**Test Plan Review**

## Test case 1:

* **Input**: file with all negative numbers
* **Objectives**: Test the functionality of the program with negative numbers.
* **Expected Output**: Sorted Table with sorted rows and columns
* **Result**: Pass

A picture containing device, meter

Description automatically generated

## Test case 2:

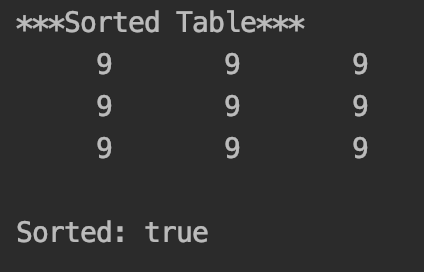
* **Input**: sorted array with ascending order values
* **Objectives**: Test the functionality of the program with an already sorted Table
* **Expected Output**: Sorted Table with sorted rows and columns
* **Result**: Pass

A picture containing device

Description automatically generated

## Test case 3:

* **Input**: sorted array with same numbers
* **Objectives**: Test the functionality of the program with input containing same value
* **Expected Output**: Sorted Table with sorted rows and columns
* **Result**: Pass



## Test case 4:

* **Input**: initialized array with random numbers
* **Objectives**: Test the functionality of the program with input containing random numbers both negative and positive.
* **Expected Output**: Sorted Table with sorted rows and columns
* **Result**: Pass

A picture containing device

Description automatically generated

## Test case 5:

* **Input**: initialized array with only ‘0’ and ‘1’
* **Objectives**: Test the functionality of the program with input containing only zeroes and ones.
* **Expected Output**: Sorted Table with sorted rows and columns
* **Result**: Pass

A picture containing device

Description automatically generated

## Test case 6:

* **Input**: initialized array with random numbers
* **Objectives**: Test the functionality of the method isSorted() with an unsorted Table
* **Expected Output**: False
* **Result**: Pass

A picture containing device

Description automatically generated

## Test case 7:

* **Input**: file containing 100 numbers (10x10)
* **Objectives**: Testing the functionality of the program with large input.
* **Expected Output**: Sorted Table with sorted rows and columns
* **Result**: Pass

A screen shot of a computer

Description automatically generated

**Clarity:**

Spelling and grammar were clear and proper. The plan was properly formatted and was easy to follow. It had a table of contents with relevant information.

**Plan:**

The plan in general was very good. It included specific test cases that would test the functionality of the TableSorter program. In addition, it provided very detailed instructions on how to run the tests. To increment functionality the test plan included JUnit test files to automatically test the test cases provided. In doing so I was able to test the functionality of the program both programmatically and manually.

**Conclusion:**

The plan was detailed and specific. The goal was to make it testable by anyone and I believe that goal was met. Even if the tester had to technical knowledge, testing files were provided to display readable results of the provided test cases. Relevant test cases were provided but, my only concern was of its completeness. I am not sure 7 test cases were sufficient to fully test the sortTable program.