# Programming (1)
# C++

Faculty of Computers and Information

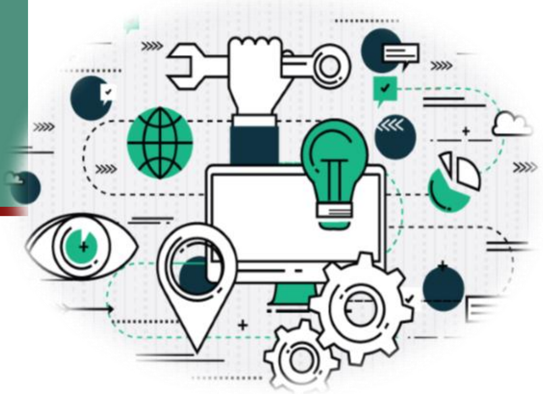
Fayoum University

2020-2021

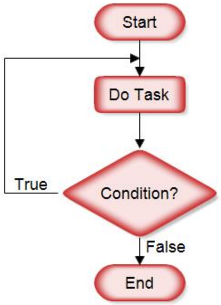
# Objectives

In this Lecture, you will:

- Become familiar with the basic components of a C++ program, including functions, special symbols, and identifiers
- Explore simple data types
- Learn how to use preprocessor directives and why they are necessary
- Explore how to properly structure a program, including using comments to document a program
- Learn how to write your first C++ program



## Design Algorithm

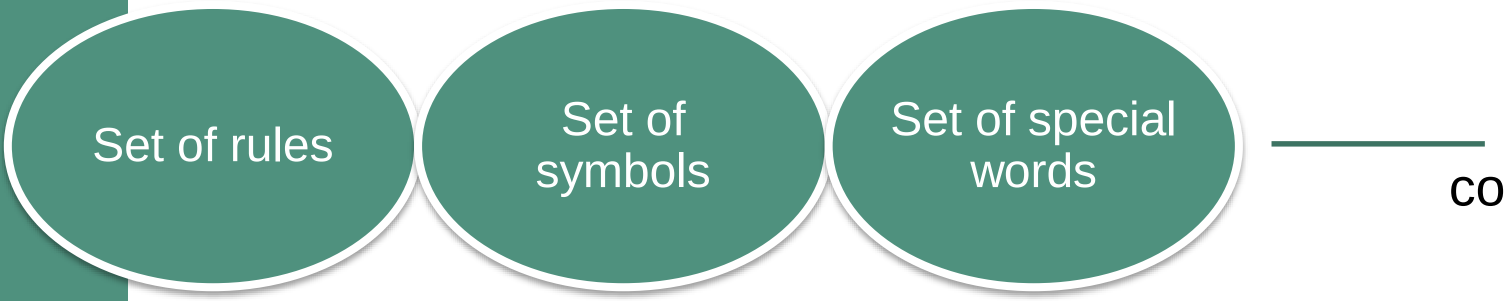


```
Follow route
Repeat
IF is the light green?
    Repeat
        Keep moving to landmark
    Until it reached the landmark
    IF is there a treasure at the landmark
        Display ‘treasure found’
        Collect Treasure
    Else
        Display ‘no treasure’
        Move to another landmark
Else
```

## Write Code

# Basic Definitions

<u>Programming language</u>



Set of rules

Set of symbols

Set of special words

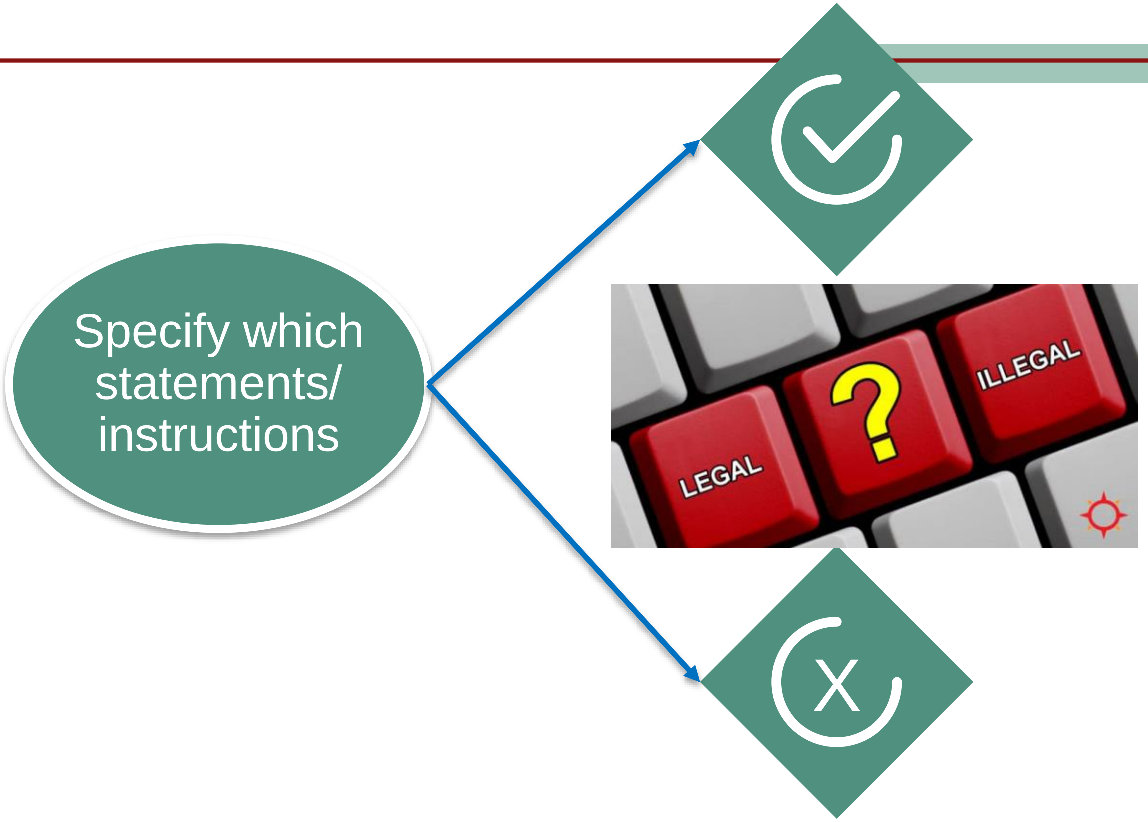
used to write computer programs

<u>Computer program</u>

Sequence of statements whose objective is to accomplish a task.

# Basic Definitions

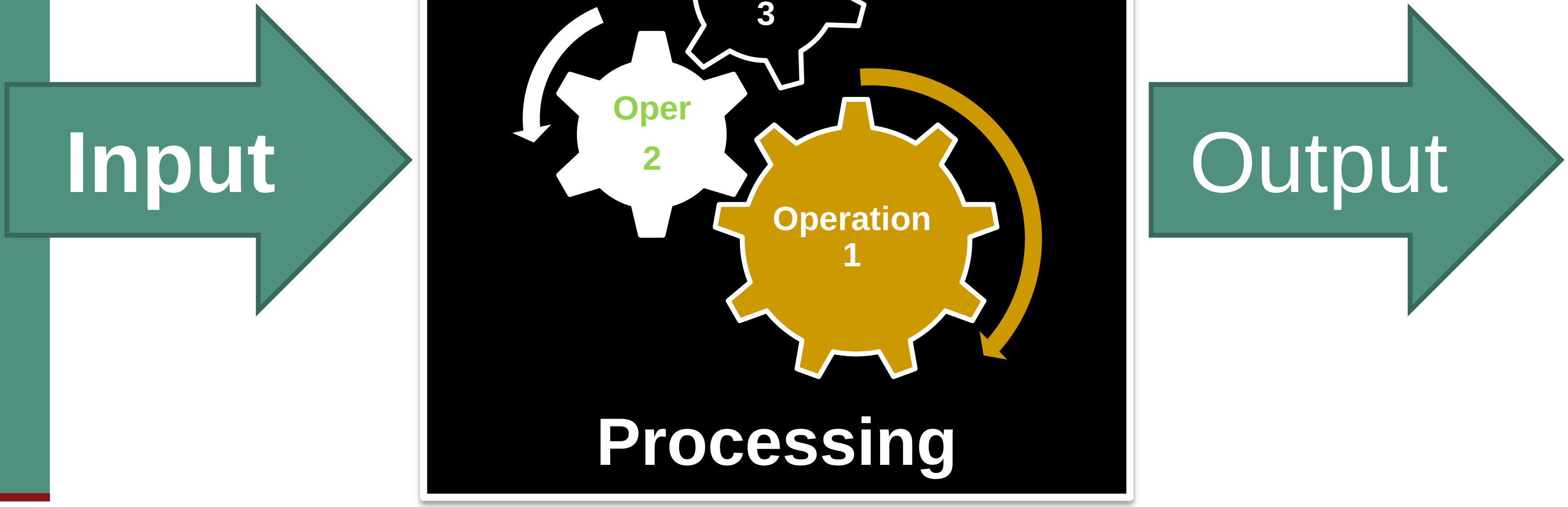
## **Syntax:**


Specify which statements/ instructions

## **Function:**

- Collection of statements; when executed, achieves a specific task.

