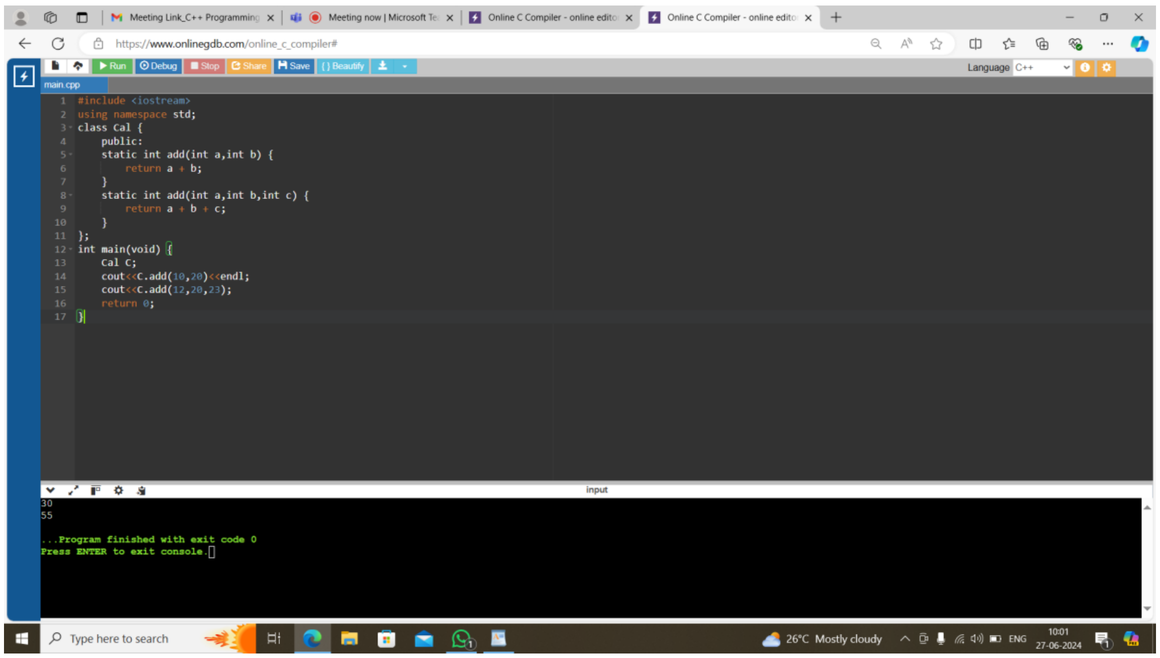
FUNCTION OVERLOADING :-

```
#include <iostream>

using namespace std;

class Cal {

    public:

    static int add(int a,int b) {

        return a + b;

    }

    static int add(int a,int b,int c) {

        return a + b + c;

    }

};

int main(void) {

    Cal C;

    cout<<C.add(10,20)<<endl;

    cout<<C.add(12,20,23);

    return 0;

}
```

