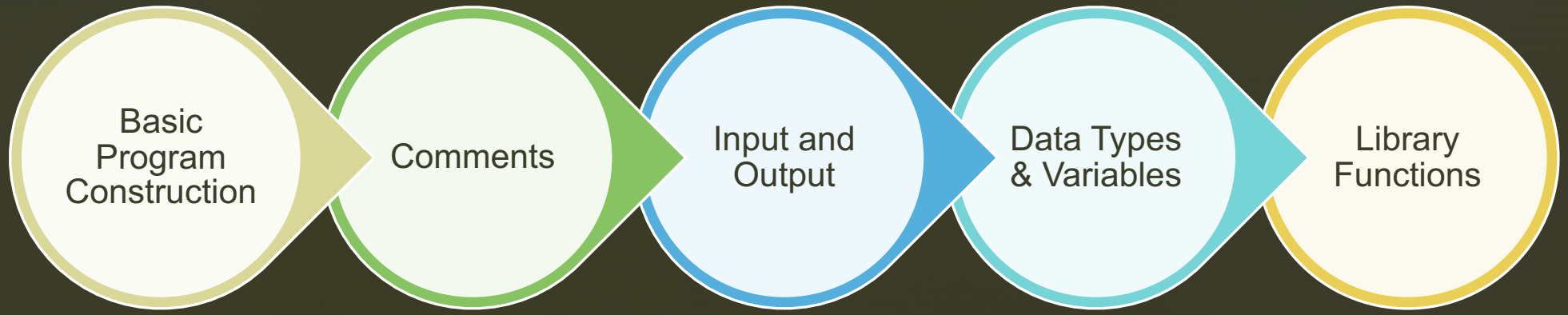
DR. (MRS) T-S.M.A. ADJAIDOO

01

C++ Programming Basics

Today's Lesson



Learning Outcomes

1

- Introduce learners to C++

2

- Equip learners with basic knowledge of C++ syntax

C++ Programming Basics

Basic Program Construction

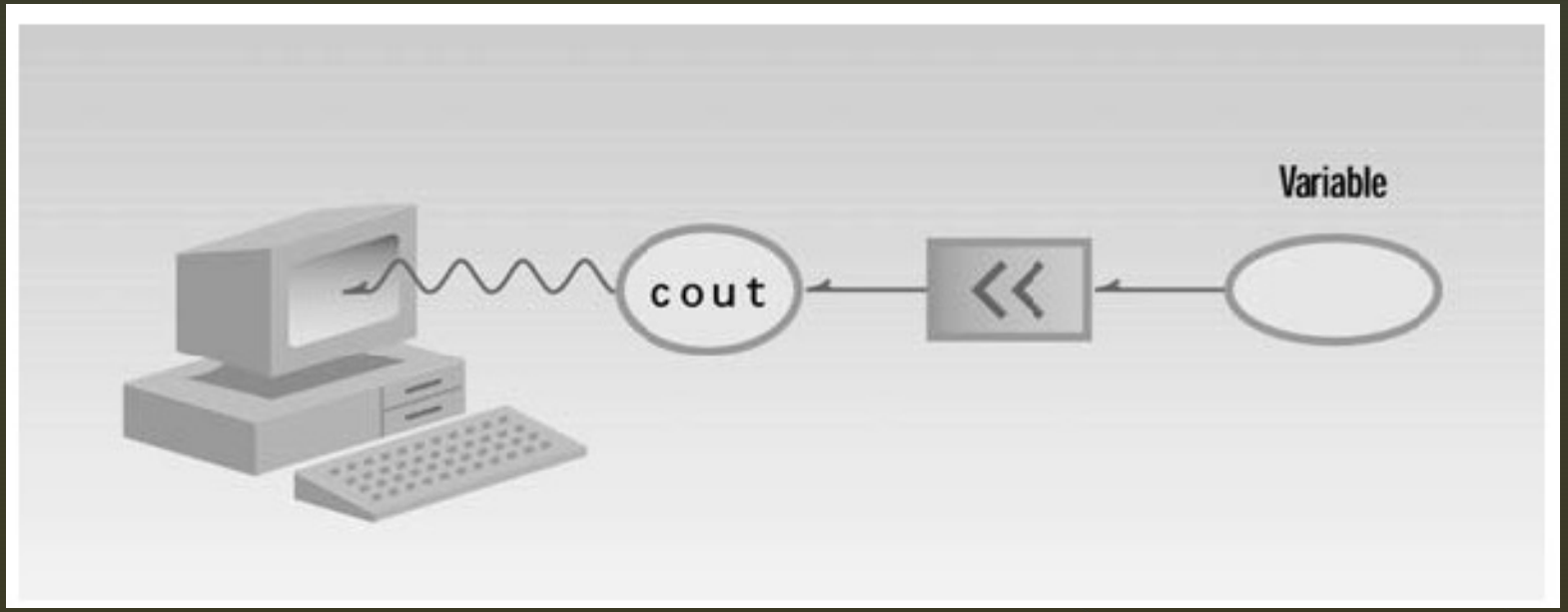
- This program is called helloworld, so its source file is helloworld.cpp. It simply prints a sentence on the screen.
- Despite its small size, this program demonstrates a great deal about the construction of C++ programs.

