```
1 #include <iostream>
 2 #include <string>
 3 using namespace std;
 4 class IntStack
 6 private:
 7
        int* stackArray;
 8
       int stackSize;
9
        int numElements;
10 public:
       IntStack(int);
11
12
       ~IntStack();
13
14
       void push(int);
       void pop(int&);
15
16
       bool isFull() const;
       bool isEmpty() const;
17
       void display() const;
18
19
       // Overloaded = operator
       const IntStack operator=(const IntStack& right)
20
21
22
            delete[] this->stackArray;
23
            this->stackSize = right.stackSize;
            this->numElements = right.numElements;
24
            this->stackArray = new int[stackSize];
25
26
            for (int i = 0; i < numElements; i++)</pre>
27
                this->stackArray[i] = right.stackArray[i];
28
            return *this;
29
       }
30
31
       // Copy constructor
       IntStack(const IntStack& obj)
32
33
       {
            this->stackSize = obj.stackSize;
34
35
            this->numElements = obj.numElements;
            this->stackArray = new int[stackSize];
36
37
            for (int i = 0; i < stackSize; i++)</pre>
38
                this->stackArray[i] = obj.stackArray[i];
39
       }
40 };
41 IntStack::IntStack(int Size)
42 {
43
        stackArray = new int[Size];
44
        stackSize = Size;
45
       numElements = 0;
46 }
47 IntStack::~IntStack()
48 {
49
       delete[] stackArray;
```

```
50 }
51 void IntStack::push(int num)
52 {
53
        if (isFull())
54
            cout << "The stack is full.\n";</pre>
55
        else
56
        {
            stackArray[numElements] = num;
57
58
            numElements++;
59
        }
60 }
61 void IntStack::pop(int &num)
62 {
63
        if (isEmpty())
64
            cout << "The stack is empty.\n";</pre>
65
        else
66
        {
67
            numElements--;
68
            num = stackArray[numElements];
69
        }
70 }
71 bool IntStack::isFull() const
72 {
73
        return numElements == stackSize;
74 }
75 bool IntStack::isEmpty() const
76 {
77
       return numElements == 0;
78 }
79 void IntStack::display() const
80 {
81
        if (isEmpty())
82
            cout << "The stack is empty.\n";</pre>
83
            for (int i = 0; i < numElements; i++)</pre>
84
85
                cout << stackArray[i] << "\t";</pre>
86
        cout << endl;</pre>
87 }
88
89 int main()
90 {
91
        IntStack stack1(6);
92
        stack1.push(5);
93
        stack1.push(10);
94
        stack1.push(15);
95
96
        IntStack stack2(6);
97
        stack2 = stack1;
        stack2.push(20);
98
```

```
...inWindows\Desktop\DC\CMPT 1209\2023-2-Labs\Lab 08.cpp
```

```
99
100
        IntStack stack3 = stack1;
        stack3.push(25);
101
102
        stack1.display();
103
        stack2.display();
104
        stack3.display();
105
106
        return 0;
107 }
108
109
110
```

3