OUTPUT :-



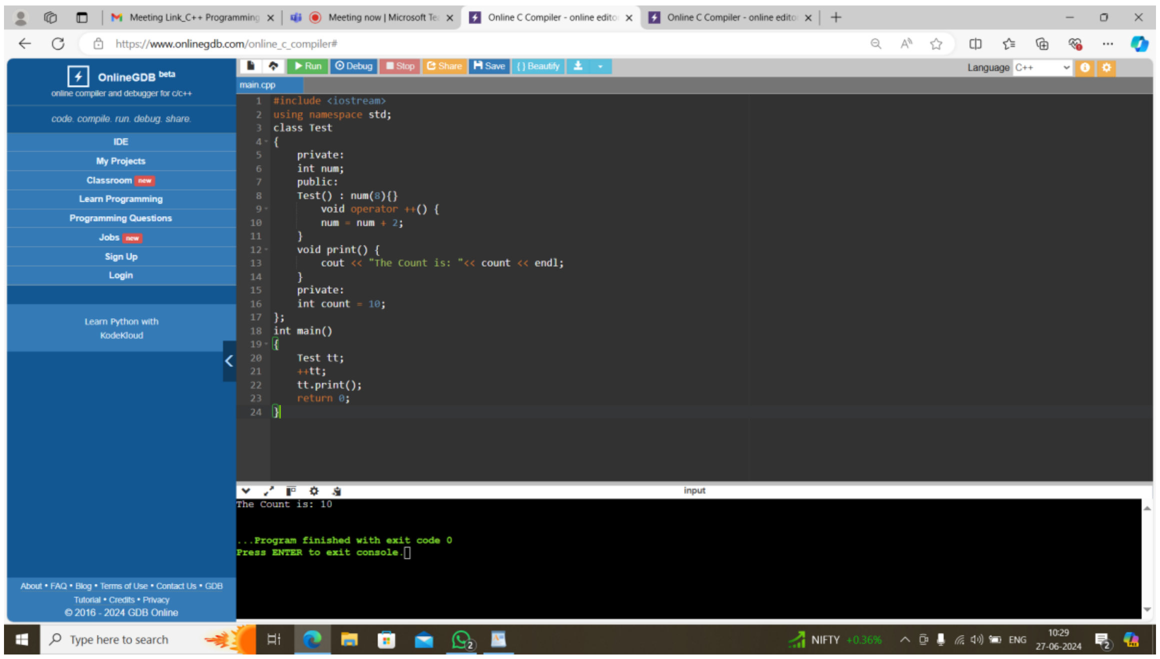
```
#include <iostream>
using namespace std;
class Cal {
    public:
    static int add(int a,int b) {
        return a + b;
    }
    static int add(int a,int b,int c) {
        return a + b + c;
    }
};
int main(void) {
    Cal C;
    cout<<C.add(10,20)<<endl;
    cout<<C.add(12,20,23);
    return 0;
}
```

```
30
55

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

PROGRAM TO OVERLOAD THE UNARY OPERATOR ++ :-

```
#include <iostream>

using namespace std;

class Test

{

    private:

    int num;

    public:

    Test() : num(8){}

        void operator ++() {

