Chapter 1: Introduction to Computers and Programming

Starting Out with C++
Early Objects
Eighth Edition

by Tony Gaddis, Judy Walters, and Godfrey Muganda

Addison-Wesley is an imprint of PEARSON

Topics

1.1 Why Program?

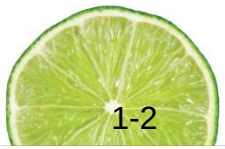
1.2 Computer Systems: Hardware and Software

1.3 Programs and Programming Languages

1.4 What Is a Program Made of?

1.5 Input, Processing, and Output

1.6 The Programming Process



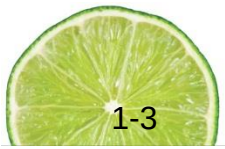
1.1 Why Program?

Computer – programmable machine designed to follow instructions

Program/Software – instructions in computer memory to make it do something

Programmer – person who writes instructions (programs) to make computer perform a task

SO, without programmers, no programs; without programs, the computer cannot do anything



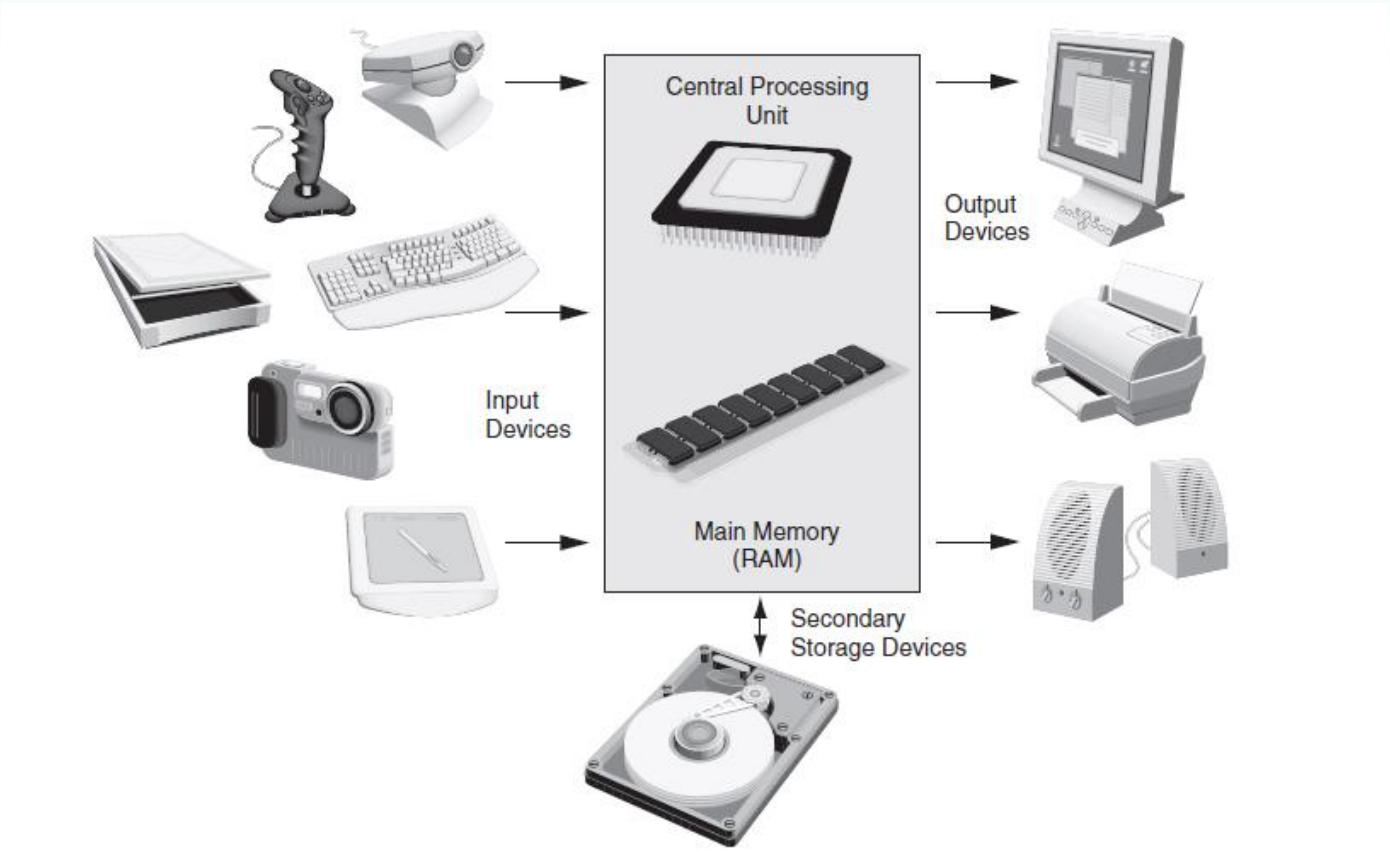
1.2 Computer Systems: Hardware and Software

