

Assignment on Jumble

Submitted by: -

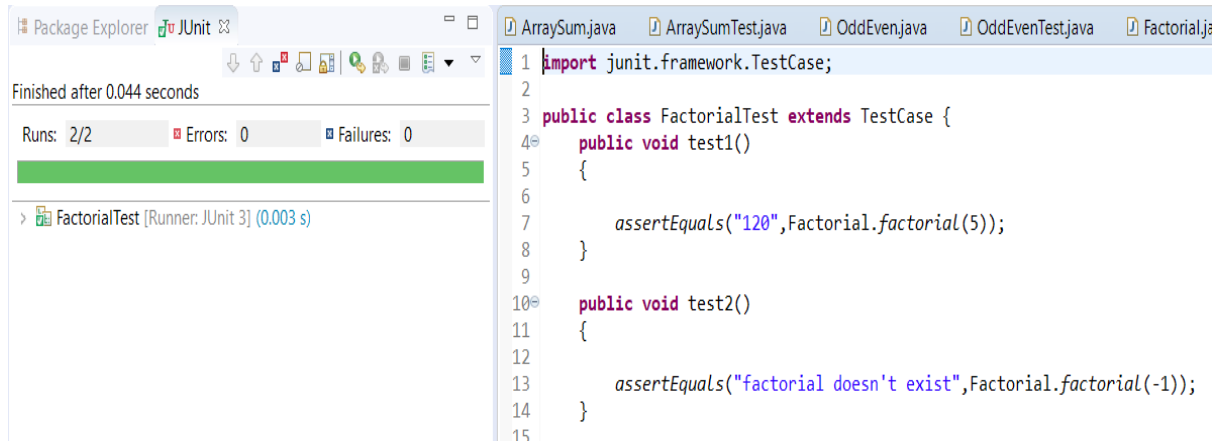
MTech, Software Engineering

Sub – Software Testing Lab

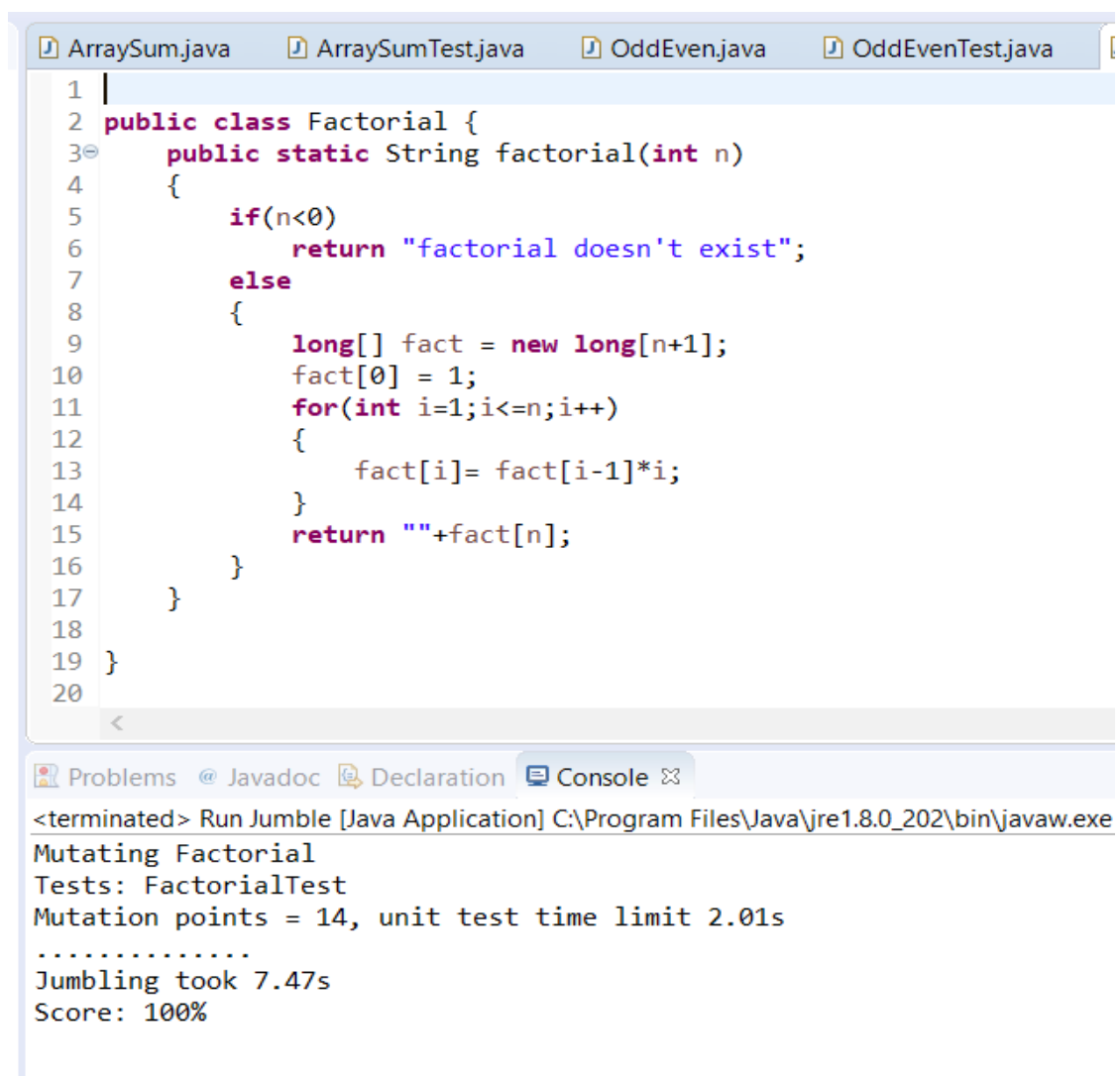


National Institute of Technology
Rourkela

1. Write a program to generate Factorial of number (where stack length should be at 3 (max)). The numbers should be 5, 3, 8 and 15.



```
1 import junit.framework.TestCase;
2
3 public class FactorialTest extends TestCase {
4     public void test1()
5     {
6         assertEquals("120",Factorial.factorial(5));
7     }
8
9     public void test2()
10    {
11
12        assertEquals("factorial doesn't exist",Factorial.factorial(-1));
13    }
14 }
15
```



```
1
2 public class Factorial {
3     public static String factorial(int n)
4     {
5         if(n<0)
6             return "factorial doesn't exist";
7         else
8         {
9             long[] fact = new long[n+1];
10            fact[0] = 1;
11            for(int i=1;i<=n;i++)
12            {
13                fact[i]= fact[i-1]*i;
14            }
15            return ""+fact[n];
16        }
17    }
18 }
19 }
20
```

