

Table of Contents

Table of Contents	1
Learning Objectives	1
Introduction.....	2
Development Environment	2
Phase 1: Creating a Program.....	3
Phase 2 and 3: Preprocessing and Compiling	4
Phase 4: Linking	4
Phase 5: Loading.....	4
Phase 6: Execution.....	4
C++ Program Basics	5
Comments	5
Preprocessor Directive	5
White Spaces.....	5
Function	6
Function Body.....	6
Statement.....	6
Stream Insertion Operator.....	6
Using Multiple Output Statements.....	6
Printing Other Data Types	6
Escape Characters	7
Program: Escape Sequence	7
Arithmetic	8
Arithmetic Operators	8
Program: Arithmetic Operators.....	8
Operator Precedence	9
Precedence of Arithmetic Operators.....	9
Example using Operator Precedence	10
Writing Algebraic expression in C++	10
Practice Exercise 1:.....	10
Reference for Further Reading.....	11

Learning Objectives

After completing this handout, you will be able to:

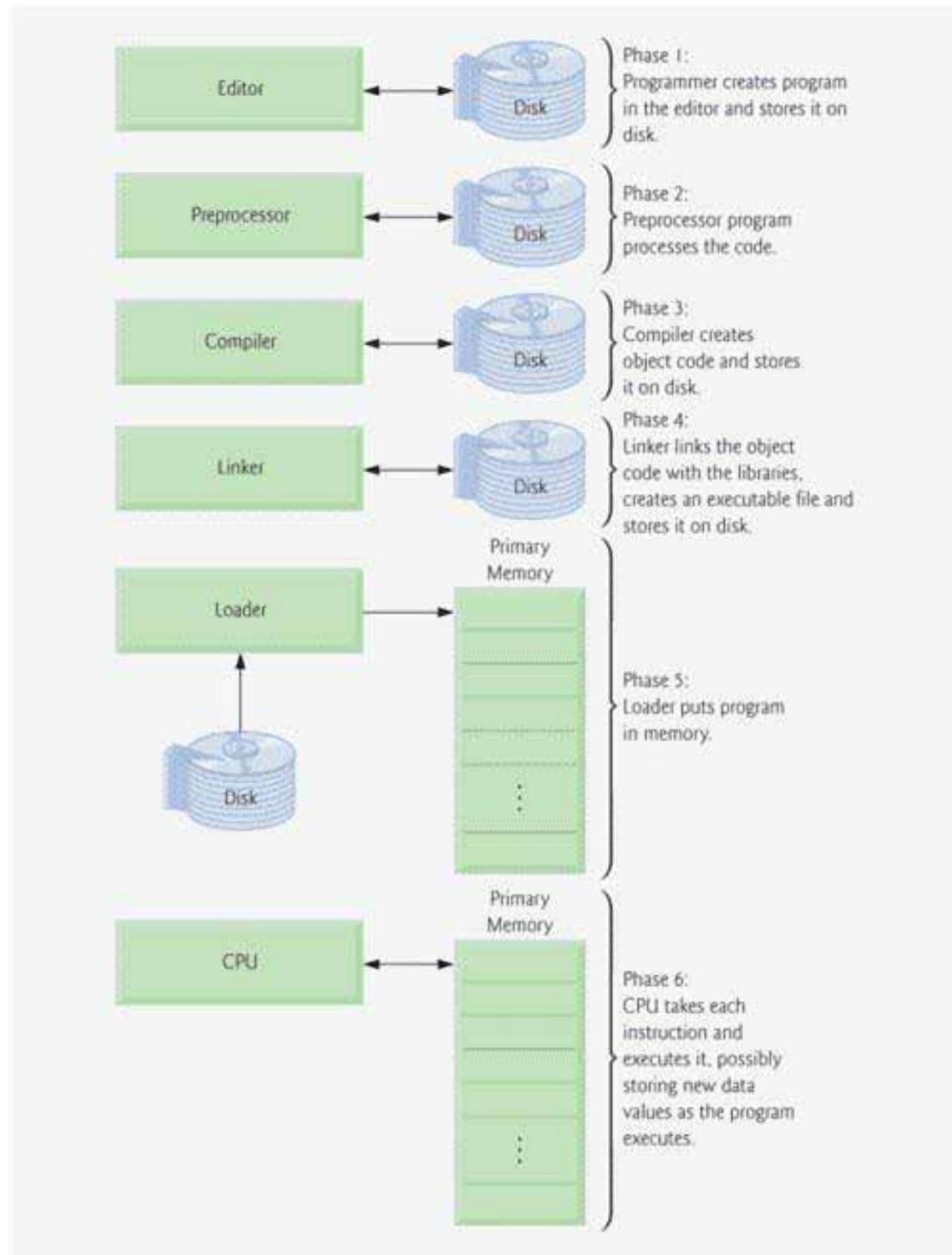
- Simple computer programs in C++.
- Write simple output statements.
- Use arithmetic operators.
- Understand the precedence of arithmetic operators.
- Use the escape sequences.

