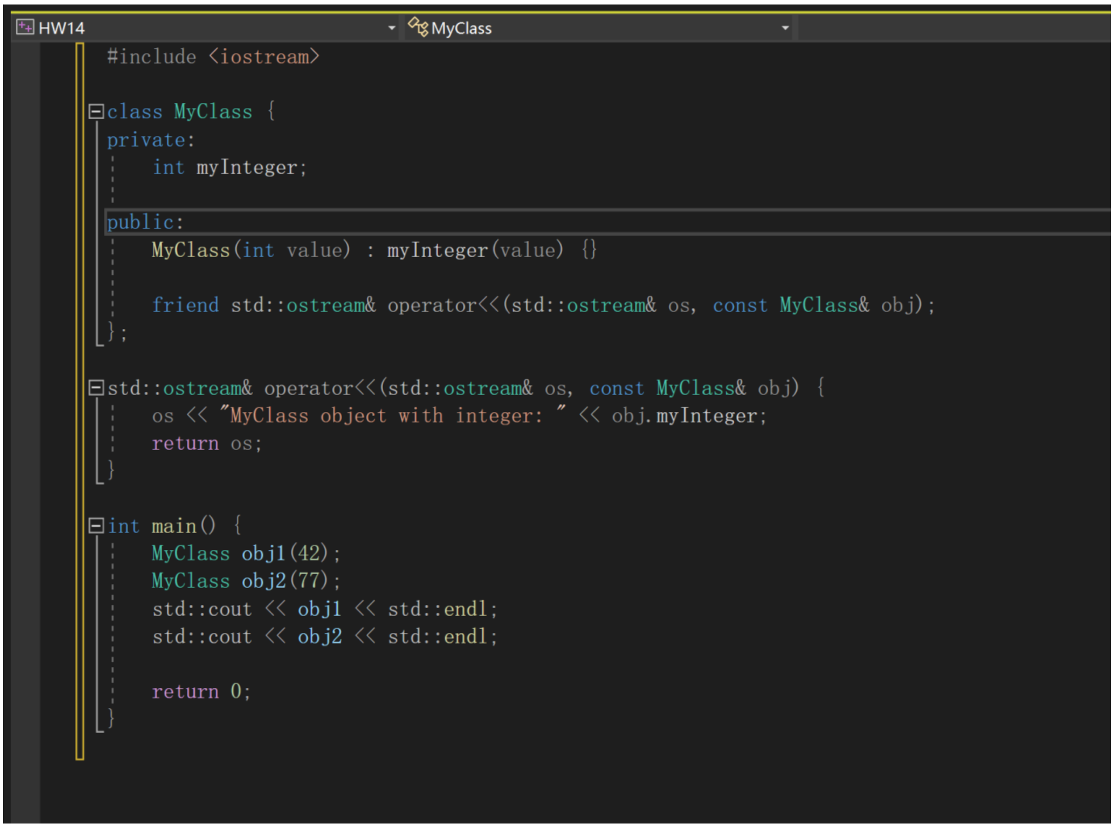
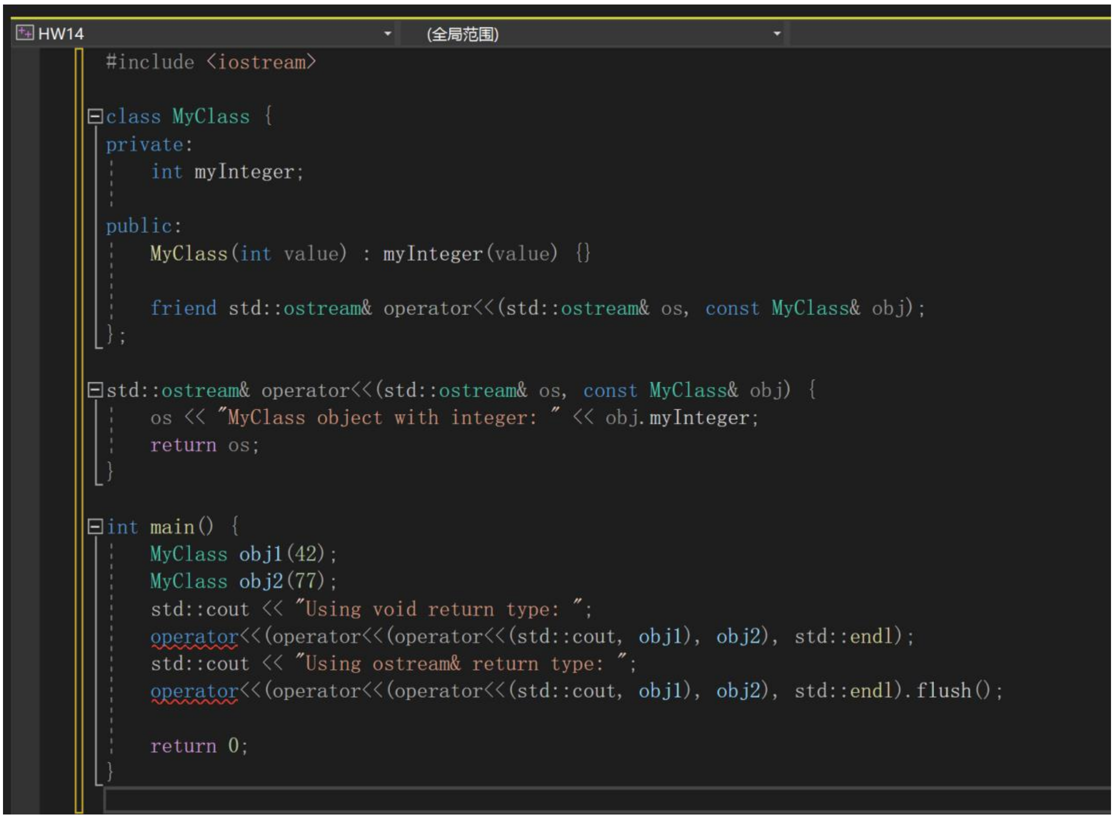
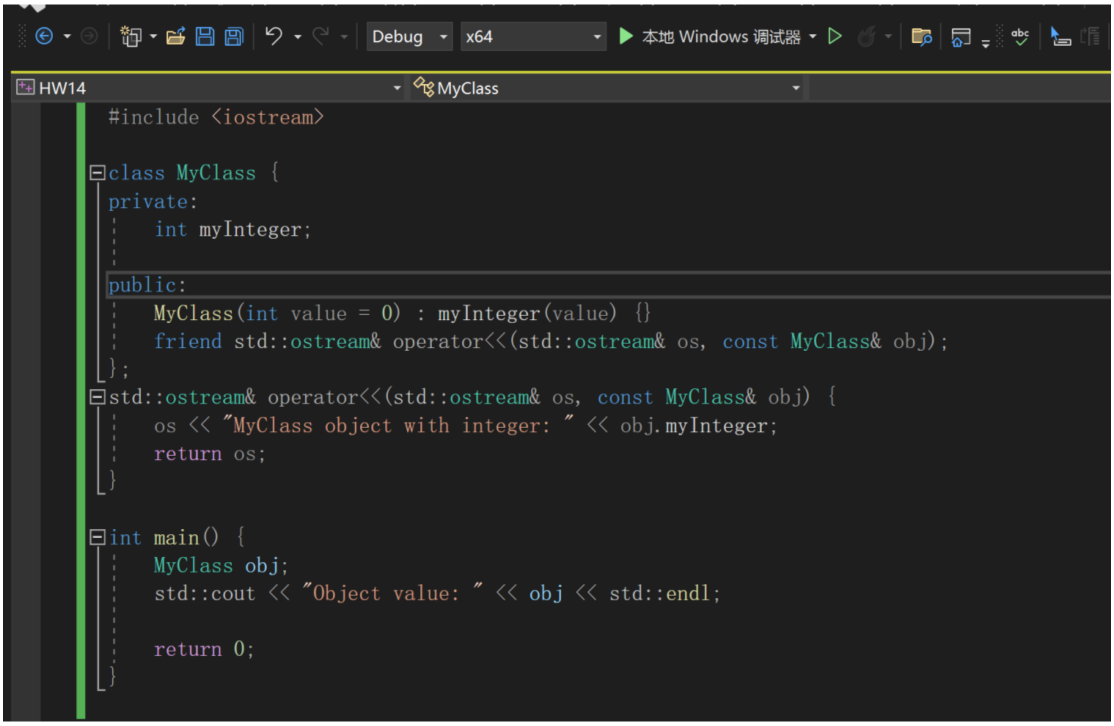


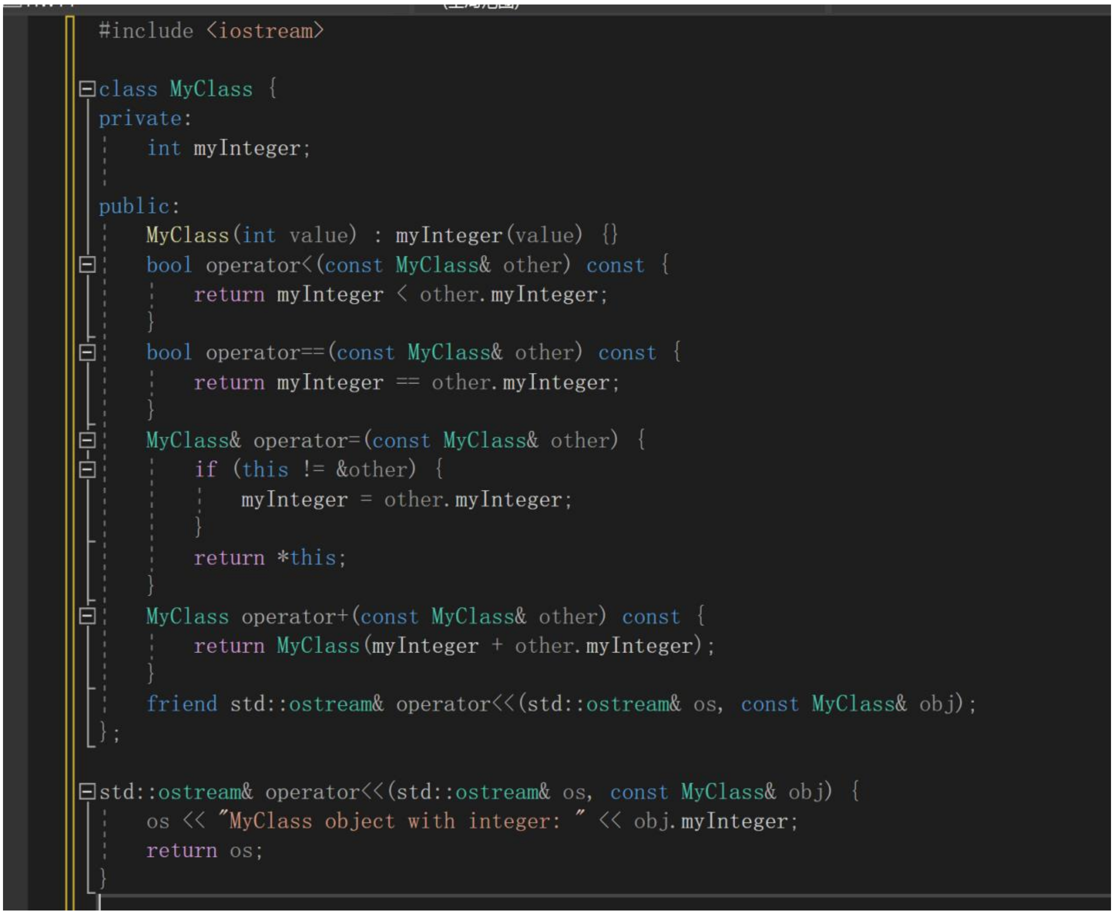
```cpp
#include <iostream>

class MyClass {
private:
    int myInteger;

public:
    MyClass(int value) : myInteger(value) {}

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

std::ostream& operator<<(std::ostream& os, const MyClass& obj) {
    os << "MyClass object with integer: " << obj.myInteger;
    return os;
}

int main() {
    MyClass obj1(42);
    MyClass obj2(77);
    std::cout << obj1 << std::endl;
    std::cout << obj2 << std::endl;

    return 0;
}
```



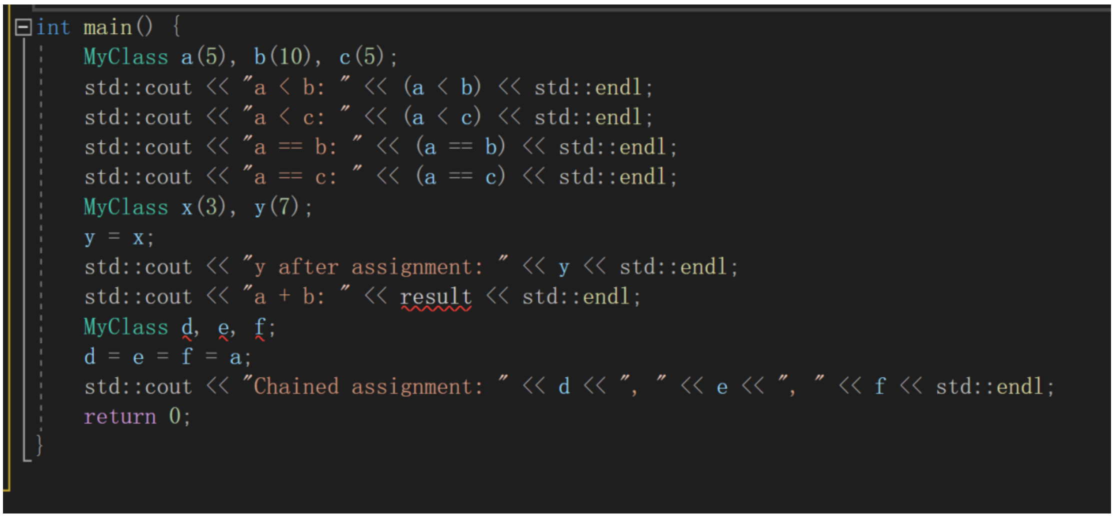
```cpp
#include <iostream>

class MyClass {
private:
    int myInteger;

