

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;

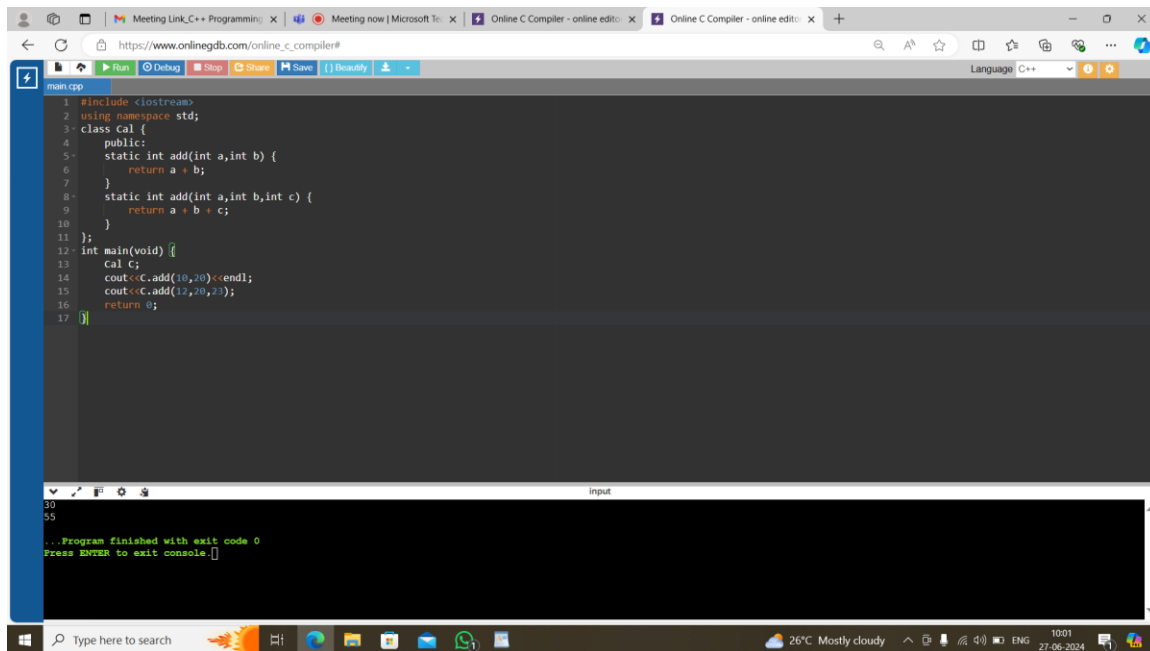
    cout<<C.add(10,20)<<endl;

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

    return 0;

}
```

OUTPUT :-



```
1 #include <iostream>
2 using namespace std;
3 class Cal {
4 public:
5     static int add(int a,int b) {
6         return a + b;
7     }
8     static int add(int a,int b,int c) {
9         return a + b + c;
10    }
11 };
12 int main(void) {
13     Cal c;
14     cout<<c.add(10,20)<<endl;
15     cout<<c.add(12,20,3);
16     return 0;
17 }
```

...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 :-

The screenshot shows the OnlineGDB web interface. The code editor contains the following C++ code:

```

1 #include <iostream>
2 using namespace std;
3 class Test
4 {
5     private:
6         int num;
7     public:
8         Test() : num(0){}
9         void operator ++() {
10             num = num + 2;
11         }
12     }
13     void print() {
14         cout << "The Count is: "<< count << endl;
15     }
16     private:
17         int count = 10;
18 };
19 int main()
20 {
21     Test tt;
22     ++tt;
23     tt.print();
24     return 0;
25 }

```

The output window at the bottom shows the result of the program execution:

