## Question 2

### b) Explanation of the code:

I created a Random TestCaseGenerator class using Java. The class generates arrays of random lengths, and fills the array with random integers.

```java
public class TestCaseGenerator {
    public static void main(String[] args) {

        // this for loop generates and test multiple random cases.
        for (int i = 0; i < 10; i++) { // number of test cases is set to 10
            // generates random length array of integers.
            Random random = new Random();
            int arrayLength = random.nextInt(15) + 2; // Array length between 2 and 15.
            int[] arrayToSort = new int[arrayLength];

            // makes array with random integers
            for (int j = 0; j < arrayLength; j++) {
                arrayToSort[j] = random.nextInt(100) - 50; // Integers between -50 and 49.
            }

            // prints the original unsorted array.
            System.out.println("Random Array: " + Arrays.toString(arrayToSort));

            // sort array using the buggy sort method (ignore last element)
            int[] sortedArray = sort(arrayToSort);

            // print the sorted array.
            System.out.println("Sorted Array: " + Arrays.toString(sortedArray));

            // Validate results
            int[] expectedArray = arrayToSort.clone();
            bubbleSort(expectedArray);
            boolean isTestPassed = Arrays.equals(sortedArray, expectedArray);
            System.out.println("Test Result: " + (isTestPassed ? "PASS" : "FAIL"));
            System.out.println("-------------------------------------");
        }
    }
```

Then I made another bubble sort class to sort the array. After sorting the arrays, we validate the results by comparing them to the original unsorted array.

```java
    public static void bubbleSort(int[] array) { bubbleSort(array, array.length); }

    //bubble sort
    2 usages
    public static void bubbleSort(int[] array, int length) {
        for (int i = 0; i < length - 1; i++) {
            for (int j = 0; j < length - i - 1; j++) {
                if (array[j] > array[j + 1]) {
                    int temp = array[j];
                    array[j] = array[j + 1];
                    array[j + 1] = temp;
                }
            }
        }
    }
}
```

To make sure my testing method works properly, I intentionally made a bug that would ignore the last integer in the array (leave it unsorted) so that the test tool reports the bug and fails.

Only way the test case would pass is when the last integer is already in the right position (sorted be default)

```java
// buggy sort method for testing purposes.
1 usage
public static int[] sort(int[] arrayToSort) {
    int[] sortedArray = arrayToSort.clone();

    // I intentionally ignored the last element during sorting so we can get some tests cases to fail.
    bubbleSort(sortedArray, length: sortedArray.length - 1);

    return sortedArray;
}
```

**d) Instructions for compiling and running your code:**

Ensure Java Development Kit (JDK) is installed on your system.

Ensure JAVA_HOME is set and the bin directory is added to the PATH in your environment variables.
Code source file is provided in the file, it is runnable on any IDE.

**e) Logs generated by the print statements & f) Logs for the random test executions**

```
Random Array: [39, 13, -19, -1, 26, -3]
Sorted Array: [-19, -1, 13, 26, 39, -3]
Test Result: FAIL
-----------------------------------
Random Array: [-32, 23, 9, -14, -1, -6, 27, -15, -33, 27]
Sorted Array: [-33, -32, -15, -14, -6, -1, 9, 23, 27, 27]
Test Result: PASS
-----------------------------------
Random Array: [-47, 18]
Sorted Array: [-47, 18]
Test Result: PASS
-----------------------------------
Random Array: [19, 30, 13, -44, -6, 33, 5, 6, -19, -30, 4, -27, -36, -16, -18]
Sorted Array: [-44, -36, -30, -27, -19, -16, -6, 4, 5, 6, 13, 19, 30, 33, -18]
Test Result: FAIL
-----------------------------------
Random Array: [13, -28, 30, 31, 19, -44, -26, 9, 29]
Sorted Array: [-44, -28, -26, 9, 13, 19, 30, 31, 29]
Test Result: FAIL
-----------------------------------
Random Array: [-5, 43, 9, 27, -11, 21, 10, -46, 31, 13, -25, -43, 36, 48]
Sorted Array: [-46, -43, -25, -11, -5, 9, 10, 13, 21, 27, 31, 36, 43, 48]
Test Result: PASS
-----------------------------------
Random Array: [-22, -13, 0, -8, -17, -34]
Sorted Array: [-22, -17, -13, -8, 0, -34]
Test Result: FAIL
-----------------------------------
Random Array: [13, -30, 23, 45, 4, -12, -22, 49, 34, -42]
Sorted Array: [-30, -22, -12, 4, 13, 23, 34, 45, 49, -42]
Test Result: FAIL
-----------------------------------
```

**B) Provide a context-free grammar to generate all the possible test-cases**

start→ test-case
test-case → array
array → [-50, 49] | array-length
array-length → 2 | ... | 15 → bubbleSort
bubbleSort → sorted array | integers
integers → -50 | -49 | ... | 0 | ... | 48 | 49 → validate results
Validate results → integer → "-50" | "-49" | "-48" | ... | "48" | "49"