public:
    MyClass(int value) : myInteger(value) {}

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

std::ostream& operator<<(std::ostream& os, const MyClass& obj) {
    os << "MyClass object with integer: " << obj.myInteger;
    return os;
}

int main() {
    MyClass obj1(42);
    MyClass obj2(77);
    std::cout << "Using void return type: ";
    operator<<(operator<<(operator<<(std::cout, obj1), obj2), std::endl);
    std::cout << "Using ostream& return type: ";
    operator<<(operator<<(operator<<(std::cout, obj1), obj2), std::endl).flush();

    return 0;
}
```



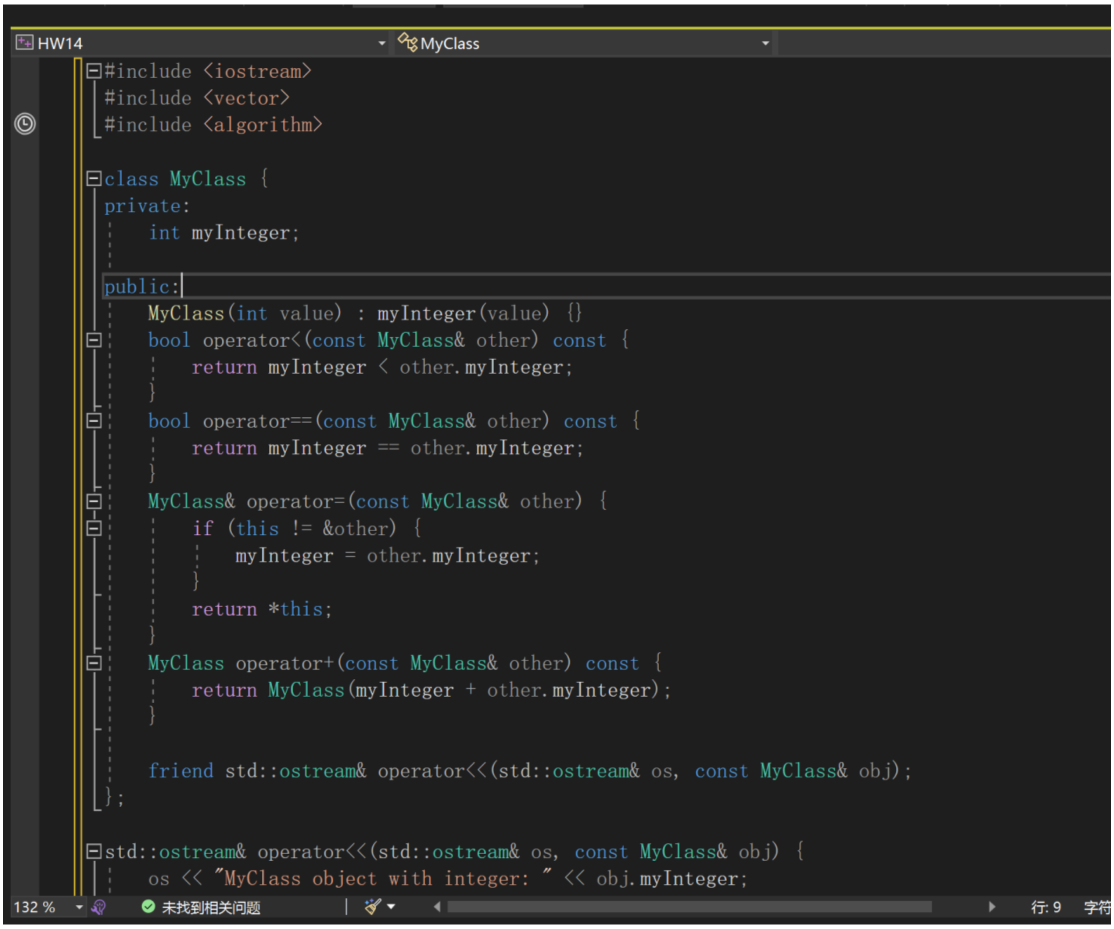
```cpp
#include <iostream>

class MyClass {
private:
    int myInteger;

public:
    MyClass(int value = 0) : myInteger(value) {}