```

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();
```

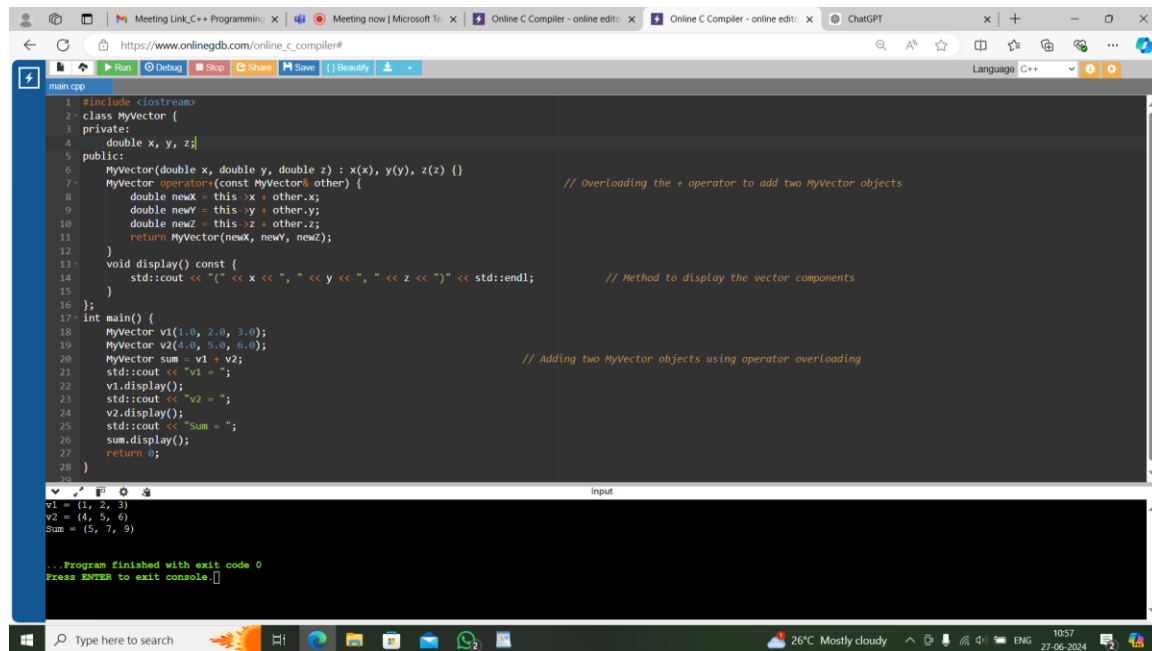
```
std::cout << "Sum = ";
```

```
sum.display();
```

```
return 0;
```

```
}
```

OUTPUT :-



```
1 #include <iostream>
2 class MyVector {
3 private:
4     double x, y, z;
5 public:
6     MyVector(double x, double y, double z) : x(x), y(y), z(z) {}
7     MyVector operator+(const MyVector& other) {
8         double newX = this->x + other.x;
9         double newY = this->y + other.y;
10        double newZ = this->z + other.z;
11        return MyVector(newX, newY, newZ);
12    }
13    void display() const {
14        std::cout << "x << x << ", " << y << ", " << z << " << std::endl;
15    }
16 };
17 int main() {
18     MyVector v1(1.0, 2.0, 3.0);
19     MyVector v2(4.0, 5.0, 6.0);
20     MyVector sum = v1 + v2;
21     std::cout << "v1 = ";
22     v1.display();
23     std::cout << "v2 = ";
24     v2.display();
25     std::cout << "Sum = ";
26     sum.display();
27     return 0;
28 }
```

Input

