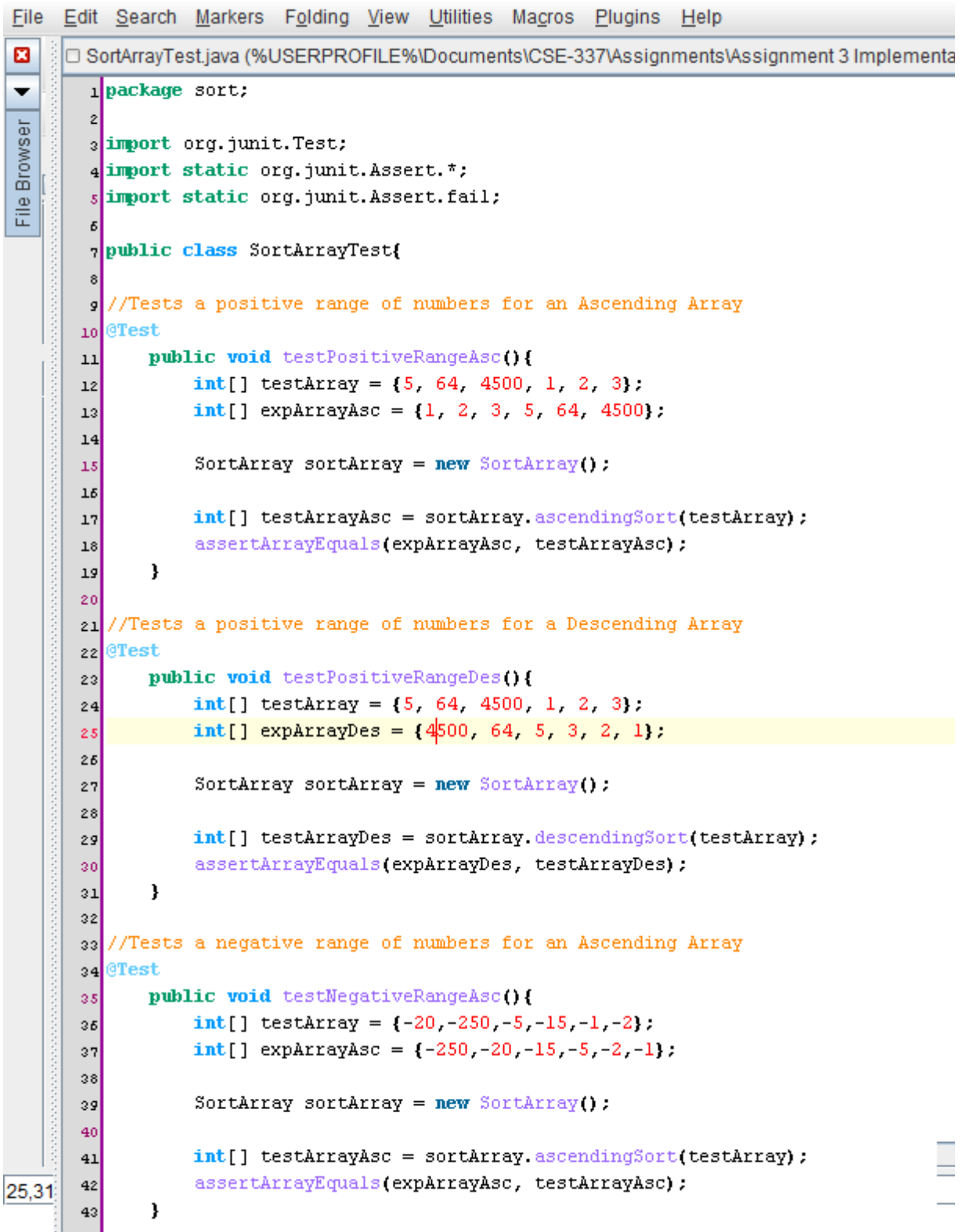


## SortArray.java

```
SortArray.java (%USERPROFILE%\Documents\CSE-337\Assignments\Assignment 3 Implementation\src\sort\
1 package sort;
2
3 import java.util.Arrays;
4
5 /**
6  * Rachel Glomski
7  * CSE 337 - Assignment 3
8  *
9  * This class has the ability to sort a given array in ascending
10 * or descending order
11 */
12
13 public class SortArray{
14
15
16     public static void main(String[] args){
17         SortArray sortArray = new SortArray();
18     }
19
20
21     public int[] ascendingSort(int[] arr){
22         Arrays.sort(arr); //built-in ascending sorting
23         return arr;
24     }
25
26     public int[] descendingSort(int[] arr){
27         Arrays.sort(arr);
28         //steps into array and swaps first and last element until
29         //reaching the middle, reversing the array
30         for(int i = 0; i < arr.length/2; i++){
31             int tempSort = arr[i];
32             arr[i] = arr[arr.length-1-i];
33             arr[arr.length-1-i] = tempSort;
34         }
35         return arr;
36     }
37 }
```

## SortArrayTest.java



```

File Edit Search Markers Folding View Utilities Macros Plugins Help
SortArrayTest.java (%USERPROFILE%\Documents\CSE-337\Assignments\Assignment 3 Implementa
1 package sort;
2
3 import org.junit.Test;
4 import static org.junit.Assert.*;
5 import static org.junit.Assert.fail;
6
7 public class SortArrayTest{
8
9 //Tests a positive range of numbers for an Ascending Array
10 @Test
11     public void testPositiveRangeAsc(){
12         int[] testArray = {5, 64, 4500, 1, 2, 3};
13         int[] expArrayAsc = {1, 2, 3, 5, 64, 4500};
14
15         SortArray sortArray = new SortArray();
16
17         int[] testArrayAsc = sortArray.ascendingSort(testArray);
18         assertEquals(expArrayAsc, testArrayAsc);
19     }
20
21 //Tests a positive range of numbers for a Descending Array
22 @Test
23     public void testPositiveRangeDes(){
