Ministry of Higher Education & Scientific Research

PAITAXT

Technical Institute-Private

1st Year Computer & Network

C++ Programming

Chapter 2:

Input/Output

Updated by: ***Asst. Prof. Dr. Tahseen G. Abdullah***

I/O Streams and Standard I/O Devices

- I/O: sequence of bytes (stream of bytes) from source to destination. Bytes are usually characters.
- Stream: sequence of characters from source to destination.
- Input stream: sequence of characters from an input device to the computer.
- Output stream: sequence of characters from the computer to an output device.
- Input stream variables: type `istream`
- Output stream variables: type `ostream`
- To use `cin` and `cout`, the preprocessor directive `#include <iostream>` must be used.

- Use `iostream` header file to extract (receive) data from keyboard and send output to the screen
  - Contains definitions of two data types:
    - `istream` - input stream
    - `ostream` - output stream
  - Has two variables:
    - `cin` - stands for common input
    - `cout` - stands for common output

Q.: The sequence of characters from the computer to an output device is:

(a) input stream **(b) output stream** **(c) common input** **(d) common output**

Q.: What is this called? `istream`

(a) input stream **(b) output stream** **(c) common input** **(d) common output**

Output Statement

- The syntax of an output statement using **cout** and the instruction operator << is :

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

  - Expression is evaluated and printed (Ex.: **cout << x+2 ;**)
  - Value is printed (Ex.: **cout << 20;**)
  - Manipulator is used to format the output

    (Ex.: **cout << endl ;**)

- Values sent to an output device
  - Usually the screen
  - Can also be a file or some device

Q.: in C++ program cout uses the operator ----------.

(a) > **(b) <** **(c) <<** **(d) >>**

Output Statement Examples

cout << variable-name; **e.g. cout << a;**

Meaning: print the value of variable to the user

cout << " Any Message "; **e.g. cout << " Hello ";**

Meaning: print the message within quotes to the user

cout << endl;

Meaning: print a new line

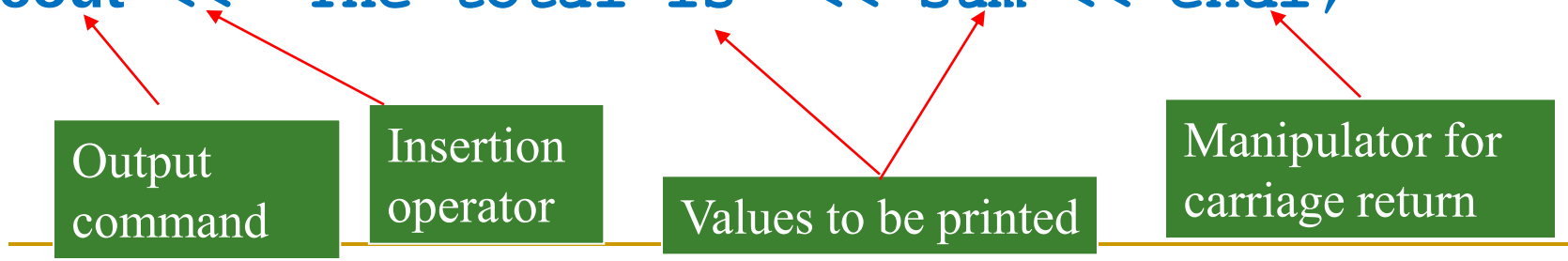
cout << 8+6/2-1;

Meaning: expression is evaluated and printed

cout << 20;

Meaning: print the value

```
cout << "The total is "<< sum << endl;
```



C++ Example Program:

```
1   // Am programa basy output dakat.
2   #include <iostream>
3   using namespace std;
4   int main()
5   {
6    int a=4 ;                                  // Output Results:
7    cout<< "  Value of a is=  "  <<a<<endl ;   // Value of a is=4
8    cout<<a+2<<endl ;                          // 6
9    cout<< 8+6/2-1<< endl ;                    // 10
10   cout<< " 9+6/3-1 "  << endl ;              // 9+6/3-1
11   cout<<a+2*a << endl;                       // 12
12   cout<<a-4;                                 // 0
13
14   return 0 ;
15   }
```

Input Statement

- The syntax of an input statement using `cin` and the extraction operator `>>` is :

```
cin >> variable >> variable...;
```

Meaning: read the **value** of the variable from the user

- There are two ways to initialize a variable:

```
int a;
```

  - By using the assignment statement

```
a= 35;
```

  - By using input statement (read statement)

```
cin >> a;
```

Ex1:

```
int a;
a= 35;
cout<<a<<endl;
```

Ex2:

```
int a;
cin >> a;
cout<<a<<endl;
```

Q.: In C++ program cin uses the operator ----------.