Problems @ Javadoc Declaration Console

<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\javaw.exe

Mutating Factorial

Tests: FactorialTest

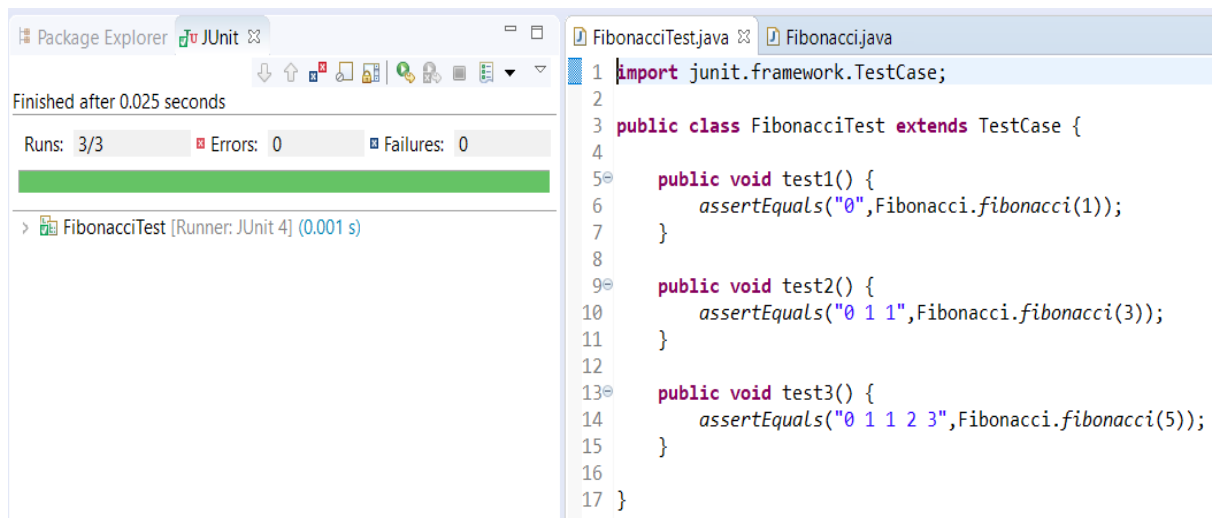
Mutation points = 14, unit test time limit 2.01s

.....

