Computer I

Lecture No. 4

C++ Types ,Variables and Functions

Al-Mustaqbal University College of Engineering & Technology
Biomedical Engineering Department
MSc. in Computer Engineering: Hamza Waleed Hamza

C++ Built-in Types

In C++, a program can use various **data types** to represent and work with different kinds of information, such as:

1) **Characters** (char)
2) **Integers** (int)
3) **Floating-point numbers** (float, double)
4) **Boolean values** (bool), etc.

Why are data types important?

Because a computer processes and stores data in **different ways**, the data type of each variable must be known. The **data type** determines:

1. **How the data is represented internally** (e.g., as binary numbers, IEEE format for floats).
2. **How much memory is allocated** for that data (e.g., int usually takes 4 bytes, char takes 1 byte).

This applies to both **data types** and **variables** when you declare a variable, you specify its type so the compiler knows how to handle it.

Examples of basic data types

Type	Size	Example	Range / Accuracy
Bool	1 byte	true, false	Logical (0 or 1)
Char	1 byte	'a', '5'	–128 to 127 (ASCII)
Short	2 bytes	-100, 300	–32,768 to 32,767
Int	4 bytes	1000, -500	–2,147,483,648 to 2,147,483,647
Long	4–8 bytes	8000000000	Depends on system (32/64-bit)
Float	4 bytes	3.14f	±3.4E+38, ~6–7 digits accuracy
Double	8 bytes	3.1415926535	±1.7E+308, ~15–16 digits accuracy

Modifiers: void, signed, and unsigned

- **void**: The void type is used when there is no value or no return value, such as in functions that don't return anything.

- **signed**: Indicates that a variable can hold both positive and negative values.

- **unsigned**: Indicates that a variable can only hold positive values (no negative values).

- **const**: The const keyword is used to create a "read-only" variable, which means its value cannot be modified later.

char and string constants

A character constant is a single character enclosed in single quotes. Characters in C++ are represented using the ASCII (American Standard Code for Information Interchange) values, which are numerical representations for characters.

For example:

'A' = 65 in ASCII

'a' = 97 in ASCII

' ' (space) = 32 in ASCII

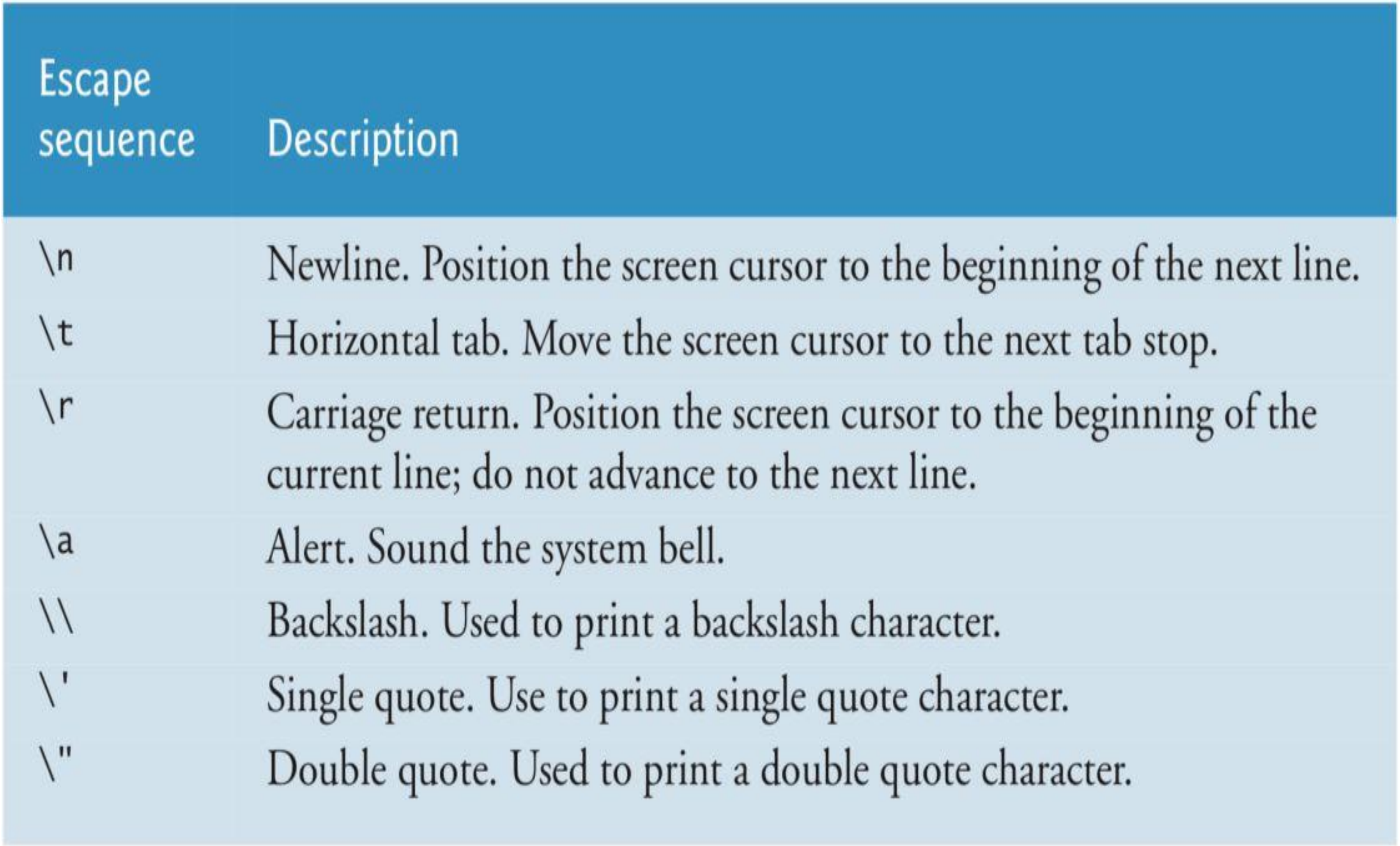
'\0' = 0 (null terminator, marks the end of a string)