# Computer System



Input

Oper 3

Oper 2

Operation 1

Processing

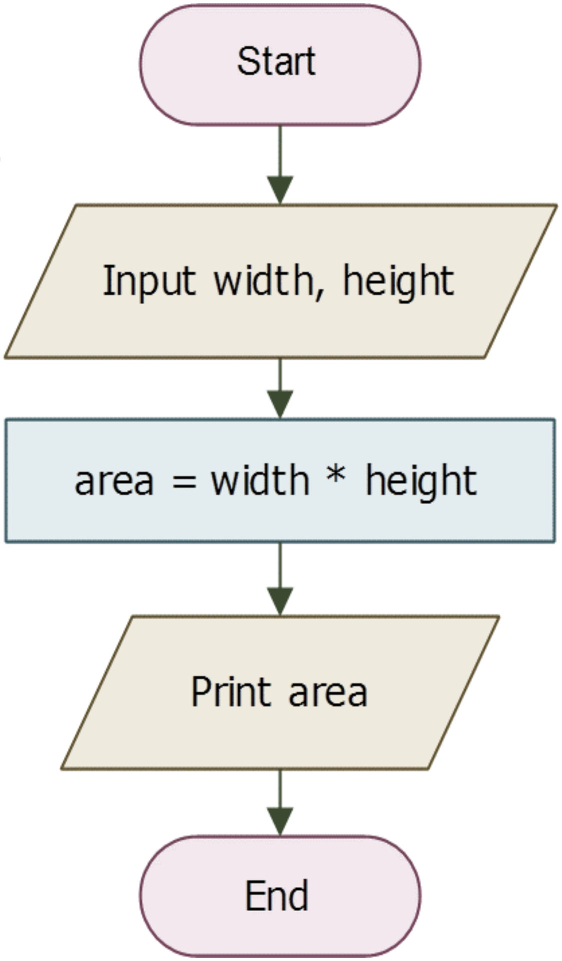
Output

# Example 1

**Write a program to find the Area of a rectangle**

The area of the Rectangle are given by t following formula:

**Area = Rectangle Length * Rectangle Width**



**Input :**

Rectangle Length , Rectangle Width.

**Processing :**

**Area** = L* W

**Output :**

Print Out The area.

# Example 2

**Write a program to compute the average of three numbers**

The average of three numbers is given by the following formulas:

**Average= (number1+number2+number3)/3**

**Input:**

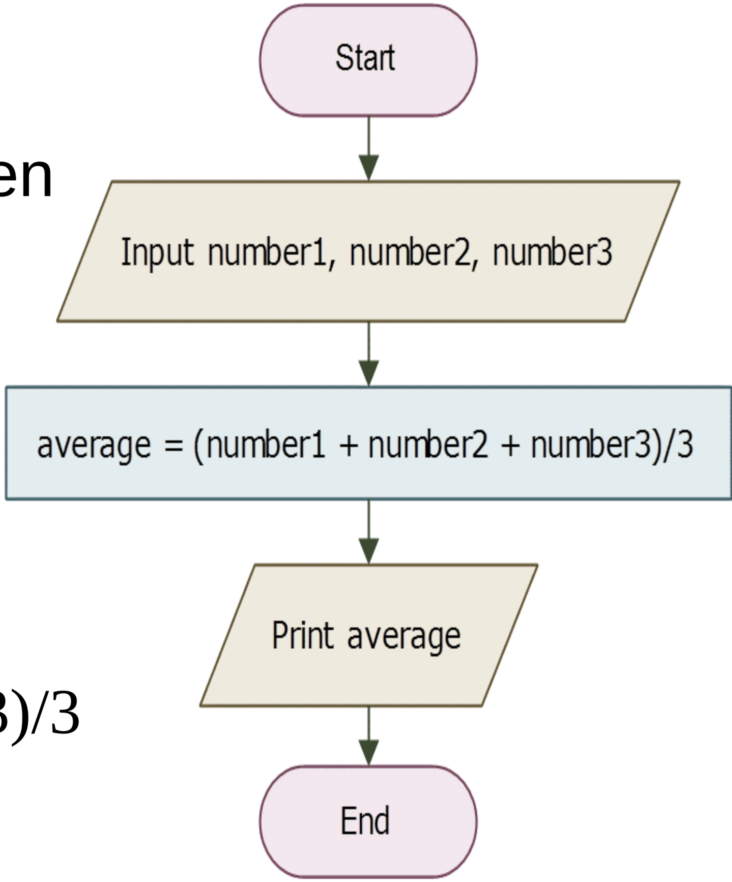
Number1, Number2, and Number3

**Processing:**

Average= (number1+number2+number3)/3

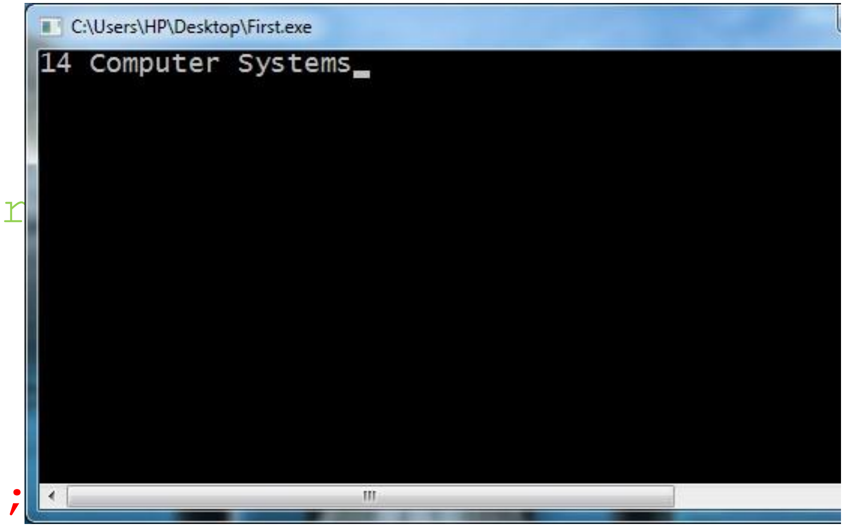
**Output:**

Print Out The Average


Start
Input number1, number2, number3
average = (number1 + number2 + number3)/3
Print average
End

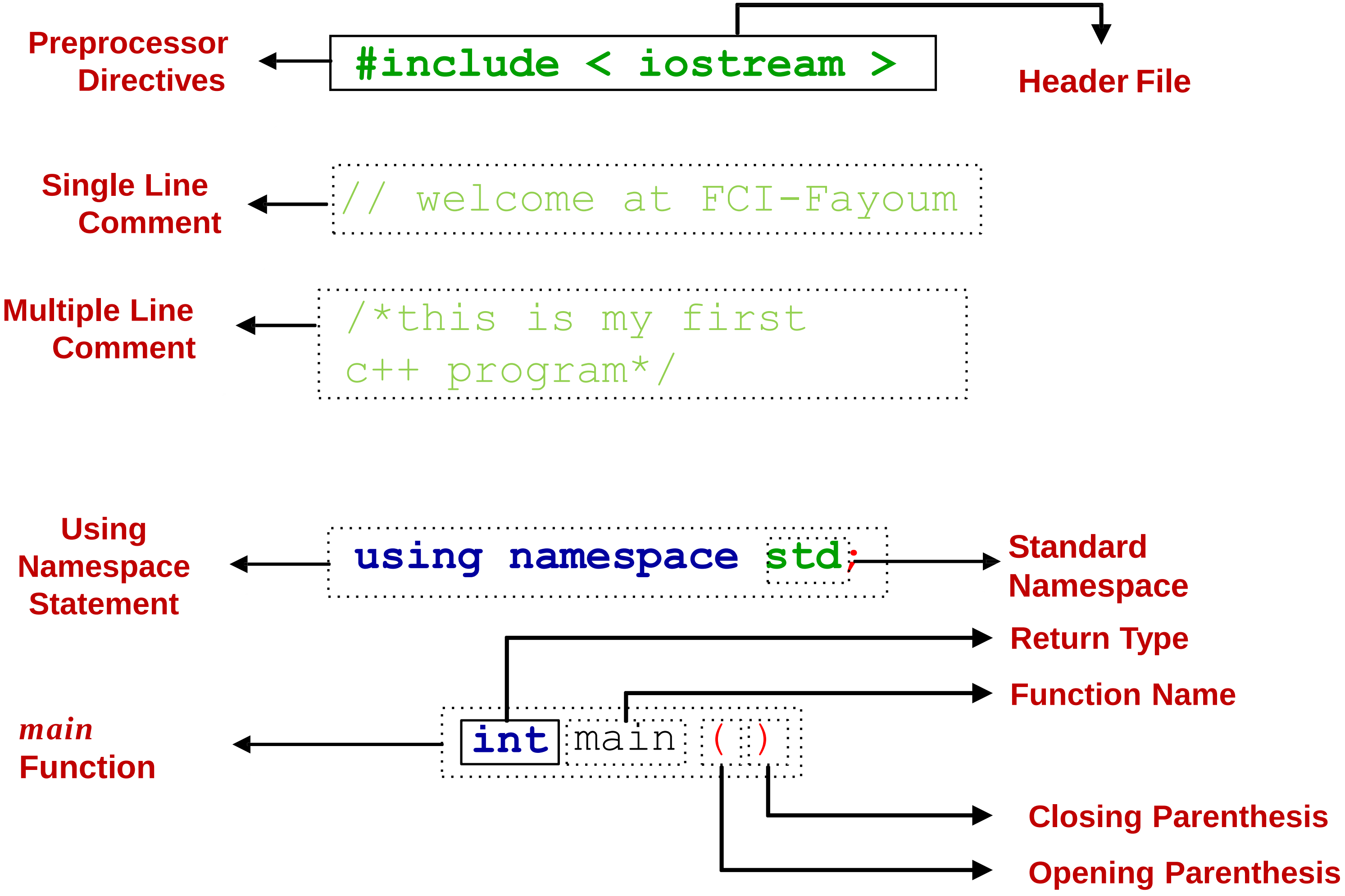
# Anatomy of C++ basic program

```
#include<iostream>

// welcome at FCI-Fayoum
/*this is my first c++ progr
using namespace std;

int    main()
{
cout<<"14 Computer Systems";
return 0;
}
```



```
14 Computer Systems_
```