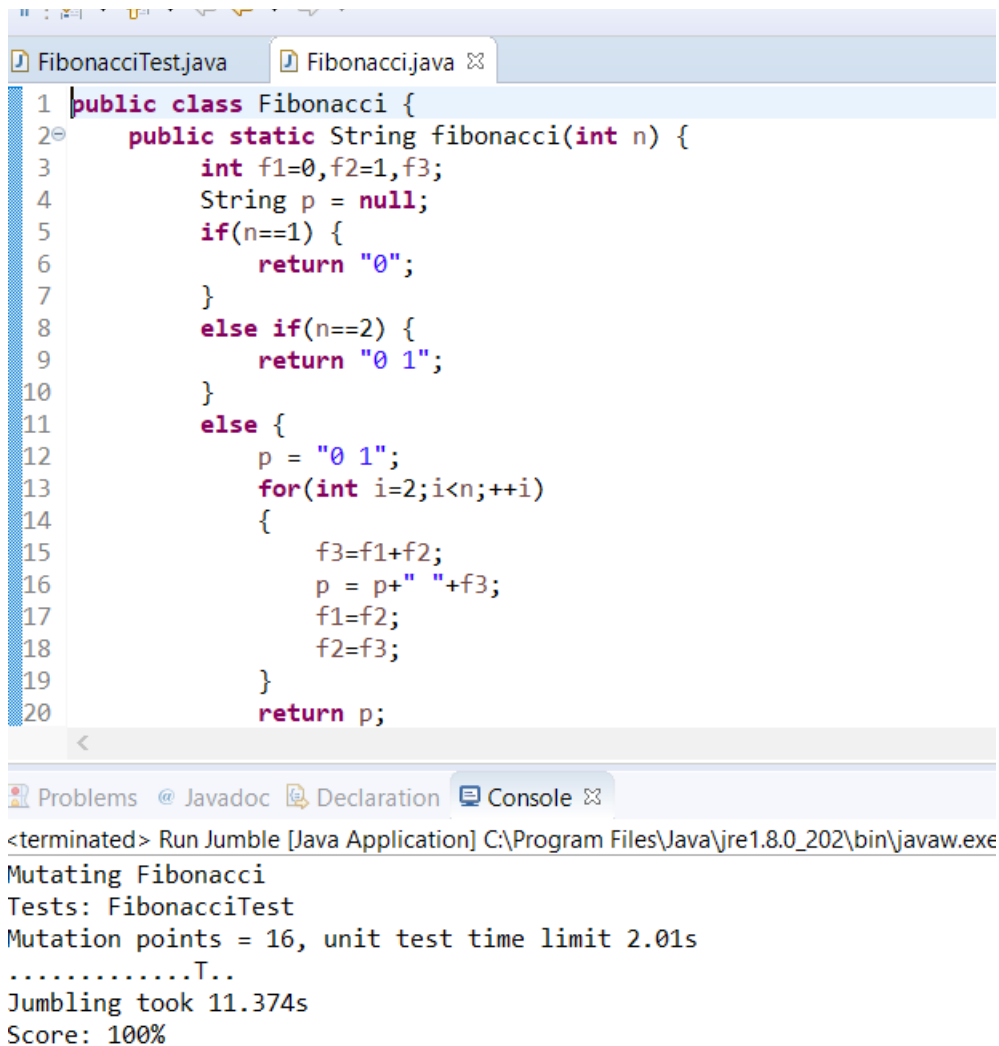
Jumbling took 7.47s

Score: 100%

2. Write a program to generate Fibonacci numbers.



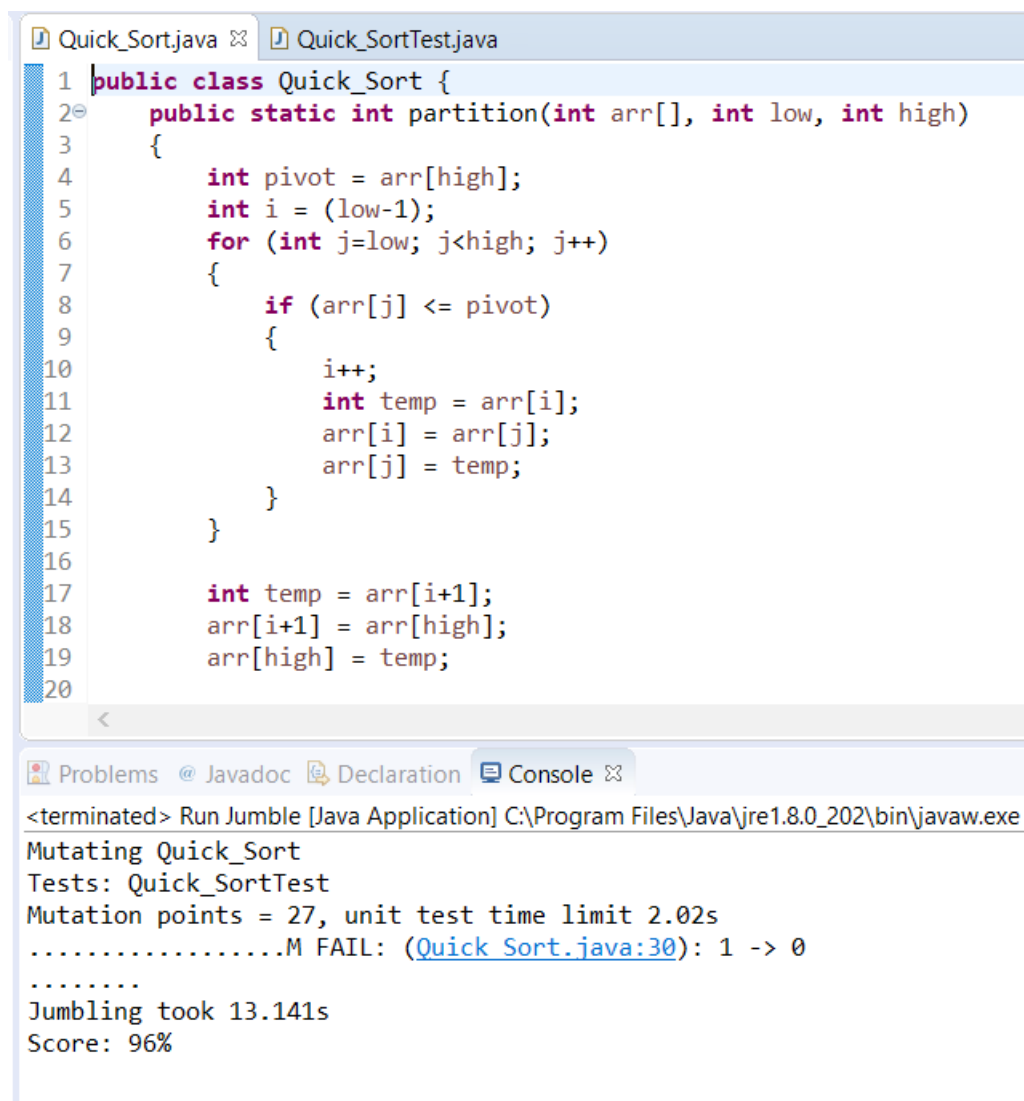
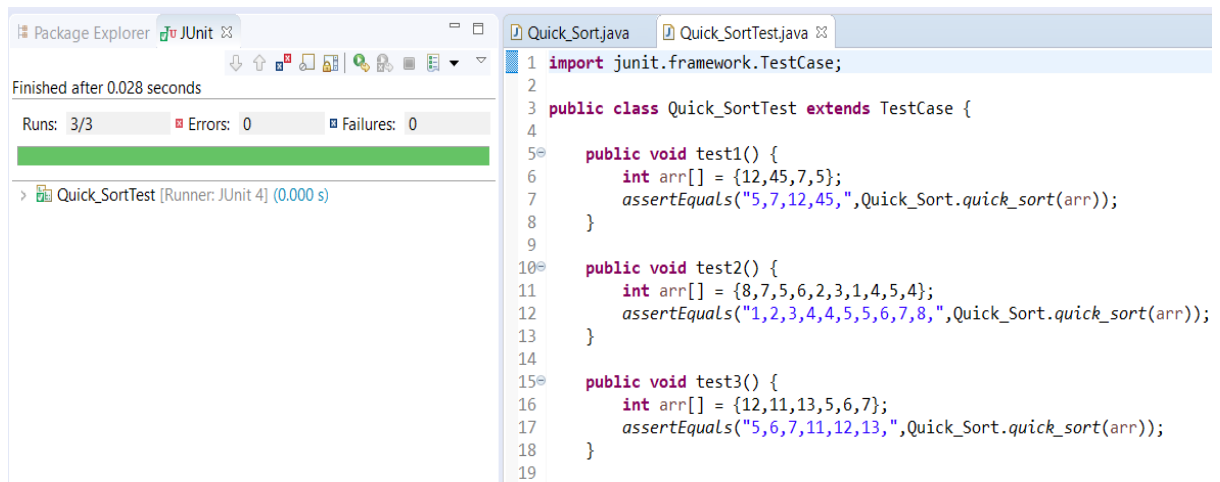
```
1 import junit.framework.TestCase;
2
3 public class FibonacciTest extends TestCase {
4
5     public void test1() {
6         assertEquals("0", Fibonacci.fibonacci(1));
7     }
8
9     public void test2() {
10        assertEquals("0 1 1", Fibonacci.fibonacci(3));
11    }
12
13    public void test3() {
14        assertEquals("0 1 1 2 3", Fibonacci.fibonacci(5));
15    }
16
17 }
```



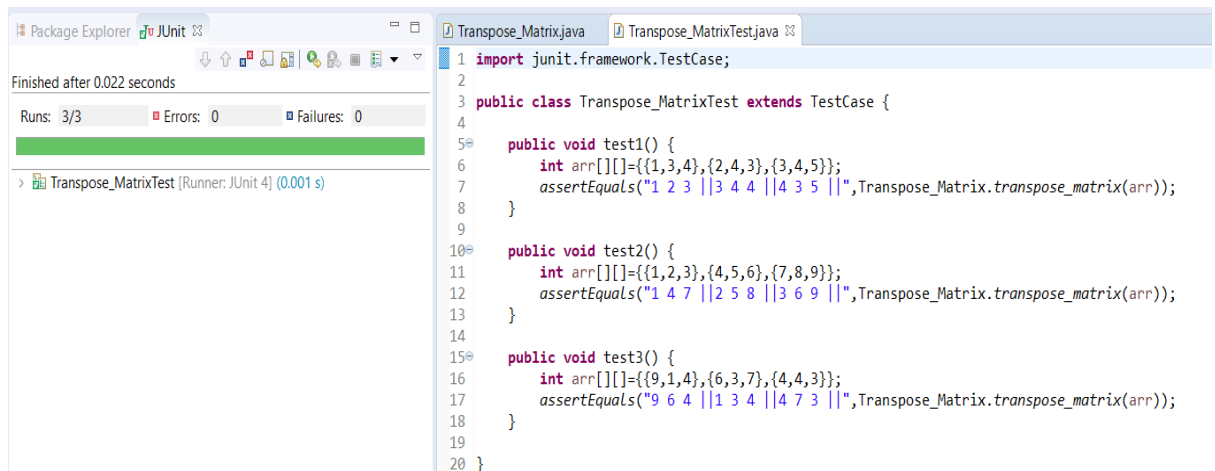
```
1 public class Fibonacci {
2     public static String fibonacci(int n) {
3         int f1=0,f2=1,f3;
4         String p = null;
5         if(n==1) {
6             return "0";
7         }
8         else if(n==2) {
9             return "0 1";
10        }
11        else {
12            p = "0 1";
13            for(int i=2;i<n;++i)
14            {
15                f3=f1+f2;
16                p = p+" "+f3;
17                f1=f2;
18                f2=f3;
19            }
20            return p;
21        }
22    }
23 }
```