        num = num + 2;

    }

    void print() {

        cout << "The Count is: "<< count << endl;

    }

    private:
        int count = 10;
};
int main()
{
        Test tt;
        ++tt;
        tt.print();
        return 0;
}
```

OUTPUT :-



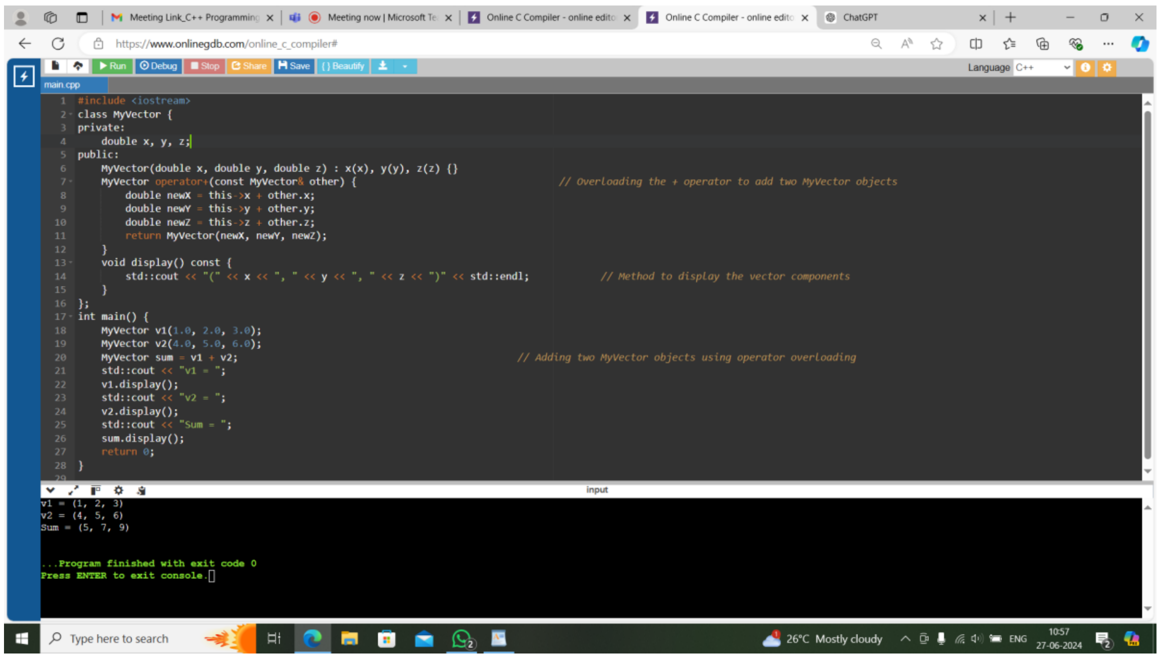
```
#include <iostream>
using namespace std;
class Test
{
    private:
    int num;
    public:
    Test() : num(8){}
        void operator ++() {
        num = num + 2;
    }
    void print() {
        cout << "The Count is: "<< count << endl;
    }
    private:
    int count = 10;
};
int main()
{
    Test tt;
    ++tt;
    tt.print();
    return 0;
}
```