    friend std::ostream& operator<<(std::ostream& os, const MyClass& obj);
};
std::ostream& operator<<(std::ostream& os, const MyClass& obj) {
    os << "MyClass object with integer: " << obj.myInteger;
    return os;
}

int main() {
    MyClass obj;
    std::cout << "Object value: " << obj << std::endl;

    return 0;
}
```



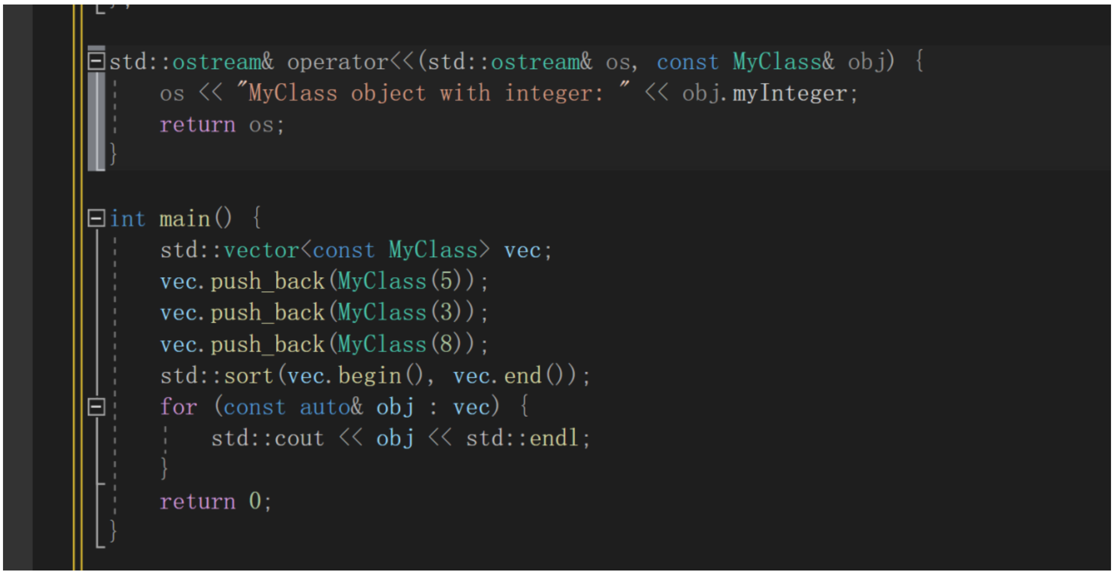
```cpp
#include <iostream>

class MyClass {
private:
    int myInteger;

public:
    MyClass(int value) : myInteger(value) {}
    bool operator<(const MyClass& other) const {
        return myInteger < other.myInteger;
    }
    bool operator==(const MyClass& other) const {
        return myInteger == other.myInteger;
    }
    MyClass& operator=(const MyClass& other) {
