Course code: CSC-284
Lab Report: 07

Submitted To
Tanzina Tasnim Bithi

Department of Computer, Science and Engineering

Submitted By
Name: Md.Pranto Ali
ID: 23303101
Program: BCSE
Section: A
Semester: Fall 2024

Submission date:

Teacher signature

1. Sum of number using function overloading:

main.cpp

Share Run

```cpp
#include <iostream>
using namespace std;
class adition_numbers {
    public:
    void sum(int a, int b) {
        cout << a + b << endl;
    };
    void sum(double a, double b, double c) {
        cout << a + b + c << endl;
    };
    void sum(float a, float b, float c) {
        cout << a + b + c << endl;
    };
};
int main() {
    cout << "Name: Md.Pranto Ali" << "Id: 23303101" << endl;
    adition_numbers obj;
    cout << "Call First Function:" << endl;
    obj.sum(5, 6);
    cout << "Call Secound Function:" << endl;
    obj.sum(5.5, 6.6, 7.7);
    cout << "Call Third Function:" << endl;
    obj.sum(8.5, 2.6, 88.7);
    return 0;
}
```

Output Clear

```
Name: Md.Pranto AliId: 23303101
Call First Function:
11
Call Secound Function:
19.8
Call Third Function:
99.8


=== Code Execution Successful ===
```

2. Finding Area of Circle , Rectangle and square Using Polymorphism:

main.cpp

Share Run

```cpp
#include<iostream>
using namespace std;
class Area {
    public:
    void find_area(double r) {
        cout << "Area of Circle: " << 3.14 * r * r << endl;
    };
    void find_area(double l, double b) {
        cout << "Area of Rectangle: " << l * b << endl;
    };
    void find_area(double s, double s1, double s2) {
        cout << "Area of Square: " << s * s1 * s2 << endl;
    };
};
int main() {
    cout << "Name: Md.Pranto Ali" << "Id: 23303101" << endl;
    Area obj;
    cout << "Call First Function:" << endl;
    obj.find_area(5);
    cout << "Call Secound Function:" << endl;
    obj.find_area(5, 6);
    cout << "Call Third Function:" << endl;
    obj.find_area(5, 5, 5);
    return 0;
};
```

Output Clear

```
Name: Md.Pranto AliId: 23303101
Call First Function:
Area of Circle: 78.5
Call Secound Function:
Area of Rectangle: 30
Call Third Function:
Area of Square: 125


=== Code Execution Successful ===
```

3. Operator Overloading Unary Minus:

main.cpp

Share Run

```cpp
#include <iostream>
using namespace std;
class Complex{
    private:
        int a, b;
    public:
        Complex(int x, int y){
            a = x;
            b = y;
        }
        void display(){
            cout << "Complex number: " << a << " + " << b << "i"
                << endl;
        }
        Complex operator -(){
            return Complex(-a, -b);
        }
};
int main() {
    cout << "Name: Md.Pranto Ali" << "Id: 23303101" << endl;
    Complex c1(2, 3);
    Complex c2 = -c1;
    c2.display();
    return 0;
}
```

Output Clear

```
Name: Md.Pranto AliId: 23303101
Complex number: -2 + -3i


=== Code Execution Successful ===
```

4. Operator Overloading Increment:

main.cpp

Share Run

```cpp
#include <iostream>
using namespace std;
class Complex{
    private:
        int a, b;
    public:
        Complex(int x, int y){
            a = x;
            b = y;
        }
        void display(){
            cout << "Complex number: " << a << " + " << b << "i"
                << endl;
        }
        Complex operator ++(){
            a++;
            b++;
            return Complex(a, b);
        }
};
int main() {
    cout << "Name: Md.Pranto Ali" << "Id: 23303101" << endl;
    Complex c1(2, 3);
    Complex c2 = ++c1; // c2 = c1.operator++();
    c2.display();
    return 0;
};
```

Output Clear

```
Name: Md.Pranto AliId: 23303101
Complex number: 3 + 4i


=== Code Execution Successful ===
```

5. Operator Overloading Decrement:

main.cpp | Share | Run | Output | Clear

```cpp
#include <iostream>
using namespace std;
class Complex{
    private:
        int a, b;
    public:
        Complex(int x, int y){
            a = x;
            b = y;
        }
        void display(){
            cout << "Complex number: " << a << " + " << b << "i"
                << endl;
        }
        Complex operator --(){
            a--;
            b--;
            return Complex(a, b);
        }
};
int main() {
    cout << "Name: Md.Pranto Ali" << "Id: 23303101" << endl;
    Complex c1(2, 3);
    Complex c2 = --c1; // c2 = c1.operator--();
    c2.display();
    return 0;
};

```

```
Name: Md.Pranto AliId: 23303101
Complex number: 1 + 2i


=== Code Execution Successful ===
```

6. Overloading Binary Operators:

main.cpp Share Run

```cpp
#include <iostream>
using namespace std;
class Complex {
    private:
        int a, b;
    public:
    Complex(int x, int y) {
        a = x;
        b = y;
    }
    void display() {
        cout << "Complex number: " << a << " + " << b << "i" <<
            endl;
    }
    Complex operator +(Complex c) {
        return Complex(a + c.a, b + c.b);
    }
};
int main() {
    cout << "Name: Md.Pranto Ali" << "Id: 23303101" << endl;
        Complex c1(2, 3);
        Complex c2(3, 4);
        Complex c3 = c1 + c2; // c3 = c1.operator+(c2);
        c3.display();
        return 0;
}
```

Output Clear

```
Name: Md.Pranto AliId: 23303101
Complex number: 5 + 7i


=== Code Execution Successful ===
```

7. Overloading Operators addition using Friend:

main.cpp

Share Run

```cpp

#include <iostream>
using namespace std;
class Complex{
    private:
        int a, b;
    public:
    Complex(int x, int y){
            a = x;
            b = y;
        }
    void display(){
            cout << "Complex number: " << a << " + " << b << "i"
                << endl;
        }
    friend Complex operator +(Complex c1, Complex c2);
};
Complex operator +(Complex c1, Complex c2){
    return Complex(c1.a + c2.a, c1.b + c2.b);
}
int main() {
    cout << "Name: Md.Pranto Ali" << endl << "Id: 23303101" <<
        endl;
    Complex c1(3, 2);
    Complex c2(4, 5);
    Complex c3 = c1 + c2; // c3 = operator+(c1, c2);
    c3.display();
    return 0;
}
```

Output Clear

```
Name: Md.Pranto Ali
Id: 23303101
Complex number: 7 + 7i


=== Code Execution Successful ===
```

8. Overloading Operator Using Friend (Add,sub):

main.cpp

Share Run

```cpp
#include<iostream>
using namespace std;
class A{
    private:
    int a;
    public:
    A(){
        a=0;
    }
    void display(){
        cout<<"Value of a: "<<a<<endl;
    }
    friend void operator -(A &obj);
};
void operator -(A &obj){
    obj.a=-obj.a;
}
int main(){
    cout << "Name: Md.Pranto Ali" << endl << "Id: 23303101" <<
        endl;
    A obj;
    obj.display();
    -obj;
    obj.display();
    return 0;
}

```

Output Clear

```
Name: Md.Pranto Ali
Id: 23303101
Value of a: 0
Value of a: 0


=== Code Execution Successful ===
```