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

int main()
{
    cout << "Hello World\n";
    return 0;
}
```

Output Using cout

- The identifier cout (pronounced “C out”) is actually an object predefined in C++ to correspond to the standard output stream.
- A stream is an abstraction that refers to a flow of data.
- The standard output stream normally flows to the screen display—although it can be redirected to other output devices.
- The operator << is called the *insertion* or *put to* operator.
- It directs the contents of the variable on its right to the object on its left.

Output Using cout



The using Directive

- A C++ program can be divided into different *namespaces.*
- *A namespace is a part of the program* in which certain names are recognized; outside of the namespace they're unknown.
- The directive using namespace std; says that all the program statements that follow are within the std namespace.
- Various program components such as cout are declared within this namespace.

The using Directive

- If we didn’t use the using directive, we would need to add the std name to many program elements.
- For example, in the FIRST program we’d need to say

  std::cout << “Every age has a language of its own.”;
- To avoid adding std:: dozens of times in programs we use the using directive instead.

Comments

- Comments are an important part of any program.
- They help the person writing a program, and anyone else who must read the source file, understand what is going on.
- The compiler ignores comments, so they do not add to the file size or execution time of the executable program.

Comments

There are two ways to comment:

The line comment and the block comment

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

/* This is a  block comment
    in C++
*/

int main()
{
    cout << "Hello World\n";
    //This is a  line comment
    return 0;
}
```

Integer Variables

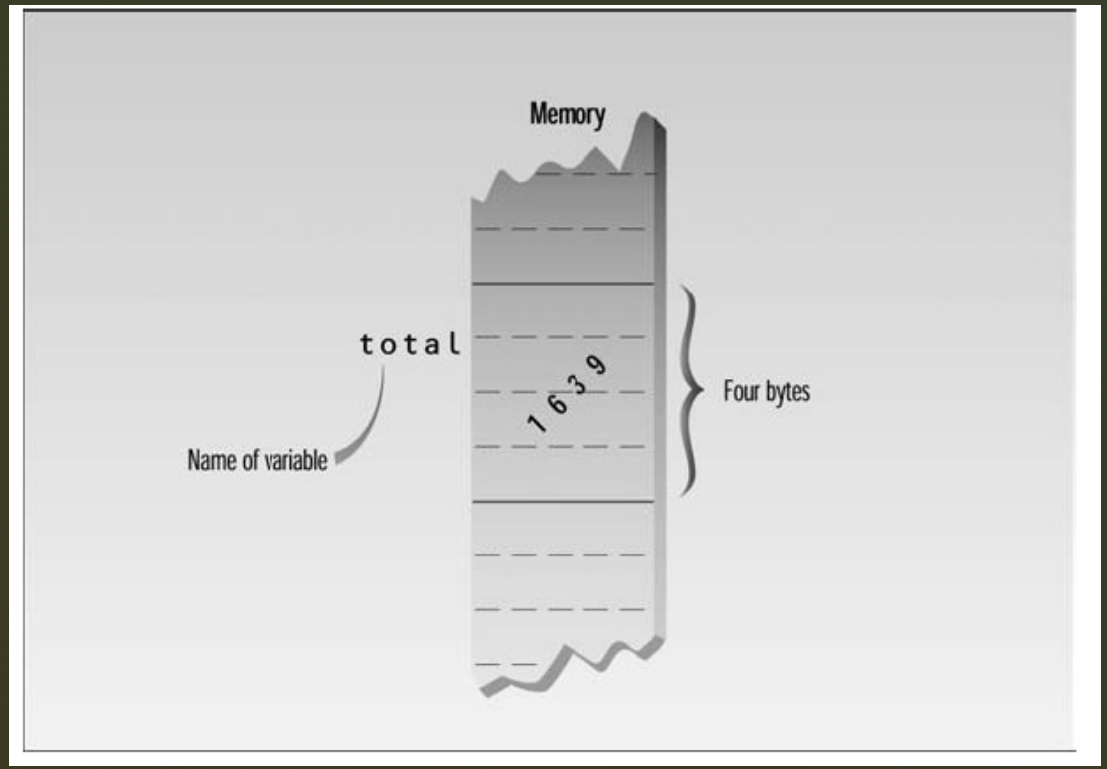
- Variables are the most fundamental part of any language.
- A variable has a symbolic name and can be given a variety of values.
- Variables are located in particular places in the computer's memory.
- When a variable is given a value, that value is actually placed in the memory space assigned to the variable.
- Integer variables represent integer numbers like 1, 30,000, and –27. Such numbers are used for counting discrete numbers of objects.

Integer Variables

- Integer variables exist in several sizes, but the most commonly used is type int.
- The amount of memory occupied by the integer types is system dependent.
- On a 32-bit system such as Windows, an int occupies 4 bytes (which is 32 bits) of memory.
- This allows an int to hold numbers in the range from –2,147,483,648 to 2,147,483,647.

