Welcome to C++

CSE 225 - Data Structures and Algorithms

Md. Mahfuzur Rahman
ECE Department
North South University

1 Using a Class [Case A]

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

class MyInfo
{
    private:
        // data hidden from outside world
        int x;

    public:
        // function to set value of variable x
        void set(int a)
        {
            x = a;
        }

        // function to return value of variable x
        int get()
        {
            return x;
        }
};

// main function
int main()
{
    MyInfo obj;

    obj.set(5);

    cout<<obj.get();  //cout is used inspite of printf( )
    return 0;
}



output:

5
```

2 Using a Class [Case B]

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

class MyInfo
{
    private:
        // data hidden from outside world
        int x;

    public:
        // function to set value of variable x
        void set(int a);

        // function to return value of variable x
        int get();
};

void MyInfo::set(int a)
        {
            x = a;
        }

int MyInfo::get()
        {
            return x;
        }


// main function
int main()
{
    MyInfo obj;

    obj.set(5);

    cout<<obj.get();  //cout is used inspite of printf( )
    return 0;
}




output:

5
```

3 Using a Class [Case C]

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

class Calculator  {

  private:
    int num1;
    int num2;

  public:
    Calculator(int n1, int n2){  //Constructor
      num1 = n1;
      num2 = n2;
    }

    int add() {
        return num1 + num2;
    }

    int multiply() {
        return num1 * num2;
    }

    void display() {
        cout<<"Numbers: "<<num1<<" and "<<num2;
        cout<<"\n\t Add:"<<add();
        cout<<"\n\t Multiply:"<<multiply();
        cout<<endl<<endl;
    }
};

int main()  {
   Calculator a(2,4);
   Calculator b(5,3);

   a.display();
   b.display();
   return 0;
}

Output:

Numbers: 2 and 4
   Add:6
   Multiply:8

Numbers: 5 and 3
   Add:8
   Multiply:15
```

4 Using a Class with Template [Case A]

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

template<class T>
class Calculator {

  private:
    T num1;
    T num2;

  public:
    Calculator(T n1, T n2){
      num1 = n1;
      num2 = n2;
    }

    T add() {
        return num1 + num2;
    }

    T multiply() {
        return num1 * num2;
    }

    void display() {
        cout<<"Numbers: "<<num1<<" and "<<num2;
        cout<<"\n\t Add:"<<add();
        cout<<"\n\t Multiply:"<<multiply();
        cout<<endl<<endl;
    }
};

int main() {
    Calculator<int> a(2,4);
    Calculator<float> b(2.2,4.1);

    a.display();
    b.display();
    return 0;
}

Output:

Numbers: 2 and 4
    Add:6
    Multiply:8

Numbers: 2.2 and 4.1
    Add:6.3
    Multiply:9.02
```

5 Using a Class with Template [Case B]

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

template<class T>
class Calculator  {

  private:
    T num1;
    T num2;

  public:
    Calculator(T n1, T n2);
    T add();
    T multiply();
    void display();

};

template<class T>
Calculator<T>::Calculator(T n1, T n2){
  num1 = n1;
  num2 = n2;
}

template<class T>
T Calculator<T>::add() {
    return num1 + num2;
}

template<class T>
T Calculator<T>::multiply() {
    return num1 * num2;
}

template<class T>
void Calculator<T>::display() {
    cout<<"Numbers: "<<num1<<" and "<<num2;
    cout<<"\n\t Add:"<<add();
    cout<<"\n\t Multiply:"<<multiply();
    cout<<endl<<endl;
}


int main()  {
   Calculator<int> a(2,4);
   Calculator<float> b(2.2,4.1);

   a.display();
   b.display();
   return 0;
}
```

6 Using a Class within Class

```cpp
#include <iostream>

using namespace std;

class Box {
    private:
      double length;      // Length of a box
      double breadth;     // Breadth of a box
      double height;      // Height of a box

    public:
      // Constructor definition
      Box(double l = 2.0, double b = 2.0, double h = 2.0) {
         cout <<"Constructor called." << endl;
         length = l;
         breadth = b;
         height = h;
      }
      double Volume() {
         return length * breadth * height;
      }
      int compare(Box box) {
         return this->Volume() > box.Volume();
      }


};

int main(void) {
   Box Box1(3.3, 1.2, 1.5);     // Declare box1
   Box Box2(8.5, 6.0, 2.0);     // Declare box2

   if(Box1.compare(Box2)) {
      cout << "Box2 is smaller than Box1" <<endl;
   } else {
      cout << "Box2 is equal to or larger than Box1" <<endl;
   }

   return 0;
}
```

7 Using a Class with Operator Overloading

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

class Complex {
    private:
      int real, imag;
    public:
       Complex(int r = 0, int i =0)  {
          real = r;
          imag = i;
        }

               // This is automatically called when '+' is used with
               // between two Complex objects
        Complex operator + (Complex &obj) {
             Complex res;
             res.real = real + obj.real;
             res.imag = imag + obj.imag;
             return res;
        }
        void print() {
           cout << real << " + i" << imag << endl;
        }
};

int main()
{
    Complex c1(10, 5);
    Complex c2(2, 4);
    Complex c3 = c1 + c2; // An example call to "operator+"
    c3.print();
}



Output:

12 + i9
```