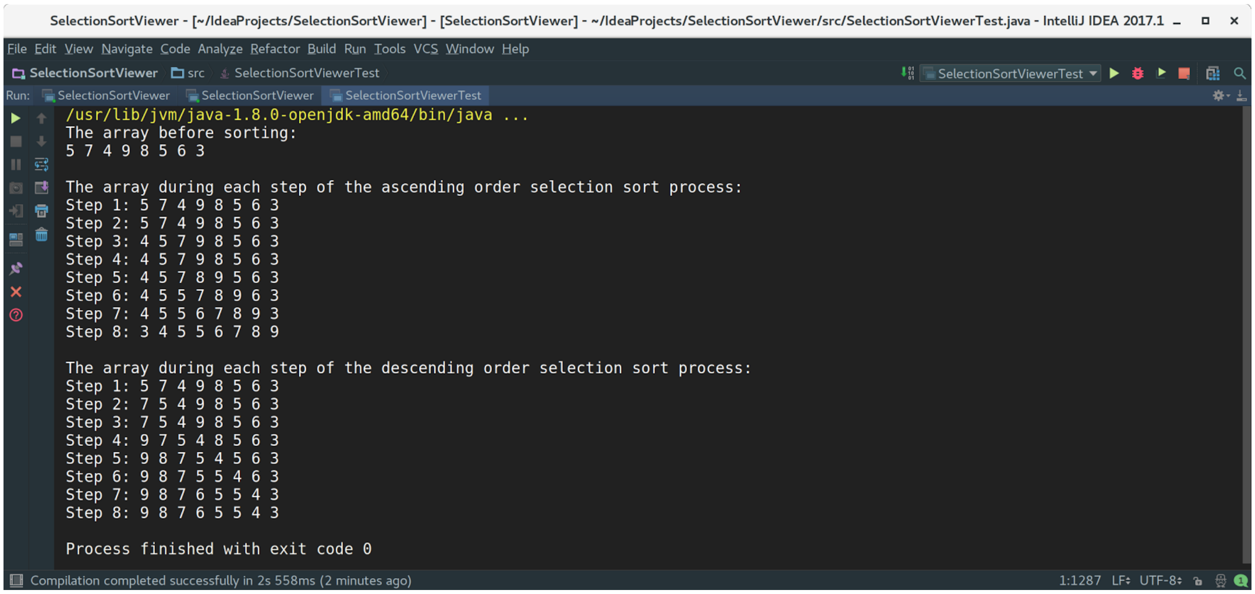
Micah Courey
CSC400: Data Structures and Algorithms
Critical Thinking Module 6
Option #1
April 28, 2017

Selection Sort Big-Oh Analysis

Although the selection sort method only contains one for loop which contains two lines of code, the two lines of code make calls to two other methods. This means in order to determine the big-Oh of selection sort we must also analyse and factor in the two methods that are called. Each call to the swap method takes constant time so this means that the calls to swap are O(n). For the indexOfMin method, it contains a for loop which runs n times where n is the size of the array. Although the loop will run n-1, n-2… with each iteration this factor isn't relevant to determining Big-Oh. When combining all the parts of the selection sort method we can determine that the Big-Oh of selection sort is O(n^2).

