<<author name redacted>>

**Black Box Testing Table Sort**

# Test Plan Version 1.0 2/12/2020

**Table of Contents**

1. Introduction
   1. Purpose
   2. Scope
   3. System Overview
   4. Document Overview
2. Requirements
   1. Functions Supported
   2. Analyze Limits
   3. Likely Errors
   4. Costly Errors
3. Testing Approach
4. Test
5. Test Schedule
6. Appendix

## 1. Introduction

### 1.1 Purpose

Identify faults in the sortable program. It needs to be able to test all sortable programs despites its execution. The program must be able to sort by both the X and the Y axis. This document will carry put a plan to be able to concretely prove the program works as it is intended to.

**1.2 Scope**

This is version 1 of the sortable program black box testing.

### 1.3 System Overview

This is black box testing, so we do not know the parameters of the program. We do know of the sortTable function and what it is intended to do. That being the case I will focus on this function and test its limitations feeding it input.

### 1.4 Document Overview

This is a black box test and must be compatible will all sortTable programs. We do not know how the program works. We know that the inputs must be sorted and that it intakes a file with numbers in the following format: 2 1 5 3 4 8 7 6 9.

### 2. Requirements

In order to undergo this test, we need a java virtual machine provided by the jdk, a test input file, table class, and lastly a sortTable.java.

**2.1 Functions supported**

Only the sortTable function will be supported.

#### 2.2 Analyze limits

Due to our limited knowledge of the tested program we only what we can input into it. We don’t know the variable type used or the implementation of the sort. **2.3 Likely Errors**

It is very likely that type errors will occur. This can include using the wrong type for input. In addition, since we don’t know the data types used a problem may also arise in the bounds for both positive and negative numbers. Also, it is likely that the sort will not work as it is supposed to.

#### 2.4 Costly Errors

Being that this is a controlled environment there is little to no risk. The machine creates a virtual environment so the computer and its surroundings is safe. The only negative effect is the output of an incorrect value.

## 3. Testing Approach

**Table 1: Test Plan**

|  |  |  |
| --- | --- | --- |
|  | **Test Plan** |  |
| **Description of Test Suite** |  |  |
| **Test Case Identifier** | **Objective** | **Criticality** |
| BB1 | Make sure it works accordingly | yes |
| BB2 | Check what data types it accepts | yes |
| BB3 | Check that it only accepts square inputs | no |
| BB4 | Check that a correct input yields a correct output | no |
| BB5 | Check that a partially correct input yields correct output | no |
| BB6 | Test ranges of input | no |
| BB7 | Check for consistent results | no |
| BB8 | Check for duplicate inputs | no |
| BB9 | Provide Empty input | no |
|  | Provide null input |  |
| BB10 | Insert negative numbers into input | no |

## 4. Test

**4.1 Test BB1**

**Objective:** Make sure the program works accordingly and produces the expected result. **Notes:** This is just a precaution to make sure that the program works correctly. This makes the test more valuable as the rest of the test will have less variables/ chances of errors. \*The following is a template of the type of tests that will be carried out since the tests have not been completed yet they would all read pending with a brief plan of attack.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test No.: BB1 | | | Current Status: Pending | |  |
| Test title: Correct functionality | | | | |  |
| Testing approach: Provide the system a normal input 3x3 9 1 8 2 7 3 6 4 6 | | | | |  |
| STEP    1 | OPERATOR ACTION    Provide the following input 9 1 8 2 7 3 6 4 6 | PURPOSE    Test the systems functionality | | EXEPCTED RESULTS    The rows and columns should be sorted | COMMENTS    This is a precaution to mitigate the points of failure. Will also explain why some inputs are inconsistent. |
| Concluding Remarks: Test, Test case… | | | | |  |
| Testing Team: | | | Date Completed: | |  |

## 5. Test Schedule

|  |  |  |
| --- | --- | --- |
| **Task and date** | **People** | **Description** |
| TBA |  | Testing will occur at a later time but will execute all testing cases acquired. |

## 6. Appendix

Blackbox Testing- A testing technique in which functionality of the application is tested without any knowledge of the code structure of the system being tested.

Test Plan- Document describing software testing scope and activities produced to formally test a system.

Test Cases- Set of conditions or variables that will be applied to a system by a tester to determine its functionality.