# Anatomy of C++ basic program



Preprocessor Directives

```
#include < iostream >
```

Header File

Single Line Comment

```
// welcome at FCI-Fayoum
```

Multiple Line Comment

```
/*this is my first
c++ program*/
```

Using Namespace Statement

```
using namespace std;
```

Standard Namespace

*main* Function

```
int main ( )
```

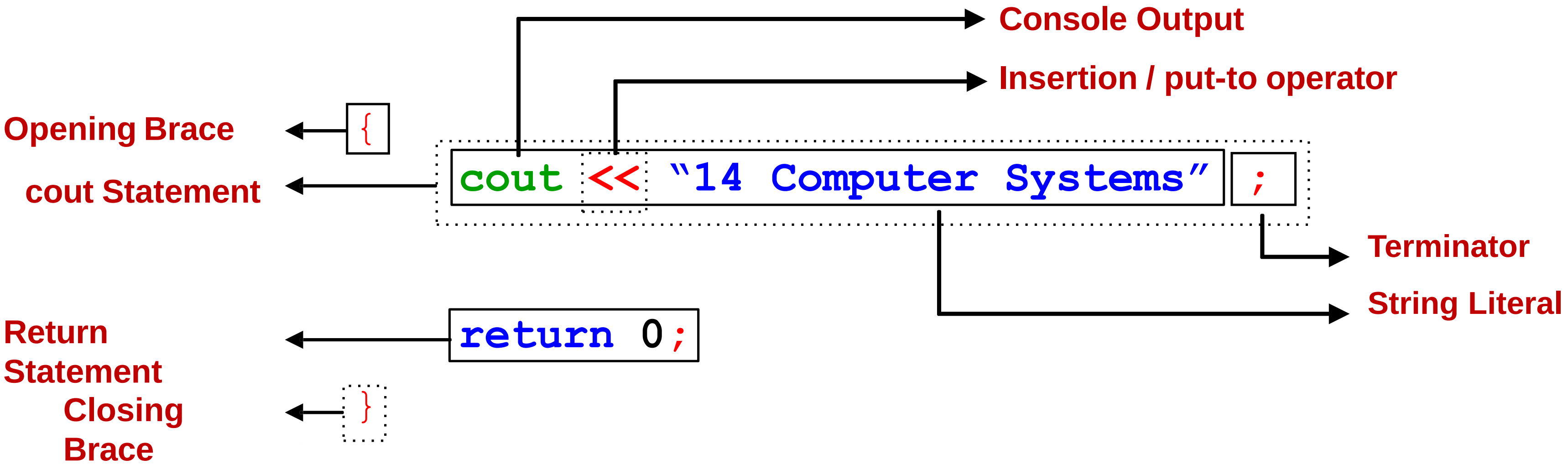
Return Type

Function Name

Closing Parenthesis

Opening Parenthesis

# Anatomy of C++ basic program



Console Output

Insertion / put-to operator

Opening Brace

```
{
```

cout Statement

```
cout << “14 Computer Systems”;
```

Terminator

String Literal

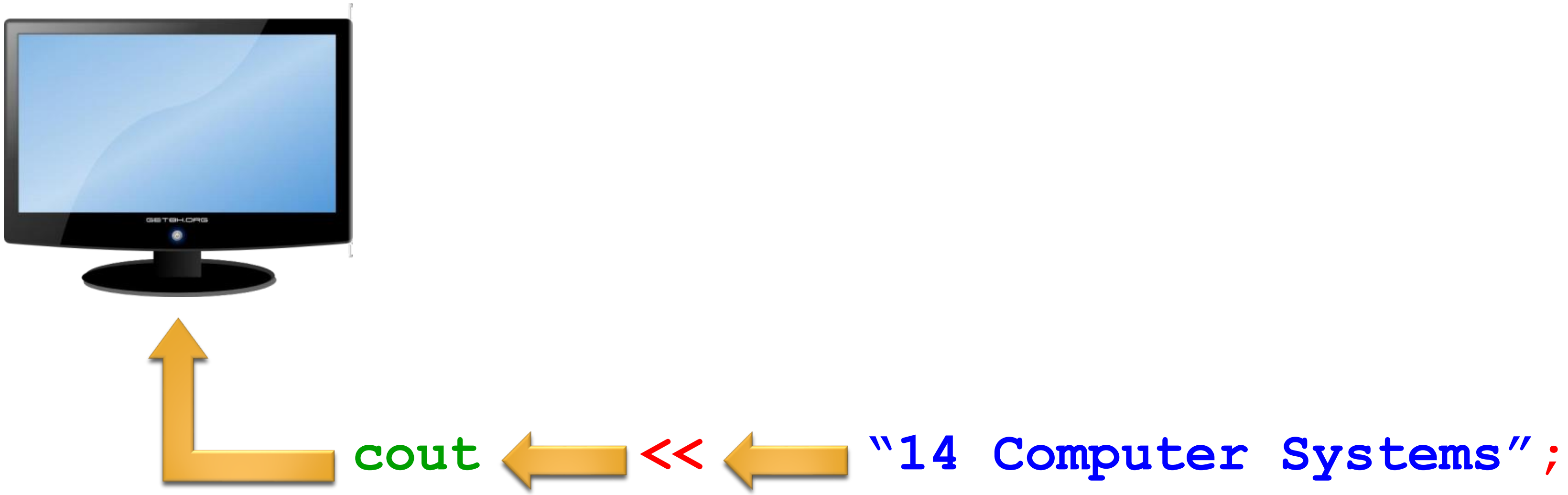
Return Statement

```
return 0;
```

Closing Brace

```
}
```

# Outputting with `cout`



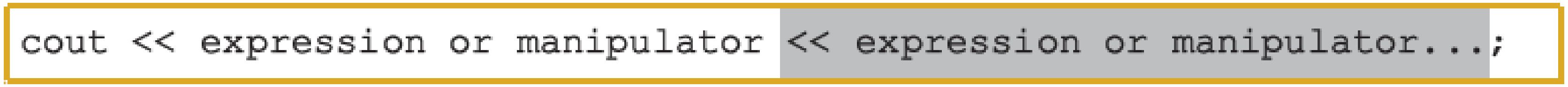
```
cout << “14 Computer Systems”;
```

# Outputting with `cout`

- In C++, cout statement is used to print/display the text or numbers on the monitor screen.
- Cout is the Standard Console Output, which is the monitor screen.
- Cout is used in conjunction with the **insertion operator** ( << ).
- Any thing on the right side of << will be displayed on monitor screen.
- **“14 Computer Systems”** is the string literal/constant to be displayed.
- **; (semicolon)** is called as **terminator**, it shows the end of any
- statement.

# Outputting with `cout`

- The syntax of cout and << is:



```
cout << expression or manipulator << expression or manipulator...;
```

  - Called an output statement
- The stream insertion operator is <<
- Expression evaluated and its value is printed at the current cursor position on the screen

# Comments

- Comments are for the reader, not the compiler.
- A comment is a line (or multiple lines) of text that are inserted into the source code to explain what the code is doing.
- The comments are always ignored (not executed) by the compiler.
- **Two types:**
  - Single line

    ```
    // This is a C++ program. It prints the sentence:
    // Welcome to C++ Programming.
    ```

  - Multiple line

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

# Comments

- Single-Line comment, comments out entire line of the code.
- It starts with double slash // .
- Any text written after // is ignored by the compiler and is considered as the comment.