```
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 (+) :-

```
#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);

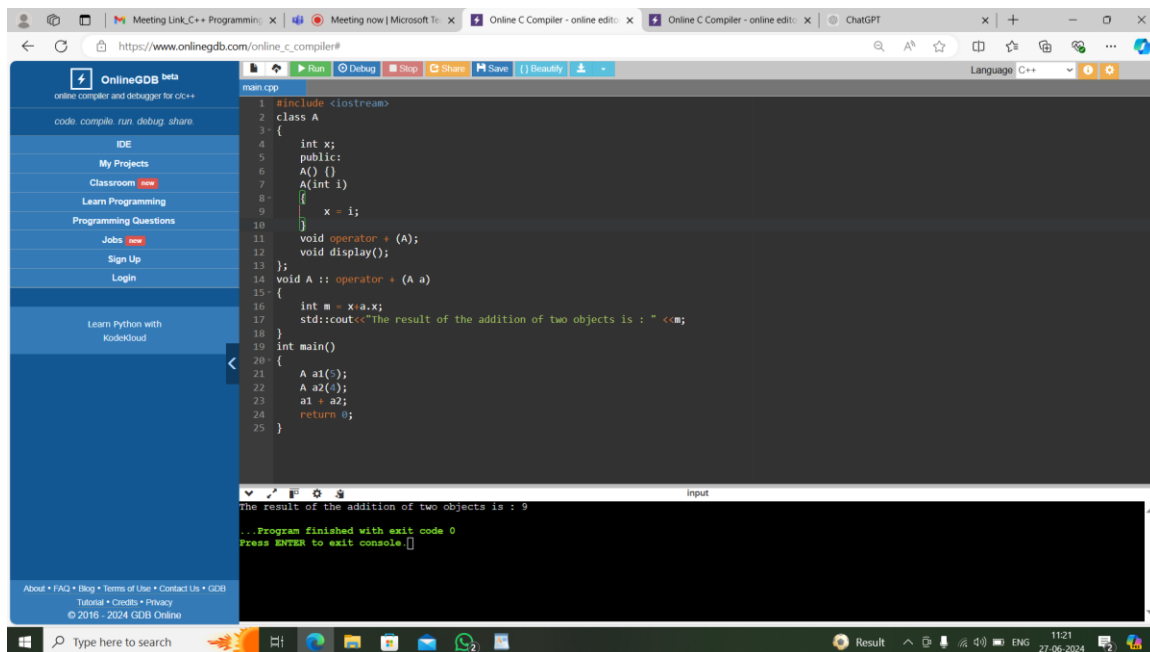
    a1 + a2;

    return 0;

}

```

OUTPUT :-



The screenshot shows the OnlineGDB IDE interface. The code editor contains the following C++ code:

```

1 #include <iostream>
2 class A
3 {
4     int x;
5     public:
6     A() {}
7     A(int i)
8     {
9         x = i;
10    }
11    void operator + (A);
12    void display();
13 };
14 void A :: operator + (A a)
15 {
16     int m = x+a.x;
17     std::cout<<"The result of the addition of two objects is : " <<m;
18 }
19 int main()
20 {
21     A a1(5);
22     A a2(4);
23     a1 + a2;
24     return 0;
25 }

```

The output window shows the result of the program execution:

```

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 subtraction 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 :-

The screenshot shows a web browser window with the URL https://www.onlinegdb.com/online_c_compiler#. The code editor contains the following C++ code:

```

1 #include<iostream>
2 using namespace std;
3 class a{
4     int x;
5     public:
6     a(){
7     a(int i)
8     {
9         x=i;
10    }
11    void operator+(a a)
12    {
13        int m= x+a.x;
14        cout<<"the result of the addition of two objects is:"<<m<<endl;
15    }
16    void operator-(a a)
17    {
18        int m= x-a.x;
19        cout<<"the result of the subtraction of two objects:"<<m<<endl;
20    }
21    void operator*(a a)
22    {
23        int m= x*a.x;
24        cout<<"the result of the multiplication of two objects:"<<m<<endl;
25    }
26    void operator/(a a)
27    {
28        int m= x/a.x;
29        cout<<"the result of the division of two objects is:"<<m<<endl;
30    }
31 }
32
33 int main()
34 {
35     a a1(9);
36     a a2(7);
37     a1+a2;
38     a1-a2;
39     a1*a2;
40     a1/a2;
41     return 0;
42 }

```

The output window shows the following results:

```

the result of the addition of two objects is:16
the result of the subtraction 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;

