Oswaldo Escobedo
CS-2401MW12

What is testing and why is so important?

The test code is a test method that helps us know if a unit of code works correctly. First, we need to decrypt and separate the code into smaller parts to see its behavior in each part. Second, we need to make use of test cases, these are methods that help us identify possible error in our code. Finally, we need to find a possible input that can lead to an error, in other words, we will make that the code crashes intentionally. Moreover, using testing code is important, because it helps us improve and fix our code, making the code resist errors.

Test Suite for the Method sortOfReverse: The purpose of this Test Suite is to verify that the sortOfReverse method does not contain typical errors of the arrays, such as that the method is not swapped correctly, does not manage to support negative numbers, that the array is empty, that the values can't be swapped because the array has an odd size, etc.

- **Test #1:** I chose this test case to verify if the method works with whatever input we choose. The purpose of using this test case is to identify if the method works with simple inputs. The output I expect is that the method runs without any problem. For instance, if we have an array with [4, 3, 2, 1] values, I expect the method to return an array with the following values [1, 2, 3, 4]. After testing it the method worked as expected.

- **Test #2:** I chose this test case to verify if the method works with negative values. The purpose of using this test case is so that (if the method has negative parameters) we identify this error and prepare the method for such situation and prevent it from throwing an exception. The output I expect is that the method runs without any problem. For example, if we have an array with [-1, -3, -4, -5] values, I expect the method to return an array with the following values [-5, -4, -3, -1]. After testing it the method worked as expected.

- **Test #3:** I chose this test case to know if the method works even despite having an empty parameter. The purpose of using this test case is to know if the method can work correctly when it has no integer in its parameters. The output I expect is that the method throws an Exception. After testing it, the method did not throw an exception and worked, this could be explained because even though we have nothing in the array we still "have" something… "nothing".

- **Test #4:** I chose this test case to know if the method can work when it has an odd length. The purpose of using this test case is to know if the method can fulfill its function even when it has an odd length. The output I expect is that the method reverses all the elements without throwing an exception or having errors. For instance, if we have an array with [3, 2, 1] values it returns [1, 2, 3]. After testing it the method worked as expected.

- **Test #5:** I chose this test case to know if the method can work when the elements in the array have the same value. The purpose of using this test case is to verify that the method supports this situation. I expect that the array returns as if nothing had changed. For example, if we have an array with [3, 3, 3] I expect [3, 3, 3]. After testing it the method worked as expected.

Test Suit for the Method cutString: The purpose of this Test Suite is to verify that the cutString method does not contain typical errors of the Strings, such as that the method tries to remove an element that does not exist, syntax errors, upper cases, etc.

- **Test #1:** I chose this case test to know if the code would work even if there is only one or no character in the string. The purpose of doing this test is to correct the error in case it throws an exception. What I hope will happen is that the method throws an indexOutOfBoundsException. After testing it, it throwed an exception, this could be explain because if we have only 1 char (neither a nor b) then the method proceeds to remove two characters, but if we remove 2 elements when we only have one element available this will result in a -1, so an indexOutOfBoundsException. Therefore, I needed to implement a try-catch statement fix the problem.

- **Test #2:** I chose this test case, because the method says to delete the first characters, except if the first character is an 'a' and the second one 'b', so what will happen if the first is a 'b' and the second an 'a'? The purpose of this test case is to know if the method will erase the first two characters or not. What I expect to happen is that the method will erase both characters, because as specified, the first one must be an 'a' and the second one must be a 'b'. After testing it the method worked as expected.

- **Test #3:** I chose this test case because I want to know if the method fulfill its function despite having ‘A’ and ‘B’ capitalized. The purpose of this test case is to know if the method erases the first two characters or not. I expect that the method returns a string that has had erased the first two characters. After testing it the method worked as expected.

- **Test #4:** I chose this test case because I want to know if the method will work even if there is neither an ‘a’ nor a ‘b’ present in the string. The purpose of this test is to know if the method will keep or remove the first two chars properly. I expect the output to remove the first two chars as it says in its description. After testing it the method worked as expected.

- **Test #5:** I chose this test case because I want to see the behavior of the method when there is two a’s in first two positions. The purpose of this test is to see if the method can handle such case. I expect the following output: “aaction” it returns a “ction”. However, the method behaved differently from what I was expecting, the output resulted is “action”, why? Because as it says in the description, we will only remove ‘a’ if it is in the first position (not on the second one).

Test suite for the method withoutTen: The purpose of this Test Suite is to verify that the withoutTen method does not contain typical errors of the arrays, such as that the method doesn't have 10's elements in the array, null values, negative numbers, that the array is empty, etc.

- **Test #1:** I chose the test assertArrayEquals because I want to verify if the code works with simple inputs, such as 1, 2, 3 (without using the number 10), if it does not work then it will return false. The purpose of this test was to make sure the code does not have any bugs. I expect the method and test to return the same array with my given values and that it returns true, so for instance, [1, 2, 3] -> [1, 2, 3]. After testing it the method worked as expected.

- **Test #2:** I chose the test assertNotNull because I want to make sure that the array does not have any null value and if it does then it must return false. The purpose of this test is to (in case the array has a negative value) improve our code so that it can be prepared for such situation. I expect the test to return true (meaning there is no null value within the array). After testing it the method worked as expected.

- **Test #3:** I chose the test assertArrayEquals because I want to verify if the method withoutTen is prepared to handle parameters with negative values. The purpose of this method is to improve our code so that it can handle such situation. I expect that the test returns true (meaning that the method can handle those values). After testing it the method worked as expected.

- **Test #4:** I chose the test assertArrayEquals because I want to make sure that the method can handle empty arrays without throwing an exception, if it does then we will proceed to improve the code. I expect that the test returns true and that runs without a problem. After testing it the method worked as expected.

- **Test #5:** I chose the test assertArrayEquals because I want to make sure that the code can fulfill its function when having only 10's in its parameters. The purpose of this test is that the code can handle that edge case. I expect [10, 10, 10] returns an array [0, 0, 0], in other words, that it returns true. After testing it the method worked as expected.

Test Suite for the Method bigArray: The purpose of this Test Suite is to verify that the bigArray method does not contain typical errors of the arrays, such as that the method creates a large array and crashes, null values, zero edge case, negative values, and having a math operation as a parameter.

- **Test #1:** I chose the test assertArrayEquals because I want to make sure that the method can handle big inputs/parameters. The purpose of this test is to (in case it can't process a big parameter) fix it and improve it. I expect (6) --> [1, 1, 2, 1, 2, 3, 1, 2, 3, 4, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 6]. After testing it the method worked as expected.

- **Test #2:** I chose the test assertNotNull because we want to make that the method is prepared in case there is a null value. The purpose of this test is to fix it and make it prepare for such situation. I expect that the test runs without a problem. After testing it the method worked as expected.

- **Test #3:** I chose the test assertArrayEquals because we need to verify if the method can handle a test case, such as 0. The purpose of this test is to analyze and fix (in case the method doesn't support this parameters) the method. I expect that the test runs without a problem and that it returns true. After testing it the method worked as expected.

- **Test #4:** I chose the test assertArrayEquals with the purpose to verify if the method can work with a negative value in its parameter. The purpose of this is to be prepared for an edge case

like this. I expect that the test returns true and that it runs without a problem. After testing it the method worked as expected.

- **Test #5:** I chose the test assertArrayEquals with the purpose that we can know that the method can work with integer division. The purpose of this method is to change the method in case it does not work. I expect that the test returns true and the expected value matches with the original. After testing it the method worked as expected.