



In this lecture you will learn:

- **The Basics of a C++ Program.**
- **Comments.**
- **Data type**

The Basics of a C++ Program

A **computer program** is a sequence of statements whose objective is to accomplish a task. **Programming** is a process of planning and creating a program.

Example 1:

```
/** *****  
  
// This is a simple C++ program. It displays four lines  
// of text, including the sum of two numbers.  
  
/** *****  
  
#include <iostream>  
  
using namespace std;  
  
int main()  
{  
  
    int num;  
  
    num = 6;  
  
    cout << "My first C++ program." << endl;  
  
    cout << "The sum of 2 and 3 = " << 5 << endl;  
  
    cout << "7 + 8 = " << 7 + 8 << endl;
```

```
cout << "Num = " << num << endl;  
  
return 0;  
  
}
```

Sample Run: (When you compile and execute this program, the following four lines are displayed on the screen.)

My first C++ program.

The sum of 2 and 3 = 5

7 + 8 = 15

Num = 6

A C++ program is a collection of one or more subprograms, called **functions**. Some functions, called predefined or standard functions, are already written and are provided as part of the system. But to accomplish most tasks, programmers must learn to write their own functions.

To write meaningful programs, you must learn the programming language's special symbols, words, and syntax rules. The syntax rules tell you which statements (instructions) are legal, or accepted by the programming language, and which are not. You must also learn semantic rules, which determine the meaning of the instructions. The programming language's rules, symbols, and special words enable you to write programs to solve problems. The syntax rules determine which instructions are valid.

Programming language: A set of rules, symbols, and special words.

Comments

The program that you write should be clear not only to you, but also to the reader of your program. Part of good programming is the inclusion of comments in the program.

Comments are for the reader, not for the compiler. So when a compiler compiles a program to check for the syntax errors, it completely ignores comments.

There are two common types of comments in a C++ program single-line comments and multiple-line comments. Single-line comments begin with `//` and can be placed anywhere in the line. Everything encountered in that line after `//` is ignored by the compiler. For example, consider the following statement:

```
cout << "7 + 8 = " << 7 + 8 << endl;
```

You can put comments at the end of this line as follows:

```
cout << "7 + 8 = " << 7 + 8 << endl; //prints: 7 + 8 = 15
```

This comment could be meaningful for a beginning programmer.

Multiple-line comments are enclosed between `/*` and `*/`. The compiler ignores anything that appears between `/*` and `*/`. For example, the following is an example of a multiple-line comment:

```
/*
```

**You can include comments that can
occupy several lines.**

```
*/
```

Special Symbols The smallest individual unit of a program written in any language is called a token. C++'s tokens are divided into **special symbols**, **word symbols**, and **identifiers**.

Following are some of the special symbols:

+ - * /

. ; ? ,

<= != == >=

The first row includes mathematical symbols for addition, subtraction, multiplication, and division. The second row consists of punctuation marks taken from English grammar. Note that the comma is also a special symbol. In C++, commas are used to separate items in a list. Semicolons are used to end a C++ statement. Note that a blank, which is not shown above, is also a special symbol. You create a blank symbol by pressing the space bar (only once) on the keyboard. The third row consists of tokens made up of two characters that are regarded as a single symbol. No character can come between the two characters in the token, not even a blank.

Reserved Words (Keywords)

A second category of tokens is word symbols. Some of the word symbols include the following:

int, float, double, char, const, void, return

Reserved words are also called keywords. The letters that make up a reserved word are always lowercase. Like the special symbols, each is considered to be a single symbol. Furthermore, word symbols cannot be

redefined within any program; that is, they cannot be used for anything other than their intended use.

Identifiers

A third category of tokens is identifiers. Identifiers are names of things that appear in programs, such as variables, constants, and functions. All identifiers must obey C++'s rules for identifiers.

Identifier: A C++ identifier consists of letters, digits, and the underscore character (`_`) and must begin with a letter or underscore.

The following are legal identifiers in C++:

first

conversion

payRate

counter1

Illegal Identifier	Description
<code>employee Salary</code>	There can be no space between <code>employee</code> and <code>Salary</code> .
<code>Hello!</code>	The exclamation mark cannot be used in an identifier.
<code>one+two</code>	The symbol <code>+</code> cannot be used in an identifier.
<code>2nd</code>	An identifier cannot begin with a digit.

Data Types

The objective of a C++ program is to manipulate data. Different programs manipulate different data.

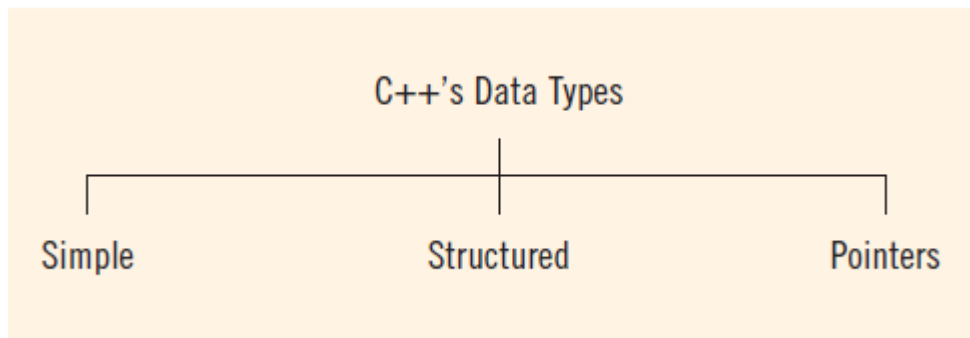
Data type: A set of values together with a set of operations.

