<<author name redacted>>

CS5387

**Black Box Test Plan Assignment**

**Assignment Purpose**: Use black box testing on the TableSort program

**Testing Notes:** Tests were written for creating tables first in all tests starting with 0 in order to see what is allowed in table so I would not have to again for both isSorted and sortTable later. Tests should be ran in order since isSorted and SortTable might rely on GetTable (tests starting with 0) tests. SortTable tests rely on isSorted tests to be completed due to the specification that it should sort a table so the isSorted is true, otherwise the sorted table could be printed and visually verified in order to eliminate the isSorted dependency.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0a | | | | Current Status: Pending | | |
| Test title: GetTable empty table | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Clear testTable.txt and save the file. Compile and run the program. Verify console output. | PURPOSE  To see if empty tables are allowed | | | EXEPCTED RESULTS  error | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0b.1 | | | | Current Status: Pending | | |
| Test title: GetTable not a number “/” | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “1 / 2 3” into testTable.txt without the parenthesis and save the file. Compile and run the program. Verify console output. | PURPOSE  To see if non-numbers or strings that may be converted to integer with ascii value close to 0-9 in tables are allowed | | | EXEPCTED RESULTS  error | COMMENTS  “/” is also ascii 47 in case of atoi |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0b.2 | | | | Current Status: Pending | | |
| Test title: GetTable not a number “:” | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “1 : 2 3” into testTable.txt without the parenthesis and save the file. Compile and run the program. Verify console output. | PURPOSE  To see if non-numbers or strings that may be converted to integer with ascii value close to 0-9 in tables are allowed | | | EXEPCTED RESULTS  error | COMMENTS  “:” is also ascii 58 in case of atoi |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0c | | | | Current Status: Pending | | |
| Test title: GetTable double type | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “1 2.1 3 4” into testTable.txt without the parenthesis and save the file. Compile and run the program. Verify console output. | PURPOSE  To see if double type in tables are allowed | | | EXEPCTED RESULTS  error | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0d | | | | Current Status: Pending | | |
| Test title: GetTable null value | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “1 “ “ 3 4” into testTable.txt without the outer parenthesis and save the file. Compile and run the program. Verify console output. | PURPOSE  To see if null values are allowed | | | EXEPCTED RESULTS  error | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0e | | | | Current Status: Pending | | |
| Test title: GetTable non-NxN (non-square) | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “1 2 3 4 5 6 7 8” into testTable.txt without the outer parenthesis and save the file. Compile and run the program. Verify console output. | PURPOSE  To see if not square tables could be allowed | | | EXEPCTED RESULTS  error | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0f | | | | Current Status: Pending | | |
| Test title: GetTable maxint and minint values | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “-2147483648 1 2 21474836647” into testTable.txt without the outer parenthesis and save the file. Enter “System.*out*.println(myTable.getSize());” to see if the console will print out the size indicating the table creation worked. Compile and run the program. Verify console output. | PURPOSE  To see if the smallest and largest integers in java will be allowed in table | | | EXEPCTED RESULTS  The console should print out “2”. | COMMENTS  Will show if negative integers are allowed.  Printing out the size is just an easier way to see the program ran through and the table is valid if there are no errors.  Part of BVA test |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0g | | | | Current Status: Pending | | |
| Test title: GetTable smaller than minint | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “-2147483649 1 2 3” into testTable.txt without the outer parenthesis and save the file. Enter “System.*out*.println(myTable.getSize());” to see if the console will print out the size indicating the table creation worked. Compile and run the program. Verify console output. | PURPOSE  To see if the table will accept an integer smaller than minint | | | EXEPCTED RESULTS  error | COMMENTS  Printing out the size is just an easier way to see the program ran through and the table is valid if there are no errors.  Part of BVA test |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 0h | | | | Current Status: Pending | | |
| Test title: GetTable larger than maxint | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “1 2 3 2147483648” into testTable.txt without the outer parenthesis and save the file. Enter “System.*out*.println(myTable.getSize());” to see if the console will print out the size indicating the table creation worked. Compile and run the program. Verify console output. | PURPOSE  To see if the table will accept an integer larger than maxint | | | EXEPCTED RESULTS  error | COMMENTS  Printing out the size is just an easier way to see the program ran through and the table is valid if there are no errors.  Part of BVA test |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 1 | | | | Current Status: Pending | | |