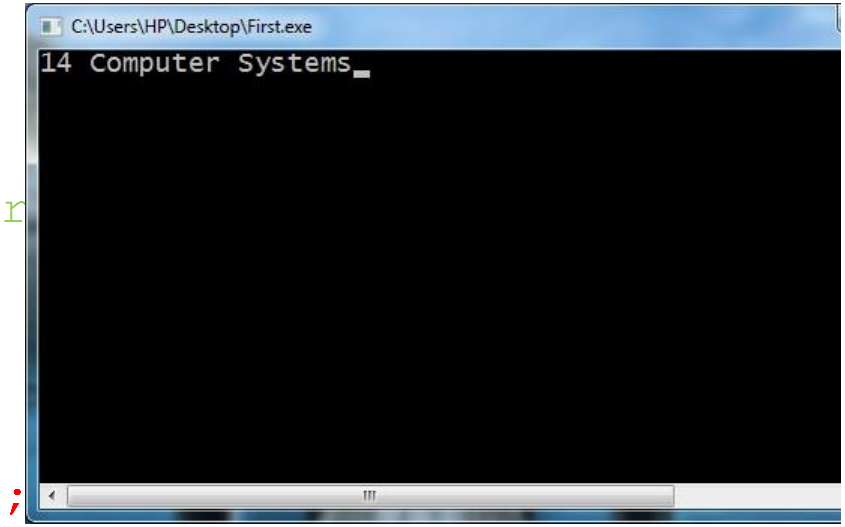
- Multi-Line comment, comments out multiple lines of the code.
- It starts with slash asterisk /* .
- It ends with asterisk slash */.
- Any text written in between /* and */ is ignored by the compiler and is considered as the comment

Single-Line Comment

Multi-Line Comment

# Example

```
#include<iostream>

// welcome at FCI-Fayoum
/*this is my first c++ progr
using namespace std;

int   main()
{
cout<<"14 Computer Systems";
return 0;
}
```



```
14 Computer Systems_
```

# Special Symbols

- **Special symbols**

```
+
-
*
/
.
;
?
,
<=
!=
==
>=
```

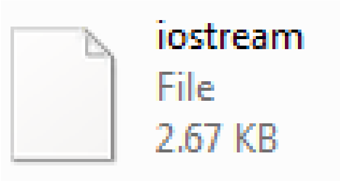
# Reserved Words (Keywords)

- **Reserved words, keywords, or word symbols**
  - Include:
    - int
    - float
    - double
    - char
    - const
    - void
    - return

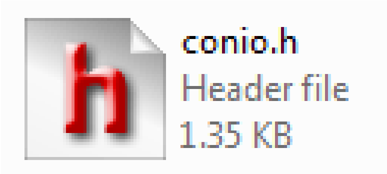
# **main()** function

- Every C++ program must contain at least one function i.e. main function.
- Main function is the gateway of any C++ program because every program starts from the main function.
- The **main** is the name of the function.
- Every function name is followed by **( )** parenthesis.
- The **int** is the return type of the function, which specifies that, at the end, the main function will return one value to the operating system whose data type will be integer.
- The **{** and **}** specify the starting and ending of the function.

# Header files


iostream
File
2.67 KB

**Input Output Stream**

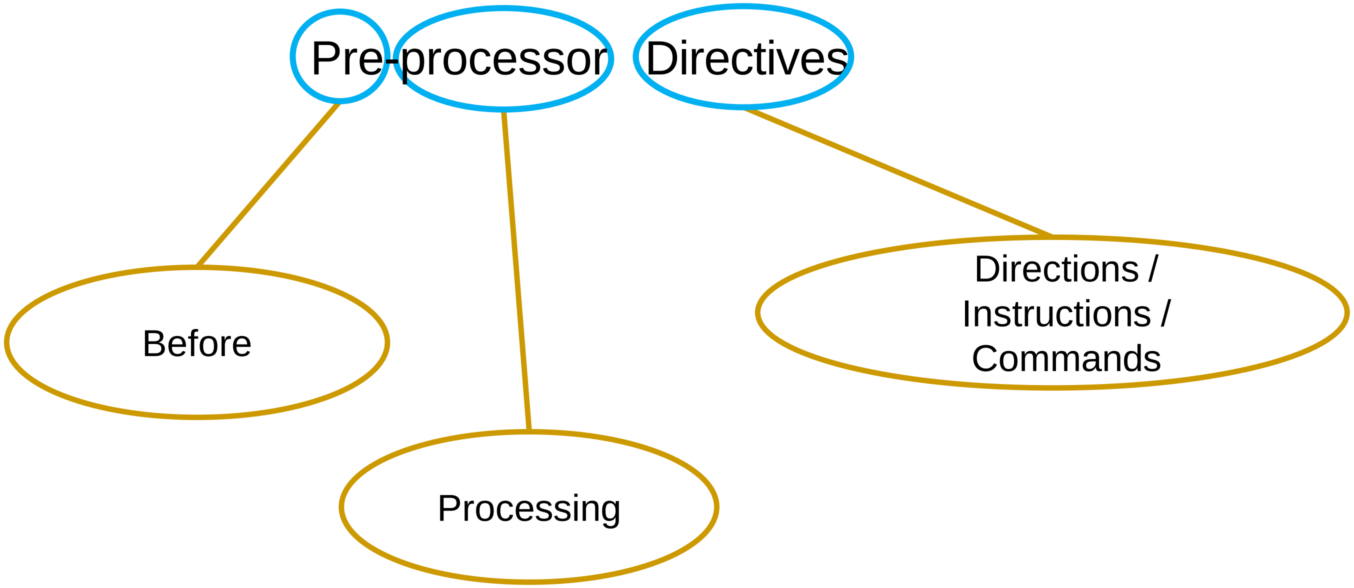

conio.h
Header file
1.35 KB

**Console Input Output**

# Header files

- Header file is the file with extension**.h**
- It contains the declarations and definitions of the functions that can be shared between various different source programs.
- When the compiler reads the **getch()** function, it does not know about how this function works.
- All the details (definition) of **getch()** function are stored in **conio.h** header file.

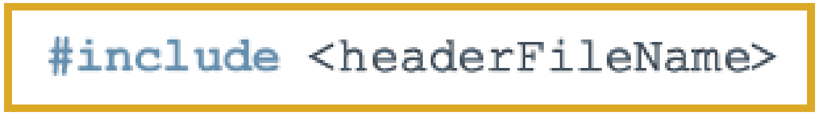
# Preprocessor Directives



**Instructions given before processing/compilation**

# Preprocessor Directives

- Syntax to include a header file:



```
#include <headerFileName>
```

- For example:

```
#include <iostream>
```

  - Causes the preprocessor to include the header file iostream in the program

# Preprocessor Directives

- In C++, there is a special program called as **Preprocessor**.
- It executes all the instructions/commands before processing the actual program in the main function.
- All the instructions/commands/direction which are given to the preprocessor are known as **Preprocessor Directives**.
- In this basic program, the first two commands at the top are known as preprocessor directives.
- Preprocessor directives start with **#** (**hash**) sign.
- **#include** is known as **include directive**, which includes the header files inside the program before processing the actual program in the main function.

# Namespaces

- A namespace can be considered as the collection of names.
- It organizes all the related names under the same namespace.
- There are many namespaces available in C++, one of them is **std** (***standard namespace***).
- The names like **cout**, **cin**, **endl** are all organized/included in **std** namespace.
- It is used to prevent name conflicts.
- It is used for categorizing the names.

# namespace and Using cin and cout in a Program

- cin and cout are declared in the header file iostream, but within std namespace
- To use cin and cout in a program, use the following two statements:

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

- The names like **cout**, **cin**, **endl** can be used in either of two ways.