(a) > **(b) <** **(c) <<** **(d) >>**

Input Statements & Values Stored in Memory

Example

```
int a, b;
double z;
char ch, ch1, ch2;
```

	Statement	Input	Value Stored in Memory
1	`cin >> ch;`	`A`	`ch = 'A'`
2	`cin >> ch;`	`AB`	`ch = 'A', 'B'` is held for later input
3	`cin >> a;`	`48`	`a = 48`
4	`cin >> a;`	`46.35`	`a = 46, .35` is held for later input
5	`cin >> z;`	`74.35`	`z = 74.35`
6	`cin >> z;`	`39`	`z = 39.0`
7	`cin >> z >> a;`	`65.78 38`	`z = 65.78, a = 38`
8	`cin >> a >> b;`	`4 60`	`a = 4, b = 60`
9	`cin >> a >> ch >> z;`	`57 A 26.9`	`a = 57, ch = 'A', z = 26.9`
10	`cin >> a >> ch >> z;`	`57 A` `26.9`	`a = 57, ch = 'A', z = 26.9`

	Statement	Input	Values Stored in Memory
11	`cin >> a >> ch >> z;`	57 A 26.9	a = 57, ch = 'A', z = 26.9
12	`cin >> a >> ch >> z;`	57A26.9	a = 57, ch = 'A', z = 26.9
13	`cin >> z >> ch >> a;`	36.78B34	z = 36.78, ch = 'B', a = 34
14	`cin >> z >> ch >> a;`	36.78 B34	z = 36.78, ch = 'B', a = 34
15	`cin >> a >> b >> z;`	11 34	a = 11, b = 34, computer waits for the next number
16	`cin >> a >> z;`	46 32.4 68	a = 46, z = 32.4, 68 is held for later input
17	`cin >> a >> z;`	78.49	a = 78, z = 0.49
18	`cin >> ch >> a;`	256	ch = '2', a = 56
19	`cin >> a >> ch;`	256	a = 256, computer waits for the input value for ch
20	`cin >> ch1 >> ch2;`	A B	ch1 = 'A', ch2 = 'B'

C++ Example Program:

```
1    // Am programa basy cin dakat.
2    #include <iostream>
3    using namespace std;
4    int main()
5    {
6     int a , b ;
7    double z ;
8    char ch , ch1, ch2 ;
9
10    cout<< "  Pet u Jmara Daxel Bka?  "  <<endl ;
11    cin>>ch>>a ;
12    cout<< "  The result of a is=  "  <<a<<endl ;
13    cout<< "  The result of ch is=  "  <<ch<<endl ;
14    return 0 ;
15    }
```

Escape sequence

- Escape sequences used to manipulate output

	Escape Sequence	Description
\n	Newline	Cursor moves to the beginning of the next line
\t	Tab	Cursor moves to the next tab stop
\ b	Backspace	Cursor moves one space to the left
\r	Return	Cursor moves to the beginning of the current line (not the next line)
\\	Backslash	Backslash is printed
\'	Single quotation	Single quotation mark is printed
\"	Double quotation	Double quotation mark is printed
\a	Alert (bell)	Sound the system bell
\v	Vertical tab	Cursor moves to the vertical tab
\?	Question mark	Question mark is printed

Q.: Which escape sequence denotes tab character in C++ program?

(a) \a **(b) \b** **(c) \t** **(d) \n**

Example1 (Escape sequence)

```
cout << " Computer Department"<<endl;;
cout << " Computer \t Department"<<endl;
cout << " Computer Department\?\!"<<endl;
cout << " Computer Department\a\a\a"<<endl;
```

- **Output**:

```
Computer Department
Computer        Department
Computer Department?!
Computer Department (with 3 bell sound)
```

Q.: Escape sequence of the **tab** character is (\t)

Note: May appear anywhere in the string

Example2 (Escape sequence)

- The new line character is '\n'
  - May appear anywhere in the string

```
cout << "Hello ALi.";
cout << " My name is Zaki.“<<endl;
```

- **Output**:

```
Hello Ali. My name is Zaki.
```

```
cout << "Hello Ali.\n";
cout << "My name is Zaki."<<endl;
```

- **Output** :

```
Hello Ali.
My name is Zaki.
```

Arithmetic Assignment Operators

Long Hand	Short Hand
x = x * y;	x *= y;
x = x / y;	x /= y;
x = x % y;	x %= y;
x = x + y;	x += y;
x = x - y;	x -= y;

i++	i=i+1	Increase by 1
i--	i=i-1	decrease by 1

Example

```
int i = 10;
int j = 15;
double k = 15.0;
```

Arithmetic operators

```
int add = i + j;
int diff = j – i;
int product = i * j;
int quotient = j / i;
double quotient = k / i;
int residual = j % i;
i++;
```