Integer Variables

int total = 1639;



Integer Variables

Here's a program that defines and uses several variables of type int

```
// demonstrates integer variables
#include <iostream>
using namespace std;

int main()
{
    int var1; //define var1
    int var2; //define var2

    var1 = 20; //assign value to var1
    var2 = var1 + 10; //assign value to var2

    cout << "var1+10 is "; //output text
    cout << var2 << endl; //output value of var2
    return 0;
}
```

The endl Manipulator

- This causes a linefeed to be inserted into the stream, so that subsequent text is displayed on the next line.
- It has the same effect as sending the '\n' character.
- It's an example of a manipulator.
- Manipulators are instructions to the output stream that modify the input/output stream in various ways.
  - Eg: setbase, ws, ends, flush etc.

Character Variables

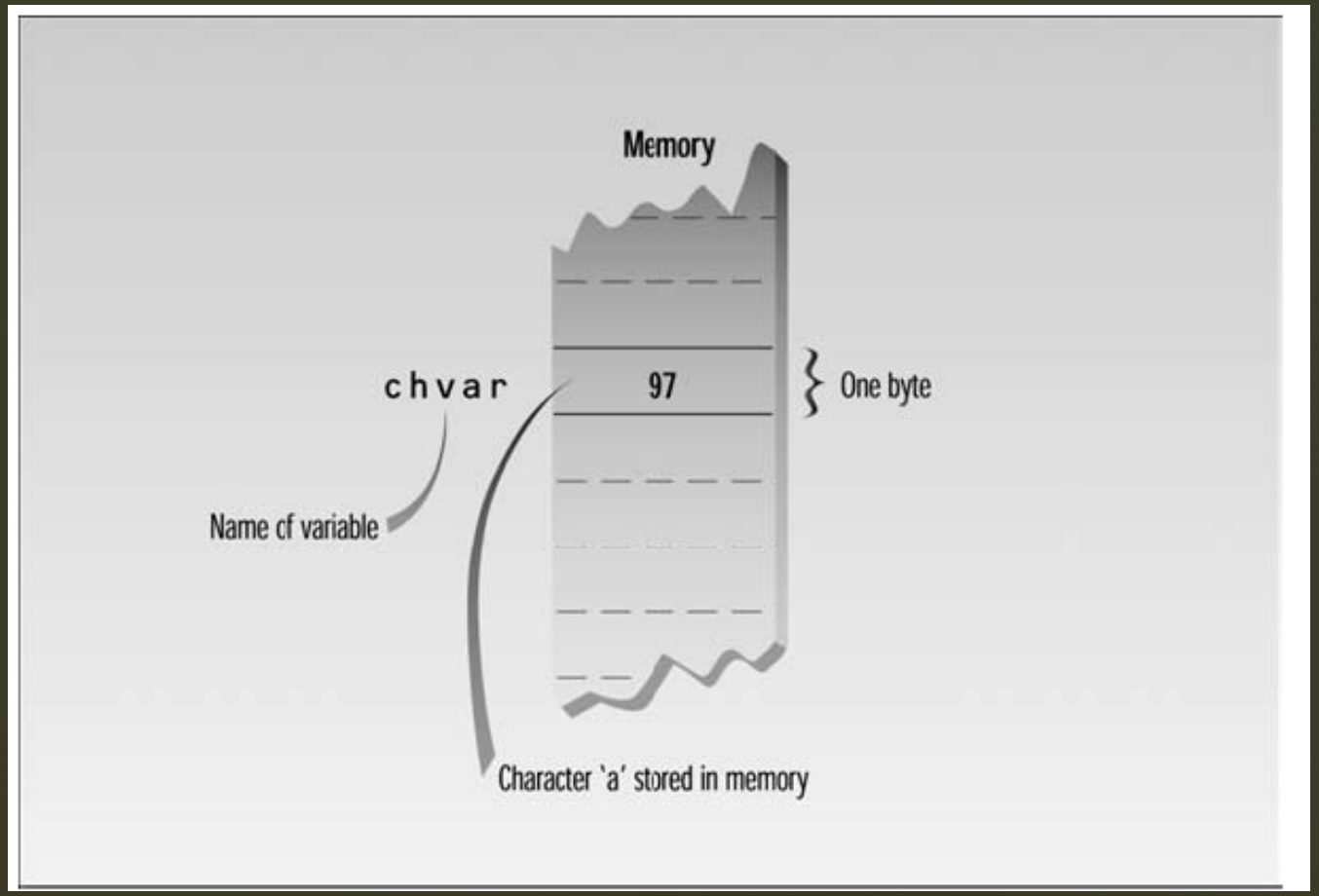
- Type char stores integers that range in value from –128 to 127.
- Variables of this type occupy only 1 byte (eight bits) of memory.
- Character variables are sometimes used to store numbers that confine themselves to this limited range, but they are much more commonly used to store ASCII characters.
- The ASCII character set is a way of representing characters such as ‘a’, ‘B’, ‘$’, ‘3’, and so on, as numbers. These numbers range from 0 to 127.