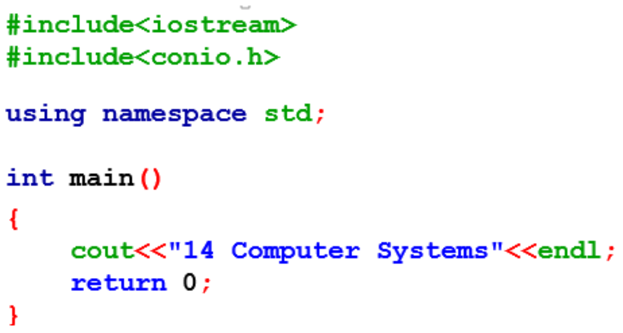
✓ **Non-fully qualified name**

✓ **Fully qualified name**

# Namespaces

## Non-fully qualified name

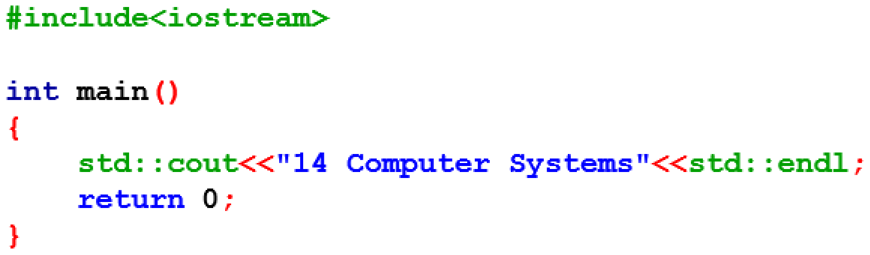
- The names can be used simply as **cout**, **cin** and **endl**.
- But we have to write the using namespace statement at the top after preprocessor directives.



```
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{
    cout<<"14 Computer Systems"<<endl;
    return 0;
}
```

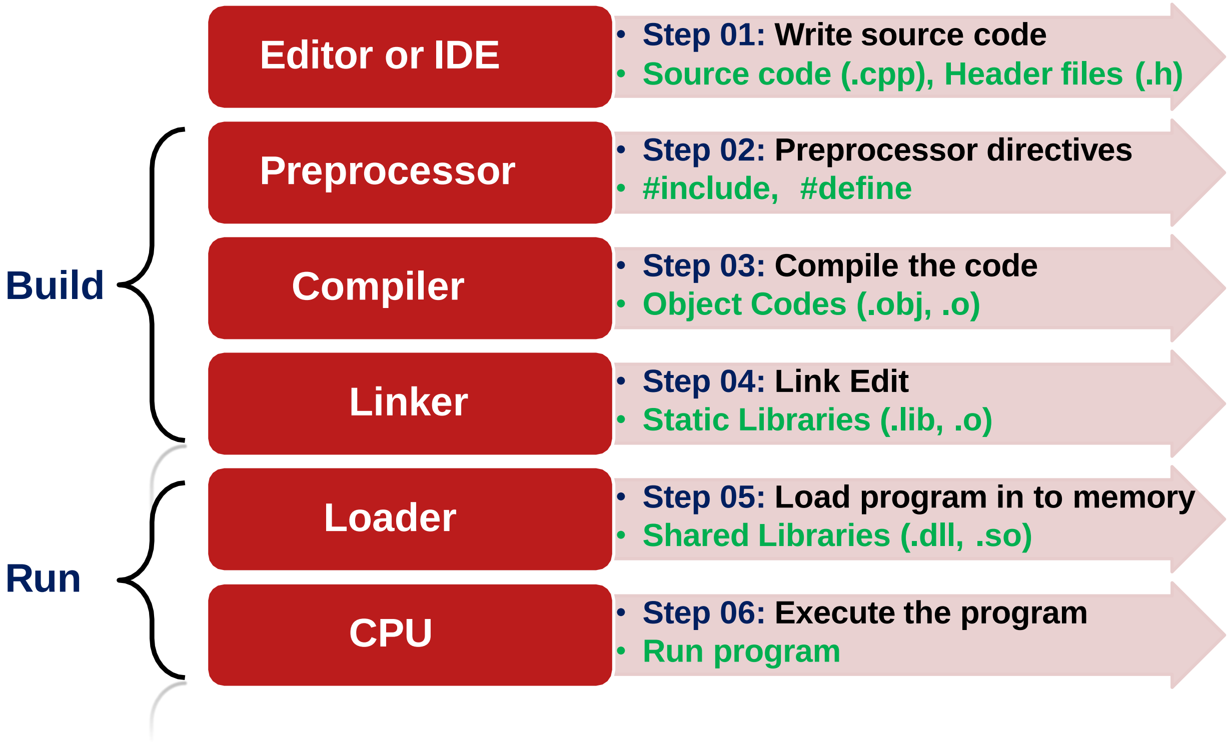
## Fully qualified name

- Every name will be prepended with namespace and **::** (double colon sign) as **std::cout**, **std::cin**, **std::endl**.
- Here we do not have to write the using namespace statement.



```
#include<iostream>

int main()
{
    std::cout<<"14 Computer Systems"<<std::endl;
    return 0;
}
```

# How C++ program works?



Build

Run

**Editor or IDE**
- Step 01: Write source code
- Source code (.cpp), Header files (.h)

**Preprocessor**
- Step 02: Preprocessor directives
- #include, #define

**Compiler**
- Step 03: Compile the code
- Object Codes (.obj, .o)

**Linker**
- Step 04: Link Edit
- Static Libraries (.lib, .o)

**Loader**
- Step 05: Load program in to memory
- Shared Libraries (.dll, .so)

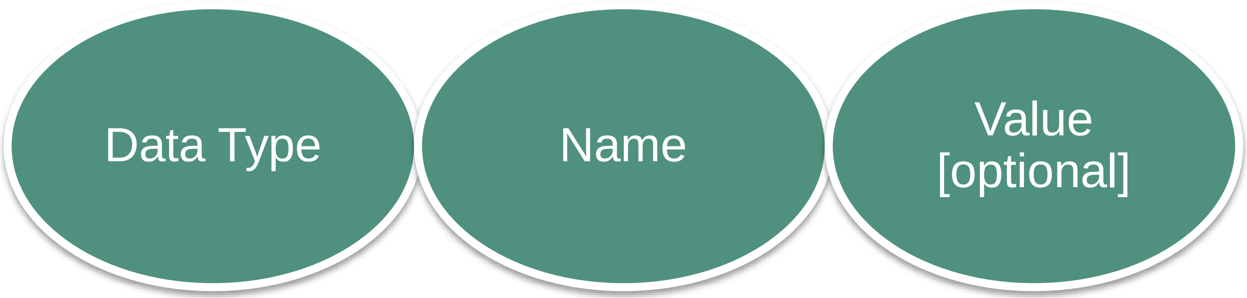
**CPU**
- Step 06: Execute the program
- Run program

# Data Types in C++

| Data Type | Memory | Range Start | Range End |
|---|---|---|---|
| char | 1 Byte | -128 | 127 |
| unsigned char | 1 Byte | 0 | 255 |
| short | 2 Bytes | -32,768 | 32,767 |
| unsigned short | 2 Bytes | 0 | 65,535 |
| int | 4 Bytes | -2,147,483,648 | 2,147,483,647 |
| unsigned int | 4 Bytes | 0 | 4,294,967,295 |
| long | 8 Bytes | -9,223,372,036,854,775,807 | 9,223,372,036,854,775,807 |
| unsigned long | 8 Bytes | 0 | 18,446,744,073,709,551,615 |
| float | 4 Bytes | $3.4 \times 10^{-38}$ | $3.4 \times 10^{38}$ |
| double | 8 Bytes | $1.7 \times 10^{-308}$ | $1.7 \times 10^{308}$ |
| bool | 1 Byte | true and false | |

# Variables in C++

- Variables are used to stored the data temporarily.
- A variable is a named piece of memory location.
- A single variable can store single value at a time.
- The value of the variable is changeable.
- In order to create a variable we need to specify three things:



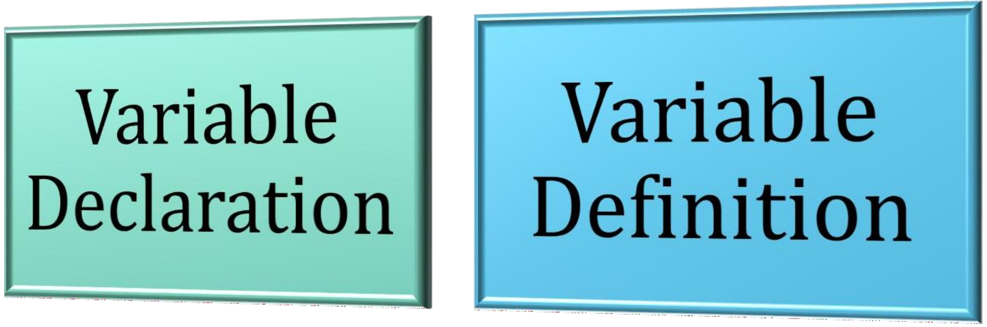
Data Type

Name

Value [optional]

# Variables in C++

- The name of the variable is called as the **identifier**.
- The data type of the variable specifies the type of the value which will be stored in it.
- The value is the actual content which will be stored in it. It is optional because some time we know the exact value which will be stored, sometimes we do not know.
- Two ways to create variables:



Variable Declaration

Variable Definition

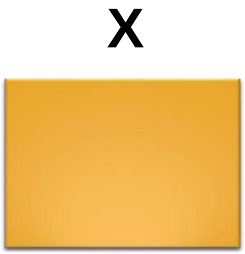
# Variable Creation in C++

## Declaration

- A variable is declared when we do not know the value to be stored in it.
- To declare the variable, we need to specify two things: **name** and **data type**.

**data_type variable_name;**

**float X;**


X

## Definition

- A variable is defined when we know the exact value to be stored in it.
- To define the variable, we need to specify three things: **name**, **data type** and**value**.

