

# **Welcome to C++**

**CSE 225 - Data Structures and Algorithms**

Md. Mahfuzur Rahman  
ECE Department  
North South University

## 1 Using a Class [Case A]

```
1 #include<iostream>
2 using namespace std;
3
4 class MyInfo
5 {
6     private:
7         // data hidden from outside world
8         int x;
9
10    public:
11        // function to set value of variable x
12        void set(int a)
13        {
14            x = a;
15        }
16
17        // function to return value of variable x
18        int get()
19        {
20            return x;
21        }
22 };
23
24 // main function
25 int main()
26 {
27     MyInfo obj;
28
29     obj.set(5);
30
31     cout<<obj.get(); //cout is used inspite of printf( )
32     return 0;
33 }
34
35
36
37 output:
38
39 5
```

## 2 Using a Class [Case B]

```
1 #include<iostream>
2 using namespace std;
3
4 class MyInfo
5 {
6     private:
7         // data hidden from outside world
8         int x;
9
10    public:
11        // function to set value of variable x
12        void set(int a);
13
14        // function to return value of variable x
15        int get();
16 };
17
18 void MyInfo::set(int a)
19 {
20     x = a;
21 }
22
23 int MyInfo::get()
24 {
25     return x;
26 }
27
28
29 // main function
30 int main()
31 {
32     MyInfo obj;
33
34     obj.set(5);
35
36     cout<<obj.get(); //cout is used inspite of printf( )
37     return 0;
38 }
39
40
41
42
43 output:
44
45 5
```

### 3 Using a Class [Case C]

```
1 #include<iostream>
2 using namespace std;
3
4 class Calculator {
5
6     private:
7         int num1;
8         int num2;
9
10    public:
11        Calculator(int n1, int n2){ //Constructor
12            num1 = n1;
13            num2 = n2;
14        }
15
16        int add() {
17            return num1 + num2;
18        }
19
20        int multiply() {
21            return num1 * num2;
22        }
23
24        void display() {
25            cout<<"Numbers: "<<num1<<" and "<<num2;
26            cout<<"\n\t Add:"<<add();
27            cout<<"\n\t Multiply:"<<multiply();
28            cout<<endl<<endl;
29        }
30 };
31
32 int main() {
33     Calculator a(2,4);
34     Calculator b(5,3);
35
36     a.display();
37     b.display();
38     return 0;
39 }
40
41 Output:
42
43 Numbers: 2 and 4
44     Add:6
45     Multiply:8
46
47 Numbers: 5 and 3
48     Add:8
49     Multiply:15
```

## 4 Using a Class with Template [Case A]

```
1 #include<iostream>
2 using namespace std;
3
4 template<class T>
5 class Calculator {
6
7     private:
8         T num1;
9         T num2;
10
11     public:
12         Calculator(T n1, T n2){
13             num1 = n1;
14             num2 = n2;
15         }
16
17         T add() {
18             return num1 + num2;
19         }
20
21         T multiply() {
22             return num1 * num2;
23         }
24
25         void display() {
26             cout<<"Numbers: "<<num1<<" and "<<num2;
27             cout<<"\n\t Add:"<<add();
28             cout<<"\n\t Multiply:"<<multiply();
29             cout<<endl<<endl;
30         }
31 };
32
33 int main() {
34     Calculator<int> a(2,4);
35     Calculator<float> b(2.2,4.1);
36
37     a.display();
38     b.display();
39     return 0;
40 }
41
42 Output:
43
44 Numbers: 2 and 4
45     Add:6
46     Multiply:8
47
48 Numbers: 2.2 and 4.1
49     Add:6.3
50     Multiply:9.02
```

## 5 Using a Class with Template [Case B]

```
1 #include<iostream>
2 using namespace std;
3
4 template<class T>
5 class Calculator {
6
7     private:
8         T num1;
9         T num2;
10
11     public:
12         Calculator(T n1, T n2);
13         T add();
14         T multiply();
15         void display();
16
17 };
18
19 template<class T>
20 Calculator<T>::Calculator(T n1, T n2){
21     num1 = n1;
22     num2 = n2;
23 }
24
25 template<class T>
26 T Calculator<T>::add() {
27     return num1 + num2;
28 }
29
30 template<class T>
31 T Calculator<T>::multiply() {
32     return num1 * num2;
33 }
34
35 template<class T>
36 void Calculator<T>::display() {
37     cout<<"Numbers: "<<num1<<" and "<<num2;
38     cout<<"\n\t Add:"<<add();
39     cout<<"\n\t Multiply:"<<multiply();
40     cout<<endl<<endl;
41 }
42
43
44 int main() {
45     Calculator<int> a(2,4);
46     Calculator<float> b(2.2,4.1);
47
48     a.display();
49     b.display();
50     return 0;
51 }
```

## 6 Using a Class within Class

```
1 #include <iostream>
2
3 using namespace std;
4
5 class Box {
6     private:
7         double length;    // Length of a box
8         double breadth;    // Breadth of a box
9         double height;    // Height of a box
10
11     public:
12         // Constructor definition
13         Box(double l = 2.0, double b = 2.0, double h = 2.0) {
14             cout << "Constructor called." << endl;
15             length = l;
16             breadth = b;
17             height = h;
18         }
19         double Volume() {
20             return length * breadth * height;
21         }
22         int compare(Box box) {
23             return this->Volume() > box.Volume();
24         }
25
26 };
27
28
29 int main(void) {
30     Box Box1(3.3, 1.2, 1.5);    // Declare box1
31     Box Box2(8.5, 6.0, 2.0);    // Declare box2
32
33     if(Box1.compare(Box2)) {
34         cout << "Box2 is smaller than Box1" << endl;
35     } else {
36         cout << "Box2 is equal to or larger than Box1" << endl;
37     }
38
39     return 0;
40 }
```

## 7 Using a Class with Operator Overloading

```
1 #include <iostream>
2 using namespace std;
3
4 class Complex {
5     private:
6         int real, imag;
7     public:
8         Complex(int r = 0, int i = 0) {
9             real = r;
10            imag = i;
11        }
12
13        // This is automatically called when '+' is used with
14        // between two Complex objects
15        Complex operator + (Complex &obj) {
16            Complex res;
17            res.real = real + obj.real;
18            res.imag = imag + obj.imag;
19            return res;
20        }
21        void print() {
22            cout << real << " + i" << imag << endl;
23        }
24 };
25
26 int main()
27 {
28     Complex c1(10, 5);
29     Complex c2(2, 4);
30     Complex c3 = c1 + c2; // An example call to "operator+"
31     c3.print();
32 }
33
34
35
36 Output:
37
38 12 + i9
```