OOPs assignment 2

Submitted by: Mehma Kaur Chadha

Roll Number 1024150080

Batch:2o13

Q1. create a structure in C++ containing the details of Students as details below and a main function to execute the structure.

Data Members(properties):

Name Roll No Degree Hostel

Current CGPA

Member Function(behavior):

addDetails(); updateDetails(); updateCGPA(); updateHostel(); displaydetails();

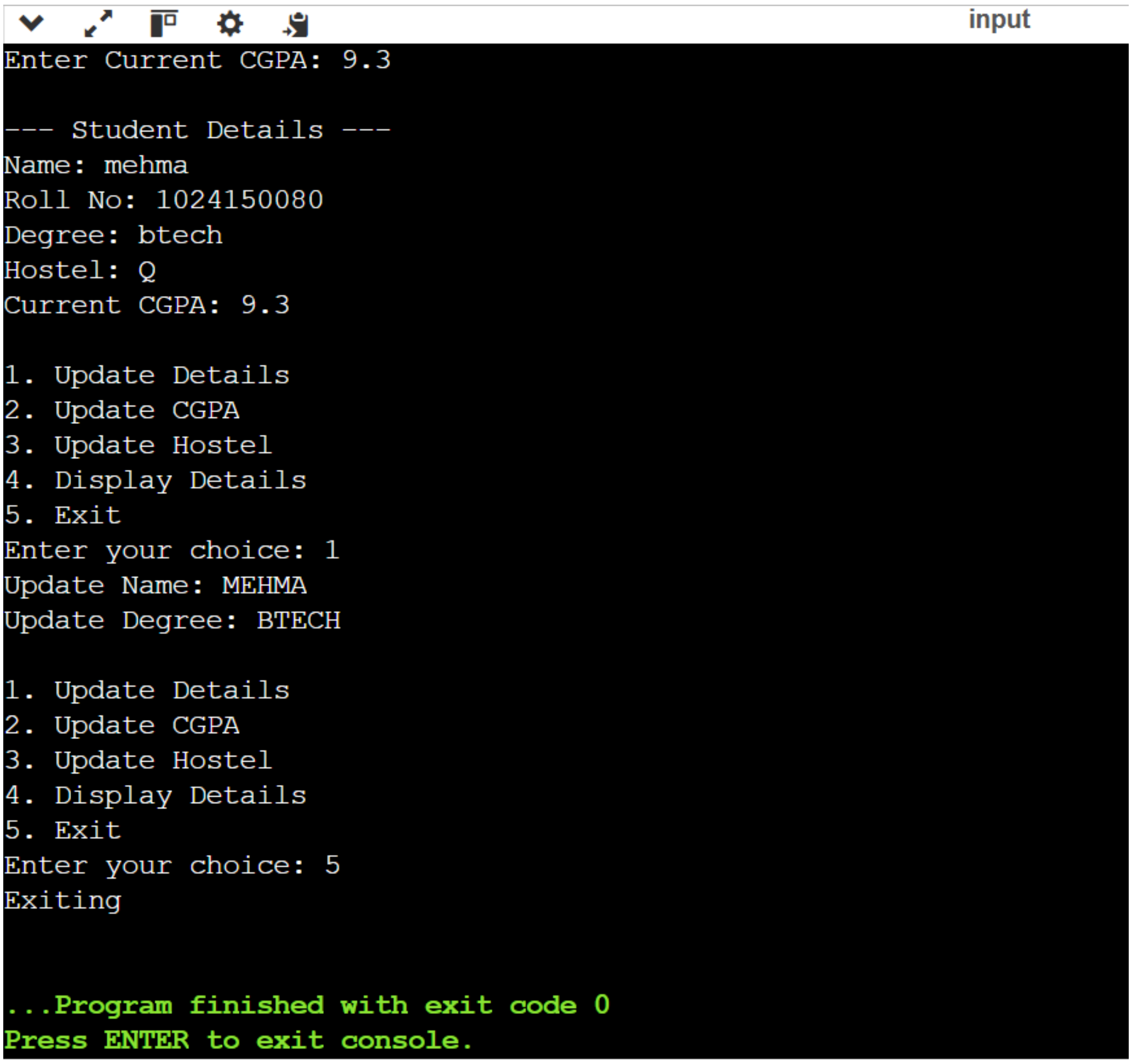
CODE:

```
#include <iostream>

#include <string>

using namespace std;


struct Student

{

   string name;

   int rollNo;

   string degree;

   string hostel;

   float currentCGPA;


   void addDetails()

   {

      cout << "Enter Name: ";

      getline(cin, name);
```

```
    cout << "Enter Roll No: ";
    cin >> rollNo;
    cin.ignore();

    cout << "Enter Degree: ";
    getline(cin, degree);

    cout << "Enter Hostel: ";
    getline(cin, hostel);

    cout << "Enter Current CGPA: ";
    cin >> currentCGPA;
    cin.ignore();
}

void updateDetails()
{
    cout << "Update Name: ";
    getline(cin, name);

    cout << "Update Degree: ";
    getline(cin, degree);
}

void updateCGPA()
{
    cout << "Enter New CGPA: ";
    cin >> currentCGPA;
    cin.ignore();
}
```

```
    void updateHostel()
    {
        cout << "Enter New Hostel: ";
        getline(cin, hostel);
    }

    void displayDetails()
    {
        cout << "\n--- Student Details ---\n";
        cout << "Name: " << name << endl;
        cout << "Roll No: " << rollNo << endl;
        cout << "Degree: " << degree << endl;
        cout << "Hostel: " << hostel << endl;
        cout << "Current CGPA: " << currentCGPA << endl;
    }
};

int main()
{
    Student s;

    s.addDetails();
    s.displayDetails();

    int choice;

    do
    {
        cout << "\n1. Update Details\n";
        cout << "2. Update CGPA\n";
        cout << "3. Update Hostel\n";
        cout << "4. Display Details\n";
        cout << "5. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;
        cin.ignore();

        switch(choice)
        {
            case 1: s.updateDetails(); break;
            case 2: s.updateCGPA(); break;
            case 3: s.updateHostel(); break;
            case 4: s.displayDetails(); break;
            case 5: cout << "Exiting\n"; break;
            default: cout << "Invalid choice!\n";
        }

    } while(choice != 5);

    return 0;
}
```