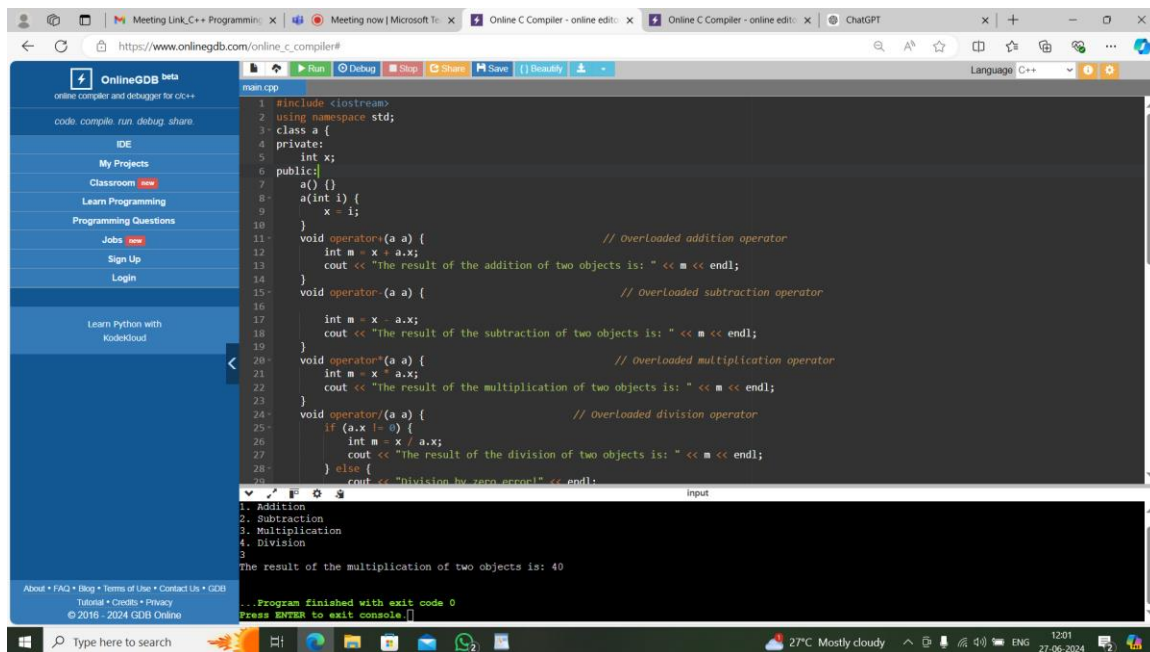
break;

}

return 0;

}

OUTPUT :-



The screenshot shows the OnlineGDB website interface. The code editor contains the following C++ code:

```
1 #include <iostream>
2 using namespace std;
3 class a {
4 private:
5     int x;
6 public:
7     a() {}
8     a(int i) {
9         x = i;
10    }
11    void operator+(a a) {                // Overloaded addition operator
12        int m = x + a.x;
13        cout << "The result of the addition of two objects is: " << m << endl;
14    }
15    void operator-(a a) {                // Overloaded subtraction operator
16        int m = x - a.x;
17        cout << "The result of the subtraction of two objects is: " << m << endl;
18    }
19    void operator*(a a) {                // Overloaded multiplication operator
20        int m = x * a.x;
21        cout << "The result of the multiplication of two objects is: " << m << endl;
22    }
23    void operator/(a a) {                // Overloaded division operator
24        if (a.x != 0) {
25            int m = x / a.x;
26            cout << "The result of the division of two objects is: " << m << endl;
27        } else {
28            cout << "Division by zero error!" << endl;
29        }
30    }
31 }
```

The output console shows the following results:

```
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 :-

Meeting Link_C++ Programming: x Meeting now | Microsoft 365: x Online C Compiler - online editor: x

https://www.onlinegdb.com/online_c_compiler#

Language C++

```
1 #include <iostream>
2 using namespace std;
3 class Test
4 {
5     private:
6         int num;
7     public:
8         Test() : num(0){}
9         void operator --() {
10             num = num - 2;
11         }
12         void print() {
13             cout << "The Count is: " << num;
14         }
15     private:
16         int count = 10;
17 };
18 int main()
19 {
20     Test tt;
21     --tt;
22     tt.print();
23     return 0;
24 }
25
```

The Count is: 6

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

Type here to search 28°C Mostly cloudy 13:15 27-06-2024