Leah Mirch CIS 285 Final:

12-14-2023

Question 1:

Original SelectionSort code:

**public** **class** SelectionSort {

**public** **void** selectionSort(**int**[] arr) {

**if** (arr == **null** || arr.length <= 1) {

**return**;

}

**for** (**int** i = 0; i < arr.length - 1; i++) {

**int** minIndex = i;

**for** (**int** j = i + 1; j < arr.length; j++) {

**if** (arr[j] < arr[minIndex]) {

minIndex = j;

}

}

**if** (minIndex <= i) {

swap(arr, i, minIndex);

}

}

}

**private** **void** swap(**int**[] arr, **int** i, **int** j) {

**int** temp = arr[i];

arr[j] = arr[i];

arr[j] = temp;

}

}

Developed SelectionSortTest code:

**import** org.junit.jupiter.api.Test;

**import** **static** org.junit.jupiter.api.Assertions.*assertArrayEquals*;

**public** **class** SelectionSortTest {

@Test

**public** **void** testPositive() {

SelectionSort selectionSort = **new** SelectionSort();

**int**[] arr = {5, 3, 1, 4, 2};

selectionSort.selectionSort(arr);

**int**[] expected = {1, 2, 3, 4, 5};

*assertArrayEquals*(expected, arr);

}

@Test

**public** **void** testNegatives() {

SelectionSort selectionSort = **new** SelectionSort();

**int**[] arr = {-5, -3, -1, -4, -2};

selectionSort.selectionSort(arr);

**int**[] expected = {-5, -4, -3, -2, -1};

*assertArrayEquals*(expected, arr);

}

@Test

**public** **void** testMixed() {

SelectionSort selectionSort = **new** SelectionSort();

**int**[] arr = {-5, 3, 0, -4, 2};

selectionSort.selectionSort(arr);

**int**[] expected = {-5, -4, 0, 2, 3};

*assertArrayEquals*(expected, arr);

}

}

Based off of the current SelectionSort code, these tests do not pass, shown below:  
A screenshot of a computer program

Description automatically generated

This is the updated SelectionSort code:  
**public** **class** SelectionSort {

**private** **int** temp;

**public** SelectionSort() {

}

**public** **int**[] selectionSort(**int**[] arr) {

**for** (**int** i = 0; i < arr.length; ++i) {

**for** (**int** j = i + 1; j < arr.length; ++j) {

**if** (arr[i] > arr[j]) {

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

**return** arr;

}

}

The corrected version simplifies the selection sort implementation by directly swapping elements in the array using a temporary variable, removing the need for a separate swap method. It returns the sorted array as the result, providing a more convenient interface. The original version had errors in the swap logic and included unnecessary conditions, making it less efficient and prone to issues.

Which shows that all the Junit Test cases now pass:  
A screenshot of a computer program

Description automatically generated