OUTPUT:



```
Enter Current CGPA: 9.3

--- Student Details ---
Name: mehma
Roll No: 1024150080
Degree: btech
Hostel: Q
Current CGPA: 9.3

1. Update Details
2. Update CGPA
3. Update Hostel
4. Display Details
5. Exit
Enter your choice: 1
Update Name: MEHMA
Update Degree: BTECH

1. Update Details
2. Update CGPA
3. Update Hostel
4. Display Details
5. Exit
Enter your choice: 5
Exiting


...Program finished with exit code 0
Press ENTER to exit console.
```

Q2. Differentiate between private and public access/scope. Perform the question no. 1 with class instead of structure with having the data members private and some member functions in private scope and some in public scope

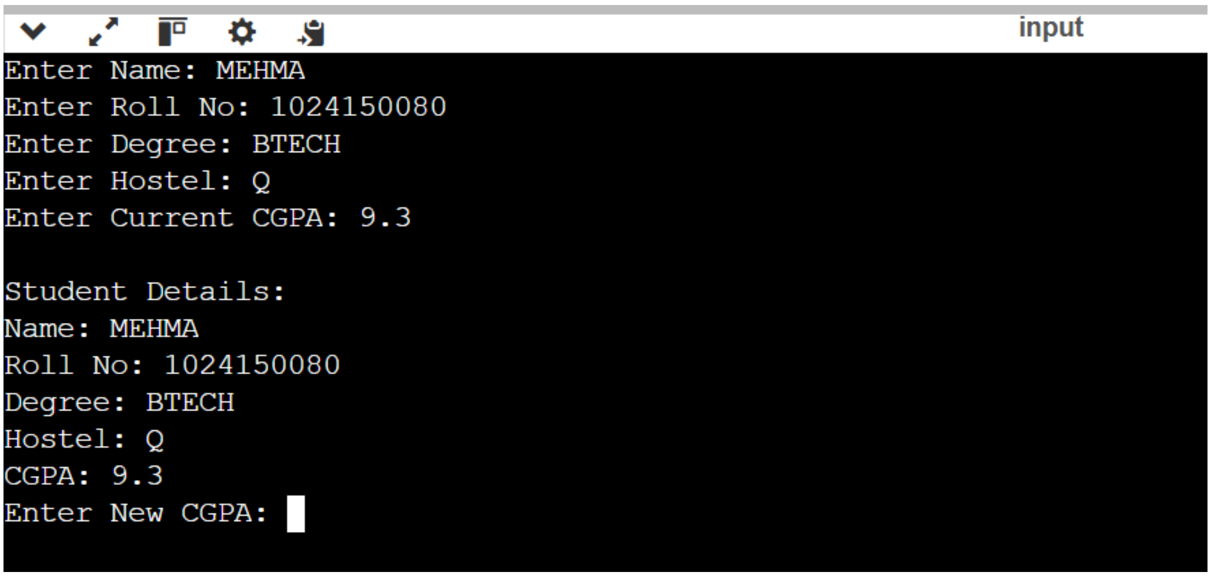
CODE:

```
#include <iostream>

#include <string>

using namespace std;


class Student

{
```

```
private:
    string name;
    int rollNo;
    string degree;
    string hostel;
    float currentCGPA;

public:
    void addDetails()
    {
        cout << "Enter Name: ";
        getline(cin, name);

        cout << "Enter Roll No: ";
        cin >> rollNo;
        cin.ignore();

        cout << "Enter Degree: ";
        getline(cin, degree);

        cout << "Enter Hostel: ";
        getline(cin, hostel);

        cout << "Enter Current CGPA: ";
        cin >> currentCGPA;
    }

    void updateDetails()
    {
        cout << "Enter New Name: ";
        cin.ignore();
        getline(cin, name);

        cout << "Enter New Degree: ";
        getline(cin, degree);
    }

    void updateCGPA()
    {
        cout << "Enter New CGPA: ";
        cin >> currentCGPA;
    }

    void updateHostel()
    {
        cout << "Enter New Hostel: ";
        cin.ignore();
        getline(cin, hostel);
    }

    void displayDetails()
    {
        cout << "\nStudent Details:\n";
        cout << "Name: " << name << endl;
        cout << "Roll No: " << rollNo << endl;
        cout << "Degree: " << degree << endl;
        cout << "Hostel: " << hostel << endl;
        cout << "CGPA: " << currentCGPA << endl;
    }
};

int main()
```

```
{
   Student s;

   s.addDetails();
   s.displayDetails();

   s.updateCGPA();
   s.updateHostel();

   s.displayDetails();

   return 0;
}
```

OUTPUT:



```
Enter Name: MEHMA
Enter Roll No: 1024150080
Enter Degree: BTECH
Enter Hostel: Q
Enter Current CGPA: 9.3

Student Details:
Name: MEHMA
Roll No: 1024150080
Degree: BTECH
Hostel: Q
CGPA: 9.3
Enter New CGPA: 
```