```
The Count is: 10


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

OPERATOR OVERLOADING :-

```
  #include <iostream>
class MyVector {
private:
        double x, y, z;
public:

        MyVector(double x, double y, double z) : x(x), y(y), z(z) {}

        MyVector operator+(const MyVector& other) {                                                    //
Overloading the + operator to add two MyVector objects

                double newX = this->x + other.x;

                double newY = this->y + other.y;

                double newZ = this->z + other.z;

                return MyVector(newX, newY, newZ);

        }

        void display() const {

                std::cout << "(" << x << ", " << y << ", " << z << ")" << std::endl;              // Method to
display the vector components

        }

};

int main() {

        MyVector v1(1.0, 2.0, 3.0);

        MyVector v2(4.0, 5.0, 6.0);

        MyVector sum = v1 + v2;                                                                   // Adding two
MyVector objects using operator overloading

        std::cout << "v1 = ";

        v1.display();

        std::cout << "v2 = ";

        v2.display();

        std::cout << "Sum = ";

        sum.display();

        return 0;

}
```

OUTPUT :-



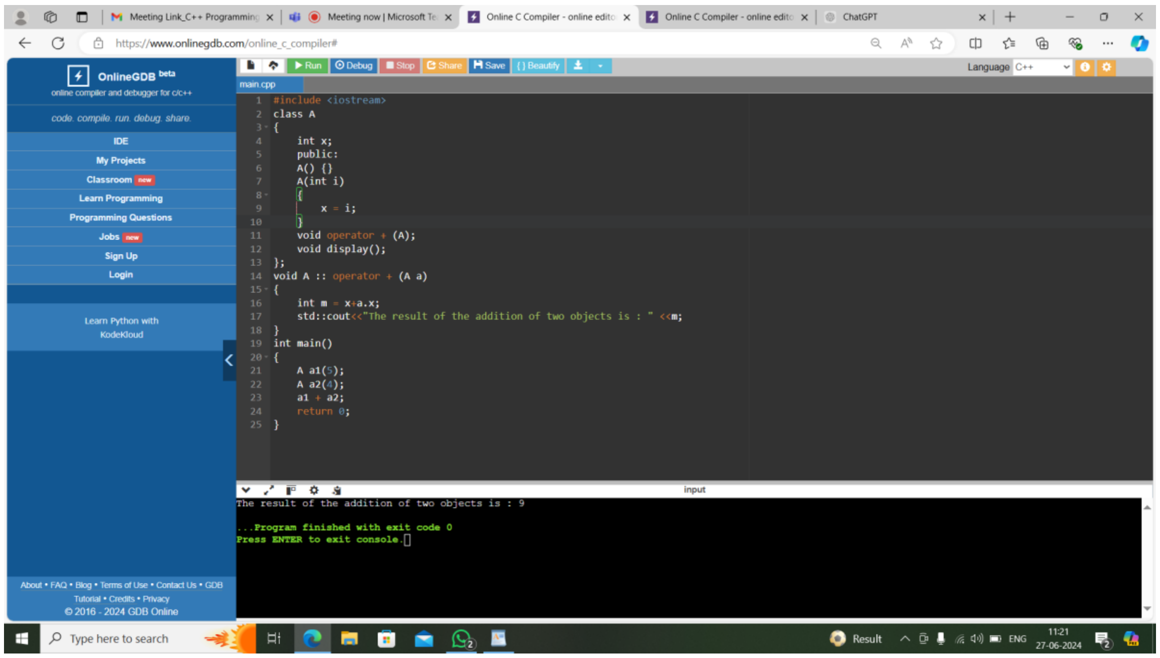
```cpp
#include <iostream>
class MyVector {
private:
    double x, y, z;
public:
    MyVector(double x, double y, double z) : x(x), y(y), z(z) {}
    MyVector operator+(const MyVector& other) {                    // Overloading the + operator to add two MyVector objects
        double newX = this->x + other.x;
        double newY = this->y + other.y;
        double newZ = this->z + other.z;
        return MyVector(newX, newY, newZ);
    }
    void display() const {
        std::cout << "(" << x << ", " << y << ", " << z << ")" << std::endl;          // Method to display the vector components
    }
};
int main() {
    MyVector v1(1.0, 2.0, 3.0);
    MyVector v2(4.0, 5.0, 6.0);
    MyVector sum = v1 + v2;                                    // Adding two MyVector objects using operator overloading
    std::cout << "v1 = ";
    v1.display();
    std::cout << "v2 = ";
    v2.display();
    std::cout << "Sum = ";
    sum.display();
    return 0;
}
```

```
v1 = (1, 2, 3)
v2 = (4, 5, 6)
Sum = (5, 7, 9)


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

//program to overload the binary operators (+) :-

```cpp
#include <iostream>

class A

{

        int x;

        public:

        A() {}

        A(int i)

        {

                x = i;

        }

        void operator + (A);

        void display();

};

void A :: operator + (A a)

{
        int m = x+a.x;
        std::cout<<"The result of the addition of two objects is : " <<m;
}
int main()
{
        A a1(5);
        A a2(4);
        a1 + a2;
        return 0;
}
```

OUTPUT :-



```
#include <iostream>
class A
{
    int x;
    public:
    A() {}
    A(int i)
    {
        x = i;
    }
    void operator + (A);
    void display();
};
void A :: operator + (A a)
{
    int m = x+a.x;
    std::cout<<"The result of the addition of two objects is : " <<m;
}
int main()
{
    A a1(5);
    A a2(4);
    a1 + a2;
    return 0;
}
```

