

```
1 /*
2  * To change this template, choose Tools | Templates
3  * and open the template in the editor.
4  */
5 package lab6.solution.src.lab06;
6
7 /**
8  *
9  * @author vanting
10 */
11 class AssertionDemo {
12
13     private static int balance = 900;
14
15     public static void main(String args[]) {
16         System.out.println("Acc balance: " + withdrawMoney(500));
17         System.out.println("Acc balance: " + withdrawMoney(500));
18     }
19
20     public static double withdrawMoney(double amount) {
21         // precondition: the balance must be enough for the withdraw amount
22         assert balance >= amount : "Insufficient Account Balance: " + (balance -
23             amount);
24
25         balance -= amount;
26         return balance;
27     }
28 }
```

```
1 /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6 package lab6.solution.src.lab06;
7
8 /**
9  *
10 * @author vanting
11 */
12 public class DimensionMismatchException extends Exception {
13
14     private Matrix firstMatrix;
15     private Matrix secondMatrix;
16
17     public Matrix getFirstMatrix() {
18         return firstMatrix;
19     }
20
21     public void setFirstMatrix(Matrix firstMatrix) {
22         this.firstMatrix = firstMatrix;
23     }
24
25     public Matrix getSecondMatrix() {
26         return secondMatrix;
27     }
28
29     public void setSecondMatrix(Matrix secondMatrix) {
30         this.secondMatrix = secondMatrix;
31     }
32 }
33 }
```

```
1 /*
2  * To change this template, choose Tools | Templates
3  * and open the template in the editor.
4  */
5 package lab6.solution.src.lab06;
6
7 /**
8  *
9  * @author wanting
10 */
11 public class Matrix {
12
13     private int[][] elements;
14
15     Matrix(int[][] e) {
16         this.elements = e;
17     }
18
19     public int getElement(int row, int col) {
20         return elements[row][col];
21     }
22
23     public void setElement(int row, int col, int value) {
24         elements[row][col] = value;
25     }
26
27     public boolean add(Matrix matrix2) {
28
29         int row1 = this.elements.length;
30         int col1 = this.elements[0].length;
31         int row2 = matrix2.elements.length;
32         int col2 = matrix2.elements[0].length;
33
34         if (row1 == row2 && col1 == col2) {
35             for (int i = 0; i < row1; i++) {
36                 for (int j = 0; j < col1; j++) {
37                     this.elements[i][j] += matrix2.elements[i][j];
38                 }
39             }
40             return true;
41         } else {
42             return false;
43         }
44     }
45
46     public void print() {
47
48         int row = this.elements.length;
49         int col = this.elements[0].length;
50
51         for (int i = 0; i < row; i++) {
52             for (int j = 0; j < col; j++) {
53                 System.out.printf("%02d", this.elements[i][j]);
54             }
55             System.out.println("");
56         }
57     }
58
59     // for Q6 in Lab 03
```

```
60 public Matrix multiply(Matrix matrix2) throws DimensionMismatchException {
61
62     int row1 = this.elements.length;
63     int col1 = this.elements[0].length;
64     int row2 = matrix2.elements.length;
65     int col2 = matrix2.elements[0].length;
66
67     int sum = 0;
68     int multiply[][] = new int[row1][col2];
69
70     if (col1 == row2) {
71         for (int i = 0; i < row1; i++) {
72             for (int j = 0; j < col2; j++) {
73                 for (int k = 0; k < col1; k++) {
74                     sum = sum + elements[i][k] * matrix2.elements[k][j];
75                 }
76                 multiply[i][j] = sum;
77                 sum = 0;
78             }
79         }
80         return new Matrix(multiply);
81     } else {
82         DimensionMismatchException ex = new DimensionMismatchException();
83         ex.setFirstMatrix(this);
84         ex.setSecondMatrix(matrix2);
85         throw ex;
86     }
87 }
88 }
89 }
```

```
1 /*
2  * To change this template, choose Tools | Templates
3  * and open the template in the editor.
4  */
5 package lab6.solution.src.lab06;
6
7 /**
8  *
9  * @author vanting
10 */
11 public class TestMatrix {
12
13     public static void main(String[] args) {
14         // initialize both matrices
15         Matrix m1 = new Matrix(new int[][] { { 5, 15, 0 }, { 2, 22, 0 } });
16         Matrix m2 = new Matrix(new int[][] { { 5, 6 }, { 7, 8 } });
17
18         System.out.println("First matrix:");
19         m1.print();
20
21         System.out.println("Second matrix:");
22         m2.print();
23
24         System.out.println("");
25         System.out.println("Result of multiplication:");
26
27         try {
28             m1.multiply(m2).print();
29         } catch (DimensionMismatchException ex) {
30             System.out.println("Invalid matrix size: ");
31             System.out.println("First matrix:");
32             ex.getFirstMatrix().print();
33             System.out.println("Second matrix:");
34             ex.getSecondMatrix().print();
35         }
36     }
37 }
38
```

```
1 /*
2  * This program should be placed in the Test Packages.
3  */
4 package lab06;
5
6 import static junit.framework.Assert.*;
7 import org.junit.Before;
8 import org.junit.Test;
9
10
11 /**
12  *
13  * @author vanting
14  */
15 public class MatrixUnitTest {
16
17     Matrix matA;
18     Matrix matB;
19     Matrix matC;
20
21     @Before
22     public void setUp() {
23         matA = new Matrix(new int[][]{{1,2,3},{4,5,6}});
24         matB = new Matrix(new int[][]{{1,1,1},{2,2,2}});
25         matC = new Matrix(new int[][]{{1},{1},{1}});
26     }
27
28     @Test
29     public void getElementTest() {
30         int[][] expected = new int[][]{{1,2,3},{4,5,6}};
31         for(int i=0; i<expected.length; i++)
32             for(int j=0; j<expected[0].length; j++)
33                 assertEquals(expected[i][j], matA.getElement(i, j));
34     }
35
36     @Test
37     public void addTest() {
38         int[][] expected = new int[][]{{2,3,4},{6,7,8}};
39         matA.add(matB);
40         for(int i=0; i<expected.length; i++)
41             for(int j=0; j<expected[0].length; j++)
42                 assertEquals(expected[i][j], matA.getElement(i, j));
43     }
44
45     //----- to be done by students
46
47     @Test
48     public void multiplyOnSuccessTest() throws DimensionMismatchException {
49         int[][] expected = new int[][]{{6},{15}};
50         Matrix result = matA.multiply(matC);
51         for(int i=0; i<expected.length; i++)
52             for(int j=0; j<expected[0].length; j++)
53                 assertEquals(expected[i][j], result.getElement(i, j));
54     }
55
56     @Test (expected = DimensionMismatchException.class)
57     public void multiplyOnFailTest() throws DimensionMismatchException {
58
59         Matrix result = matA.multiply(matB); // throw exception
60     }
61 }
62
```

```
60  
61     }  
62  
63     }  
64
```