

```
1  #include <iostream>
2  using namespace std;
3
4  /*class Box {
5  private:
6      int x;
7      int y;
8  public:
9      Box()
10     {
11         x = 0;
12         y = 0;
13     }
14     void setX(int a)
15     {
16         x = a;
17     }
18     void setY(int a)
19     {
20         y = a;
21     }
22     int getX()
23     {
24         return x;
25     }
26     int getY()
27     {
28         return y;
29     }
30 };
31 int main()
32 {
33     Box a, b;
34     a.setX(10);
35     a.setY(20);
36     b = a;
37     cout << "After assignment statement" << endl;
38     cout << a.getX() << " " << a.getY() << endl;
39     cout << b.getX() << " " << b.getY() << endl;
40     a.setX(30);
41     a.setY(40);
42     b.setX(50);
43     b.setY(60);
44     cout << "After updating a and b" << endl;
45     cout << a.getX() << " " << a.getY() << endl;
46     cout << b.getX() << " " << b.getY() << endl;
47     return 0;
48 }*/
49
```

```
50 class StudentTestScores
51 {
52 private:
53     string studentName; // The student's name
54     double* testScores; // Points to array of test scores
55     int numTestScores; // Number of test scores
56 public:
57     StudentTestScores(int size)
58     {
59         studentName = " ";
60         numTestScores = size;
61         testScores = new double[size];
62         for (int i = 0; i < size; i++)
63             testScores[i] = 0;
64     }
65
66     /*~StudentTestScores()
67     {
68         delete[] testScores;
69     }*/
70
71     void setTestScore(double score, int index)
72     {
73         testScores[index] = score;
74     }
75
76     void setStudentName(string name)
77     {
78         studentName = name;
79     }
80
81     string getStudentName() const
82     {
83         return studentName;
84     }
85
86     int getNumTestScores()
87     {
88         return numTestScores;
89     }
90
91     double getTestScore(int index) const
92     {
93         return testScores[index];
94     }
95     void displayStudent()
96     {
97         cout << "Name: " << studentName << endl;
98         cout << "Test Scores: ";
```

```
99     for (int i = 0; i < numTestScores; i++)
100         cout << testScores[i] << " ";
101     cout << endl;
102 }
103
104 // Overloaded = operator
105 const StudentTestScores operator=(const StudentTestScores& right)
106 {
107     delete[] testScores;
108     studentName = right.studentName;
109     numTestScores = right.numTestScores;
110     testScores = new double[numTestScores];
111     for (int i = 0; i < numTestScores; i++)
112         testScores[i] = right.testScores[i];
113     return *this;
114 }
115
116 // Copy constructor
117 StudentTestScores(const StudentTestScores& obj)
118 {
119     studentName = obj.studentName;
120     numTestScores = obj.numTestScores;
121     testScores = new double[numTestScores];
122     for (int i = 0; i < numTestScores; i++)
123         testScores[i] = obj.testScores[i];
124 }
125 };
126
127 int main()
128 {
129     StudentTestScores student1(3);
130     student1.setStudentName("Jack");
131     student1.setTestScore(100.0, 0);
132     student1.setTestScore(95.0, 1);
133     student1.setTestScore(80, 2);
134     student1.displayStudent();
135
136     StudentTestScores student2(3);
137     student2 = student1;
138     student2.setStudentName("David");
139     student2.displayStudent();
140
141     StudentTestScores student3 = student1;
142     student3.setStudentName("Adam");
143     student3.displayStudent();
144
145     student1.setTestScore(88, 0);
146     student2.setTestScore(77, 1);
147     student3.setTestScore(66, 2);
```

```
148     cout << "After updating students 1,2, and 3" << endl;
149     student1.displayStudent();
150     student2.displayStudent();
151     student3.displayStudent();
152
153     return 0;
154 }
```