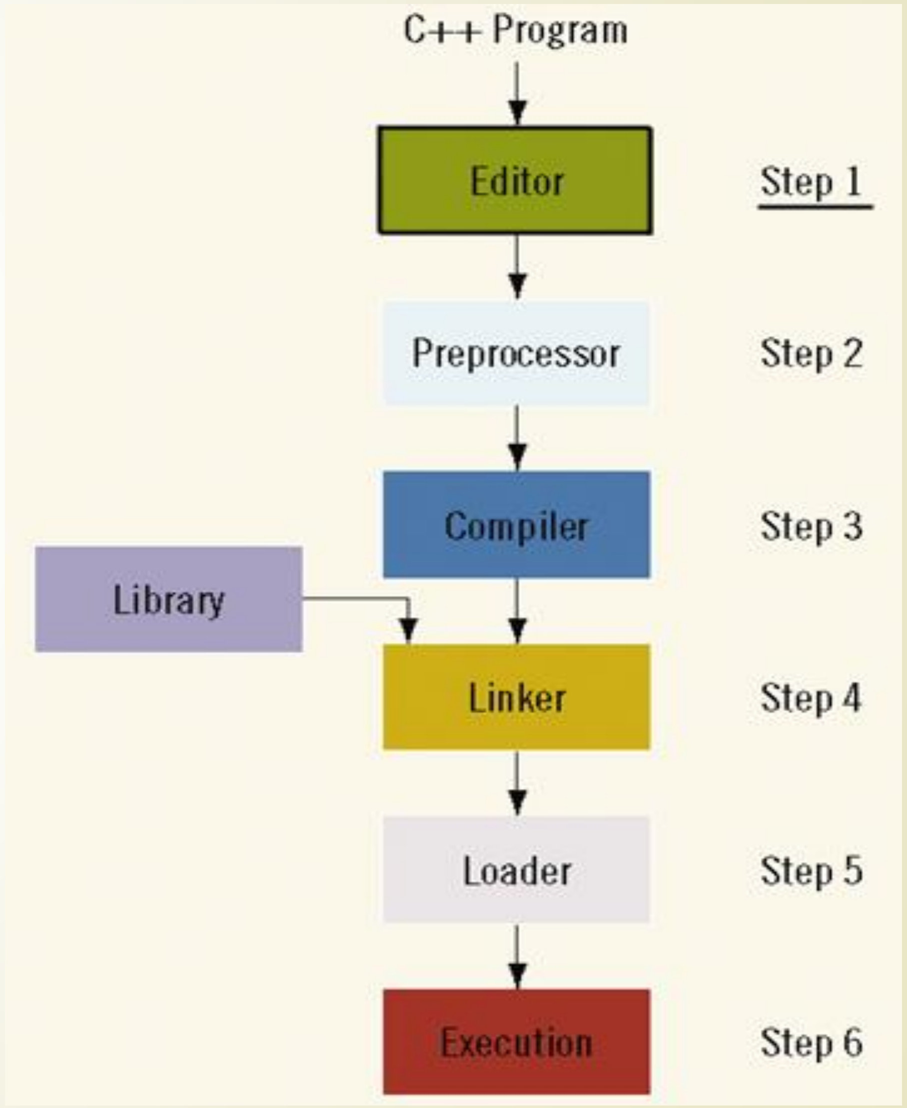
- SDK contains programs that are useful in creating your program such as mathematical functions.
- The prewritten code resides in a library.
- **Linker** combines the object code with the program from libraries.



Pre-processing a Program

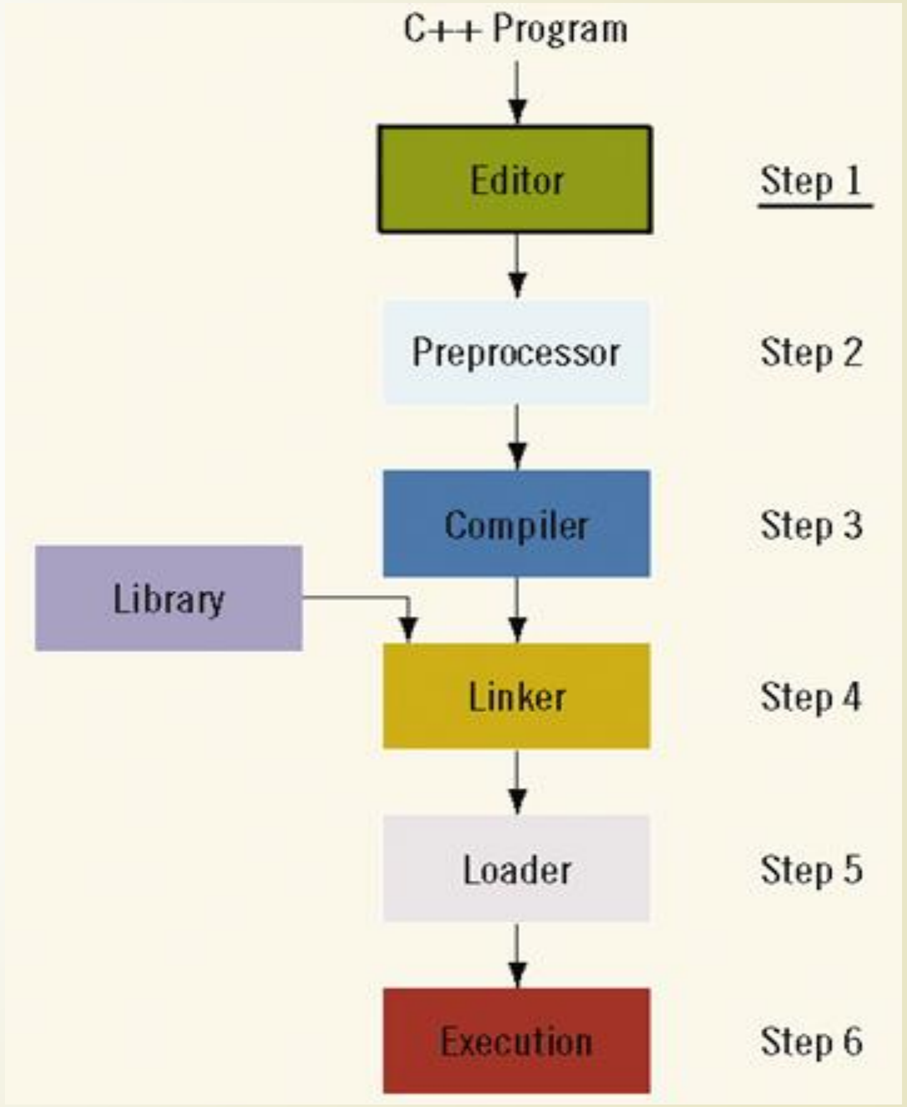
Step 5: You must load the executable program into main memory for execution.

