Don Bosco Institute of Technology, Kurla(W)
Department of Electronics and Tele-Communication Engineering
ECL304 - Skill Lab: C++ and Java Programming
Sem III
2021-22

Lab Number:	**7**
Student Name:	**Suraj Kumar Das**
Roll No :	**42**

Title:

1. To write a program to demonstrate friend function in C++.
2. To write a program to demonstrate friend class in C++.

Learning Objective:

- Students will be able to implement friend function and friend classes in C++.

Learning Outcome:

- To understand how to use the private members using friend function and friend class.

Course Outcome:

ECL304.6	Percept the Utility and applicability of OOP

Theory:

- **Explain in details about access specifiers: public, private and protected.**
  There are 3 access specifiers for a class/struct/Union in C++. These access specifiers define how the members of the class can be accessed. Of course, any member of a class is accessible within that class(Inside any member function of that same class). Moving ahead to type of access specifiers, they are:

  1.Public - The members declared as Public are accessible from outside the Class through an object of the class.

  2.Protected - The members declared as Protected are accessible from outside the class **BUT** only in a class derived from it.

3.Private - These members are only accessible from within the class. No outside Access is allowed.

- **Explain about friend function and friend classes in C++.**

Friend Function
A friend function is a function that is specified outside a class but has the ability to access the class members' protected and private data. A friend can be a member's function, function template, or function, or a class or class template, in which case the entire class and all of its members are friends

Friend Class
Friend Class A friend class can access private and protected members of other class in which it is declared as friend. It is sometimes useful to allow a particular class to access private members of other class. For example, a LinkedList class may be allowed to access private members of Node.

1. **To write a program to demonstrate friend function in C++.**

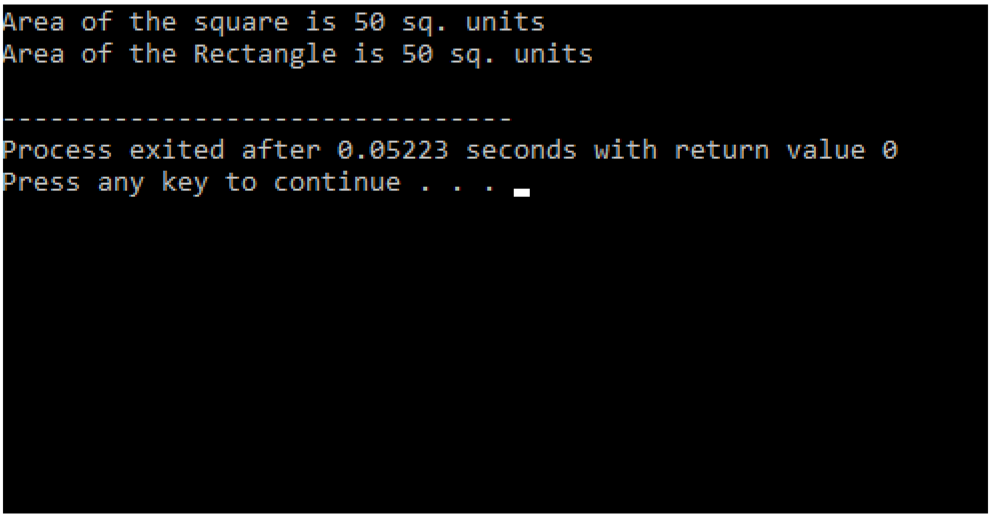
Algorithm :	Step 1: Create a class 'class1' and inside that class put the various attributes of the class (in private), calculation method for calculating the area of the square and rectangle and also declare the friend function inside this class. Step 2: The body of the friend function is outside of the class1. Step 3: Using the friend function we will access the private data attributes of the class and will input the values of length and breadth as 10 using call by reference method. Step 4: In the main function pass the object a in the method input values. Step 5: In the main function create the object of the class1 and using that object call the calculation method for getting the output.