C++ data types fall into the following three categories and are illustrated in Figure below:

1. Simple data type

2. Structured data type

3. Pointers



Simple Data Types

The simple data type is the fundamental data type in C++ because it becomes a building block for the structured data type. There are three categories of simple data:

1. Integral, which is a data type that deals with integers, or numbers without a decimal part
2. Floating-point, which is a data type that deals with decimal numbers
3. Enumeration, which is a user-defined data type

Integral data types are further classified into the following nine categories: **char, short, int, long, bool, unsigned char, unsigned short, unsigned int, and unsigned long.**

Integral types	Floating-point types
<code>bool</code>	<code>float</code>
<code>char</code>	<code>double</code>
<code>signed char</code>	<code>long double</code>
<code>unsigned char</code>	
<code>short int</code>	
<code>unsigned short int</code>	
<code>int</code>	
<code>unsigned int</code>	
<code>long int</code>	
<code>unsigned long int</code>	
<code>wchar_t</code>	

Allocating Memory with Constants and Variables

Named constant: A memory location whose content is not allowed to change during program execution.

To allocate memory, we use C++'s declaration statements. The syntax to declare a named constant is:

```
const dataType identifier = value;
```

```
const double CONVERSION = 2.54;
```

```
const int NO_OF_STUDENTS = 20;
```

```
const char BLANK = ' ';
```

Variable: A memory location whose content may change during program execution. The syntax for declaring one variable or multiple variables is:

dataType identifier, identifier, . . . ;

Consider the following statements:

double amountDue;

int counter;

char ch;

int x, y;

string name;

Example 2:

#include <iostream>

using namespace std;

int main()

{

int feet;

int inches;

cout << "Enter two integers separated by spaces: ";

cin >> feet >> inches;

cout << endl;


```
cout << "Feet = " << feet << endl;  
  
cout << "Inches = " << inches << endl;  
  
return 0;  
  
}
```

Reviewing the Basics

- What are the modules of C++ programs called?
- What does the following preprocessor directive do?

```
#include <iostream>
```

- What does the following statement do?

```
using namespace std;
```

- Suppose your main() function has the following line:

```
cout << "Please enter your PIN: ";
```

And suppose the compiler complains that cout is an unknown identifier. What is the likely cause of this complaint, and what are three ways to fix the problem?

Programming Exercises

- Write a C++ program that displays your name and address.
- Write a C++ program that asks for a distance in furlongs and converts it to yards. (One furlong is 220 yards.)
- Write a program that asks the user to enter his or her age. The program then should display the age in months:

Enter your age: 29

Your age in months is 384.

- The following program has syntax mistakes. Correct them. On each successive line, assume that any preceding error has been corrected.

```
#include <iostream>

const int SECRET_NUM = 11,213;

const PAY_RATE = 18.35

main()

{

int one, two;

double first, second;

one = 18;

two = 11;

first = 25;

second = first * three;

second = 2 * SECRET_NUM;

SECRET_NUM = SECRET_NUM + 3;

cout << first << " " << second << SECRET_NUM << endl;

paycheck = hoursWorked * PAY_RATE

cout << "Wages = " << paycheck << endl;

return 0; }
```

- Suppose a, b, and sum are int variables and c is a double variable.
What value is assigned to each variable after each statement executes? Suppose a = 3, b = 5, and c = 14.1.

	<u>a</u>	<u>b</u>	<u>c</u>	<u>sum</u>
sum = a + b + c;	_____	_____	_____	_____
c /= a;	_____	_____	_____	_____
b += c - a;	_____	_____	_____	_____
a *= 2 * b + c;	_____	_____	_____	_____

- What is printed by the following program? Suppose the input is: 2015

```
#include <iostream>
```

```
using namespace std;
```

```
const int NUM = 10;
```

```
const double X = 20.5;
```

```
int main()
```

```
{
```

```
int a, b;
```

```
double z;
```

```
char grade;
```

```
a = 25;
```

```
cout << "a = " << a << endl;
```

```
cout << "Enter two integers: ";

cin >> a >> b;

cout << endl;

cout << "The numbers you entered are "

<< a << " and " << b << endl;

z = X + 2 * a - b;

cout << "z = " << z << endl;

grade = 'A';

cout << "Your grade is " << grade << endl;

a = 2 * NUM + z;

cout << "The value of a = " << a << endl;

return 0;

}
```

- Write a program that produces the following output:

```
*****

*      Programming Assignment 1      *

*      Computer Programming I        *

*      Author: your name             *

*      Due Date: Wednesday, Jan. 7   *

*****
```

- Write a program that prompts the user to enter five test scores and then prints the average test score. (Assume that the test scores are decimal numbers.)

References:

- [1] Deitel, P. & Deitel, R., "C++ How to Program", Eighth Edition, Pearson Education Inc., 2012.
- [2] Stefan B., " C++ Primer Plus ", Sixth Edition, Pearson Education Inc., 2012.