24         int[] testArray = {5, 64, 4500, 1, 2, 3};
25         int[] expArrayDes = {4500, 64, 5, 3, 2, 1};
26
27         SortArray sortArray = new SortArray();
28
29         int[] testArrayDes = sortArray.descendingSort(testArray);
30         assertEquals(expArrayDes, testArrayDes);
31     }
32
33 //Tests a negative range of numbers for an Ascending Array
34 @Test
35     public void testNegativeRangeAsc(){
36         int[] testArray = {-20,-250,-5,-15,-1,-2};
37         int[] expArrayAsc = {-250,-20,-15,-5,-2,-1};
38
39         SortArray sortArray = new SortArray();
40
41         int[] testArrayAsc = sortArray.ascendingSort(testArray);
42         assertEquals(expArrayAsc, testArrayAsc);
43     }

```

(continued)

```

45 //Tests a negative range of numbers for a Descending Array
46 @Test
47     public void testNegativeRangeDes(){
48         int[] testArray = {-20,-250,-5,-15,-1,-2};
49         int[] expArrayDes = {-1,-2,-5,-15,-20,-250};
50
51         SortArray sortArray = new SortArray();
52
53         int[] testArrayDes = sortArray.descendingSort(testArray);
54         assertEquals(expArrayDes, testArrayDes);
55     }
56
57 //Tests a negative and positive range of numbers for an Ascending Array
58 @Test
59     public void testBothAsc(){
60         int[] testArray = {-5, 3, 3, -2, 0, 1};
61         int[] expArrayAsc = {-5, -2, 0, 1, 3, 3};
62
63         SortArray sortArray = new SortArray();
64
65         int[] testArrayAsc = sortArray.ascendingSort(testArray);
66         assertEquals(expArrayAsc, testArrayAsc);
67     }
68
69 //Tests a negative and positive range of numbers for a Descending Array
70 @Test
71     public void testBothDes(){
72         int[] testArray = {-5, 3, 3, -2, 0, 1};
73         int[] expArrayDes = {3, 3, 1, 0, -2, -5};
74
75         SortArray sortArray = new SortArray();
76
77         int[] testArrayDes = sortArray.descendingSort(testArray);
78         assertEquals(expArrayDes, testArrayDes);
79     }
80 }

```