Character Constants

- Character constants use single quotation marks around a character, like ‘a’ and ‘b’.
- *When the C++ compiler* encounters such a character constant, it translates it into the corresponding ASCII code.
- The constant ‘a’ appearing in a program, for example, will be translated into 97, as shown.

Character Variables

char chvar = ‘a’;



Character Variables

Character variables can be assigned character constants as values.

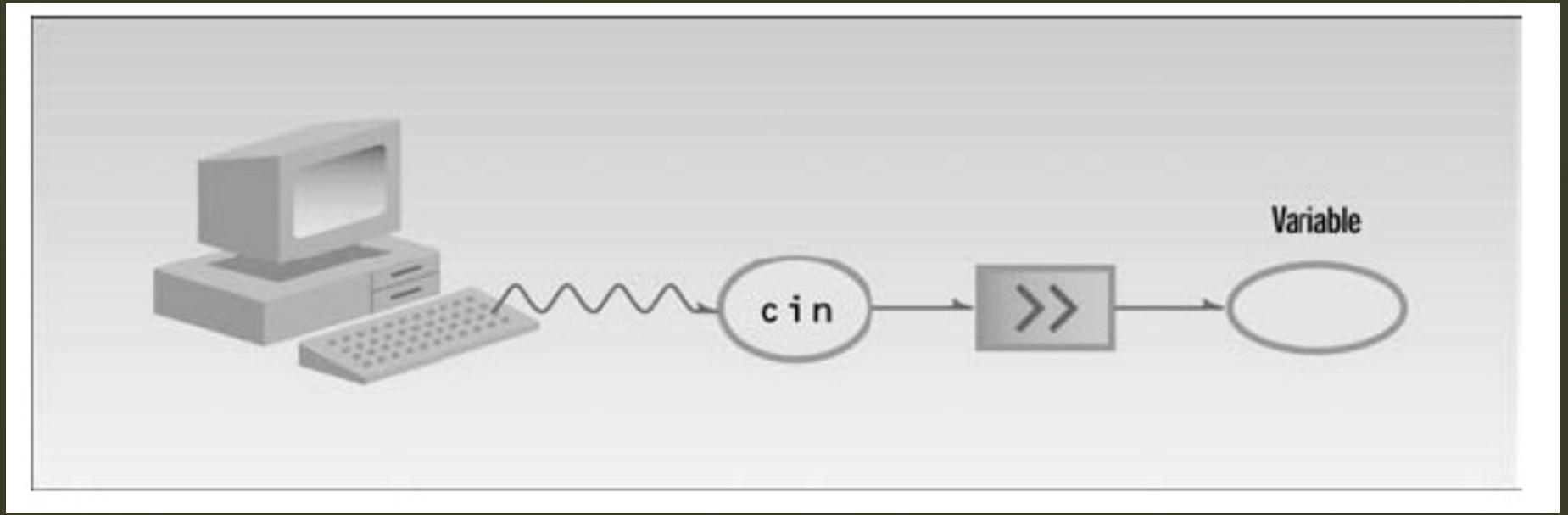
The following program shows some examples of character constants and variables.

```
// demonstrates character variables
#include <iostream> //for cout, etc.
using namespace std;

int main()
{
    char charvar1 = 'A'; //define char variable as character
    char charvar2 = '\t'; //define char variable as tab

    cout << charvar1; //display character
    cout << charvar2; //display character

    charvar1 = 'B'; //set char variable to char constant
    cout << charvar1; //display character

    cout << '\n'; //display newline character
    return 0;
}
```

Input with cin

- The keyword cin (pronounced “C in”) is an object, predefined in C++ to correspond to the standard input stream.
- This stream represents data coming from the keyboard (unless it has been redirected).
- The >> is the *extraction* or *get from* operator.
- *It takes the value* from the stream object on its left and places it in the variable on its right.

Input with cin



Input with cin

```
// demonstrates cin, newline
#include <iostream>
using namespace std;

int main()
{
    int ftemp; //for temperature in fahrenheit
    cout << "Enter temperature in fahrenheit: ";
    cin >> ftemp;
    int ctemp = (ftemp-32) * 5 / 9;
    cout << "Equivalent in Celsius is: " << ctemp << '\n';
    return 0;
}
```