Hardware – Physical components of a computer

Main Hardware Component Categories

1. Central Processing Unit (CPU)
2. Main memory (RAM)
3. Secondary storage devices
4. Input Devices
5. Output Devices

Main Hardware Component Categories

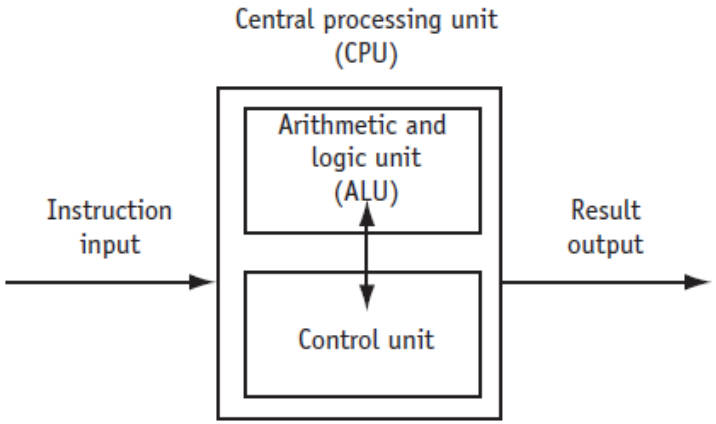


Central Processing Unit (CPU)

CPU – Hardware component that runs programs

Includes

- Control Unit
  - Retrieves and decodes program instructions
  - Coordinates computer operations
- Arithmetic & Logic Unit (ALU)
  - Performs mathematical operations



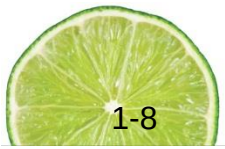
The CPU's Role in Running a Program

Cycle through:

- Fetch: get the next program instruction from main memory
- Decode: interpret the instruction and generate a signal
- Execute: route the signal to the appropriate component to perform an operation

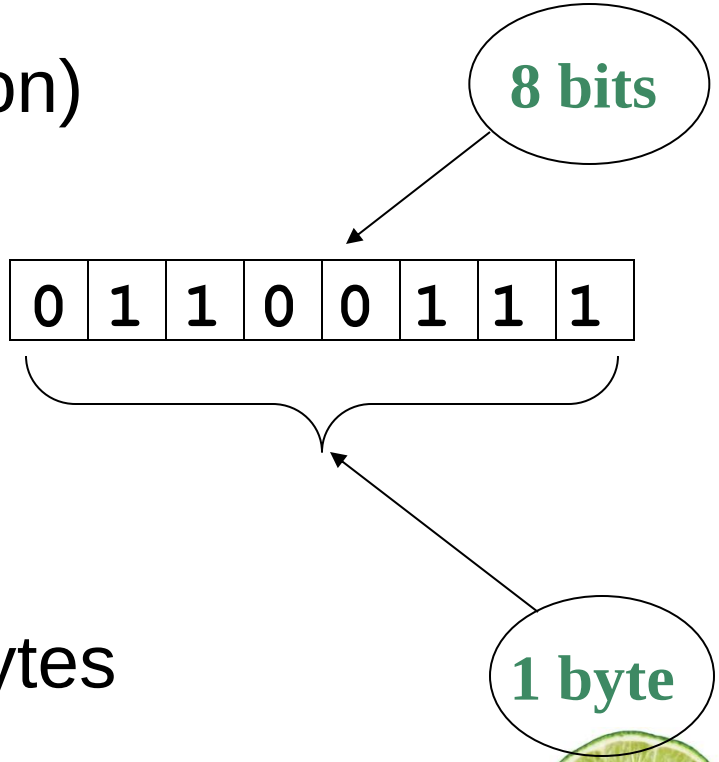
Main Memory

- Holds both program instructions and data
- Volatile – erased when program terminates or computer is turned off
- Also called Random Access Memory (RAM)

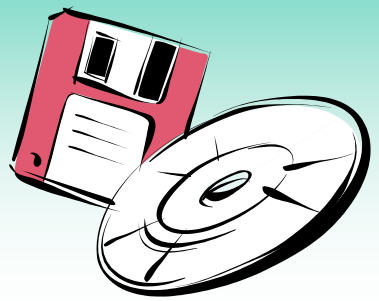


Main Memory Organization

- Bit
  - Smallest piece of memory
  - Stands for **b**inary dig**it**
  - Has values 0 (off) or 1 (on)
- Byte
  - Is 8 consecutive bits
  - Has an address
- Word
  - Usually 4 consecutive bytes