<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\javaw.exe
Mutating Fibonacci
Tests: FibonacciTest
Mutation points = 16, unit test time limit 2.01s
.....T..
Jumbling took 11.374s
Score: 100%

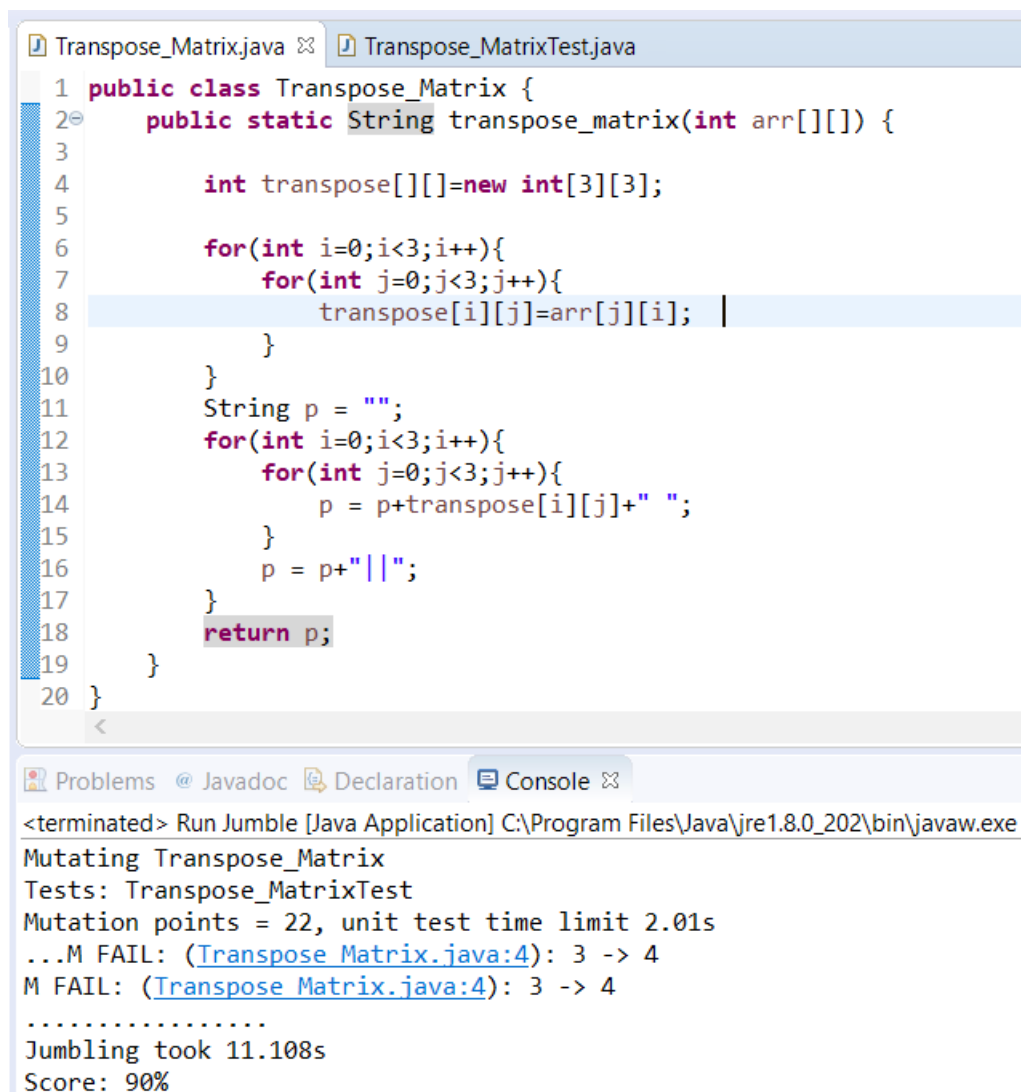
3. Write a program which performs sorting of group of integer values using quick sort technique.



4. Write a program that accepts elements of a matrix and display its transpose.



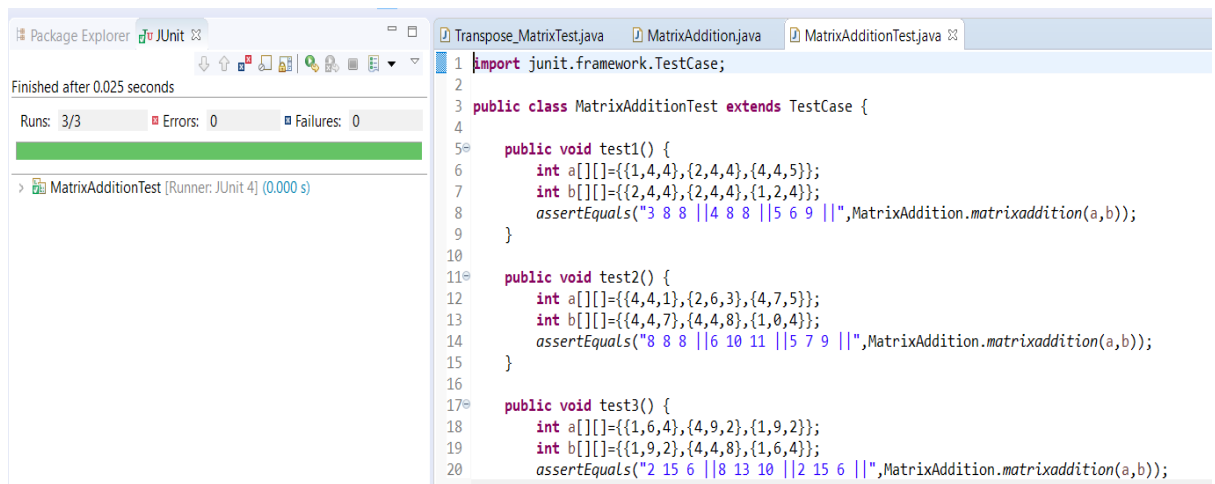
```
1 import junit.framework.TestCase;
2
3 public class Transpose_MatrixTest extends TestCase {
4
5     public void test1() {
6         int arr[][]={{1,3,4},{2,4,3},{3,4,5}};
7         assertEquals("1 2 3 || 3 4 4 || 4 3 5 ||",Transpose_Matrix.transpose_matrix(arr));
8     }
9
10    public void test2() {
11        int arr[][]={{1,2,3},{4,5,6},{7,8,9}};
12        assertEquals("1 4 7 || 2 5 8 || 3 6 9 ||",Transpose_Matrix.transpose_matrix(arr));
13    }
14
15    public void test3() {
16        int arr[][]={{9,1,4},{6,3,7},{4,4,3}};
17        assertEquals("9 6 4 || 1 3 4 || 4 7 3 ||",Transpose_Matrix.transpose_matrix(arr));
18    }
19
20 }
```



```
1 public class Transpose_Matrix {
2     public static String transpose_matrix(int arr[][]) {
3
4         int transpose[][]=new int[3][3];
5
6         for(int i=0;i<3;i++){
7             for(int j=0;j<3;j++){
8                 transpose[i][j]=arr[j][i];
9             }
10        }
11        String p = "";
12        for(int i=0;i<3;i++){
13            for(int j=0;j<3;j++){
14                p = p+transpose[i][j]+" ";
15            }
16            p = p+"||";
17        }
18        return p;
19    }
20 }
```

