**Test Plan: 5679**

**Test Case 1:**

* Input: 2,147,483,647
* Expected Output: 2,147,483,647 should pass through the program without error and terminate

Reason: This test case was given to test the boundaries that can be given to the program.

*The program does not accept files, but rather has the user go into the program and write the array. For this test, the program returns true, indicating that this test passed, since the input was just one value.*

**Test Case 2:**

* Input: 2,147,483,646
* Expected Output: 2,147,483,646 should pass through the program without error and terminate

Reason: This test case was given to test the boundaries that can be given to the program.

*For this test, the program returns true, indicating that this test passed, since the input was just one value.*

**Test Case 3:**

* Input: 2,147,483,648
* Expected Output: 2,147,483,648 the program should fail with an error and terminate

Reason: This test case was given to test the boundaries that can be given to the program with the integer value.

*This test case failed and got a number format exception, because the largest possible integer that java accepts is 2,147,483,647, and the input was too high of an integer.*

**Test Case 4:**

* Input: {0, 1, -2147483648}
* Expected Output: {0, 1, -2147483648} should pass through the program without error and terminate.

Reason: This test case was given to test the boundaries and to test if an unsorted array will be sorted with the smallest negative value an integer can have.

*This test case failed and returned “ERROR!: Input must be square.” This test failed, because the array was a 1D array and the table requires a square matrix.*

**Test Case 5:**

* Input: {0, 1, -2147483647}
* Expected Output: {0, 1, -2147483647} should pass through the program without error and terminate.

Reason: This test case was given to test the boundaries and to test if an unsorted array will be sorted with the second smallest negative value an integer can have.

*This test case failed and returned “ERROR!: Input must be square.” This test failed, because the array was a 1D array and the table requires a square matrix.*

**Test Case 6:**

* Input: {0, 1, -2147483649}
* Expected Output: {0, 1, -2147483649} the program should fail with an error and terminate.

Reason: This test case was given to test the boundaries and to test if an unsorted array will be sorted with the smallest negative value an integer can have.

*This test case failed and got a number format exception, because the smallest possible integer that java accepts is -2,147,483,648, and the input was too small of an integer.*

**Test Case 7:**

* Input: {0, 1, 2147483647}
* Expected Output: {0, 1, 2147483647} should pass through the program without error and terminate

Reason: This test case was given to test the boundaries and to test if a sorted array will be left sorted with the largest value an integer can have.

*This test case failed and returned “ERROR!: Input must be square.” This test failed, because the array was a 1D array and the table requires a square matrix.*

**Test Case 8:**

* Input: {0, 1, 2147483646}
* Expected Output: {0, 1, 2147483646} should pass through the program without error and terminate

Reason: This test case was given to test the boundaries and to test if a sorted array will be left sorted with the second largest value an integer can have.

*This test case failed and returned “ERROR!: Input must be square.” This test failed, because the array was a 1D array and the table requires a square matrix.*

**Test Case 9:**

* Input: {0, 1, 2147483648}
* Expected Output: {0, 1, 2147483648} the program should fail with an error and terminate.

Reason: This test case was given to test the boundaries and to test if a sorted array will be left sorted with a very large integer value.

*This test case failed and got a number format exception, because the largest possible integer that java accepts is 2,147,483,647, and the input was too high of an integer.*

**Test Case 10:**

* Input: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
* Expected Output: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10} should pass through the program without error and terminate

Reason: This test case was given to test if a sorted array with regular integers will be left sorted.

*This test case failed and returned “ERROR!: Input must be square.” This test failed, because the array was a 1D array and the table requires a square matrix.*

**Test Case 11:**

* Input: {0, 1, 2, 3, 4, a, b, c, d, e, f}
* Expected Output: {0, 1, 2, 3, 4, a, b, c, d, e, f} the program should fail with an error and terminate

Reason: This test case was given to test if an array with some letters will be sorted.

*This test case failed and got a number format exception, because there are letters in the array and not just integer values.*

**Test Case 12:**

* Input: {a, b, c, d, e, f, g, h, i, j, k, l}
* Expected Output: {a, b, c, d, e, f, g, h, i, j, k, l} the program should fail with an error and terminate

Reason: This test case was given to test if an array with all letters will be sorted.

*This test case failed and got a number format exception, because the array is letters and not an integer array.*

**Test Case 13:**

* Input: {1, 2, 3, 4, 5, !, @, #, $, %}
* Expected Output:{1, 2, 3, 4, 5, !, @, #, $, %} the program should fail with an error and terminate

Reason: This test case was given to test if an array with some special characters will be sorted.

*This test case failed and got a number format exception, because there are special characters in the array and not just integer values.*

**Test Case 14:**

* Input: {!, @, #, $, %, ^, &, \*, (, ), \_}
* Expected Output: {!, @, #, $, %, ^, &, \*, (, ), \_} the program should fail with an error and terminate

Reason: This test case was given to test if an array with all special characters will be sorted.

*This test case failed and got a number format exception, because the array made of special characters and not of integers.*

**The Test Plan Review:**

The test plan did not do what was intended. The test cases failed to test any square matrices, so it is unknown if the values would have been sorted in ascending order by the column. The test cases also tested arrays with integer values and that contained letters and special characters. It made the test cases that were completely letters and special characters unnecessary, because it was already established that the array had to be comprised of integers. The first test cases checked the boundaries of the integers being used in the code but kept using more test cases with those values, which already determined they would not work.