Introduction

- C++ is a powerful computer programming language that is appropriate for technically oriented people with little or no programming experience and for experienced programmers to use in building substantial information systems.
- C++ is one of today's most popular software development languages.
- C++ evolved from C, which evolved from two previous programming languages, BCPL and B.
- C++, an extension of C, was developed by Bjarne Stroustrup in the early 1980s at Bell Laboratories. C++ provides a number of features that "spruce up" the C language, but more importantly, it provides capabilities for object-oriented programming.
- C++ programs consist of pieces called classes and functions.

Development Environment

- C++ programs typically go through six phases: edit, preprocess, compile, link, load and execute.



Phase 1: Creating a Program

- Phase 1 consists of editing a file with an editor program (normally known simply as an editor).

- You type a C++ program (typically referred to as source code) using the editor.
- C++ source code file names often end with the .cpp.
- C++ software packages for Microsoft Windows such as Borland C++, Metrowerks CodeWarrior, and Microsoft Visual C++ have editors integrated into the programming environment. You can also use a simple text editor, such as Notepad in Windows, to write your C++ code.

Phase 2 and 3: Preprocessing and Compiling

- In phase 2, the programmer gives the command to compile the program.
- In a C++ system, a preprocessor program executes automatically before the compiler's translation phase begins.
- The C++ preprocessor obeys commands called preprocessor directives, which indicate that certain manipulations are to be performed on the program before compilation.

Phase 4: Linking

- Phase 4 is called linking.
- C++ programs typically contain references to functions and data defined elsewhere, such as in the standard libraries or in the private libraries of groups of programmers working on a particular project.
- The object code produced by the C++ compiler typically contains "holes" due to these missing parts.
- A linker links the object code with the code for the missing functions to produce an executable image.

Phase 5: Loading

- Phase 5 is called loading.
- Before a program can be executed, it must first be placed in memory.
- This is done by the loader, which takes the executable image from disk and transfers it to memory.

Phase 6: Execution

- Finally, the computer, under the control of its CPU, executes the program one instruction at a time.

C++ Program Basics

```
1 /*
2 Author: Fawad Ishaq Ch
3 Program: Hello World
4 Description: Prints a simple text on the screen.
5 Creation Date: Jan 04, 2008
6 */
7 #include <iostream>
8
9 int main()
10 {
11     std::cout << "Hello World!";
12     return 0;
13 }
```

Comments

- Lines beginning with // indicate that the remainder of the line is a comment.
- Programmers insert comments to document programs and also help people read and understand them.
- Comments do not cause the computer to perform any action when the program is run they are ignored by the C++ compiler and do not cause any machine-language object code to be generated.
- A comment beginning with // is called a single-line comment because it terminates at the end of the current line.
- Multi-line comments can be inserted by using the pair of /* and */. Anything that comes in between the pair is a comment.

Preprocessor Directive

- Lines that begin with # are processed by the preprocessor before the program is compiled.
- This line notifies the preprocessor to include in the program the contents of the input/output stream header file <iostream>.
- This file must be included for any program that outputs data to the screen or inputs data from the keyboard using C++-style stream input/output.

White Spaces

- Programmers use blank lines, space characters and tab characters (i.e., "tabs") to make programs easier to read. Together, these characters are known as white space.
- White-space characters are normally ignored by the compiler.

Function

- Functions are one of the fundamental building blocks of C++.
- C++ programs typically consist of one or more functions and classes.
- Exactly one function in every program must be main.
- C++ programs begin executing at function main, even if main is not the first function in the program.
- The keyword `int` indicates that main "returns" an integer (whole number) value.
- A keyword is a word in code that is reserved by C++ for a specific use.

Function Body

- The left brace `{` must begin the body of every function. A corresponding right brace `}` must end each function's body.

Statement

- The entire `std::cout << "Hello World!\n";` is called a statement.
- Every C++ statement must end with a semicolon (also known as the statement terminator).
- Output and input in C++ are accomplished with streams of characters.

Stream Insertion Operator

- The `<<` operator is referred to as the stream insertion operator. When this program executes, the value to the right of the operator, the right operand, is inserted in the output stream.

Using Multiple Output Statements

- **This is my first C++ program** can be printed in many ways.

- **Example 1:**

```
cout << "This is my";  
cout << "first C++ Program";
```

- **Example 2:**

```
cout << "This is my\n first C++ program";
```

- **Example 3:**

```
cout << "This is my" << endl << " first C++ program";
```

Printing Other Data Types

- **Printing numbers:**

```
cout << 100;
```



```
24     return 0;
25 }
```

Arithmetic

- The arithmetic operators are all binary operators, i.e., operators that take two operands.
- For example, the expression `number1 + number2` contains the binary operator `+` and the two operands `number1` and `number2`.