Program Screenshots

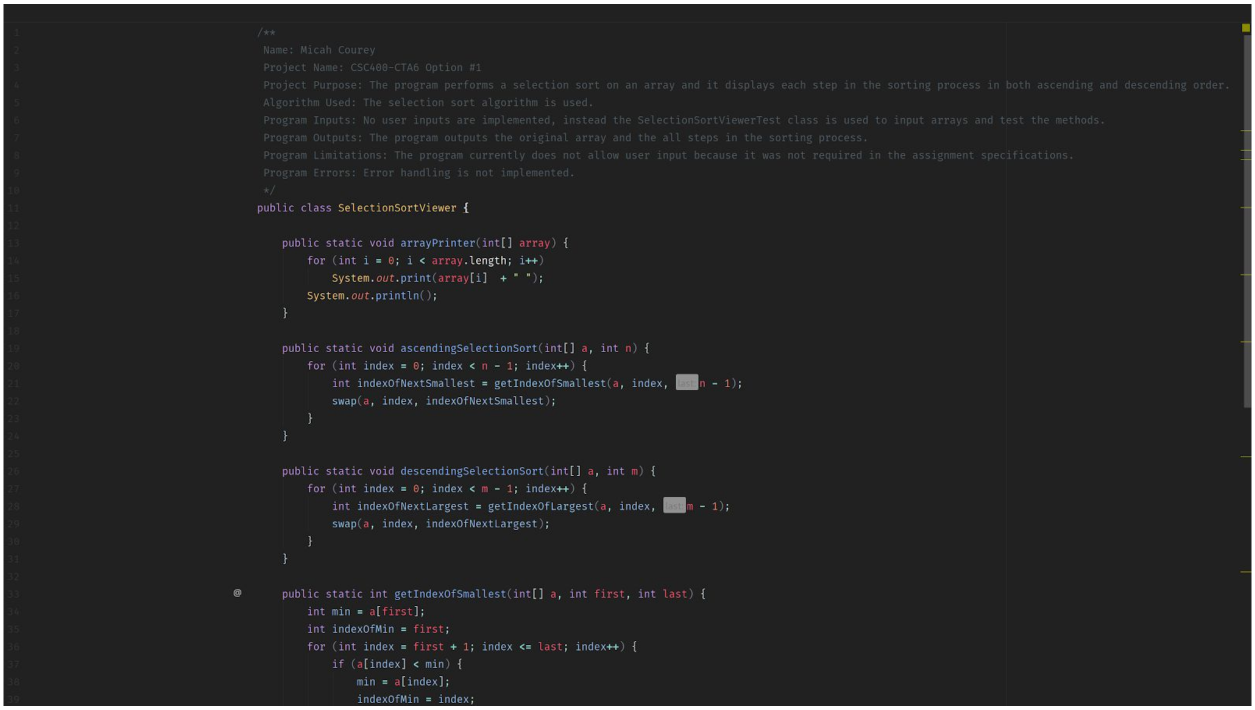
Program Output


SelectionSortViewer - [~/IdeaProjects/SelectionSortViewer] - [SelectionSortViewer] - ~/IdeaProjects/SelectionSortViewer/src/SelectionSortViewerTest.java - IntelliJ IDEA 2017.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
SelectionSortViewer
src
SelectionSortViewerTest
SelectionSortViewerTest
Run:
SelectionSortViewer
SelectionSortViewer
SelectionSortViewerTest
1:1287 LF÷ UTF-8÷

```
/usr/lib/jvm/java-1.8.0-openjdk-amd64/bin/java ...
The array before sorting:
5 7 4 9 8 5 6 3

The array during each step of the ascending order selection sort process:
Step 1: 5 7 4 9 8 5 6 3
Step 2: 5 7 4 9 8 5 6 3
Step 3: 4 5 7 9 8 5 6 3
Step 4: 4 5 7 9 8 5 6 3
Step 5: 4 5 7 8 9 5 6 3
Step 6: 4 5 5 7 8 9 6 3
Step 7: 4 5 5 6 7 8 9 3
Step 8: 3 4 5 5 6 7 8 9

The array during each step of the descending order selection sort process:
Step 1: 5 7 4 9 8 5 6 3
Step 2: 7 5 4 9 8 5 6 3
Step 3: 7 5 4 9 8 5 6 3
Step 4: 9 7 5 4 8 5 6 3
Step 5: 9 8 7 5 4 5 6 3
Step 6: 9 8 7 5 5 4 6 3
Step 7: 9 8 7 6 5 5 4 3
Step 8: 9 8 7 6 5 5 4 3

Process finished with exit code 0
```

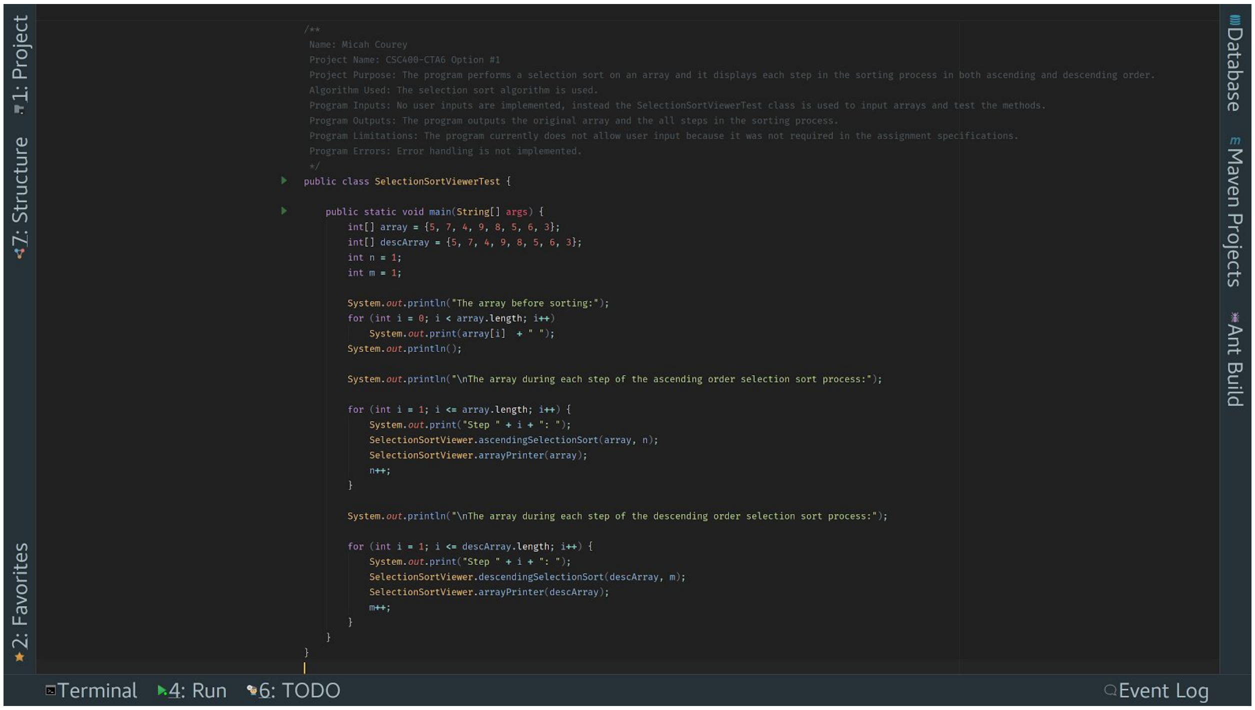
Compilation completed successfully in 2s 558ms (2 minutes ago)

