Input/Output (I/O)

The standard C++ library includes the header file iostream, which can be used to feed new data into the computer or obtain output on an output device such as: VDU, printer etc. The following C++ stream objects can be used for the input/output purpose.

**cout** console output  
**cin** console input

### cout object

cout is used to print message on screen in conjunction with the insertion operator `<<`

```cpp
cout << "Hello World"; // prints Hello world on screen  
cout << 250; // prints number 250 on screen  
cout << sum; // prints the content of variable sum on screen
```

To print constant strings of characters we must enclose them between double quotes (").

If we want to print out a combination of variables and constants, the insertion operator (`<<`) may be used more than once in a single statement

```cpp
cout << "Area of rectangle is " << area << " square meter" ;
```

If we assume the area variable to contain the value 24 the output of the previous statement would be:
Area of rectangle is 24 square meter

### cin object

cin can be used to input a value entered by the user from the keyboard. However, the extraction operator `>>` is also required to get the typed value from cin and store it in the memory location.

Let us consider the following program segment:
int marks;
cin >> marks;

In the above segment, the user has defined a variable marks of integer type in the first statement and in the second statement he is trying to read a value from the keyboard.

// input output example
#include <iostream>
using namespace std;

int main ()
{
    int length;
    int breadth;
    int area;

    cout << "Please enter length of rectangle: ";
cin >> length;
    cout << "Please enter breadth of rectangle: ";
cin >> breadth;

    area = length * breadth;
    cout << "Area of rectangle is " << area;
    return 0;
}

Output :

Please enter length of rectangle: 6
Please enter breadth of rectangle: 4
Area of rectangle is 24

You can also use cin to request more than one input from the user:

cin >> length >> breadth;

is equivalent to:

cin >> length;
cin >> breadth;
cin and strings

We can use cin to get strings with the extraction operator (>>) as we do with fundamental data type variables:

```cpp
cin >> mystring;
```

However, cin extraction stops reading as soon as if finds any blank space character, so in this case we will be able to get just one word for each extraction.

For example, if we want to get a sentence from the user, this extraction operation would not be useful. In order to get entire lines, we can use the function getline, which is the more recommendable way to get user input with cin:

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

int main ()
{
    string name;
    cout << "Enter your name";
    getline (cin, name);
    cout << "Hello " << name << "!
";
    return 0;
}
```

Output

Enter your name : Aniket Rajput
Hello Aniket Rajput!