Arithmetic Operators

C++ operation	C++ arithmetic operation
Addition	<code>+</code>
Subtraction	<code>-</code>
Multiplication	<code>*</code>
Division	<code>/</code>
Modulus	<code>%</code>

- C++ provides the modulus operator, `%`, that yields the remainder after integer division.
- The modulus operator can be used only with integer operands.
- The expression `x % y` yields the remainder after `x` is divided by `y`.

Program: Arithmetic Operators

```
1 /*
2 Author: Fawad Ishaq Ch
3 Program: Arithmetic Operators
4 Description: Shows the use of basic arithmetic operators
5 Creation Date: Jan 04, 2008
6 */
7 #include <iostream>
8
9 int main()
10 {
11     // Addition
12     std::cout << "5 + 2 = " << 5 + 2 << "\n";
13     // Subtraction
14     std::cout << "5 - 2 = " << 5 - 2 << "\n";
15     // Multiplication
16     std::cout << "5 * 2 = " << 5 * 2 << "\n";
17     // Division
```

```

18     std::cout << "5 / 2 = " << 5 / 2 << "\n";
19     // Modulus
20     std::cout << "5 % 2 = " << 5 % 2 << "\n";
21     return 0;
22 }

```

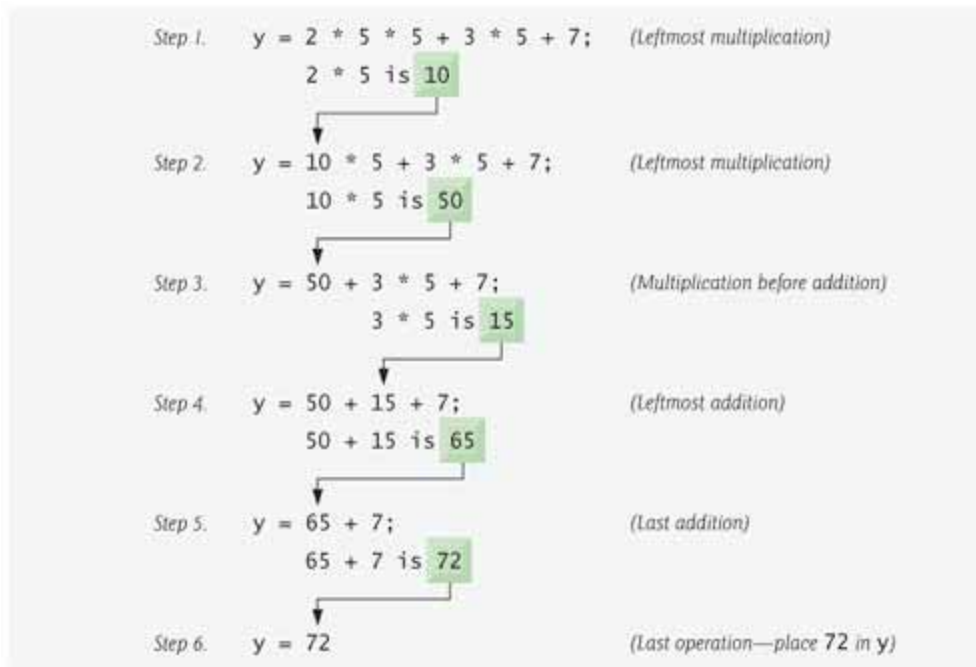
Operator Precedence

- Operators in expressions contained within pairs of parentheses () are evaluated first.
- Multiplication, division and modulus operations are applied next. If an expression contains several multiplication, division and modulus operations, operators are applied from left to right. Multiplication, division and modulus are said to be on the same level of precedence.
- Addition and subtraction operations are applied last. If an expression contains several addition and subtraction operations, operators are applied from left to right. Addition and subtraction also have the same level of precedence.

Precedence of Arithmetic Operators

Operator(s)	Operation(s)	Order of evaluation (precedence)
()	Parentheses	Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses "on the same level" (i.e., not nested), they are evaluated left to right.
* / %	Multiplication Division Modulus	Evaluated second. If there are several, they are evaluated left to right.
+ -	Addition Subtraction	Evaluated last. If there are several, they are evaluated left to right.

Example using Operator Precedence



Writing Algebraic expression in C++

Algebra: $m = \frac{a + b + c + d + e}{5}$

C++: $m = (a + b + c + d + e) / 5;$

Algebra: $y = mx + b$

C++: $y = m * x + b;$

Algebra: $z = pr \% q + w/x - y$

C++: $z = p * r \% q + w / x - y;$

Practice Exercise 1:

Write a single C++ statement to accomplish each of the following (assume that using declarations have not been used):

- Print the message "This is a C++ program" on one line.
- Print the message "This is a C++ program" on two lines. End the first line with C++.
- Print the message "This is a C++ program" with each word on a separate line.

- Print the message "This is a C++ program" with each word separated from the next by a tab.

Reference for Further Reading

- C++ How to Program, Fifth Edition, By H.M. Deitel, Chapter # 2.
- Object-Oriented Programming in C++, Third Edition, By Robert Lafore, Chapter # 3.