SelectionSortViewer Class



```java
/**
 Name: Micah Courey
 Project Name: CSC400-CTA6 Option #1
 Project Purpose: The program performs a selection sort on an array and it displays each step in the sorting process in both ascending and descending order.
 Algorithm Used: The selection sort algorithm is used.
 Program Inputs: No user inputs are implemented, instead the SelectionSortViewerTest class is used to input arrays and test the methods.
 Program Outputs: The program outputs the original array and the all steps in the sorting process.
 Program Limitations: The program currently does not allow user input because it was not required in the assignment specifications.
 Program Errors: Error handling is not implemented.
 */
public class SelectionSortViewer {

    public static void arrayPrinter(int[] array) {
        for (int i = 0; i < array.length; i++)
            System.out.print(array[i]  + " ");
        System.out.println();
    }

    public static void ascendingSelectionSort(int[] a, int n) {
        for (int index = 0; index < n - 1; index++) {
            int indexOfNextSmallest = getIndexOfSmallest(a, index, last: n - 1);
            swap(a, index, indexOfNextSmallest);
        }
    }

    public static void descendingSelectionSort(int[] a, int m) {
        for (int index = 0; index < m - 1; index++) {
            int indexOfNextLargest = getIndexOfLargest(a, index, last: m - 1);
            swap(a, index, indexOfNextLargest);
        }
    }

    public static int getIndexOfSmallest(int[] a, int first, int last) {
        int min = a[first];
        int indexOfMin = first;
        for (int index = first + 1; index <= last; index++) {
            if (a[index] < min) {
                min = a[index];
                indexOfMin = index;
```

SelectionSortViewerTest Class



```java
/**
 Name: Micah Courey
 Project Name: CSC400-CTA6 Option #1
 Project Purpose: The program performs a selection sort on an array and it displays each step in the sorting process in both ascending and descending order.
 Algorithm Used: The selection sort algorithm is used.
 Program Inputs: No user inputs are implemented, instead the SelectionSortViewerTest class is used to input arrays and test the methods.
 Program Outputs: The program outputs the original array and the all steps in the sorting process.
 Program Limitations: The program currently does not allow user input because it was not required in the assignment specifications.
 Program Errors: Error handling is not implemented.
 */
public class SelectionSortViewerTest {

    public static void main(String[] args) {
        int[] array = {5, 7, 4, 9, 8, 5, 6, 3};
        int[] descArray = {5, 7, 4, 9, 8, 5, 6, 3};
        int n = 1;
        int m = 1;

        System.out.println("The array before sorting:");
        for (int i = 0; i < array.length; i++)
            System.out.print(array[i]  + " ");
        System.out.println();

        System.out.println("\nThe array during each step of the ascending order selection sort process:");

        for (int i = 1; i <= array.length; i++) {
            System.out.print("Step " + i + ": ");
            SelectionSortViewer.ascendingSelectionSort(array, n);
            SelectionSortViewer.arrayPrinter(array);
            n++;
        }

        System.out.println("\nThe array during each step of the descending order selection sort process:");

        for (int i = 1; i <= descArray.length; i++) {
            System.out.print("Step " + i + ": ");
            SelectionSortViewer.descendingSelectionSort(descArray, m);
            SelectionSortViewer.arrayPrinter(descArray);
            m++;
        }
    }
}
```