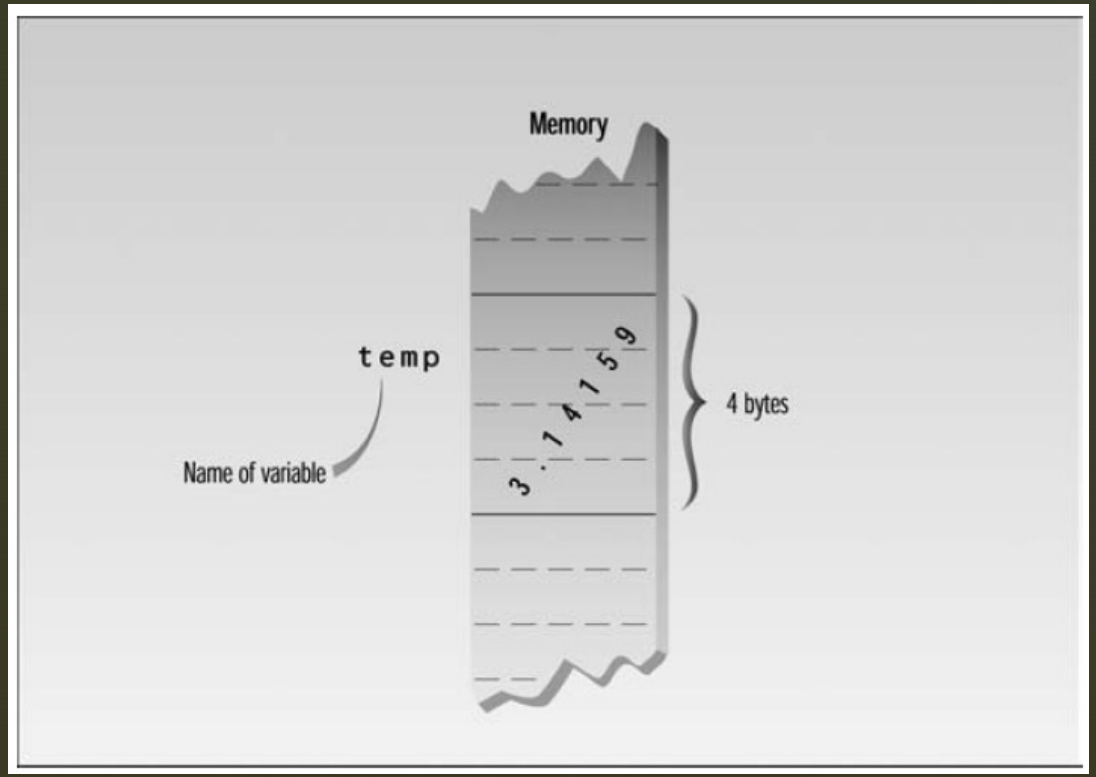
Floating Point Types

- Floating-point variables represent numbers with a decimal place—like 3.1415927, 0.0000625, and –10.2.
- They have both an integer part, to the left of the decimal point, and a fractional part, to the right.
- Floating-point variables represent what mathematicians call *real numbers, which* are used for measurable quantities such as distance, area, and temperature.
- There are three kinds of floating-point variables in C++:
  - type float, type double, and type long double.