```
The result of the addition of two objects is : 9

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

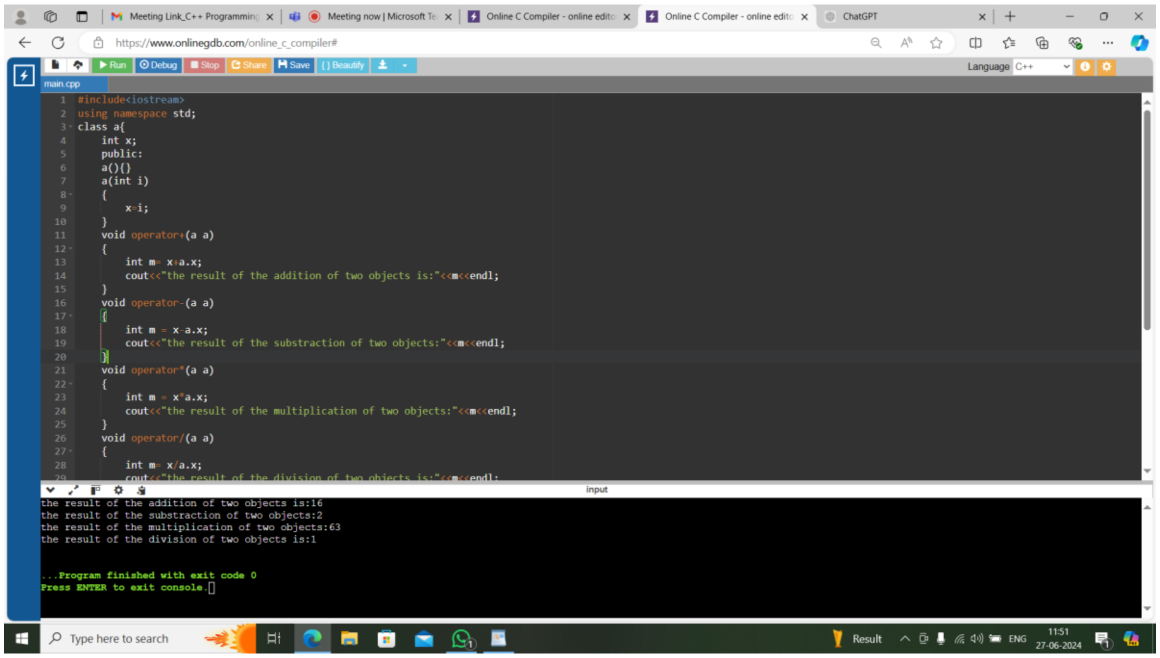
//program to overload the binary operators (+,-,*,/) :-

```
#include<iostream>
using namespace std;
class a{
        int x;
```

```
public:
a(){}
a(int i)
{
	x=i;
}
void operator+(a a)
{
	int m= x+a.x;
	cout<<"the result of the addition of two objects is:"<<m<<endl;
}
void operator-(a a)
{
	int m = x-a.x;
	cout<<"the result of the substraction of two objects:"<<m<<endl;
}
void operator*(a a)
{
	int m = x*a.x;
	cout<<"the result of the multiplication of two objects:"<<m<<endl;
}
void operator/(a a)
{
	int m= x/a.x;
	cout<<"the result of the division of two objects is:"<<m<<endl;
}
void display()
{
```

```
            cout<<x;
        }

};
int main()
{
        a a1(9);
        a a2(7);
        a1+a2;
        a1-a2;
        a1*a2;
        a1/a2;
        return 0;
}
```

