

main.cpp

Run

1 //3.overload the << operator to print contents of a user defined class

2

3 #include <iostream>

4

5 class MyClass {

6 int a, b;

7 public:

8 MyClass(int x, int y) : a(x), b(y) { }

9

10 friend std::ostream& operator<<(std::ostream& os, const MyClass& obj);

11 };

12

13 std::ostream& operator<<(std::ostream& os, const MyClass& obj) {

14 os << "a: " << obj.a << ", b: " << obj.b;

15 return os;

16 }

17

18 int main() {

19 MyClass myObj(10, 20);

20 std::cout << myObj << std::endl;

21

22 return 0;

23 }

Output

Clear

/tmp/HqOWG3SWV2.o

a: 10, b: 20

[illegible]

[illegible]

main.cpp

Run

Clear

```
1 //4.overload the == operator to compare two objects of a user defined class
2
3 #include <iostream>
4 class MyClass {
5     int x, y;
6 public:
7     MyClass(int a, int b) : x(a), y(b) { }
8
9     bool operator==(const MyClass& other) const {
10         return (x == other.x && y == other.y);
11     }
12 };
13 int main() {
14     MyClass a(5, 6), b(5, 6), c(7, 8);
15     if (a == b)
16         std::cout << "a and b are equal" << std::endl;
17     else
18         std::cout << "a and b are not equal" << std::endl;
19     if (a == c)
20         std::cout << "a and c are equal" << std::endl;
21     else
22         std::cout << "a and c are not equal" << std::endl;
23     return 0;
24 }
```

Output

Clear

```
/tmp/HqOWG3SWV2.o
a and b are equal
a and c are not equal
```

```
main.cpp  Run  Output  Clear

1 // Online C++ compiler to run C++ program online
2
3 #include <iostream>
4 class MyNumbers {
5 public:
6     MyNumbers(int a, int b) : num1(a), num2(b) {}
7     MyNumbers operator-(const MyNumbers& other) {
8         return MyNumbers(num1 - other.num1, num2 - other.num2);
9     }
10    int num1, num2;
11 };
12 int main() {
13     MyNumbers num1(10, 5);
14     MyNumbers num2(3, 2);
15     MyNumbers result = num1 - num2;
16     std::cout << "Result: " << result.num1 << ", " << result.num2 << std
        ::endl;
17     return 0;
18 }
```

/tmp/8nwzAVmfJ6.o
Result: 7, 3

main.cpp





























Run

Output



Clear

```
1 //6. [] operator  
2  
3 #include <iostream>  
4  
5 class MyArray {  
6 private:  
7     int* data;  
8     int size;  
9  
10 public:  
11     MyArray(int size) : size(size) {  
12         data = new int[size];  
13     }  
14  
15     int& operator[](int index) {  
16         if (index < 0 || index >= size) {  
17             std::cout << "Index out of bounds!" << std::endl;  
18             exit(1);  
19         }  
20         return data[index];  
21     }  
22  
23     // Other methods and destructor  
24 };  
25  
26 int main() {  
27     MyArray arr(5);  
28     arr[0] = 10;  
29     arr[1] = 20;  
30     std::cout << arr[1] << std::endl;
```

/tmp/XnuzMEVlyg.o
20



main.cpp



Run




























Output

Clear



```
1 //7.() to call a function with arguments
2
3 #include <iostream>
4
5 class MyFunction {
6 public:
7     int operator()(int a, int b) {
8         return a + b;
9     }
10 };
11
12 int main() {
13     MyFunction func;
14     std::cout << func(5, 7) << std::endl;
15     return 0;
16 }
```

```
/tmp/XnuzMEVlyg.o
12
```

Waiting for aax-fe-sin.amazon-adsystem.com...



main.cpp



Run

1 //9.overload a function to add two integer numbers and two floating point
number separately

2

3 #include <iostream>

4

5 class MyMath {

6 public:

7 int addInt(int a, int b) {

8 return a + b;

9 }

10

11 double addFloat(double a, double b) {

12 return a + b;

13 }

14 };

15

16 int main() {

17 MyMath math;

18 std::cout << math.addInt(5, 7) << std::endl;

19 std::cout << math.addFloat(5.5, 7.7) << std::endl;

20 return 0;

21 }

Output

Clear

/tmp/XnuzMEVlyg.o

12

13.2