<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\javaw.exe
Mutating Transpose_Matrix
Tests: Transpose_MatrixTest
Mutation points = 22, unit test time limit 2.01s
...M FAIL: (Transpose_Matrix.java:4): 3 -> 4
M FAIL: (Transpose_Matrix.java:4): 3 -> 4
.....
Jumblng took 11.108s
Score: 90%

5. Write a program to add two matrices and display sum matrix.



The screenshot shows an IDE with a JUnit test runner on the left and a Java source file on the right. The test runner indicates that the tests passed successfully. The Java file, `MatrixAdditionTest.java`, contains three test methods: `test1()`, `test2()`, and `test3()`, each testing the `matrixaddition` method with different input matrices.

```
1 import junit.framework.TestCase;
2
3 public class MatrixAdditionTest extends TestCase {
4
5     public void test1() {
6         int a[][]={{1,4,4},{2,4,4},{4,4,5}};
7         int b[][]={{2,4,4},{2,4,4},{1,2,4}};
8         assertEquals("3 8 8 ||4 8 8 ||5 6 9 ||",MatrixAddition.matrixaddition(a,b));
9     }
10
11     public void test2() {
12         int a[][]={{4,4,1},{2,6,3},{4,7,5}};
13         int b[][]={{4,4,7},{4,4,8},{1,0,4}};
14         assertEquals("8 8 8 ||6 10 11 ||5 7 9 ||",MatrixAddition.matrixaddition(a,b));
15     }
16
17     public void test3() {
18         int a[][]={{1,6,4},{4,9,2},{1,9,2}};
19         int b[][]={{1,9,2},{4,4,8},{1,6,4}};
20         assertEquals("2 15 6 ||8 13 10 ||2 15 6 ||",MatrixAddition.matrixaddition(a,b));
21     }
22 }
```



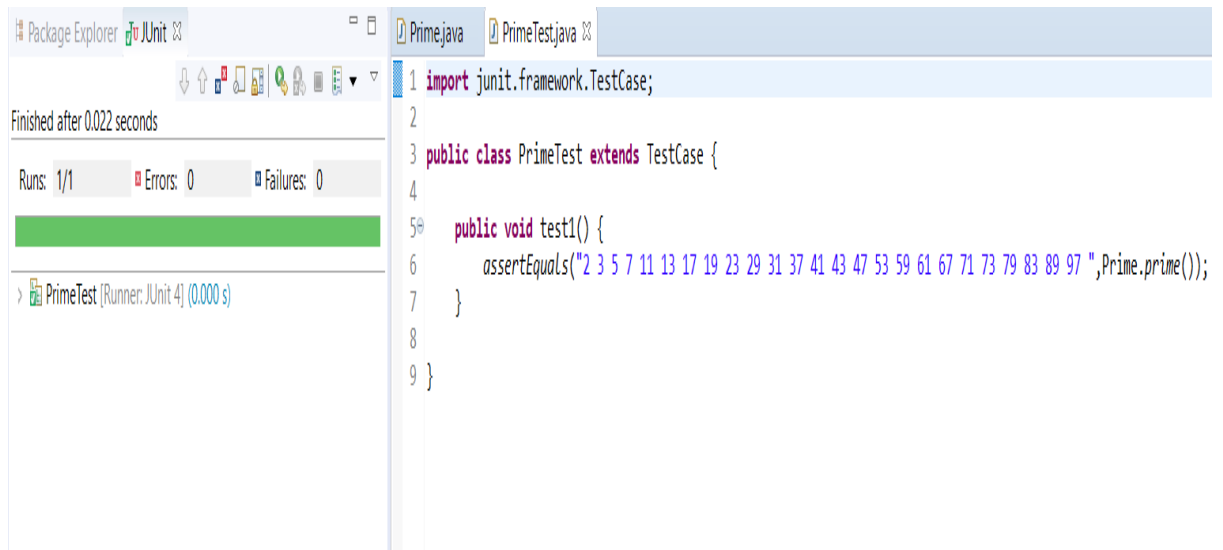
The screenshot shows the implementation of the `MatrixAddition` class in `MatrixAddition.java`. The class has a static method `matrixaddition` that takes two 3x3 integer matrices and returns a string representation of their sum. The test results at the bottom show that the tests failed due to a mutation in the output string.

```
1 public class MatrixAddition {
2     public static String matrixaddition(int a[][],int b[][]) {
3
4         int c[][]=new int[3][3];
5         String p = "";
6         for(int i=0;i<3;i++){
7             for(int j=0;j<3;j++){
8                 c[i][j]=a[i][j]+b[i][j];
9                 p = p+c[i][j]+" ";
10            }
11            p = p+"||";
12        }
13        return p;
14    }
15 }
16
```

Problems @ Javadoc Declaration Console

```
<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\javaw.exe
Mutating MatrixAddition
Tests: MatrixAdditionTest
Mutation points = 15, unit test time limit 2.01s
...M FAIL: (MatrixAddition.java:4): 3 -> 4
M FAIL: (MatrixAddition.java:4): 3 -> 4
.....
Jumbling took 8.013s
Score: 86%
```

