Al-Mustaqbal University
College of Sciences
Department of Cybersecurity
المرحلة الاولى- اساسيات البرمجة

كليـــة العلـــوم
قسم الأمن السيبراني

Subject: Programming Fundamentals

First Stage

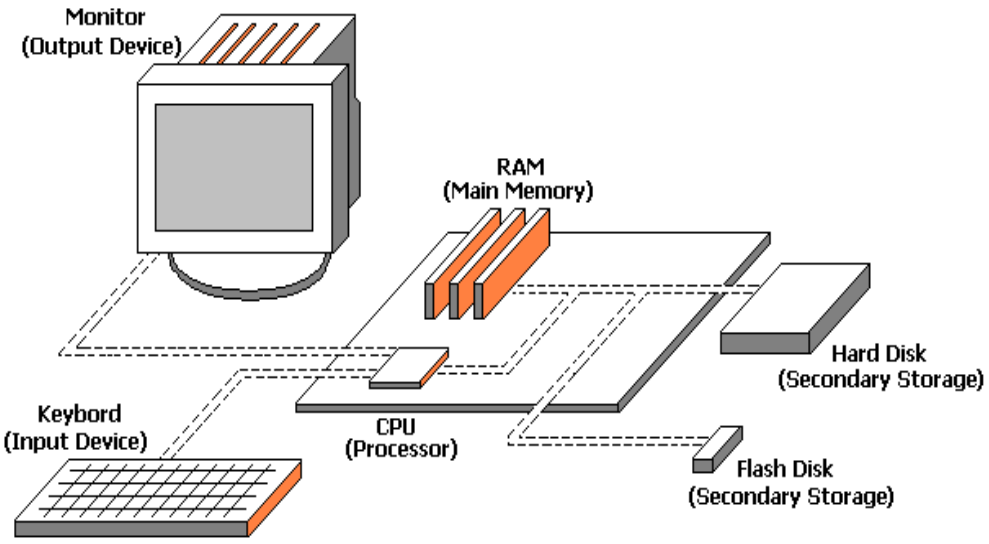
Assistant Lecturer: Jaber Baqer Al-Hamdani

Lecture (1)

Introduction to Programming Fundamentals

2024-2025

1. Introduction to Computers and Programming



hardware components

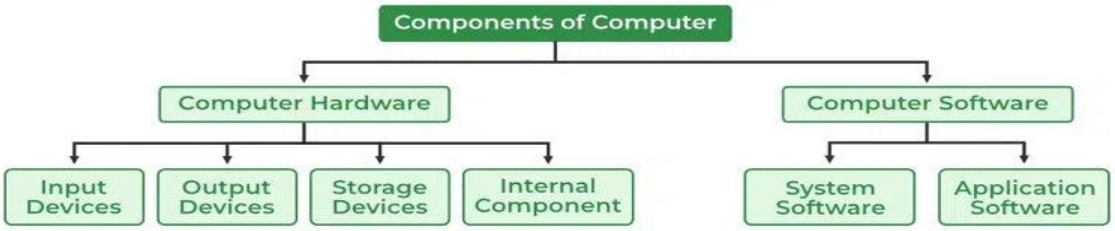
A **computer** is a powerful device capable of performing computations and making logical decisions at incredibly fast speeds—millions, or even billions, of times faster than humans. These computations are guided by a set of instructions known as **computer programs**.

- **Computer Program:** A set of instructions that a computer follows to perform a task.
- **Programming:** The process of writing instructions for a computer in a specific order to solve a problem.

2. Software vs. Hardware

- **Software (SW):** Refers to the programs and applications that run on a computer.
- **Hardware (HW):** Refers to the physical components of the computer system, such as the CPU, memory, and input/output devices.

Computers process data under the control of these programs, which tell the computer how to operate and what tasks to complete. Writing software involves programming languages that simplify instructions into a form that humans can understand.



3. History of Programming Languages

The evolution of programming languages has been a long journey, with each new language providing tools to make programming more efficient and powerful.

.

- 1954: Fortran.
- 1957: Cobol.
- 1958: Algol (*Base for Simula*).
- 1958: Lisp.
- 1961: B1000.
- 1962: Sketchpad.
- 1964: Basic.
- 1967: Simula67.
- 1968: FLEX.
- 1970: Pascal (*From Algol*).
- ***1971: C*** *(From a language called B)*.
- 1972: Smalltalk72 (*Based on Simula67 and Lisp*).
- 1976: Smalltalk76.
- 1979: ADA (*From Pascal*).
- ***1980: C with classes*** *(experimental version)*.
- **1983: C++ (by Bjarne Stroustrup).**
- 1986: Objective-C (*from C and Smalltalk*).
- 1986: Eiffel (*from Simula*).
- 1991: Sather (*From Eiffel*).
- 1991: Java.
- 2000: C#.


