BB\_Test\_Teams

**3rd Review**

Program reviewed 7284

Test plan used 6314

The document provided looks more of a template to follow rather than a solid test plan. Test plan should have a step by step procedure so anyone with a basic background on programming and unit testing can implement it. This template seems to be provided by a course. Hence, anyone who has not taken such course will get lost on how to correctly document and follow it. It is a good template to use, but training or instructions from the expert user of this template is required.

I actually had to make adjustments to the code in order to follow the test plan. That is a contradiction to what a black box test plan is about.

1. **Testing Approach:**

Comments: This section is telling me to fill out some description of what the test is about. This type of work should’ve already been done so someone can have a better understanding of the test itself that is about to implement.

Table 1. TEST SUITs are well described and defined about the objectives of the test plan. The identifier IsSortedSorted, IsSortedCol and IsSortedRows are not shown in the code, if I were a tester I would be confused about what those actually are. Are they methods to test their veracity? I’m not sure, at this point it is hard to follow along. Same goes to other identifiers in the second TEST SUIT.

1. **Test XX:**

This sections specifies to test the isSorted() method, then I now realize what those identifiers mean. In the first description says what to do, but doesn’t say how to do it. Someone that’s not familiar with the code (Table.java) won’t be able to know how to instantiate a Table object, link files to run the code. Tester won’t know that it’s required to use Table.Gettable() method to link the txt file to the object and test it. Since, this is a blackbox testing, I believe it’s crucial to specify instructions on how to test without having to understand the code.

**Test Case 1 (testing isSorted()):**

* + Input:
    - 3x3: 1,2,3;4,5,6;7,8,9;
  + Expected Output: True

*The program was able to deliver the expected result from the documentation, test case passed.*

**Test Case 2 (testing isSorted()):**

* + Input:
    - 3,2,1; 3,2,1; 3,2,1 ;
  + Expected Output: False
  + Actual output: False

*The program was able to deliver the expected result from the documentation, test case passed.*

**Test Case 3 (testing isSorted()):**

* + Input:
    - 3,3,3;2,2,2;1,1,1;
  + Expected Output: sorted by human inspection.
  + Actual output: False

*The program was able to deliver the expected result from the documentation, test case passed.*

**Test Case 6 (testing sortable()):**

* + Input:
    - 1,2,3;4,5,6;7,8,9;
  + Expected Output: sorted by human inspection

*The program was able to deliver the expected result from the documentation, test case passed.*

**Test Case 7 (testing sortable()):**

* + Input:
    - 3,2,1;3,2,1;3,2,1;
  + Expected Output: sorted by human inspection

*The program was able to deliver the expected result from the documentation, test case passed.*

**Test Case 8 (testing sortable()):**

* + Input:
    - 3,3,3;2,2,2;1,1,1;
  + Expected Output: true
  + Reason: The sortable method should be able to sort tables that are unsorted.

*The program was able to deliver the expected result from the documentation, test case passed.*

***Conclusion:*** The overall test plan was good, although I’d like to remark that I had to modify the code in order to make use of the method in the code. Also, the template given can easily confuse the tester as it’s a template and not a series of execution steps.

Test case used do not cover cases where the code might fail such as empty files, 1D arrays, one number in the file, etc. in order to tackle failures, we must have cases where the code shows vulnerabilities.