6. Write a program to Print Prime Numbers from 1 to 100 using Scanner Class and For Loop.

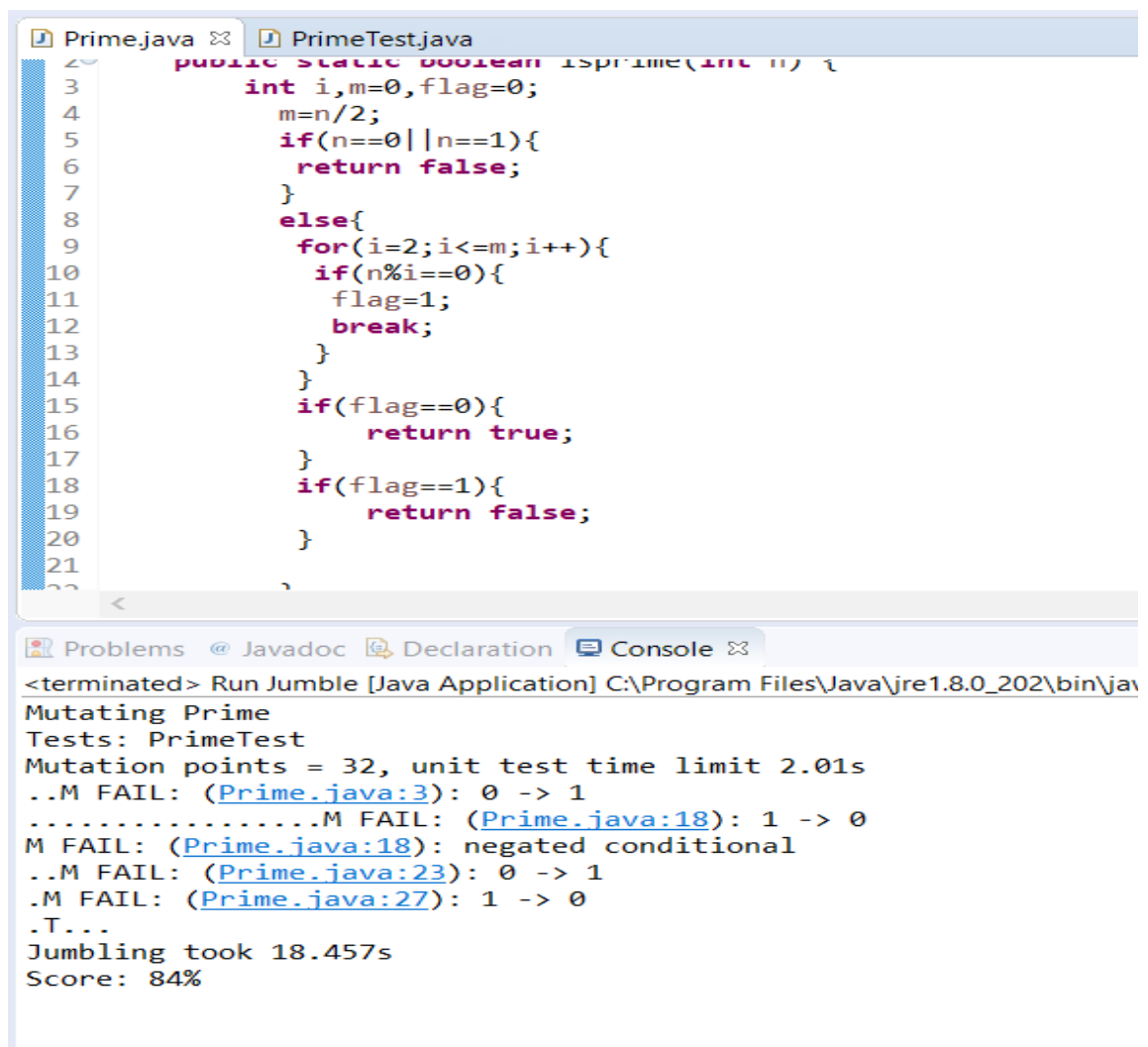


```
1 import junit.framework.TestCase;
2
3 public class PrimeTest extends TestCase {
4
5     public void test1() {
6         assertEquals("2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 ", Prime.prime());
7     }
8
9 }
```

Finished after 0.022 seconds

Runs: 1/1 Errors: 0 Failures: 0

> PrimeTest (Runner: JUnit 4) (0.000 s)

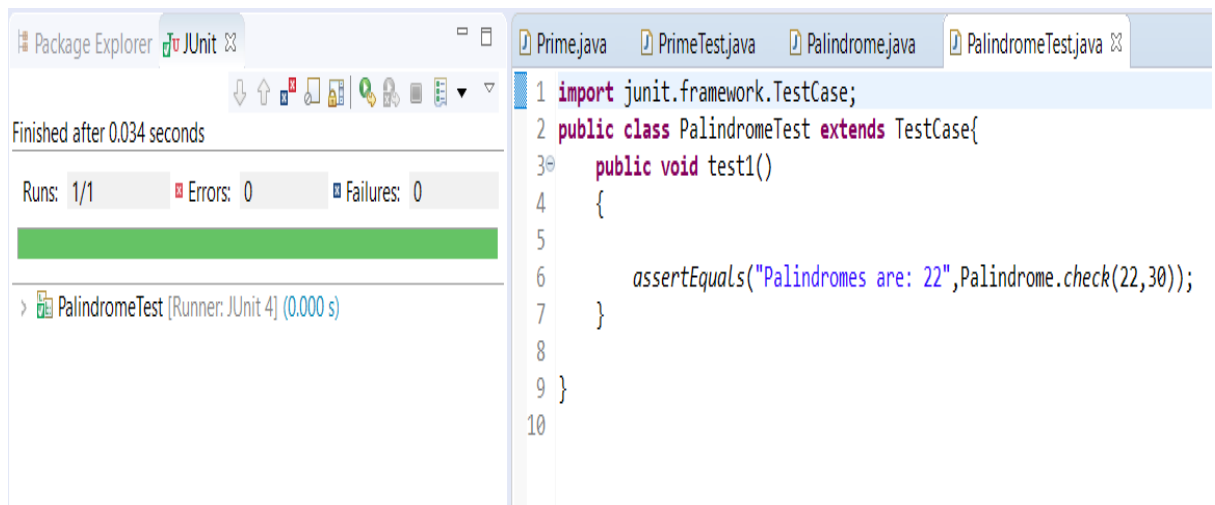


```
2 public static boolean isPrime(int n) {
3     int i, m=0, flag=0;
4     m=n/2;
5     if(n==0 || n==1){
6         return false;
7     }
8     else{
9         for(i=2; i<=m; i++){
10             if(n%i==0){
11                 flag=1;
12                 break;
13             }
14         }
15         if(flag==0){
16             return true;
17         }
18         if(flag==1){
19             return false;
20         }
21     }
22 }
```

