

# Lecture 4

## 1 Statements:

A statement in a computer carries out some action. There are three types of statements used in C++; they are expression statement, compound statement and control statement.

Expression statement	Compound statement	Control statement
<pre>x=y; sum=x+y;</pre>	<pre>{ a=b+c; x=x*x; y=a+x; }</pre>	<pre>If (a&gt;b) { a=l; k=a+1; }</pre>

## 2 Getting Started with C++:

The skeleton of a typical C++ program structure is given below:

```
Program heading
Begin
Type or variable declaration
Statements of operation
Results
end
```

The keyboard and screen I/O instructions in C++ are:

(a): **COU**T/ display an object onto the video screen:

```
Cout<<var.1<<var2<<...<<var.n;
```

(b): **Cin**/ It is used to read an object from a standard input device (keyboard):

```
Cin>>var.1>>var.2>>...>>var.n;
```

To begin learning C++ lets examine our first C++ Program:

## Example 1

```
#include<iostream.h>
void main( )
{
    // A program to print welcome
    cout << "Welcome";
}
```

Output:

Welcome

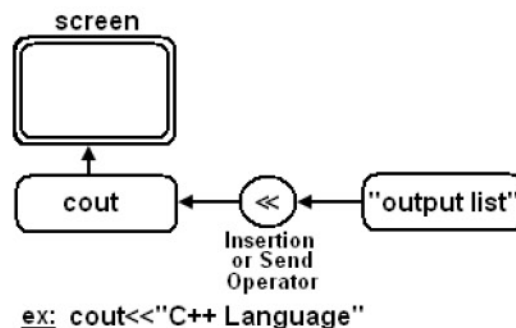
- ✓ **#include<iostream.h>** this line is for pre-processor directive. Any begins with # is processed before the program is compiled. C++ programs must be start with #include.

Every group of related functions is stored in a separate library called (header file).To use the *cin* and *cout*, must include the header file *iostream*.

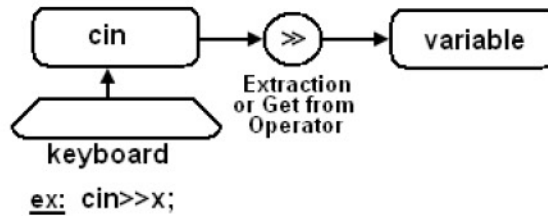
- ✓ **main( )**, is the name of C++ function. Every C++ program must have a function called main.
- ✓ **void**, is the return type of the main function. When the return type of a function is void, this function will not passes back any value to the calling function.

Some programmers use *int* as a return type for the main function, in this case a **return(0)** statement must be written as a last statement of the main function-body.

- ✓ **{**, introducing the statements that define the function.
- ✓ **}**, indicates the end of the statements in the function.
- ✓ **//**, text after these symbols is a comment. It does not affect the program code, and compilers normally ignore it.
- ✓ **cout**, the input stream object. It passes the characters quotes (") to the terminal screen.



- ✓ **cin**, the input stream object. It reads the input values from the keyboard.



- ✓ <<, the stream insertion operator (or send operator).
- ✓ >>, the stream extraction operator (or get from operator).
- ✓ ;, semicolon, the terminator of every C++ statement.

The **endl** is used in c++ to represent a new line, as shown in the following example:

Example 2	
<pre> #include&lt;iostream.h&gt; void main( ) {     cout &lt;&lt; "hallow" &lt;&lt; endl;     cout &lt;&lt; "students"; }           </pre>	<p><b>Output:</b></p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: auto;">             hallow students           </div>

The **\n** is a special escape code, also used in C++ to represent a new line, as shown in the following example:

Example 3	
<pre> #include&lt;iostream.h&gt; void main( ) {     cout &lt;&lt; "hallow \n";     cout &lt;&lt; "students"; }           </pre>	<p><b>Output:</b></p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: auto;">             hallow students           </div>



#### Example 4



The following program reads three different inputs and outputs it.

```
#include<iostream.h>
void main( )
{
    int num=3;
    cout << "number="<<num<<"\n";
    char ch='a';
    cout << "character="<<ch<<"\n";
    float fa=-34.45;
    cout<<"real number="<<fa<<"\n";
}
```

#### Output:

```
Number=3
Character=a
Real number=34.45
```

#### Example 5



The following program reads three different inputs and outputs it.

```
#include<iostream.h>
void main( )
{
    int n; float f; char c;
    cout << "input integer number:";
    cin>>n;
    cout<<endl;

    cout << "input decimal number:";
    cin>>f;
    cout<<endl;

    cout << "input character:";
    cin>>c;
}
```

#### Output:

```
input integer number: 5
input decimal number: 4.2
input character: A
```

### 4 Constants:

Like variables, constants are data storage locations. Unlike variables, and as the name implies, constants don't change.

```
const int myage=23;
const double pi=3.14;
const float salary=20.5;
```

## Example 6

 Write a program that reads the radius of a circle, then computes and outputs its area.

```
#include<iostream.h>
void main( )
{
    const float pi = 3.14;
    int r; float c;
    cout << "enter the radius of circle:";
    cin>>r;
    cout<<endl;
    c = r * r * pi;
    cout << "the area of circle:" << c;
}
```

### Output:

```
enter the radius of circle: 5
the area of circle: 78.5
```

## Example 7

 The following program computes the arithmetic operators.

```
#include<iostream.h>
void main( )
{
    int a,b,sum,sub,mul,div;
    cout << "enter any two numbers<<endl;
    cin>> a>>b;
    sum=a+b;
    sub=a-b;
    mul=a*b;
    div=a/b;
    cout<<"a="<<a<<"b="<<b<<"sum="<<sum<<endl;
    cout<<"sub="<<sub<<endl;
    cout<<"mul="<<mul<<endl;
    cout<<"div="<<div<<endl;
}
```

### Output:

```
Enter any two numbers
10 20
A=10 b=20 sum=30
Sub=10
Mul=200
Div=0
```

### Example 8

 The following program computes different division operators.

```
#include<iostream.h>
void main( )
{
    int x, y, z, r ;
    x= 7 / 2;
    cout << "x=" << x <<endl;
    y=17/(-3);
    cout << "y="<< y <<endl;
    z=-17/3;
    cout << "z="<< z <<endl;
    r=-17/(-3);
    cout << "r="<< r <<endl;
}
```

Output:

```
x= 3
y= -5
z= -5
r= 5
```

The modulus operator “%” is used with integer operands (int, short, long, unsigned). It can't be used with float or double operands.

### Example 9

```
#include<iostream.h>
void main( )
{
    int y1, y2;
    y1 = 8 % 3;
    y2 = -17 % 3;
    cout << "y1="<< y1 <<endl;
    cout << "y2="<< y2 <<endl;
}
```

Output:

```
y1=2
y2=-2
```