Bjarne Stroustrup
at: AT&T Labs

This brief timeline shows how programming languages evolved from specialized tasks to general-purpose and object-oriented solutions.

4. Introduction to C++

C++ is one of the most widely used programming languages today, known for its power, flexibility, and efficiency. It is an extension of the **C** language, adding support for **Object-Oriented Programming (OOP)**, which allows developers to model real-world entities using classes and objects.

Key Reasons for C++ Popularity:

1. **Combination of Structured and Object-Oriented Programming:** It supports both procedural programming (like C) and object-oriented paradigms.
2. **Efficiency and Performance:** C++ allows fine control over system resources, making it popular for system programming, game development, and applications requiring high performance.
3. **Functionality:** C++ includes features such as function overloading, templates, and exception handling, making it a versatile and powerful tool.

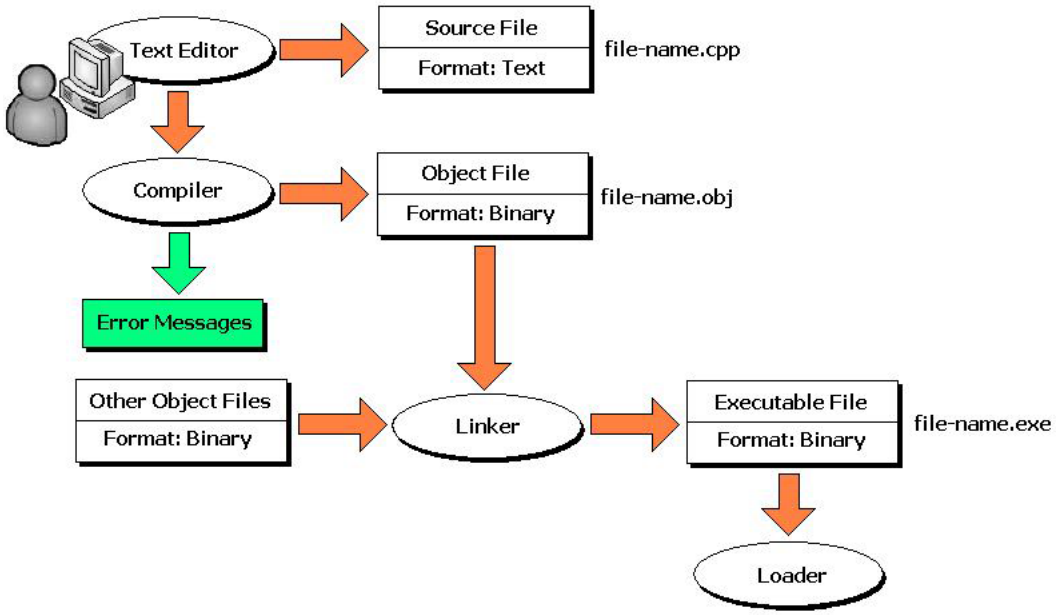
5. The Process of Developing a C++ Program

The development of a C++ program involves several key steps, each ensuring the code is correctly written, compiled, and executed.

Step-by-Step Development Process:

1. **Edit:** Write the C++ source code in a text editor and save it with a `.cpp` file extension.
2. **Pre-process:** The pre-processor handles operations like including libraries and defining macros.
3. **Compile:** The compiler converts the source code into **object code** (machine-readable code).
4. **Link:** The linker combines object code with library files to produce an executable file.
5. **Load:** The loader places the executable code into memory for the CPU to access.
6. **Execute:** The CPU runs the program, performing computations and making decisions based on the code.

Diagram of C++ Program Development Process:



- **Source Code:** `file-name.cpp` (Text format)
- **Object Code:** `file-name.obj` (Binary format)
- **Executable Code:** `file-name.exe` (Binary format)

6. Simple Example of a C++ Program

Let's now write a simple C++ program that outputs "Welcome" to the screen.

```
#include <iostream>     // Include the iostream library for input/output
using namespace std;    // Use the standard namespace

int main() {           // Main function where the program starts
    cout << "Welcome to C++ Programming!";         // Print the message
    return 0;          // Exit the program with success status
}
Output:
Welcome to C++ Programming!
```

Explanation of Code:

- **#include <iostream>:** Includes the library that allows input and output operations.
- **using namespace std;**: Allows the use of the standard C++ library without needing to prefix everything with `std::`.
- **int main() {...}:** The `main` function is the entry point for every C++ program.
- **cout:** This is the output stream object that displays data to the screen.
- **return 0;**: Signals successful execution of the program.

7. Conclusion

In this lecture, we introduced the fundamental concepts of **computers** and **programming**. We looked at the history and evolution of programming languages, with a focus on **C++**. C++ is a powerful language that combines structured and object-oriented programming, providing many tools for developing efficient programs. Finally, we covered the basic process of developing a C++ program and saw a simple example that outputs text to the screen.