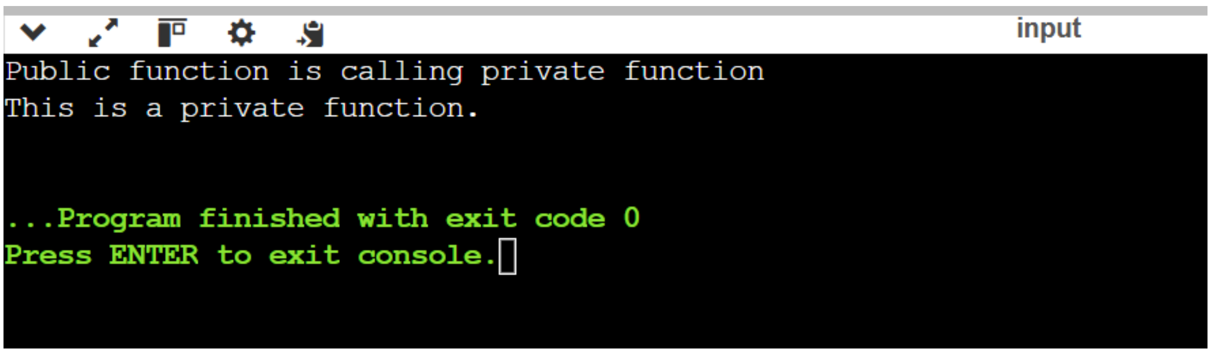
Q3. Create a code snippet that illustrates the following: Calling of private member functions inside public member function

CODE:

```
#include <iostream>
```

```
using namespace std;

class Demo
{
private:
   void privateFunction()
   {
      cout << "This is a private function." << endl;
   }

public:
   void publicFunction()
   {
      cout << "Public function is calling private function" << endl;
      privateFunction();
   }
};

int main()
{
   Demo obj;
   obj.publicFunction();
}
```

OUTPUT:



```
Public function is calling private function
This is a private function.


...Program finished with exit code 0
Press ENTER to exit console.
```

1. *Q4. Define a class Rectangle with variables width and height of integer typwe along with following:*

   (a) *void getdata() to initialize object values*

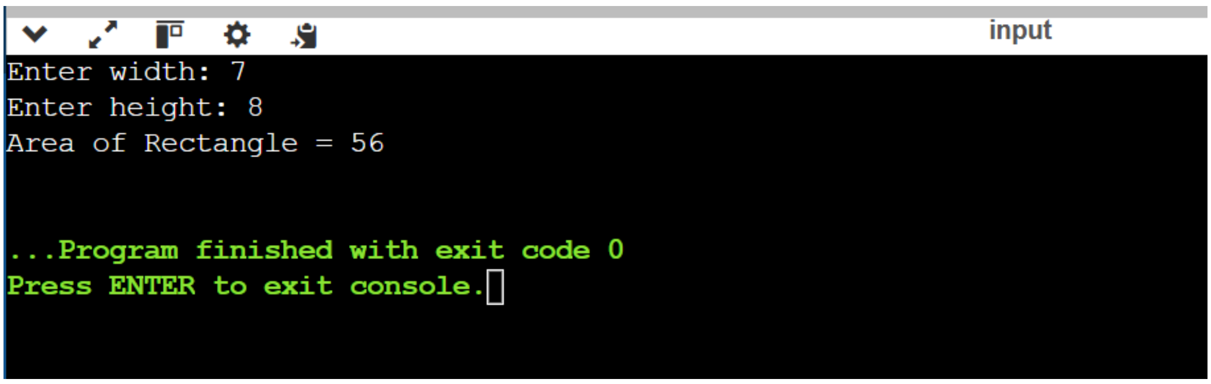
void calculatearea() to calculate and display the area.

CODE:

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

class Rectangle
{
private:
   int width;
   int height;

public:
   void getdata()
   {
      cout << "Enter width: ";
      cin >> width;

      cout << "Enter height: ";
      cin >> height;
   }

   void calculatearea()
   {
      int area = width * height;
      cout << "Area of Rectangle = " << area << endl;
   }
};
```

```
int main()
{
   Rectangle r;

   r.getdata();
   r.calculatearea();
}
```

OUTPUT:



```
Enter width: 7
Enter height: 8
Area of Rectangle = 56


...Program finished with exit code 0
Press ENTER to exit console.
```

Q5. Define a class Complex with variables real and imaginary along with following: (a)void setComplex (float, float) to initialize object values.

(b) void displayComplex() to show the complex number

(c) Pass and return objects to calculate sum of two complex numbers. Display the sum.

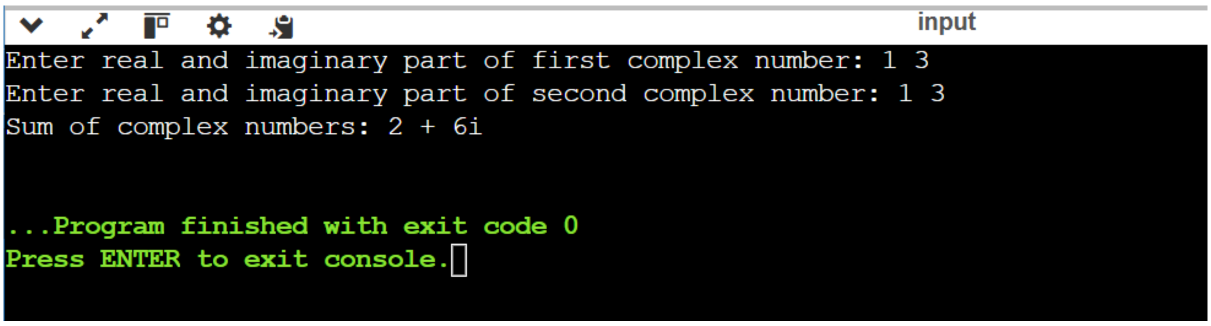
CODE:

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

class Complex
{
private:
   float real;
   float imaginary;
```

```
public:
   void setComplex(float r, float i)
   {
      real = r;
      imaginary = i;
   }

   void displayComplex()
   {
      cout << real << " + " << imaginary << "i" << endl;
   }

   Complex addComplex(Complex c)
   {
      Complex temp;
      temp.real = real + c.real;
      temp.imaginary = imaginary + c.imaginary;
      return temp;
   }
};

int main()
{
   Complex c1, c2, sum;
   float r, i;

   cout << "Enter real and imaginary part of first complex number: ";
   cin >> r >> i;
   c1.setComplex(r, i);

   cout << "Enter real and imaginary part of second complex number: ";
```

```
    cin >> r >> i;
    c2.setComplex(r, i);

    sum = c1.addComplex(c2);

    cout << "Sum of complex numbers: ";
    sum.displayComplex();

    return 0;
}
```

OUTPUT:



```
Enter real and imaginary part of first complex number: 1 3
Enter real and imaginary part of second complex number: 1 3
Sum of complex numbers: 2 + 6i


...Program finished with exit code 0
Press ENTER to exit console.
```

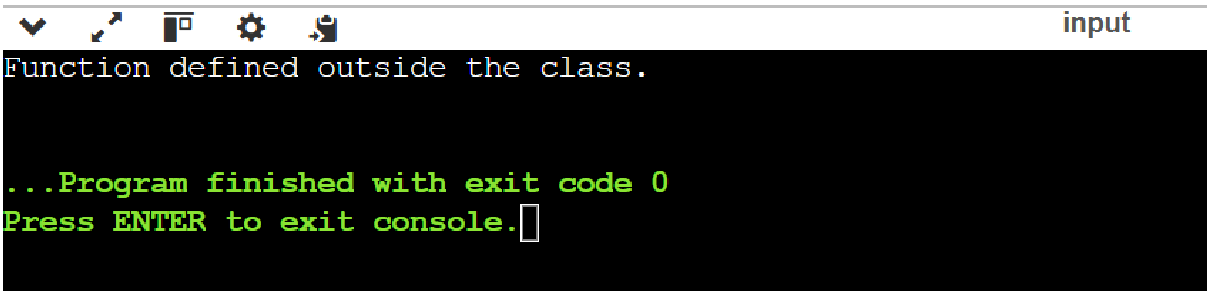
~~1~~✓. Q6. Implement *scope resolution operator* : : for the following uses:

