[author name redacted]

CS 5387

2/19/20

Code Test Plan Review

**Code: 5064**

**Test Plan: 1301**

*Immediate Observations:* Test Plan 1301 requires Junit testing however Code 5064 is not designed to work with Junit testing because Code 5064 TableSorter.java is a main class where TableSorter objects cannot be created for Junit testing.

*Moving forward with Test Plan 1301 via hardcoding and no JUnit testing:*

**U01: Pass Empty Array (SortTable)**

Description: This test requires declaring a Table object without initializing the values so they are null. TableSorter is then tested on the null Table.

Expected Result: “Error is logged, but program continues processing.”

Actual Result: A compile error occurs because the Table class does not allow Table objects to be created without both an integer to indicate the size of the table and an integer array to indicate the values of the table.

***Observations:*** This test could not be completed because the Table class does not allow an object to be created without initializing the parameter values.

**U02: Pass a Table with only position [0][0] (SortTable)**

Description: This test requires the creation of a Table object of size 1 with "only one valid int value inside”. TableSorter is then tested on a Table of size 1x1.

Expected Result: “Method completes successfully”

Actual Result: Method compiles and runs without errors.

***Observations:*** During black box testing, the definition of “one valid int” is not obvious. The test should specify which integer value should be used during this particular test. Also, a definition of what a successful completion of a method looks like should be established. How do I know if it completed successfully? What does success look like in this case? Perhaps including a print statement of the table.toString() that shows the table as an expected result would give the tester a more accurate description of what the expected result should look like.

**U03: Pass a Table with a size of 4 (SortTable)**

Description: This test requires the creation of a Table object of size 4 with “valid int values”. It then requires the table to be passed through the sortable method then the isSorted method. The proof that the sortable method sorted the table correctly is determined by the output of the isSorted method.

Expected Result: sortable “method completes successfully” and the isSorted method returns “True.”

Actual Result: sortable method compiled and ran without errors. IsSorted method returned “true”.

***Observations:*** During black box testing, the definition of “valid int values” is not obvious. The test should specify which integer values should be used during this particular test. Also, a definition of what a successful completion of a method looks like should be established. How do I know if it completed successfully? What does success look like in this case? Moreover, relying on the isSorted method output to test the validity of the sortable method can be faulty because it assumes the isSorted method works correctly. This also seems to test the isSorted method more than the sortable method because the output of the sortable method is not observed directly.

**U04: Pass Empty Array (isSorted)**

Description: This test requires declaring a Table object without initializing the values so they are null. isSorted is then tested on the null Table.

Expected Result: “Error is logged, but program continues processing.”

Actual Result: A compile error occurs because the Table class does not allow Table objects to be created without both an integer to indicate the size of the table and an integer array to indicate the values of the table.

***Observations:*** This test could not be completed because the Table class does not allow an object to be created without initializing the parameter values.

**U05: Pass a Table with only position [0][0] (isSorted)**

Description: This test requires the creation of a Table object of size 1 with "only one valid int value inside”. isSorted is then tested on a Table of size 1x1.

Expected Result: “Method completes successfully and returns a value of ‘True’.”

Actual Result: Method compiles and runs without errors and returns “true”.

***Observations:*** During black box testing, the definition of “one valid int” is not obvious. The test should specify which integer value should be used during this particular test.

**U06: Pass a sorted Table with a size of 4 (isSorted)**

Description: This test requires the creation of a Table object of size 4 with the integer array values [1, 2, 3, 4]. It then tests the isSorted method only.

Expected Result: “Method completes succesfully” and isSorted returns True.

Actual Result: The isSorted compiles and runs without errors and returns “true”.

***Observations:*** This is a good test to determine if the isSorted method can detect a sorted table.

**U07: Pass an unsorted Table with a size of 4 (isSorted)**

Description: This test requires the creation of a Table object of size 4 with the integer array values [2, 3, 1, 4]. It then tests the isSorted method only.

Expected Result: isSorted “method completes successfully and returns a ‘False’ value.”

Actual Result: The isSorted method compiles and runs without errors and returns “false”.

***Observations:*** This is a good test to determine if the isSorted method can detect an unsorted table of positive values.

**U08: Pass an unsorted Table with a size of 4 (isSorted)**

Description: This test requires the creation of a Table object of size 4 with the integer array values [-3, 4, -1, 2]. It then tests the isSorted method only.

Expected Result: isSorted “method completes successfully and returns a ‘False’ value.”

Actual Result: isSorted method compiles and runs without errors and returns “true.” Test failed.

***Observations:*** This a good test to determine if the isSorted method can detect an unsorted table with negative values. This test has shown that there is an error in the isSorted method.

**Further Observations:**

The sortable method sorting algorithm was not actually tested in any of these test cases:

* + There were only three tests for sortable:
    - The first test could not be completed because a table object without parameters could not be created.
    - The second test only tested whether sortable could handle a table of size 1x1 with one integer value and required no actual sorting.
    - The third test tested whether sortable could handle a 2x2 table with “valid int values”. Then the expected result analyzed actually came from the isSorted method and not from the sortable method directly. So if the isSorted method is faulty, then this test did not actually test the accuracy of the sortable method. Moreover, there was no specification of what values to use or if the values were sorted correctly.

Thus, the tests designed to test the sortable method don’t test the sorting algorithm and there are no further tests to show how sortable sorts sorted arrays, unsorted arrays, arrays containing duplicates, arrays containing negative values, or how it handles non integer values, etc.

Regarding the isSorted method tests: The U08 isSorted method test was successful in finding an error in the isSorted method.

**Conclusion:**

1. Does the software work as intended?
   1. These tests have proven that the isSorted method does not work as intended when given an unsorted array containing negative integers. For the reasons explained above, the accuracy of the sortable method is undetermined. However, because the isSorted method does not work as intended, it can be said that the software itself does not work as intended.
2. Does the test plan work as intended?
   1. As mentioned above, the test plan does not test the sortable method sorting algorithm. So it can be said that the test plan does not work as intended.