A string constant is a sequence of characters enclosed in double quotes. Unlike a character constant, which is just one character, a string constant is a full sequence of characters (including spaces) followed by a special null character '\0' at the end.

For example:

"Hello!" is a string constant that consists of the characters H, e, l, l, o, !, and ends with the \0 character.

Escape Sequences



Escape sequence	Description
\n	Newline. Position the screen cursor to the beginning of the next line.
\t	Horizontal tab. Move the screen cursor to the next tab stop.
\r	Carriage return. Position the screen cursor to the beginning of the current line; do not advance to the next line.
\a	Alert. Sound the system bell.
\\	Backslash. Used to print a backslash character.
\'	Single quote. Use to print a single quote character.
\"	Double quote. Used to print a double quote character.

Defining &Initializing of Variables

❑ Defining Variable

❖ A variable must be defined before you can use it in a program.

❖ When you define a variable the type is specified and an appropriate amount of memory reserved.

SYNTAX: type name1 [,name2 ...];

EXAMPLES: char c; int i, counter; double x, y, size;

❑ Initializing a Variable

❖ A variable can be initialized, i.e. a value can be assigned to the variable, during its definition.

❖ You can assign a value to a variable after defining it in one of these two ways:

- ✓ an equals sign (=) and an initial value for the variable.
- ✓ round brackets containing the value of the variable.

EXAMPLES: char c = 'a’; float x(1.875);

Global and Local Variables

- A variable defined **outside** of each function is global, i.e., it can be used by all functions.
- A variable defined **within** a function is local, i.e., it can be used only in that function.
- If you do not initialize a global variable, it defaults to 0.
- If you do not initialize a local variable, its value will be random (undefined).

A Sample Program : Circumference and area of a circle with radius 2.5

```
#include <iostream>
using namespace std;
const double pi = 3.141593;
int main() {
    double area, circuit, radius = 1.5;
    area = pi * radius * radius;
    circuit = 2 * pi * radius;
    cout << "\nTo Evaluate a Circle\n" << endl;
    cout << "Radius: " << radius
         << " Circumference: " << circuit
         << " Area: " << area;
    return 0;
}
```

Constant Objects

Output is :