Problems @ Javadoc Declaration Console

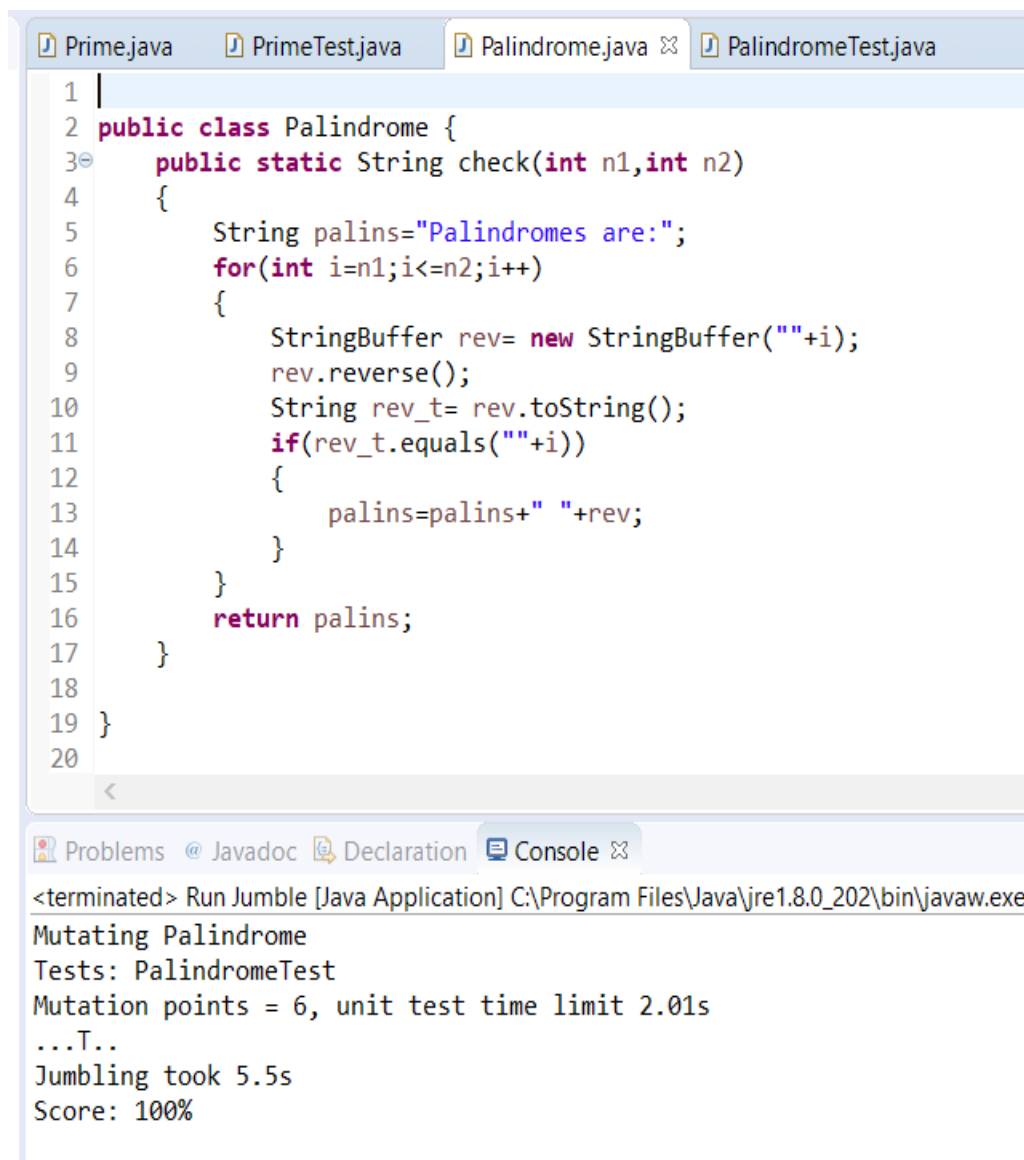
<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\jav
Mutating Prime
Tests: PrimeTest
Mutation points = 32, unit test time limit 2.01s
..M FAIL: (Prime.java:3): 0 -> 1
.....M FAIL: (Prime.java:18): 1 -> 0
M FAIL: (Prime.java:18): negated conditional
..M FAIL: (Prime.java:23): 0 -> 1
..M FAIL: (Prime.java:27): 1 -> 0
..T...
Jumbling took 18.457s
Score: 84%

7. Write a program to generate palindrome of numbers.



The screenshot shows an IDE with a Package Explorer on the left and a code editor on the right. The Package Explorer shows a JUnit test run for PalindromeTest, which finished after 0.034 seconds with 1/1 runs, 0 errors, and 0 failures. The code editor shows the PalindromeTest.java file with the following code:

```
1 import junit.framework.TestCase;
2 public class PalindromeTest extends TestCase{
3     public void test1()
4     {
5
6         assertEquals("Palindromes are: 22", Palindrome.check(22, 30));
7     }
8 }
9
10
```



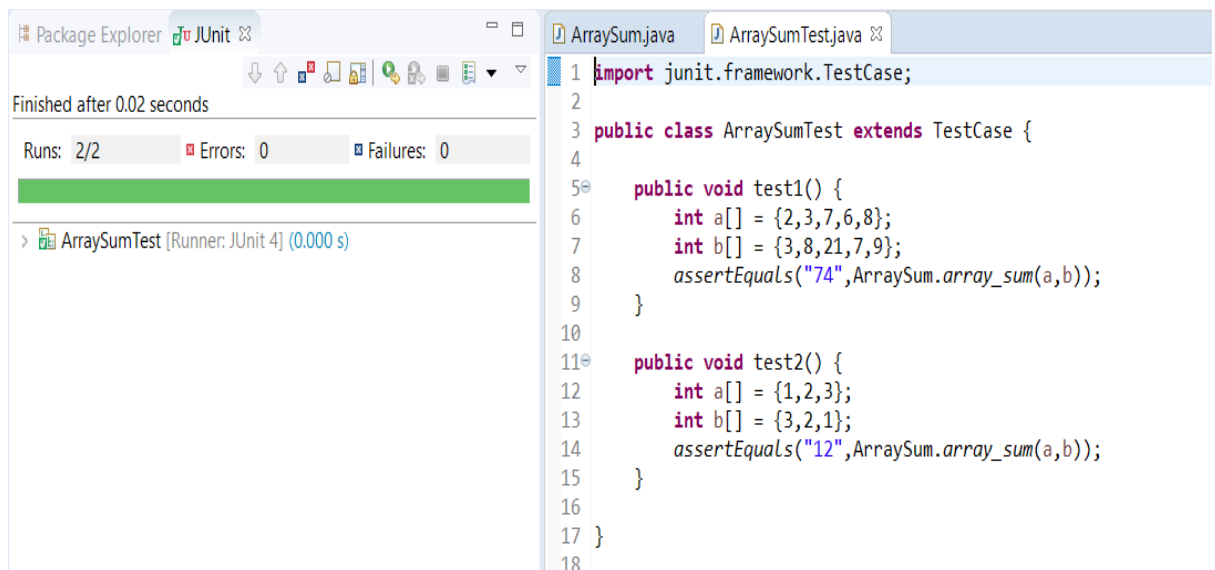
The screenshot shows an IDE with a code editor displaying the Palindrome.java file. The code is as follows:

```
1
2 public class Palindrome {
3     public static String check(int n1, int n2)
4     {
5         String palins="Palindromes are: ";
6         for(int i=n1; i<=n2; i++)
7         {
8             StringBuffer rev= new StringBuffer(""+i);
9             rev.reverse();
10            String rev_t= rev.toString();
11            if(rev_t.equals(""+i))
12            {
13                palins=palins+" "+rev;
14            }
15        }
16        return palins;
17    }
18 }
19
20
```