OUTPUT :-



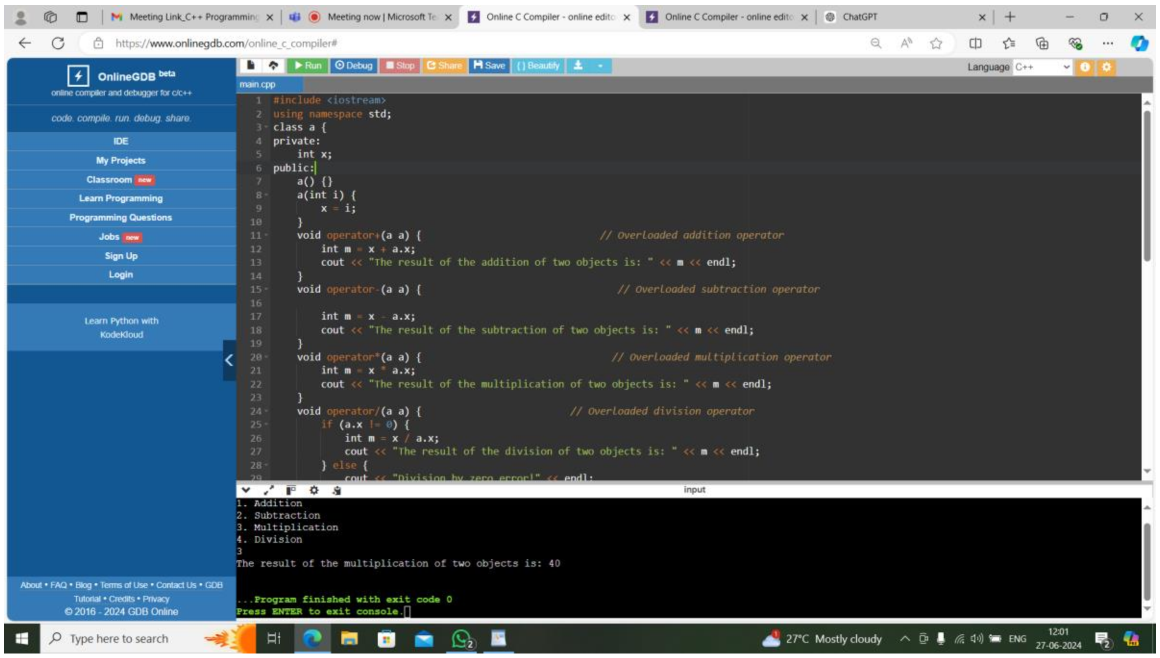
```
#include<iostream>
using namespace std;
class a{
    int x;
    public:
    a(){}
    a(int i)
    {
        x=i;
    }
    void operator+(a a)
    {
        int m= x+a.x;
        cout<<"the result of the addition of two objects is:"<<m<<endl;
    }
    void operator-(a a)
    {
        int m = x-a.x;
        cout<<"the result of the substraction of two objects:"<<m<<endl;
    }
    void operator*(a a)
    {
        int m = x*a.x;
        cout<<"the result of the multiplication of two objects:"<<m<<endl;
    }
    void operator/(a a)
    {
        int m= x/a.x;
        cout<<"the result of the division of two objects is:"<<m<<endl;
```

```
the result of the addition of two objects is:16
the result of the substraction of two objects:2
the result of the multiplication of two objects:63
the result of the division of two objects is:1


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

//program to overload the binary operators with options :-

```
#include <iostream>

using namespace std;

class a {

private:

    int x;

public:

    a() {}

    a(int i) {

        x = i;

    }

    void operator+(a a) {                                        // Overloaded addition operator

        int m = x + a.x;

        cout << "The result of the addition of two objects is: " << m << endl;

    }

    void operator-(a a) {                                          // Overloaded subtraction operator


        int m = x - a.x;

        cout << "The result of the subtraction of two objects is: " << m << endl;

    }

    void operator*(a a) {                                          // Overloaded multiplication operator

        int m = x * a.x;

        cout << "The result of the multiplication of two objects is: " << m << endl;

    }

    void operator/(a a) {                                  // Overloaded division operator

        if (a.x != 0) {

            int m = x / a.x;

            cout << "The result of the division of two objects is: " << m << endl;

        } else {
            cout << "Division by zero error!" << endl;
        }
    }
    void display() {                                   // Display function to show the value of x
        cout << x;
    }
};
int main() {
    a a1(8);
    a a2(5);
    int choice;
    cout << "Enter your choice:" << endl;
    cout << "1. Addition" << endl;
    cout << "2. Subtraction" << endl;
    cout << "3. Multiplication" << endl;
    cout << "4. Division" << endl;
    cin >> choice;

    switch (choice) {
        case 1:
            a1 + a2;
            break;
        case 2:
            a1 - a2;
            break;
        case 3:
            a1 * a2;
            break;
        case 4:
            a1 / a2;
            break;
        default:
            cout << "Invalid choice!" << endl;
            break;
    }