(a)Class functions defined outside the class

Code:

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

class Sample
{
public:
    void display();   // Function declaration
};

// Function defined outside using scope resolution operator
void Sample::display()
{
    cout << "Function defined outside the class." << endl;
}

int main()
{
    Sample obj;
    obj.display();
    return 0;
}
```

OUTPUT:



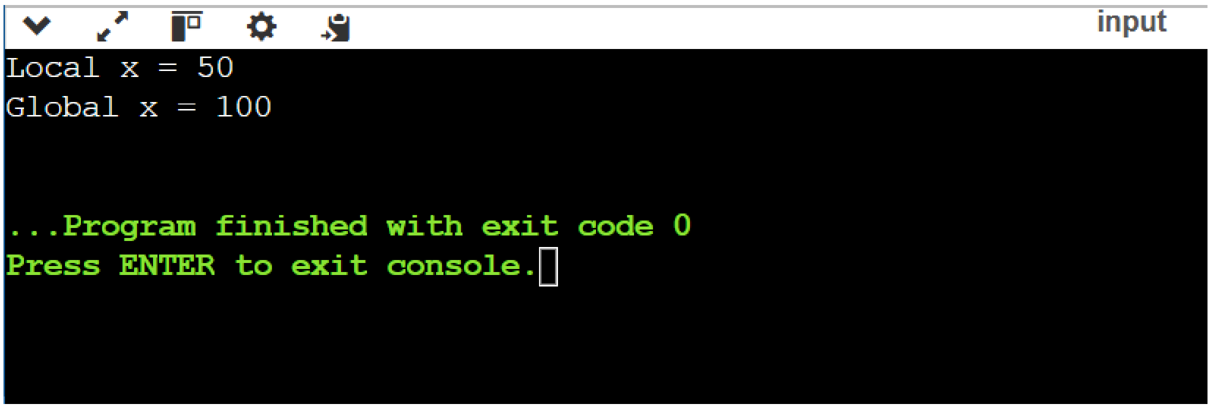
```
Function defined outside the class.


...Program finished with exit code 0
Press ENTER to exit console.
```

(b) Access a global variable with same name as a local variable

CODE:

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

int x = 100;

int main()
{
    int x = 50;

    cout << "Local x = " << x << endl;
    cout << "Global x = " << ::x << endl;

    return 0;
}
```

OUTPUT:



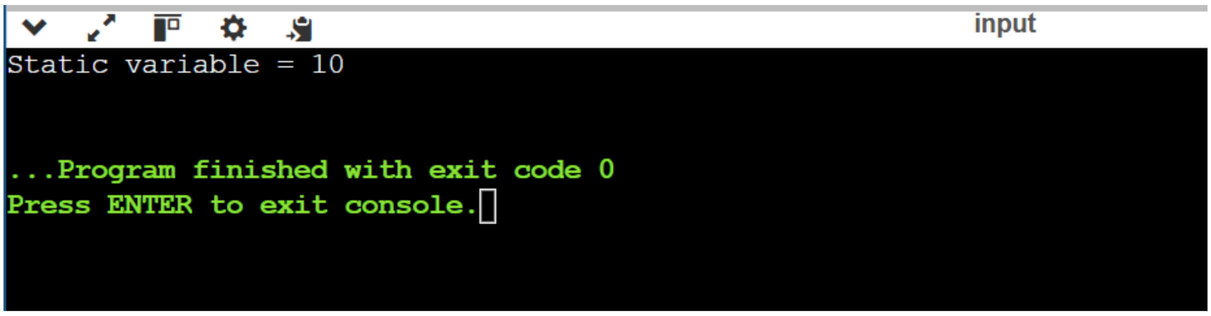
```
Local x = 50
Global x = 100


...Program finished with exit code 0
Press ENTER to exit console.
```

(C) Access a static variables

Code:

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

class Demo
{
public:
    static int count;
};
int Demo::count = 10;

int main()
{
    cout << "Static variable = " << Demo::count << endl;
    return 0;
}
```

OUTPUT:



```
Static variable = 10


...Program finished with exit code 0
Press ENTER to exit console.
```

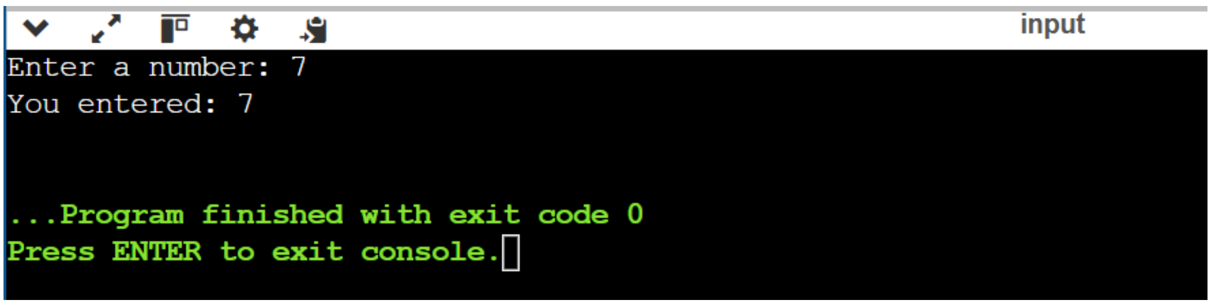
(D) Use inbuilt libraries (cin cout with scope resolution operator)

Code:

#include <iostream>

```
int main()
{
   int num;

   std::cout << "Enter a number: ";
   std::cin >> num;

   std::cout << "You entered: " << num << std::endl;

}
```

Output:



```
Enter a number: 7
You entered: 7


...Program finished with exit code 0
Press ENTER to exit console.
```

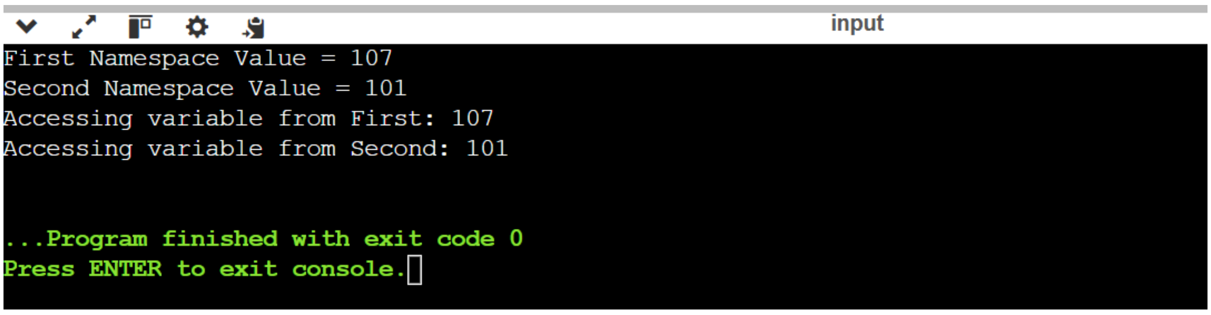
Q7 Create a code to implement the *namespace* and use similar variables and functions defined in different code sections.

Code:

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

namespace First
{
   int value = 107;
```

```
    void display()
    {
        cout << "First Namespace Value = " << value << endl;
    }
}

namespace Second
{
    int value = 101;

    void display()
    {
        cout << "Second Namespace Value = " << value << endl;
    }
}

int main()
{
    First::display();
    Second::display();

    cout << "Accessing variable from First: " << First::value << endl;
    cout << "Accessing variable from Second: " << Second::value << endl;

}
```

Output:



```
First Namespace Value = 107
Second Namespace Value = 101
Accessing variable from First: 107
Accessing variable from Second: 101


...Program finished with exit code 0
Press ENTER to exit console.
```