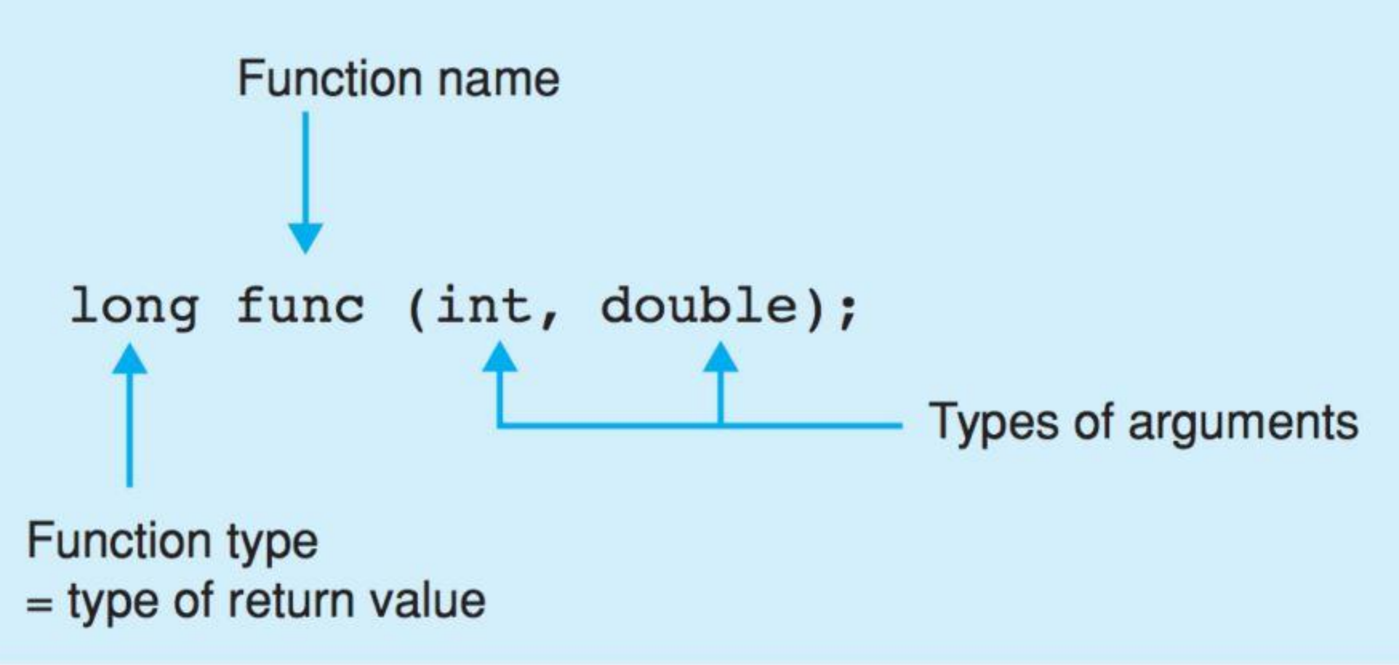
```
To Evaluate a Circle
Radius: 1.5
Circumference: 9.42478
Area: 7.06858
```

Constant Objects: The const keyword is used to create a "read only" variable.

- This means that it cannot be modified at a later stage
- A constant variable must be initialized during its definition.

EXAMPLE: const double pi = 3.1415947;

Declaring Functions



Function name

```
long func (int, double);
```

Types of arguments

Function type
= type of return value

In programming, the function declaration provides the compiler with key information:

❑ **Function name:** The identifier used to call the function (e.g., func).

❑ **Arguments:** These are the values or data types the function will receive (e.g., the first argument is of type int, and the second is of type double).

❑ **Return Type:** The type of value the function returns, such as long.

Example: **double pow(double base, double exponent);**

Mathematical Standard Functions

The following are some standard mathematical functions available in many programming languages:

- double sin(double); // Sine
- double cos(double); // Cosine
- double tan(double); // Tangent
- double atan(double); // Arc tangent
- double cosh(double); // Hyperbolic Cosine
- double sqrt(double); // Square Root
- double pow(double, double); // Power
- double exp(double); // Exponential Function
- double log(double); // Natural Logarithm
- double log10(double); // Base-ten Logarithm

Example: Calculating Powers with the Standard Function pow()

To use the pow() function, you need to include the required header files (<cmath>), as it is part of the math library.

```
#include <iostream>
#include <cmath>
using namespace std;
int main() {
    double x = 2.5, y;
    y = pow(x, 3.0); // Calculates 2.5 raised to the power of 3
    cout << "2.5 raised to the power of 3 yields: " << y << endl;
    cout << "2 + (5 raised to the power of 2.5) yields: " << 2.0 + pow(5.0, x) << endl;
    return 0;
}
```

Output is :

```
2.5 raised to the power of 3 yields: 15.625
2 + (5 raised to the power of 2.5) yields: 57.9017
```

Functions without Return Value

Functions without Return :

- ❑ You can write functions that perform a certain action but do not return a value to the function that called them.
- ❑ The type void is available for functions of this type, which are also referred to as procedures in other programming languages.

Example: void srand(unsigned int seed);

- ❑ The standard function srand() initializes an algorithm that generates random numbers.
- ❑ Since the function does not return a value, it is of type void.
- ❑ An unsigned value is passed to the function as an argument to seed the random number generator. The value is used to create a series of random numbers.

Functions without Arguments

- If a function does not expect an argument, the function prototype must be declared as void.
- Or the braces following the function name must be left empty.

Example:

```
int rand( void );
int rand();
```

- The standard function rand() is called without any arguments and returns a random number between 0 and 32767.
- A series of random numbers can be generated by repeating the function call.