Secondary Storage

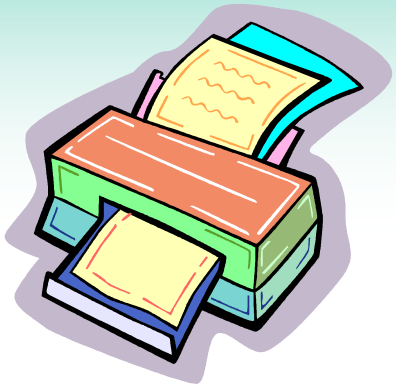


- Non-volatile - data retained when program is not running or computer is turned off
- Comes in a variety of media
  - magnetic: floppy or hard disk drive, internal or external
  - optical: CD or DVD drive
  - flash: USB flash drive

Input Devices

- Used to send information to the computer from outside
- Many devices can provide input
  - keyboard, mouse, microphone, scanner, digital camera, disk drive, CD/DVD drive, USB flash drive

Output Devices



- Used to send information from the computer to the outside
- Many devices can be used for output
  - Computer screen, printer, speakers, disk drive, CD/DVD recorder, USB flash drive

Software Programs That Run on a Computer

- System software
  - programs that manage the computer hardware and the programs that run on the computer
  - Operating Systems
    - Controls operation of computer
    - Manages connected devices
    - Runs programs
  - Utility Programs
    - Support programs that enhance computer operations
    - Examples: anti-virus software, data backup, data compression
  - Software development tools
    - Used by programmers to create software
    - Examples: compilers, integrated development environments (IDEs)

1.3 Programs and Programming Languages

- **Program**

  a set of instructions directing a computer to perform a task

- **Programming Language**

  a language used to write programs

Algorithm

Algorithm: a set of steps to perform a task or to solve a problem

Order is important. Steps must be performed sequentially

Programs and Programming Languages

Types of languages

- Low-level: used for communication with computer hardware directly.
- High-level: closer to human language

From a High-level Program to an Executable File

a) Create file containing the program with a text editor.

b) Run preprocessor to convert source file directives to source code program statements.

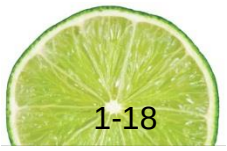
c) Run compiler to convert source program statements into machine instructions.

From a High-level Program to an Executable File

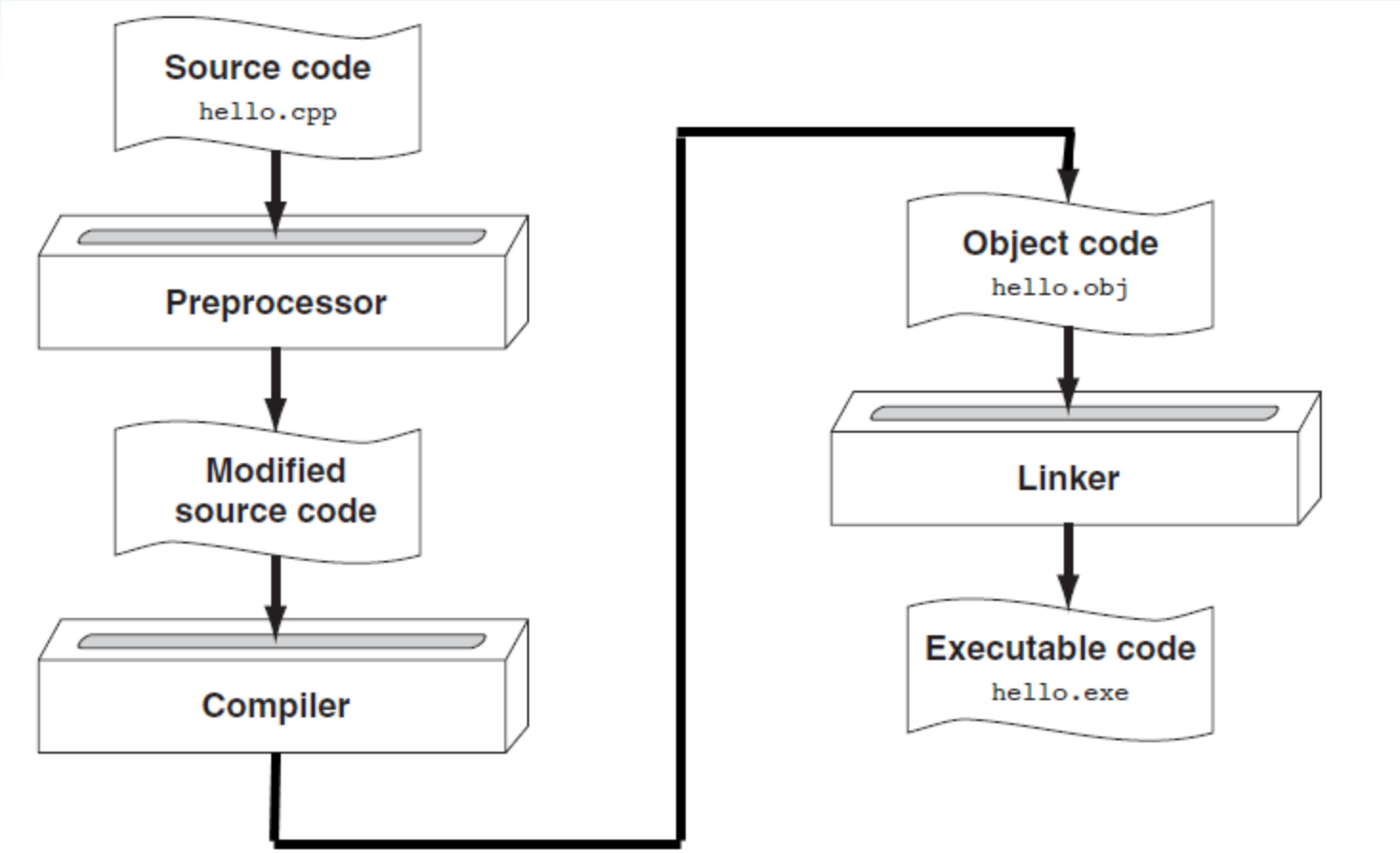
d) Run linker to connect hardware-specific library code to machine instructions, producing an executable file.