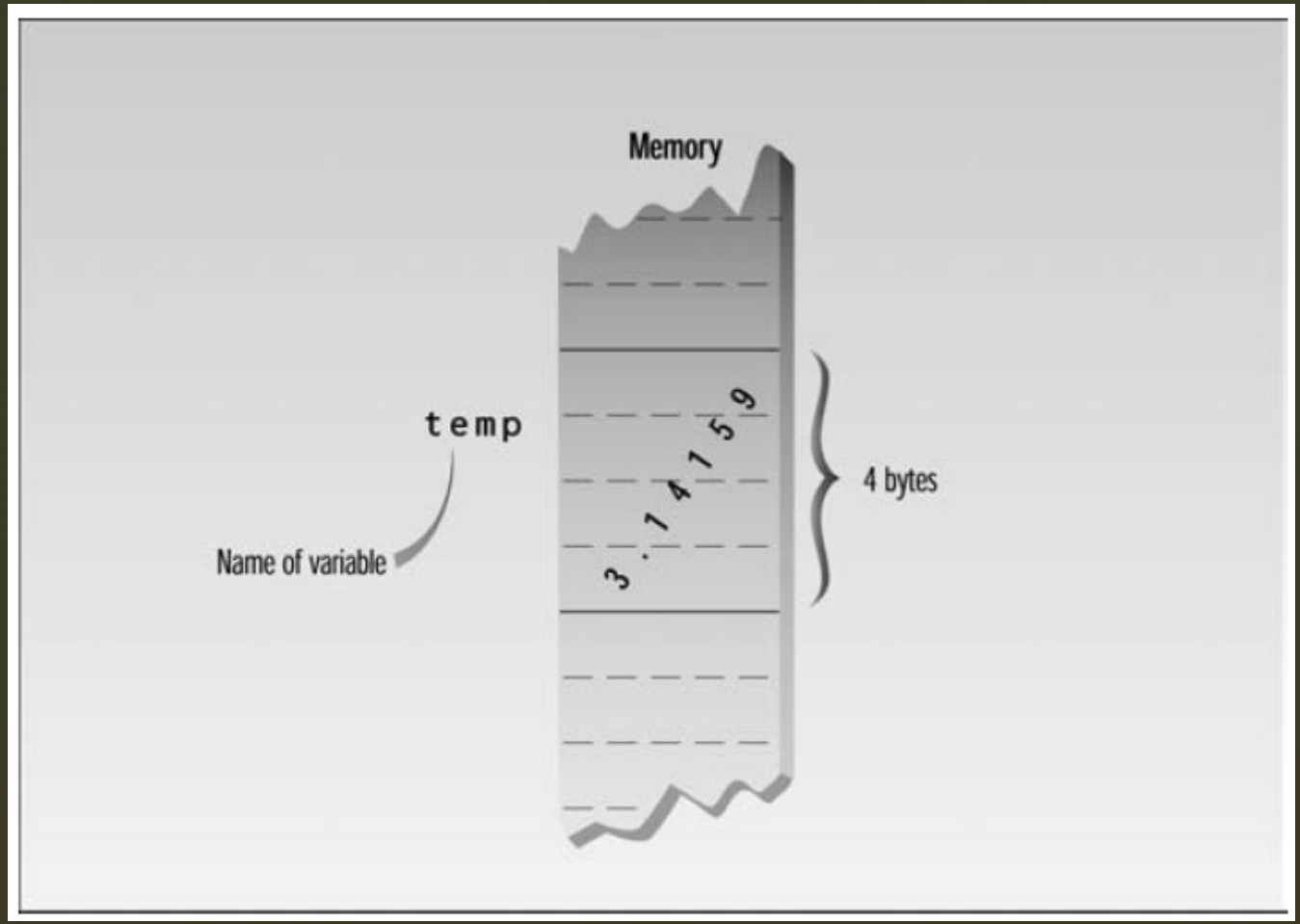
Floating Point Types

Float

- Type float stores numbers in the range of about 3.4×10^{-8} to 3.4×10^{8}, with a precision of seven digits.
- It occupies 4 bytes (32 bits) in memory.



Floating Point Types

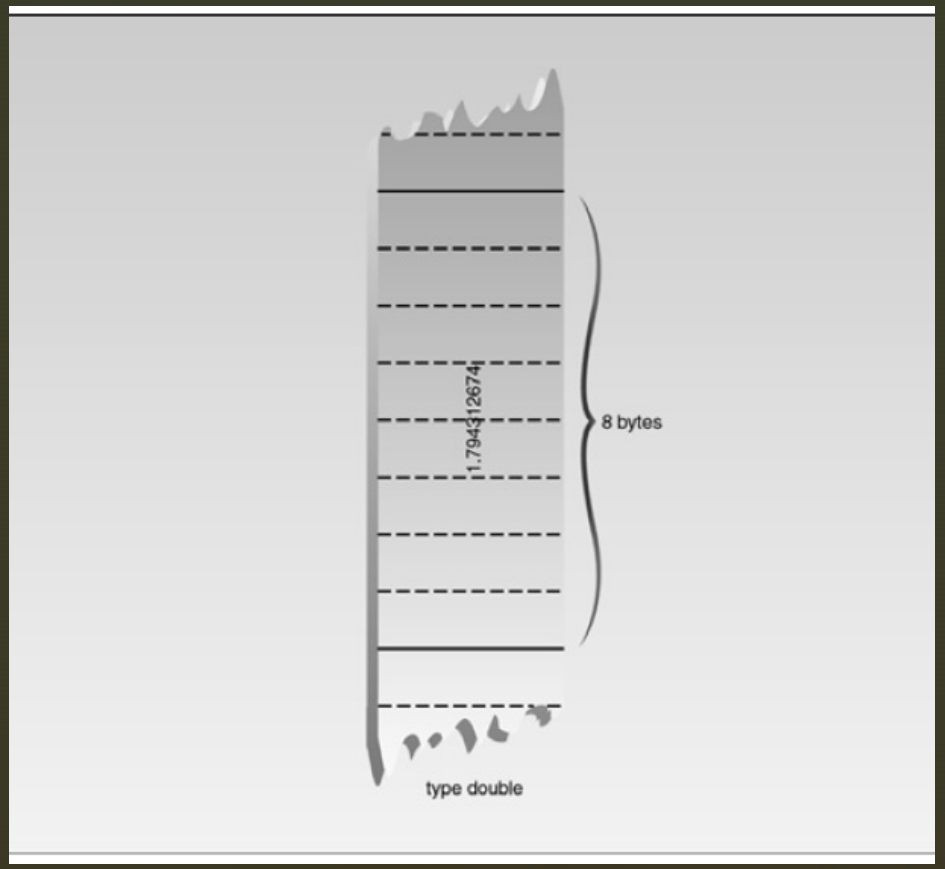


Floating Point Types

```
// demonstrates floating point variables
#include <iostream>  //for cout, etc.
using namespace std;

int main()
{
    float rad;  //variable of type float
    const float PI = 3.14159F;  //type const float

    cout << "Enter radius of circle: ";  //prompt
    cin >> rad; //get radius

    float area = PI * rad * rad;    //find area
    cout << "Area is " << area << endl;     //display answer
    return 0;
}
```

Floating Point Types

- The larger floating point types, double and long double, are similar to float except that they require more memory space and provide a wider range of values and more precision.
- Type double requires 8 with a precision of 15 digits.
- Type long double is compiler-dependent but is often the same as double.