Example: Generating Random Numbers

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main() {
    unsigned int seed;
    int z1, z2, z3;
    cout << " --- Random Numbers --- \n";
    cout << "To initialize the random number generator, please enter an integer value: ";
    cin >> seed;
    srand(seed); // Initializes the random number generator with the entered seed
    z1 = rand(); // Generates random numbers
    z2 = rand();
    z3 = rand();
    cout << "\nThree random numbers: " << z1 << " " << z2 << " " << z3 << endl;
    return 0;
}
```

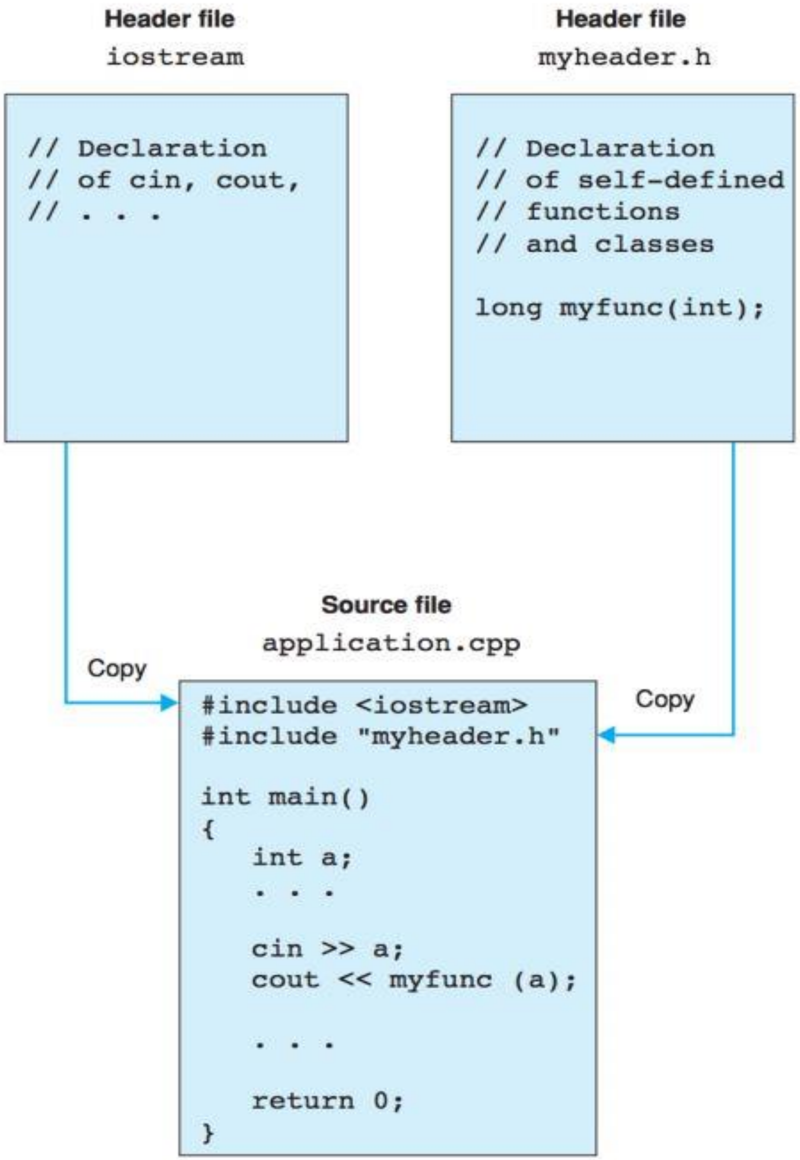
Output is :

```
--- Random Numbers ---
To initialize the random number generator, please enter an integer value: 7777
Three random numbers: 25435 6908 14579
```

Header Files

Header files contain declarations and macros that are necessary for the proper functioning of the program. They are included at the start of the program using the #include directive.

- You can only include one header file per #include directive.
- The file name can be enclosed in either angled brackets < > or double quotes " ".



```
// Declaration
// of cin, cout,
// . . .
```

```
// Declaration
// of self-defined
// functions
// and classes

long myfunc(int);
```

```
#include <iostream>
#include "myheader.h"

int main()
{
   int a;
   . . .

   cin >> a;
   cout << myfunc (a);

   . . .

   return 0;
}
```

Using Strings in C++

To use strings in C++, you need to include the <string> header file. The string class allows you to work with text data.

Example:

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string prompt("What is your name: "), name, line(40, '-'), total = "Hello ";
    cout << prompt;
    cin >> name;
    total = total + name;
    cout << line << endl << total << endl;
    cout << line << endl;
    return 0;
}
```

Home work

Homework 1 : Write a C++ program that defines two variables for floating-point numbers and initializes them with the values 123.456 and 76.543. Then display the sum and the difference of these two numbers on screen.

Homework 2 : Write a program to calculate the square roots of the numbers 4, 12.25, and 0.0121 using the sqrt() function from the cmath library.

The function prototype is:

```
double sqrt(double x);
```

The return value of sqrt() is the square root of x.