Loader: a program that loads an executable program into main memory.



Pre-processing a Program

Step 6: The final step is to execute the program.



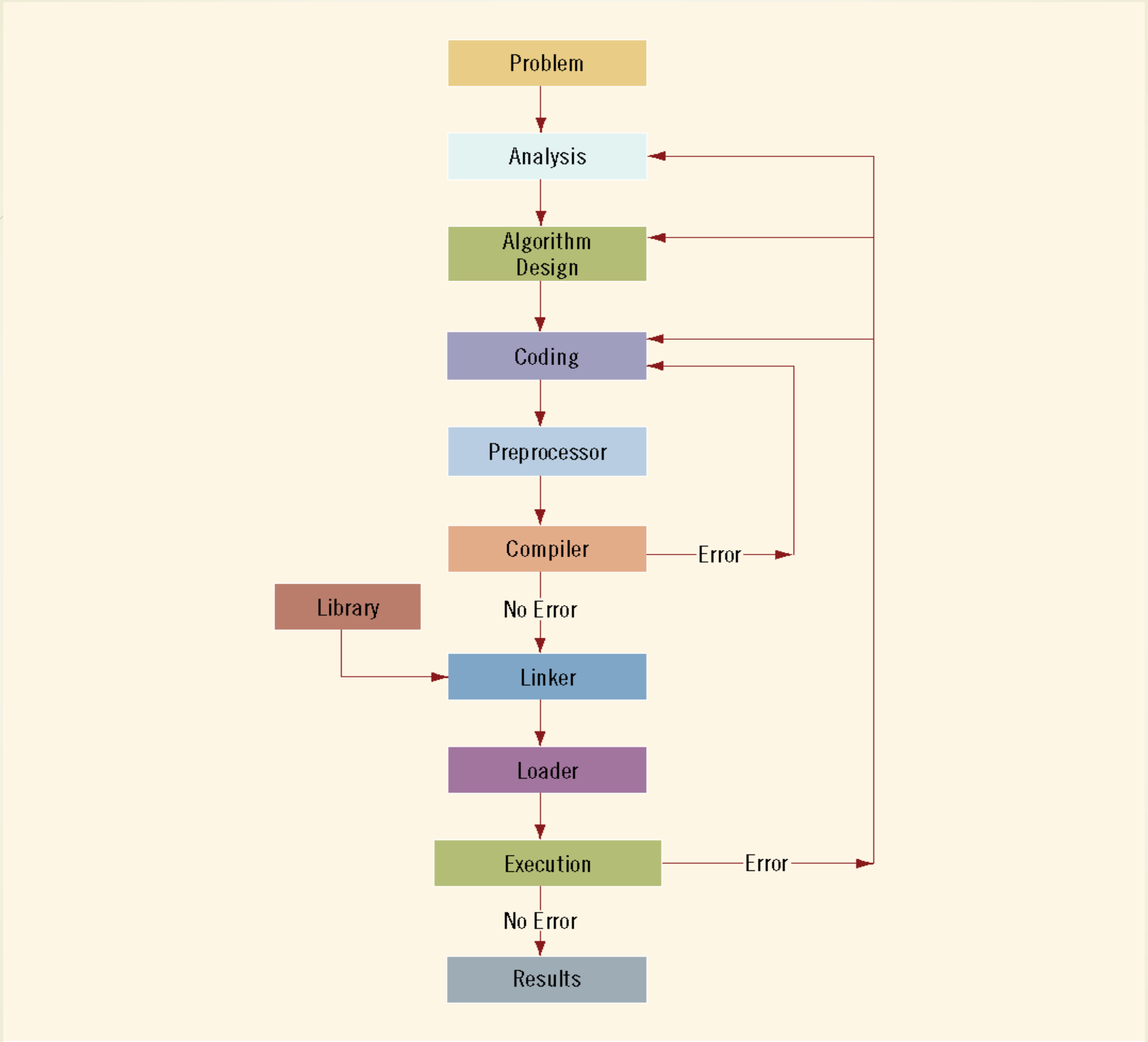


FIGURE 1-4 Problem analysis–coding–execution cycle

References

1. C++ Programming: From Problem Analysis to Program Design, Third Edition
2. https://www.just.edu.jo/~yahya-t/cs115/