**data_type variable_name= value;**

**float X= 56.214;**

X

56.214

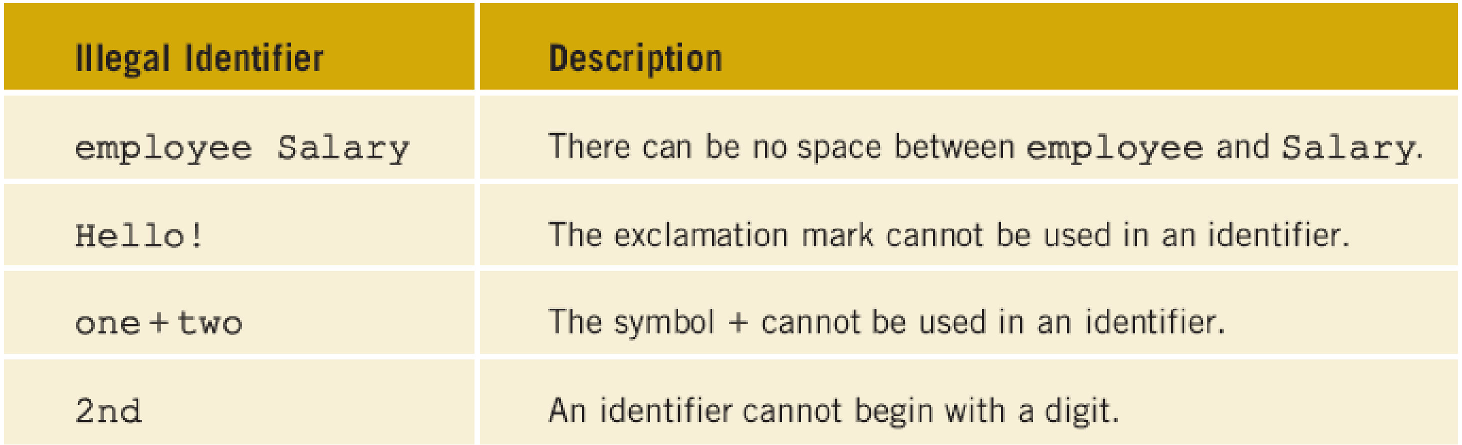
# Identifiers

- Consist of letters, digits, and the underscore character (_)
- Must begin with a letter or underscore
- C++ is case sensitive
    - NUMBER is not the same as number
- Two predefined identifiers are cout and cin
- Reserved words, can not be used.
- Does’t contain spaces.

# Identifiers (continued)

- The following are legal identifiers in C++:
  - `first`
  - `conversion`
  - `payRate`

TABLE 2-1 Examples of Illegal Identifiers



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

# Whitespaces

- Every C++ program contains whitespaces
  - Include blanks, tabs, and newline characters
- Used to separate special symbols, reserved words, and identifiers
- Proper utilization of whitespaces is important
  - Can be used to make the program readable

# Variables as Memory Locations

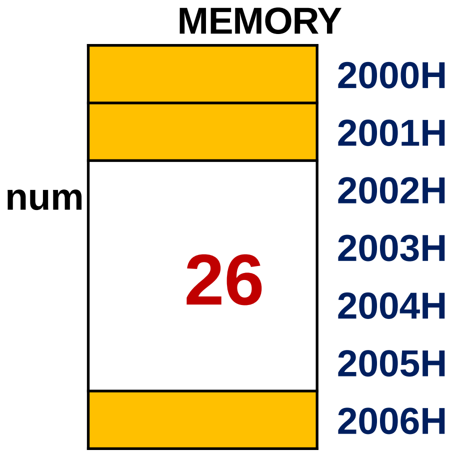
- A variable is defined as a named piece of memory location.

  .

- When we create a variable, some bytes of memory are reserved according to the data type, and are given a name of the variable.

- Entire memory is divided in to pieces of 1 byte and each of the byte has address defined in hexadecimal number.

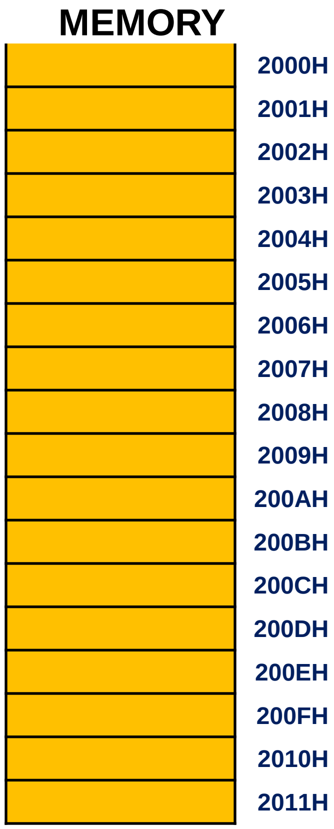
```
int num = 26 ;
```



# Variables as Memory Locations

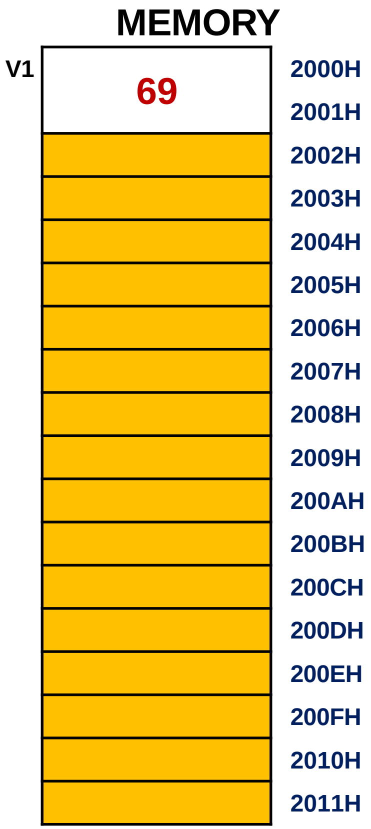
- Consider following empty memory  map:
- Execute the following variable definition statements:

```
short V1 = 69 ;

char V2 = ‘w’ ;

int V3 = 5478 ;

float V4 = 87.245 ;
```