        if (this != &other) {
            myInteger = other.myInteger;
        }
        return *this;
    }
    MyClass operator+(const MyClass& other) const {
        return MyClass(myInteger + other.myInteger);
    }
    friend std::ostream& operator<<(std::ostream& os, const MyClass& obj);
};

std::ostream& operator<<(std::ostream& os, const MyClass& obj) {
    os << "MyClass object with integer: " << obj.myInteger;
    return os;
}
```



```
int main() {
    MyClass a(5), b(10), c(5);
    std::cout << "a < b: " << (a < b) << std::endl;
    std::cout << "a < c: " << (a < c) << std::endl;
    std::cout << "a == b: " << (a == b) << std::endl;
    std::cout << "a == c: " << (a == c) << std::endl;
    MyClass x(3), y(7);
    y = x;
    std::cout << "y after assignment: " << y << std::endl;
    std::cout << "a + b: " << result << std::endl;
    MyClass d, e, f;
    d = e = f = a;
    std::cout << "Chained assignment: " << d << ", " << e << ", " << f << std::endl;
    return 0;
}
```



```
#include <iostream>
#include <vector>
#include <algorithm>

class MyClass {
private:
    int myInteger;

public:
    MyClass(int value) : myInteger(value) {}
    bool operator<(const MyClass& other) const {
        return myInteger < other.myInteger;
    }
    bool operator==(const MyClass& other) const {
        return myInteger == other.myInteger;
    }
    MyClass& operator=(const MyClass& other) {
        if (this != &other) {
            myInteger = other.myInteger;
        }
        return *this;
    }
    MyClass operator+(const MyClass& other) const {
        return MyClass(myInteger + other.myInteger);
    }

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

std::ostream& operator<<(std::ostream& os, const MyClass& obj) {
    os << "MyClass object with integer: " << obj.myInteger;
```



```
};

std::ostream& operator<<(std::ostream& os, const MyClass& obj) {
    os << "MyClass object with integer: " << obj.myInteger;
    return os;
}

int main() {
    std::vector<const MyClass> vec;
    vec.push_back(MyClass(5));
    vec.push_back(MyClass(3));
    vec.push_back(MyClass(8));
    std::sort(vec.begin(), vec.end());
    for (const auto& obj : vec) {
        std::cout << obj << std::endl;
    }
    return 0;
}
```