| Test title: isSorted single integer matrix | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” Create a TableSorter object with “TableSorter mySorter = **new** TableSorter();” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “1” into testTable.txt without the parenthesis and save the file. Use “**boolean** sorted = mySorter.isSorted(myTable);” to hold the result of isSorted. Use “System.out.println(sorted);” to display the result. Compile and run the program. Verify console output. | PURPOSE  A 1x1 matrix is the smallest size and is automatically sorted. | | | EXEPCTED RESULTS  The console should print out “true”. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 2 | | | | Current Status: Pending | | |
| Test title: isSorted sorted 3 X 3 integer matrix | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” Create a TableSorter object with “TableSorter mySorter = **new** TableSorter();” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “-2147483648 0 1 2 3 4 5 6 2147483647” into testTable.txt without the parenthesis and save the file. Use “**boolean** sorted = mySorter.isSorted(myTable);” to hold the result of isSorted. Use “System.out.println(sorted);” to display the result. Compile and run the program. Verify console output. | PURPOSE  Test with both rows and columns sorted. | | | EXEPCTED RESULTS  The console should print out “true”. | COMMENTS  May also catch common descending instead of ascending error if it doesn’t pass. |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 3 | | | | Current Status: Pending | | |
| Test title: isSorted 3 X 3 integer matrix with row not sorted | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” Create a TableSorter object with “TableSorter mySorter = **new** TableSorter();” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter “-3 -4 1 1 2 3 4 5 6” into testTable.txt without the parenthesis and save the file. Use “**boolean** sorted = mySorter.isSorted(myTable);” to hold the result of isSorted. Use “System.out.println(sorted);” to display the result. Compile and run the program. Verify console output. | PURPOSE  To check if it will catch that a row is not in ascending order | | | EXEPCTED RESULTS  The console should print out “false”. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 4 | | | | Current Status: Pending | | |
| Test title: isSorted 3 X 3 integer matrix with column not sorted | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” Create a TableSorter object with “TableSorter mySorter = **new** TableSorter();” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter  “-1 2 3 4 -2 5 6 7 8” into testTable.txt without the parenthesis and save the file. Use “**boolean** sorted = mySorter.isSorted(myTable);” to hold the result of isSorted. Use “System.out.println(sorted);” to display the result. Compile and run the program. Verify console output. | PURPOSE  To check if it will catch that a column is not in ascending order | | | EXEPCTED RESULTS  The console should print out “false”. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 5 | | | | Current Status: Pending | | |
| Test title: sortTable single integer matrix | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” Create a TableSorter object with “TableSorter mySorter = **new** TableSorter();” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter  “1” into testTable.txt without the parenthesis and save the file. Use “**boolean** sorted = mySorter.isSorted(myTable);” to hold the result of isSorted. Use “System.out.println(sorted);” to display the result.  Compile and run the program. Verify console output. | PURPOSE  To see how it handles trying to sort the one value matrix and if it considers a one value matrix to be sorted. | | | EXEPCTED RESULTS  The console should print out true. | COMMENTS  Assignment specifications state sortTable() should sort a table so that isSorted will be true, so this test relies on all isSorted tests to be completed and working correctly beforehand.  Printing out the table would allow testing without being dependent on isSorted being tested/correct |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: 6 | | | | Current Status: Pending | | |
| Test title: sortTable row and columns not sorted to sorted | | | | | | |
| Testing approach: Move Table.java and TableSorter.java into a file named “cs5387.” Create a Java class named “Tester”. Import package cs5387 using “import package cs5387;”. Create a main method using “**public** **static** **void** main(String[] Args) **throws** Exception { }”. Create a text file in a file named “testTable.txt” in the same package/file location “cs5387” to store integers separated by a space for the tables. Create the Table in the main method using “Table myTable = Table.*GetTable*("testTable.txt");” Create a TableSorter object with “TableSorter mySorter = **new** TableSorter();” | | | | | | |
| STEP  1 | OPERATOR ACTION  Enter  “9 8 7 4 21 4 3 15 -1” into testTable.txt without the parenthesis and save the file. Use “**boolean** sorted = mySorter.isSorted(myTable);” to hold the result of isSorted. Use “System.out.println(sorted);” to display the result. Compile and run the program. Verify console output. | PURPOSE  To have a matrix with out of order rows and columns sorted to a matrix with ascending rows and columns | | | EXEPCTED RESULTS  The console should print out “true” | COMMENTS  Assignment specifications state sortTable() should sort a table so that isSorted will be true, so this test relies on all isSorted tests to be completed and working correctly beforehand.  Printing out the table would allow testing without being dependent on isSorted being tested/correct.  Will show negatives get sorted correctly. Should imply if it passes that if only rows sorted and not cols or vice versa then it would also sort it and be true. |
| Concluding Remarks: | | | | | | |
| Testing Team: | | | Date Completed: | | | |