# Variables as Memory Locations

```
short  V1 = 69 ;
```



MEMORY

V1

69

2000H

2001H

2002H

2003H

2004H

2005H

2006H

2007H

2008H

2009H

200AH

200BH

200CH

200DH

200EH

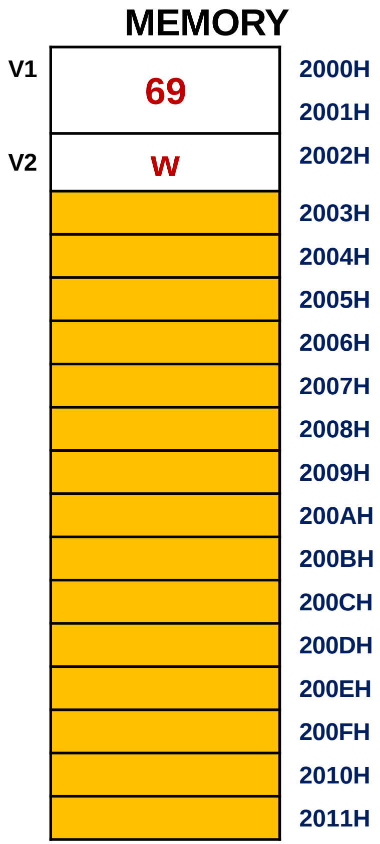
200FH

2010H

2011H

# Variables as Memory Locations

```
char  V2 = ‘w’ ;
```



MEMORY

V1 69 2000H 2001H

V2 w 2002H

2003H

2004H

2005H

2006H

2007H

2008H

2009H

200AH

200BH

200CH

200DH

200EH

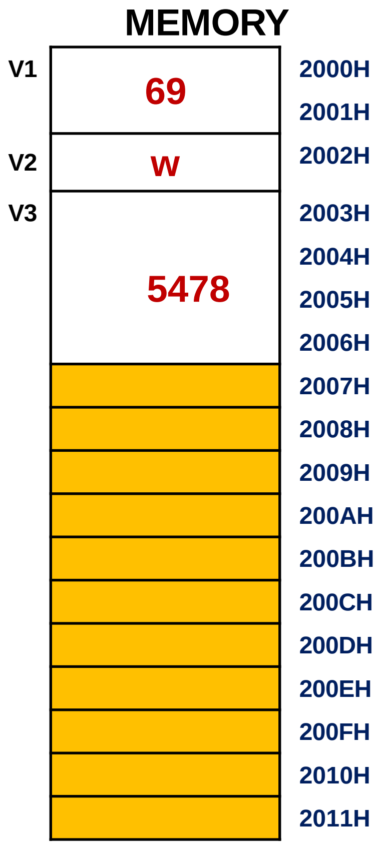
200FH

2010H

2011H

# Variables as Memory Locations

```
int  V3 = 5478 ;
```



MEMORY

V1 69 2000H 2001H

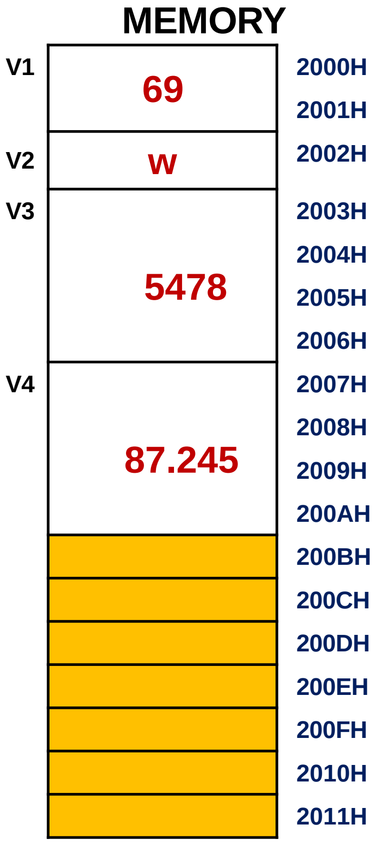
V2 w 2002H

V3 5478 2003H 2004H 2005H 2006H

2007H
2008H
2009H
200AH
200BH
200CH
200DH
200EH
200FH
2010H
2011H

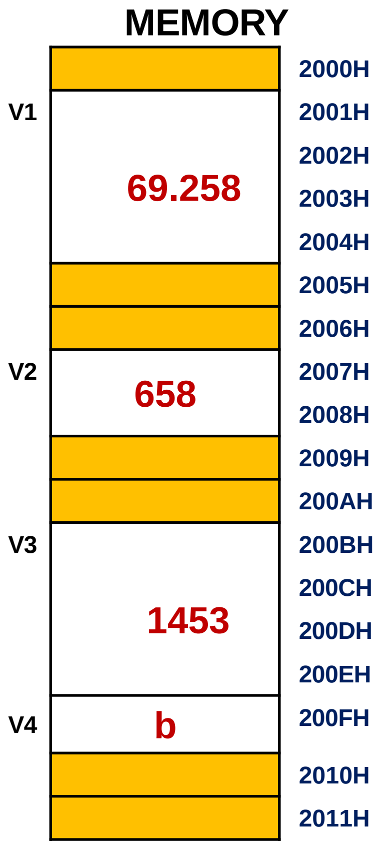
# Variables as Memory Locations

float V4 = 87.245;



# Variables as Memory Locations

- Consider following memory map:
- Write down the variable definition statements for the variables created in the memory.



MEMORY

| | | |
|---|---|---|
| | | 2000H |
| V1 | 69.258 | 2001H |
| | | 2002H |
| | | 2003H |
| | | 2004H |
| | | 2005H |
| | | 2006H |
| V2 | 658 | 2007H |
| | | 2008H |
| | | 2009H |
| | | 200AH |
| V3 | 1453 | 200BH |
| | | 200CH |
| | | 200DH |
| | | 200EH |
| V4 | b | 200FH |
| | | 2010H |
| | | 2011H |

# Variables as Memory Locations

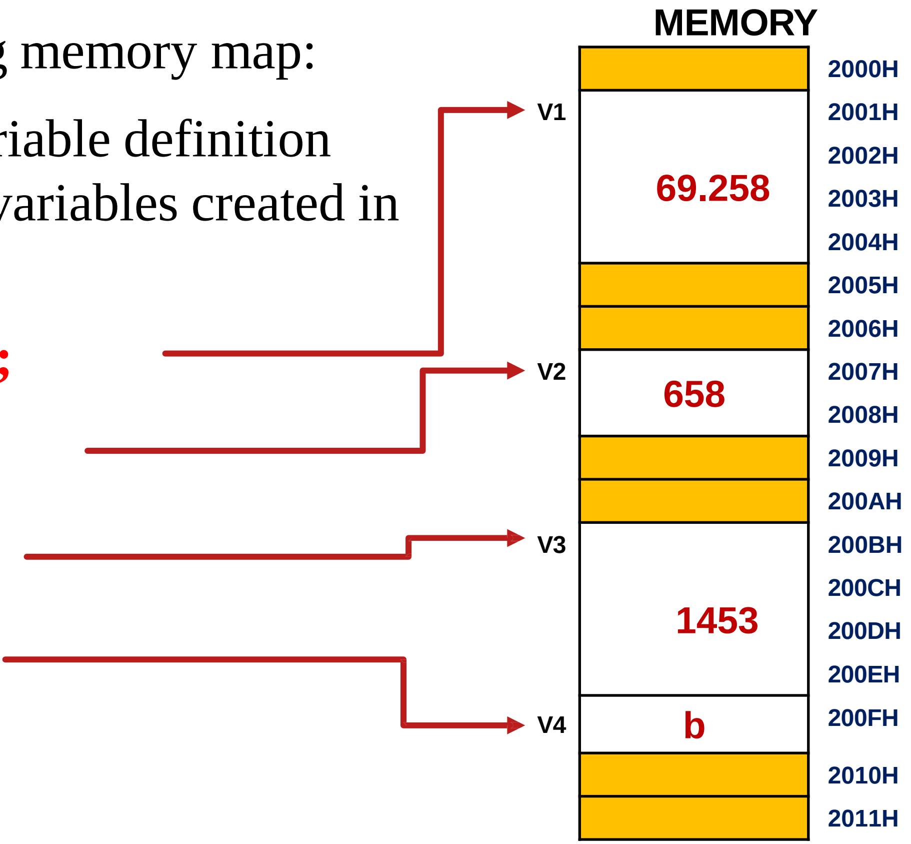
- Consider following memory map:
- Write down the variable definition statements for the variables created in memory.

**float V1 = 69.258 ;**

**short V2 = 658 ;**

**int V3 = 1453 ;**

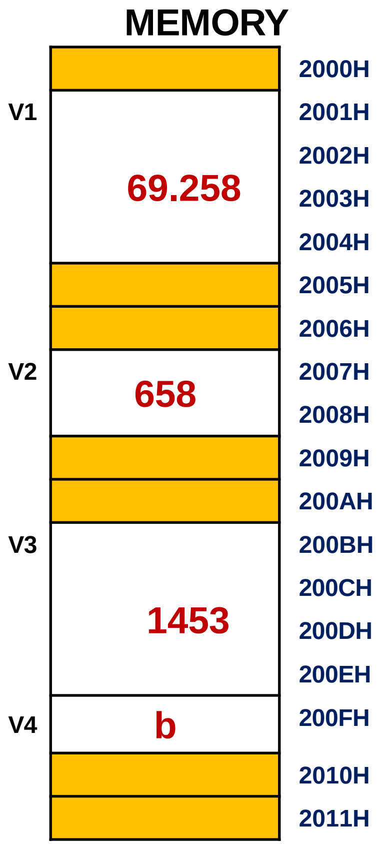
**char V4 = ‘b’ ;**



# Variables as Memory Locations

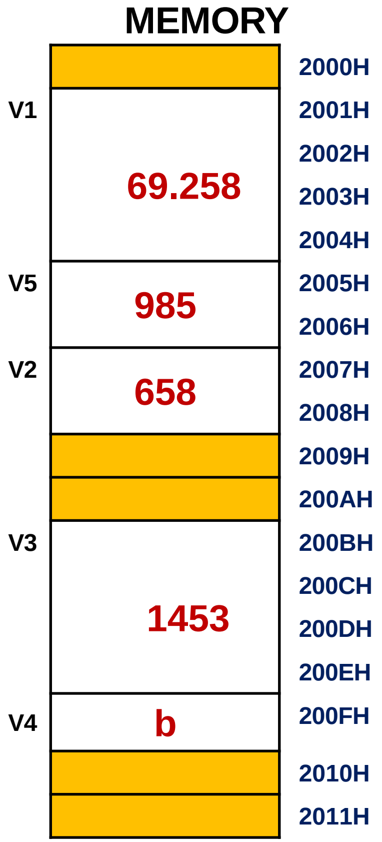
- Execute the following variables definition statements in the same memory map:

```
short  V5 = 985;
char  V6 = ‘r’ ;
short V7 = 74 ;
```



# Variables as Memory Locations

```
short   V5  = 985 ;
```

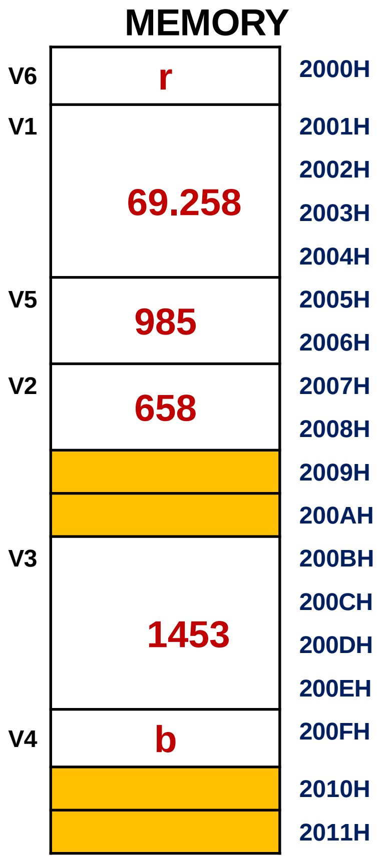


MEMORY

| | | |
|---|---|---|
| | | 2000H |
| V1 | 69.258 | 2001H |
| | | 2002H |
| | | 2003H |
| | | 2004H |
| V5 | 985 | 2005H |
| | | 2006H |
| V2 | 658 | 2007H |
| | | 2008H |
| | | 2009H |
| | | 200AH |
| V3 | 1453 | 200BH |
| | | 200CH |
| | | 200DH |
| | | 200EH |
| V4 | b | 200FH |
| | | 2010H |
| | | 2011H |