    return 0;
}
```

OUTPUT :-



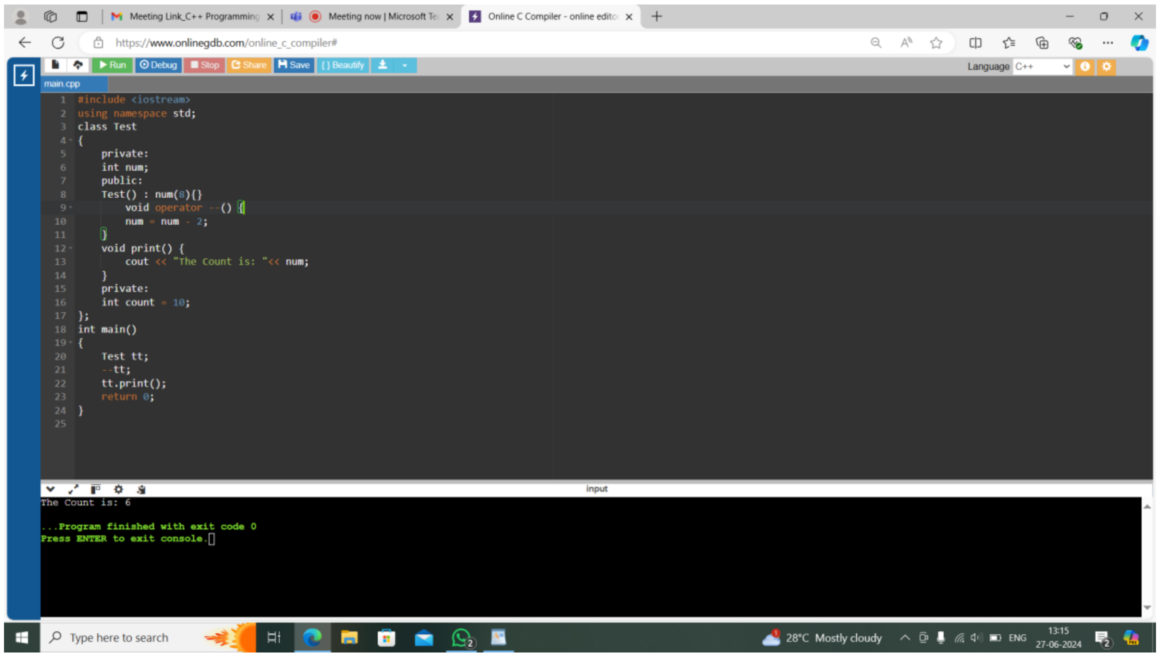
```
#include <iostream>
using namespace std;
class a {
private:
    int x;
public:
    a() {}
    a(int i) {
        x = i;
    }
    void operator+(a a) {                               // Overloaded addition operator
        int m = x + a.x;
        cout << "The result of the addition of two objects is: " << m << endl;
    }
    void operator-(a a) {                                 // Overloaded subtraction operator

        int m = x - a.x;
        cout << "The result of the subtraction of two objects is: " << m << endl;
    }
    void operator*(a a) {                                // Overloaded multiplication operator
        int m = x * a.x;
        cout << "The result of the multiplication of two objects is: " << m << endl;
    }
    void operator/(a a) {                            // Overloaded division operator
        if (a.x != 0) {
            int m = x / a.x;
            cout << "The result of the division of two objects is: " << m << endl;
        } else {
            cout << "Division by zero error!" << endl;
```

```
1. Addition
2. Subtraction
3. Multiplication
4. Division
3
The result of the multiplication of two objects is: 40


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

FUNCTION OVERLOADING :-

```
#include <iostream>
using namespace std;
class Test
{
    private:
    int num;
    public:
    Test() : num(8){}
        void operator --() {
        num = num - 2;
    }
    void print() {
        cout << "The Count is: "<< num;
    }
    private:
    int count = 10;
};
int main()
{
    Test tt;
    --tt;
    tt.print();
    return 0;
}
```

OUTPUT :-



```cpp
#include <iostream>
using namespace std;
class Test
{
    private:
    int num;
    public:
    Test() : num(8){}
        void operator --() {
        num = num - 2;
    }
    void print() {
        cout << "The Count is: "<< num;
    }
    private:
    int count = 10;
};
int main()
{
    Test tt;
    --tt;
    tt.print();
    return 0;
}
```

```
The Count is: 6

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