Steps b) through d) are often performed by a single command or button click.

Errors occuring at any step will prevent execution of the following steps.



From a High-level Program to an Executable File



1.4 What Is a Program Made Of?

Common elements in programming languages:

- Key Words
- Programmer-Defined Identifiers
- Operators
- Punctuation
- Syntax

Example Program

```
#include <iostream>
using namespace std;

int main()
{
   double num1 = 5,
          num2, sum;
    num2 = 12;

    sum = num1 + num2;
    cout << "The sum is " << sum;
    return 0;
}
```


Key Words

- Also known as reserved words
- Have a special meaning in C++
- Can not be used for another purpose
- Written using lowercase letters
- Examples in program (shown in green):

```
using namespace std;
int main()
```


Programmer-Defined Identifiers

- Names made up by the programmer
- Not part of the C++ language
- Used to represent various things, such as variables (memory locations)
- Example in program (shown in green):

```
double num1
```


Operators

- Used to perform operations on data
- Many types of operators
  - Arithmetic: `+, -, *, /`
  - Assignment: `=`
- Examples in program (shown in green):

```
num2 = 12;
sum = num1 + num2;
```


Punctuation

- Characters that mark the end of a statement, or that separate items in a list
- Example in program

```
double num1 = 5,
       num2, sum;
num2 = 12;
```


Lines vs. Statements

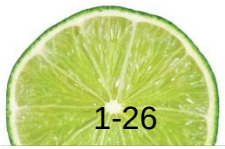
In a source file,

A line is all of the characters entered before a carriage return.

Blank lines improve the readability of a program.

Here are four sample lines. Line 3 is blank:

```
1. double num1 = 5, num2, sum;
2. num2 = 12;
3.
4. sum = num1 + num2;
```



Lines vs. Statements

In a source file,

- A statement is an instruction to the computer to perform an action.
- A statement may contain keywords, operators, programmer-defined identifiers, and punctuation.
- A statement may fit on one line, or it may occupy multiple lines.
- Here is a single statement that uses two lines:

```
double num1 = 5,
       num2, sum;
```

Variables

- A variable is a named location in computer memory (in RAM)
- It holds a piece of data. The data that it holds may change while the program is running.
- The name of the variable should reflect its purpose
- It must be *defined* before it can be used. Variable definitions indicate the variable name and the type of data that it can hold.
- Example variable definition:

```
double num1;
```

1.5 Input, Processing, and Output

Three steps that many programs perform

1) Gather input data
   - from keyboard
   - from files on disk drives
2) Process the input data
3) Display the results as output
   - send it to the screen or a printer
   - write it to a file

1.6 The Programming Process

1. Define what the program is to do.
2. Visualize the program running on the computer.
3. Use design tools to create a model of the program.

   Hierarchy charts, flowcharts, pseudocode, etc.
4. Check the model for logical errors.
5. Write the program source code.
6. Compile the source code.

The Programming Process (cont.)

7. Correct any errors found during compilation.
8. Link the program to create an executable file.
9. Run the program using test data for input.
10. Correct any errors found while running the program.

Repeat steps 4 - 10 as many times as necessary.

11. Validate the results of the program.
    Does the program do what was defined in step 1?

Chapter 1: Introduction to Computers and Programming

The End!!!