The const Qualifier

- The keyword const (for constant) precedes the data type of a variable.
- It specifies that the value of a variable will not change throughout the program.
- Any attempt to alter the value of a variable defined with this qualifier will elicit an error message from the compiler.

Type bool

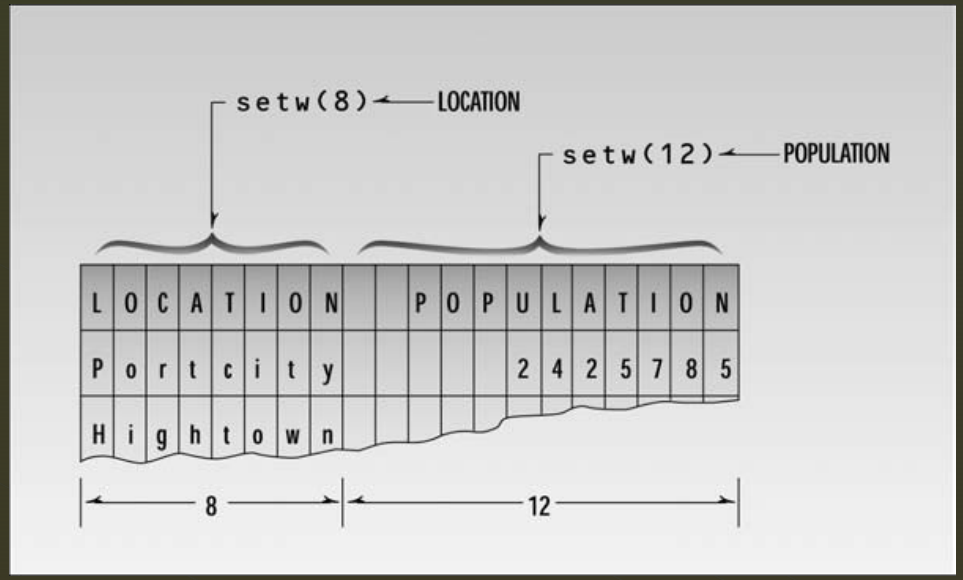
- Variables of type bool can have only two possible values:
  - true and false.
- In theory a bool type requires only one bit (not byte) of storage.
- In practice compilers often store them as bytes because a byte can be quickly accessed, while an individual bit must be extracted from a byte, which requires additional time.

The setw Manipulator

- setw changes the field width of output.
- You can think of each value displayed by cout as occupying a field: an imaginary box with a certain width.
- The default field is just wide enough to hold the value.
- However, in certain situations this may not lead to optimal results.
- The setw manipulator causes the number (or string) that follows it in the stream to be printed within a field n characters wide, where n is the argument to setw(n).
- The value is right justified within the field.

The setw Manipulator

```
// demonstrates setw manipulator
#include <iostream>
#include <iomanip> // for setw
using namespace std;

int main()
{
    long pop1 = 2425785, pop2 = 47, pop3 = 9761;

    cout << setw(8) << "LOCATION" << setw(12)
    << "POPULATION" << endl
    << setw(8) << "Portcity" << setw(12) << pop1 << endl
    << setw(8) << "Hightown" << setw(12) << pop2 << endl
    << setw(8) << "Lowville" << setw(12) << pop3 << endl;
    return 0;
}
```



Library Functions

- Many activities in C++ are carried out by library functions.
- These functions perform file access, mathematical computations, and data conversion, among other things.

Library Functions

```
// demonstrates sqrt() library function
#include <iostream> //for cout, etc.
#include <cmath> //for sqrt()
using namespace std;

int main()
{
    double number, answer; //sqrt() requires type double
    cout << "Enter a number: ";
    cin >> number; //get the number
    answer = sqrt(number); //find square root
    cout << "Square root is "<< answer << endl; //display it
    return 0;
}
```

Header Files

- As with cout and other such objects, you must #include a header file that contains the declaration of any library functions you use.
- In the documentation for the sqrt() function, you'll see that the specified header file is CMATH.
- In SQRT the preprocessor directive #include <cmath> takes care of incorporating this header file into our source file.
- If you don't include the appropriate header file when you use a library function, you'll get an error message like this from the compiler: *'sqrt' unidentified identifier.*