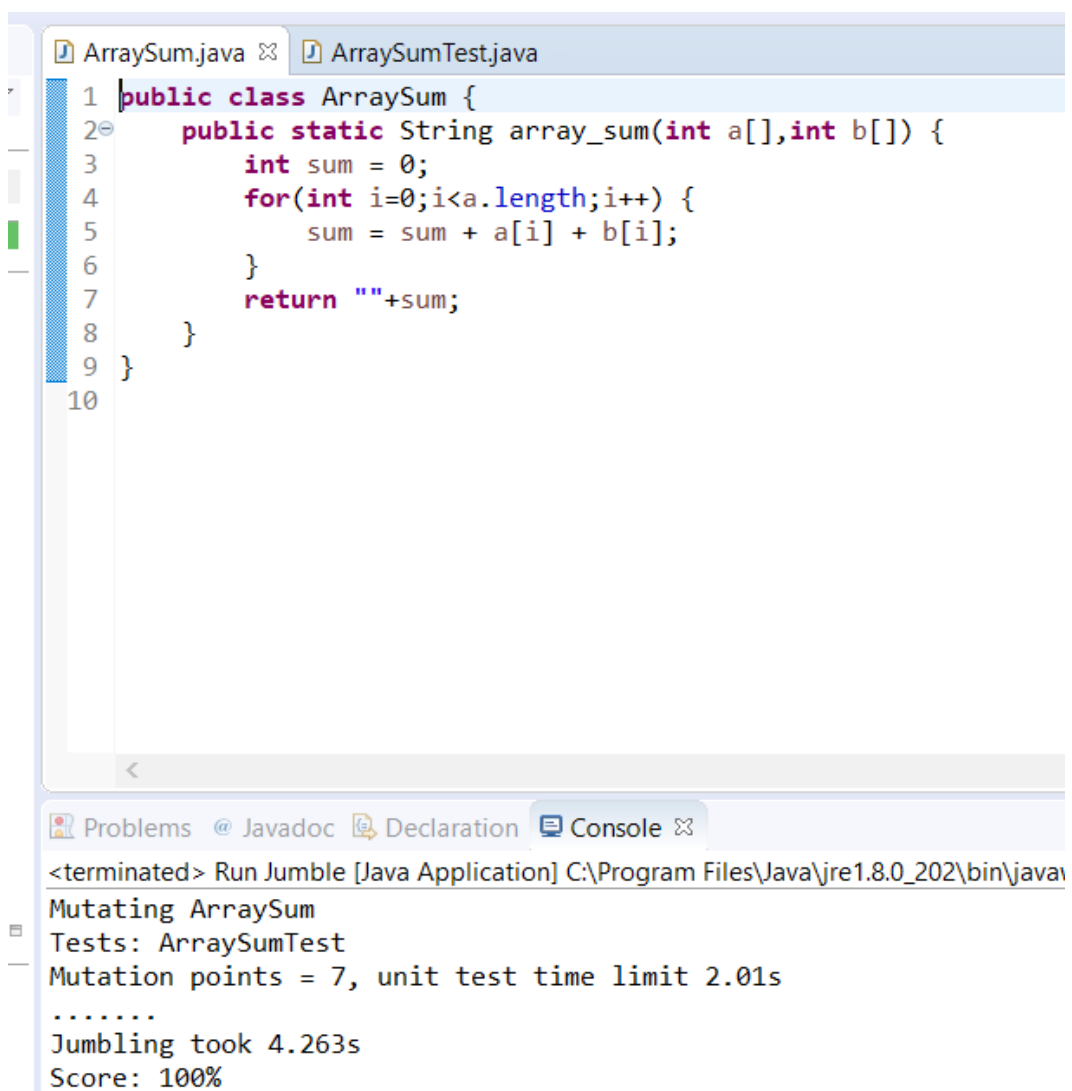
Below the code editor, the Console tab shows the output of the program:

```
<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\javaw.exe
Mutating Palindrome
Tests: PalindromeTest
Mutation points = 6, unit test time limit 2.01s
...T..
Jumbling took 5.5s
Score: 100%
```


8. Write a program to find out sum of two array.



```
1 import junit.framework.TestCase;
2
3 public class ArraySumTest extends TestCase {
4
5     public void test1() {
6         int a[] = {2,3,7,6,8};
7         int b[] = {3,8,21,7,9};
8         assertEquals("74",ArraySum.array_sum(a,b));
9     }
10
11     public void test2() {
12         int a[] = {1,2,3};
13         int b[] = {3,2,1};
14         assertEquals("12",ArraySum.array_sum(a,b));
15     }
16
17 }
18
```



```
1 public class ArraySum {
2     public static String array_sum(int a[],int b[]) {
3         int sum = 0;
4         for(int i=0;i<a.length;i++) {
5             sum = sum + a[i] + b[i];
6         }
7         return ""+sum;
8     }
9 }
10
```

<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\java
Mutating ArraySum
Tests: ArraySumTest
Mutation points = 7, unit test time limit 2.01s
.....
Jumbling took 4.263s
Score: 100%

9. Write a program to check whether the number is even or odd.



The screenshot shows an IDE with two tabs: `OddEven.java` and `OddEvenTest.java`. The `OddEvenTest.java` file contains the following code:

```
1 import junit.framework.TestCase;
2
3 public class OddEvenTest extends TestCase {
4
5     public void test1() {
6         assertEquals("Even", OddEven.add_even(44));
7     }
8
9     public void test2() {
10        assertEquals("Odd", OddEven.add_even(77));
11    }
12
13 }
```

The Package Explorer on the left shows the test runner `OddEvenTest [Runner: JUnit 4] (0.001 s)`. The JUnit progress bar indicates "Finished after 0.02 seconds" with "Runs: 2/2", "Errors: 0", and "Failures: 0".



The screenshot shows the `OddEven.java` file with the following code:

```
1 public class OddEven {
2     public static String odd_even(int n) {
3         if(n%2==0) {
4             return "Even";
5         }
6         else {
7             return "Odd";
8         }
9     }
10 }
11
```

The Console window at the bottom shows the output of the JUnit test run:

```
<terminated> Run Jumble [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\javaw.exe
Mutating OddEven
Tests: OddEvenTest
Mutation points = 7, unit test time limit 2.0s
.....
Jumbling took 3.834s
Score: 100%
```

10. Write a program for binary to hexa-decimal conversion.