# Variables as Memory Locations

```
char   V6  = 'r' ;
```

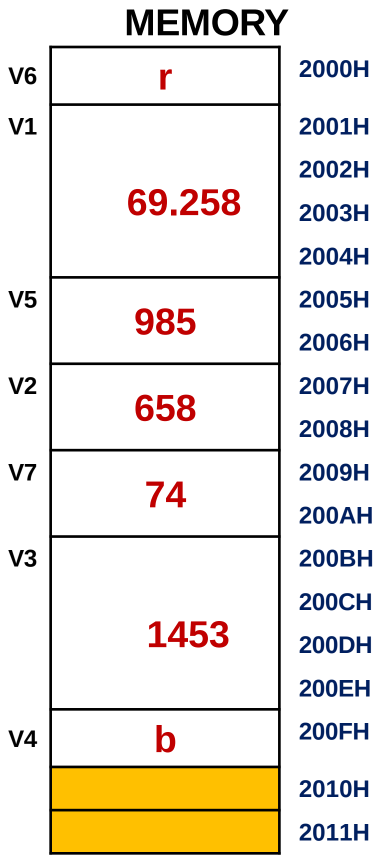


MEMORY

| | | |
|---|---|---|
| V6 | r | 2000H |
| V1 | 69.258 | 2001H |
| | | 2002H |
| | | 2003H |
| | | 2004H |
| V5 | 985 | 2005H |
| | | 2006H |
| V2 | 658 | 2007H |
| | | 2008H |
| | 🟨 | 2009H |
| | 🟨 | 200AH |
| V3 | 1453 | 200BH |
| | | 200CH |
| | | 200DH |
| | | 200EH |
| V4 | b | 200FH |
| | 🟨 | 2010H |
| | 🟨 | 2011H |

# Variables as Memory Locations

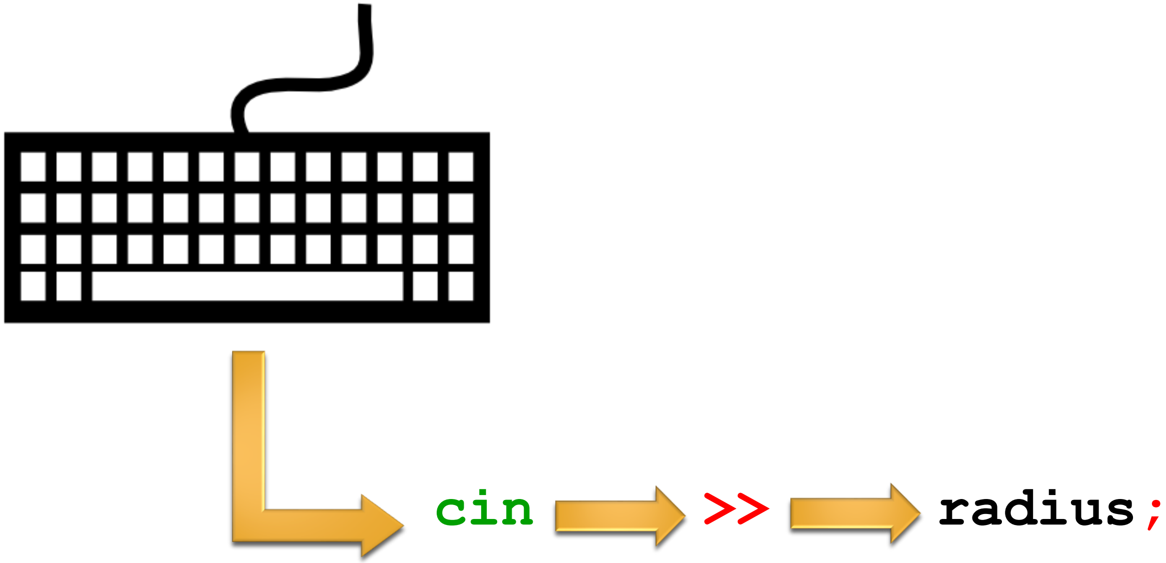
**short V7 = 74;**



MEMORY

| | | |
|---|---|---|
| V6 | r | 2000H |
| V1 | 69.258 | 2001H |
| | | 2002H |
| | | 2003H |
| | | 2004H |
| V5 | 985 | 2005H |
| | | 2006H |
| V2 | 658 | 2007H |
| | | 2008H |
| V7 | 74 | 2009H |
| | | 200AH |
| V3 | 1453 | 200BH |
| | | 200CH |
| | | 200DH |
| | | 200EH |
| V4 | b | 200FH |
| | 🟨 | 2010H |
| | 🟨 | 2011H |

# Inputting with cin



```
cin >> radius;
```

# Inputting with cin

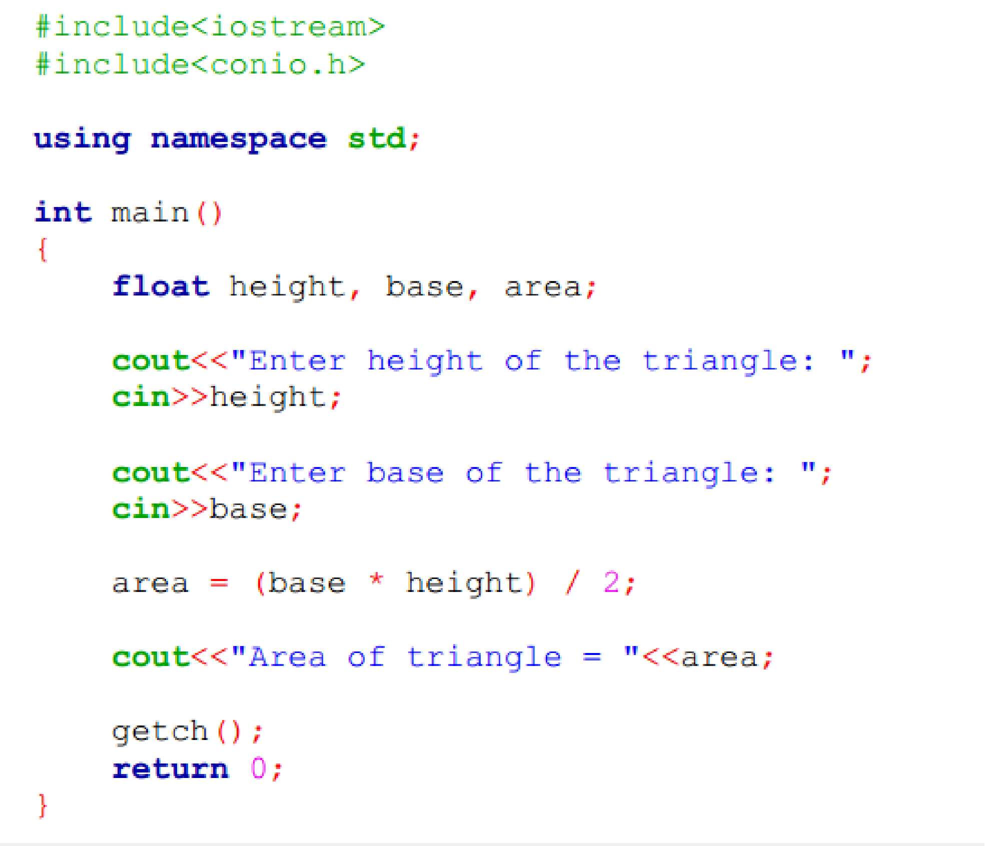
- In C++, cin statement is used to get input from the keyboard.
- Cin is the Standard Console Input, which is the keyboard.
- Cin is used in conjunction with the **extraction operator** ( >> ).
- Anything inputted from keyboard will go to the variable to the right side of >> extraction operator.

# Examples

# Program Example 01

**Problem Statement:**

Write a computer program in C++ that accepts the base and height of a right  angle triangle from the user and displays the area of the triangle.



```cpp
#include<iostream>
#include<conio.h>

using namespace std;

int main()
{
    float height, base, area;

    cout<<"Enter height of the triangle: ";
    cin>>height;

    cout<<"Enter base of the triangle: ";
    cin>>base;

    area = (base * height) / 2;

    cout<<"Area of triangle = "<<area;

    getch();
    return 0;
}
```