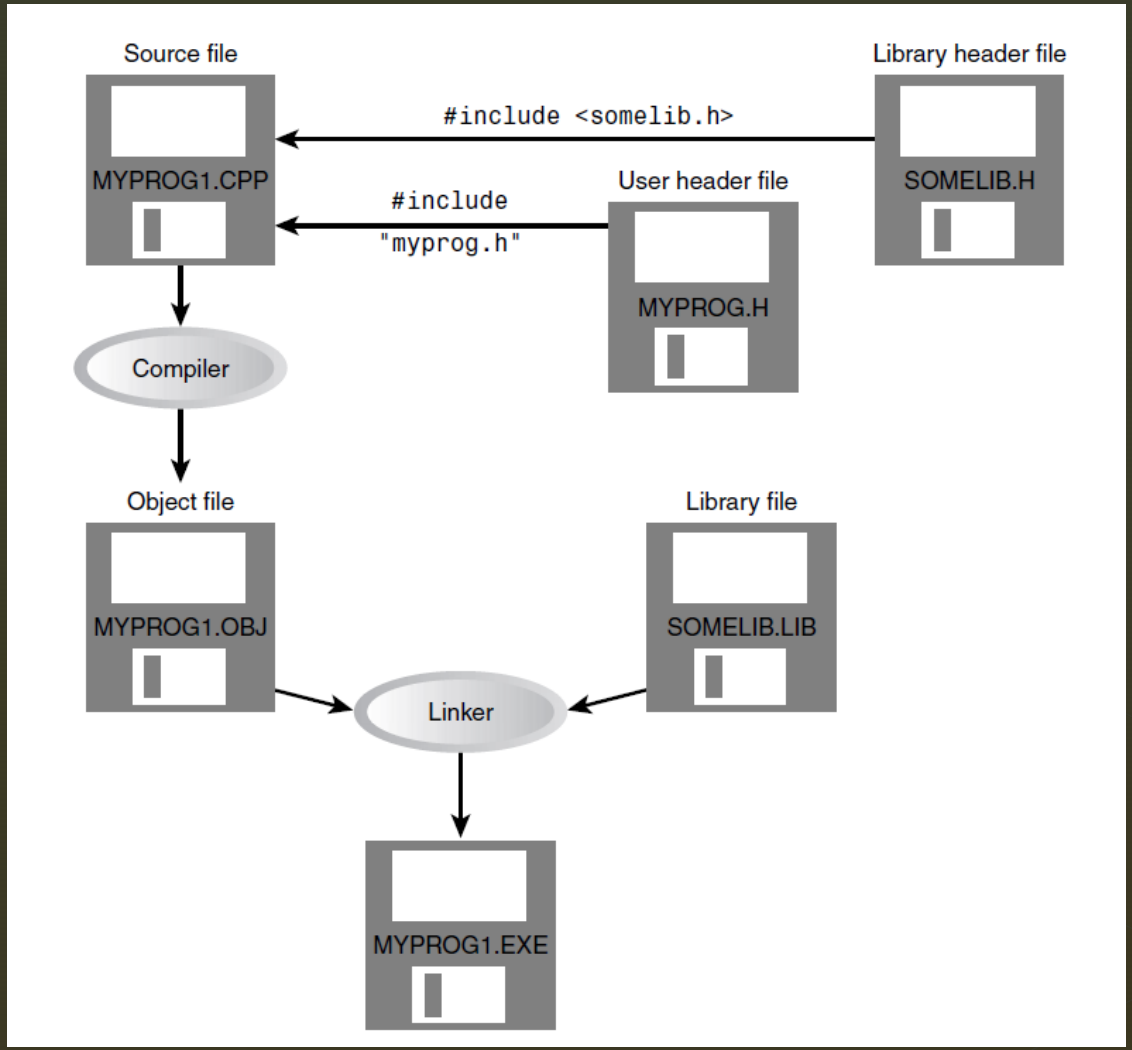
Library Files

- Various files containing library functions and objects link to your program to create an executable file.
- These files contain the actual machine-executable code for the functions.
- Such library files often have the extension .LIB.
- The sqrt() function is found in such a file. It is automatically extracted from the file by the linker, and the proper connections are made so that it can be called (that is, invoked or accessed) from the SQRT program.
- Your compiler takes care of all these details for you, so ordinarily you don’t need to worry about the process.

Header Files and Library Files

- To use a library function like sqrt(), you must link the library file that contains it to your program.
- The appropriate functions from the library file are then connected to your program by the linker.
- The functions in your source file need to know the names and types of the functions and other elements in the library file. They are given this information in a header file.
- Each header file contains information for a particular group of functions.
- The functions themselves are grouped together in a library file, but the information about them is scattered throughout a number of header files.

Header Files and Library Files



Two Ways to Use #include

You can use #include in two ways:

- The angle brackets < > indicate that the compiler should begin searching for files in the standard INCLUDE directory. This directory holds the header files supplied by the compiler manufacturer for the system.
- Instead of angle brackets around the filename, you can also use quotation marks, as in #include “myheader.h”
- Quotation marks instruct the compiler to begin its search for the header file in the current directory; this is usually the directory that contains the source file.
- You normally use quotation marks for header files you write yourself.
- Quotation marks or angle brackets work in any case, but making the appropriate choice speeds up the compilation process slightly by giving the compiler a hint about where to find the file.

Exercises

- Assuming there are 7.481 gallons in a cubic foot, write a program that asks the user to enter a number of gallons, and then displays the equivalent in cubic feet.
- Write a program that allows the user to enter a floating-point number representing degrees Celsius, and then displays the corresponding degrees Fahrenheit.
  - You can convert temperature from degrees Celsius to degrees Fahrenheit by multiplying by 9/5 and adding 32.

Any Questions?

The End

Contact: tsadjaidoo@knust.edu.gh

Office: Caesar Building, Room 413