Program:

```
//To write a program to demonstrate friend function in C++.


#include <iostream>

using namespace std;

class class1{

  int length , breadth ;

  float square_area , rectangle_area;

  public:

          class1(){


                  length = 0 , breadth = 0 ;


            }


            void calculation(){

                   square_area=length*breadth;

                   rectangle_area=length*breadth;

          cout<<"Area of the square is "<<square_area<<" sq.
units"<<"\n"<<"Area of the Rectangle is "<<rectangle_area<<" sq.
units"<<"\n";

}


friend void input_values(class1 &a);  //friend function


};

void input_values(class1 &a){  /

          a.length=5;

          a.breadth=10;

}
```

	int main()
	{
	    class1 a;
	    input_values(a);
	    a.calculation();
	    return 0;
	}

Input given:	Length=4
	Breath=5

Output Screenshot:	 Area of the square is 50 sq. units Area of the Rectangle is 50 sq. units -------------------------------- Process exited after 0.05223 seconds with return value 0 Press any key to continue . . .

2.To write a program to demonstrate friend class in C++.

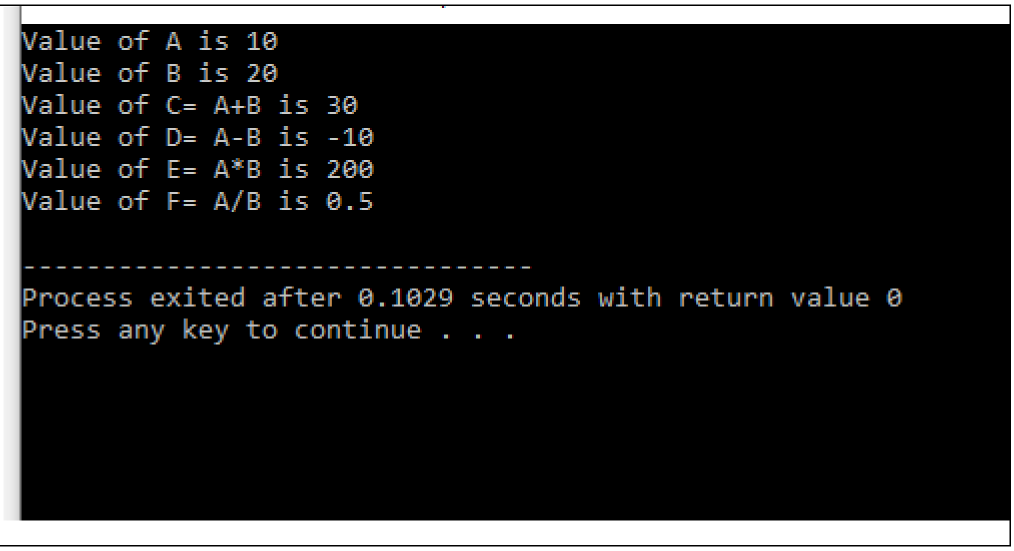
Algorithm :	Step 1: Create a class named first_class and declare its attributes in private form as int A and int B and initialize them for values 100 and 200 respectively. Step 2: Create different variables, each representing a mathematical operation such as multiplication , division, addition and subtraction of the two numbers A and B. Step 3: Creating the friend class of the first_class name ‘second_class’ and declaring its body outside of the class. Step 4: The body of the friend class include the display_output method through which the output is to be displayed using the object of the class first_class. Step 5: In the main function we create the objects of both the classes and we call the method for getting the output

Program:	/* To write a program to demonstrate friend class in C++. */
	
	#include <iostream>
	using namespace std;
	class first_class{
	private:
	   int A = 10;
	   int B = 20;
	   int C = A+B;
	   int D = A-B;
	   int E = A*B;
	   float F = (float)A/B ;
	
	public: //access specifier

```
  friend class second_class;  //declaring the friend class
};


class second_class {
public:  //access specifier
  void display_output(first_class object1){


    cout<<"Value of A is "<<object1.A<<endl;   //performinng various mathematical operations sucha as multiplication , addition, division ,subtraction
    cout<<"Value of B is "<<object1.B<<endl;
    cout<<"Value of C= A+B is "<<object1.C<<endl;
    cout<<"Value of D= A-B is "<<object1.D<<endl;
    cout<<"Value of E= A*B is "<<object1.E<<endl;
    cout<<"Value of F= A/B is "<<object1.F<<endl;
  }
};
int main() {
  second_class object1; //object creation
  first_class object2;
  object1.display_output(object2);  //calling the method using the object of the class
  return 0;
}
```

Input given:	A=10
	B=20

Output Screenshot :



```
Value of A is 10
Value of B is 20
Value of C= A+B is 30
Value of D= A-B is -10
Value of E= A*B is 200
Value of F= A/B is 0.5

--------------------------------
Process exited after 0.1029 seconds with return value 0
Press any key to continue . . .
```