The Answer

```
//25
//5
// 150
// 1
// 1.5
// 5
// 11
```

Increment and Decrement Operators

- Pre-increment `++x;`
  equivalent to `x = x + 1;`
  - Pre-decrement `--x;`
  - Changes the value before execution of a statement `y = ++x;`
- Post-increment `x++;`
  equivalent to `x = x + 1;`
  - Post-decrement `x--;`
  - Changes the value after execution of the statement `y = x++;`

operator	easy	easiest
x = x+1;	x += 1	x++
x = x-1;	x -= 1	x--

Example Program: Increment & Decrement Operators

```
#include < iostream >
using namespace;
int main ()
{
int n = 2, k =4;                          //Output Results:
cout << n << endl;                          // line1:
cout << ++n << endl;                        // line2:
cout << n << endl;                          // line3:
cout << k++ << endl;                        // line4:
cout << k<< endl;                           // line5:
cout << n+10 << endl;                       // line6:
cout << -n << endl;                         // line7:
cout <<  n << endl;                         // line8:
cout << --k << endl;                        // line9:
cout << n-- << endl;                        // line10:
cout << n + k << endl;                      // line11:
cout << n << "\t" <<k << endl;              // line12:
cout << "\n\n\a" << endl;                   // line13:
cout << " k * k = "<< k * k;                // line14:
return 0;
}
```

Example Program: (Arithmetic Assignment Operators)

```
1   // Arithmetic Assignment Operator
2   #include <iostream>
3   using namespace std;
4
5    int main()
6    {
7      int a,b=3;
8
9       a=b;
10      a+=2;          // equivalent to a=a+2
11      b++;           // equivalent to b=b+1
12     cout << "a=" << a << endl;
13     cout << "b=" << b << endl;
14     return 0;
15  }
```

Output Results:

```
a=5
b=4
```

Define & Predefined Functions in a Program

- Define function is organized in he header files as follows:

  #define name ------

- Predefined functions are organized as a collection of libraries called header files:
  - When activated, it accomplishes a task
  - Header file may contain several functions
  - To use a predefined function, you need the name of the appropriate header file
  - You also need to know:
    - Function name
    - Number of parameters required
    - Type of each parameter
    - What the function is going to do

e.g.

- To use `pow` (power), we need:
  - `# include <cmath>` in the header file
  - Two numeric parameters
  - Syntax: `pow(x,y)` = x^y
    - `x` and `y` are the arguments or parameters
  - In `pow(2,3)`, the parameters are `2` and `3`
- `sqrt` is the square root
- `str.length()` is the length of string

Q.: What are the result of pow(2,3)?

(a) 6 **(b) 8** **(c) 9** **(d) 0**

Example Program: Using #define function

```
1    //  Using #define --
2    #include <iostream>
3    using namespace std;
4
5   #define Ali  ' \t '
6   #define NEWLINE ' \n'
7
8    int main()
9    {
10        int  n=10;
11       cout << n << Ali << n+2 << NEWLINE;
12       cout << ++n << Ali << n++ << NEWLINE;
13       cout << n<< Ali << ++n << NEWLINE;
14       cout << n<< NEWLINE<< NEWLINE;
15       cout << n+6 << Ali << n-6 << NEWLINE;
16       return 0;
17   }
```

Output Results:

Sample Program: #define & Predefined Functions

```
1    // How to use Predefined Functions.
2    #include <iostream>
3    #include <cmath>
4    #include <string>
5    #define Ali ' \t '
6    #define Ok ' \a '
7    using namespace std;
8     int main()
9     {
10        string str= “Hiwa Yasin Karim";
11      cout << " pow(2,3)=" <<pow(2,3)<< Ali ;
12      cout <<sqrt(25)<< endl;
13      cout << " Length of string str=" <<str.length()<<endl;
14      cout <<Ok<<Ok<< endl;
15      return 0;
16    }
```

Output Results:

```
pow(2,3)=8          5
Length of string str=16
```

Sample Program: Predefined Functions

```
1   // How to use Predefined Functions.
2   #include <iostream>
3   #include <cmath>
4   #include <string>
5   using namespace std;
6    int main()
7    {
8      int a=25,b=2;
9      double x=16,p=0,y;
10     string str1= "Programming With C++";
11     cout << " pow(2,3)=" <<pow(2,3)<< endl;                  // line11
12     cout << " pow(a,b)=" <<pow(a,b)<< endl;                  // line12
13     cout << " square root of x is =" <<sqrt(x)<< endl;       // line13
14     y=cos(p);
15     cout << " y=cos(p)=" <<y<< endl;                         // line15
16     cout << " Length of string str1=" <<str1.length()<<endl; // line16
17     return 0;
18  }
```

Output Results:

```
Line 11:  pow(2,3)=8
Line 12:  pow(a,b)=625
Line 13:  square root of x is =4
Line 15:  y=cos(p)=1
Line 16:  Length of string str1=20
```

End of the Lecture

Let Learning Continue

Example: What are the results of the following `cout` statements?

```
1)  cout << "Hello ALi.";
2)  cout << "Hello";
3)  cout << 20;
4)  cout << 8+6/2-1;
5)  cout << 8+6/(2-1);
6)  cout << 9/2;
7)  cout << 9.0/2.0;
8)  cout << 4+2*3-5;
9)  cout << 2<<endl<<4<<endl<<6;
10) cout << 2<<4<<6;
```