Tyler Stovsky

705512370

1. One notable obstacle I overcame was writing the countFloatingPointValues function. To me, it felt the most similar to project 3 because I had to loop through a string and make sure each character was valid, in this case either a digit or a decimal point. After testing certain edge cases, such as multiple decimal points, I completed the function.

Another notable obstacle was the replaceFirstandLastOccurrences function. I struggled for a long time trying to figure out how to find only the first and last occurrence, but eventually I came up with a solution to store the index in a variable and update it every time it sees a second occurrence, which will eventually find the last occurrence.

In identicalValuesTogether, I also had trouble with the logic. I came up with the idea in the end to subtract the indices of duplicates and checked if that difference was equal to 1, which would mean they are next to each other.

shiftLeft was also tricky because I had trouble figuring out how to replace the last element with the new placeholder string, but I came up with a solution to shift the string just once and replace the end with the new placeholder, and then loop that shift the desired amount of times.

b.

General tests for all functions:

Empty strings ([“”, “”, “”, “”], 4)

Size value equal to zero ([“a”, “B”, “c”, “D”], 0)

Negative size value ([“A”, “b”, “C”, “Dd”], -60)

Array of zero elements ([], 0)

locateMaximum:

One empty string ([“C”, “E”, “”, “D”], 4)

Largest at the end ([“a”, “b”, “c”, “d”], 4)

Duplicate largest value ([“A”, “D”, “D”, “B”, “C"], 5)

Largest value in the middle ([“A”, “D”, “C”, “5”], 4)

Longer strings (([“bkepake”, “&oea”, “lop6%”, “Dooe30”], 4)

Size value shorter than array length ([“h”, “O”, “j”, “p”], 2)

countFloatingPointValues:

Empty string ([“503”, “”, “12.602”, “Joe”], 4)

No floating point values ([“Hello”, “b”, “j'], 3)

Floating point value with a plus or minus ([“+12.5”, “h”, “-14”], 3)

Floating point value with a comma ([“1200”, “1,000”, “14.6”, “hello”], 4)

Leading zeros ([“00.1234”, “12345”], 2)

Beginning with a decimal ([“j”, “.5803”, “42”], 3)

Only a decimal ([“56”, “48.8301”, “.”, “A”, “aogeoa”], 5)

Ending in a decimal ([“cs31”, “1234.”, “hi”], 3)

hasNoCapitals:

All capitals ([“H”, “O”, “A”, “Y”], 4)

All capitals longer strings ([“MSO”, “KGOWOA”, “EM”], 3)

Digits ([“123”, “90210”, “8”], 3)

One capital ([“b”, “g”, “A”, “d”], 4)

All lowercase (“n”, “r”, “p”], 3)

Longer all lowercase ([“abcd”, “hello”, “m”, “bam”], 4)

Capitals mixed in with lowercase ([“njlOq”, “iioT”, “yto”], 3)

Other symbols mixed in with lowercase ([“yotr$”, “89&h”, “m”, “jme4#”], 4)

One empty string all lowercase ([“mop”, “”, “bar”], 3)

identicalValuesTogether:

No identical values (“1234”, “Tyler”, “hello”, “can”, “ABC”], 5)

One identical value not together ([“a”, “b”, “a”, “d”], 4)

Set of identical value together and one separate ([“B”, “B”, “h”, “B”, “M”], 5)

One identical value together ([“cool”, “hi”, “hi”, “Joe”], 4)

Multiple identical values not together ([“v”, “F”, “v”, “F”, “u”], 5)

Multiple identical values, one together one not ([“up”, “boo”, “boo”, “up”], 4)

Multiple identical values together ([“foo”, “wow”, “wow”, “wow”, “b”, “b”], 6)

Two sets of the same identical value together ([“yes”, “yes”, “no”, “yes”, “yes”], 5)

Duplicate empty strings together ([“bar”, “”, “”, “no”], 4)

hasTwoOrMoreDuplicates:

No duplicates ([“789”, “chris”, “water”], 3)

One set of duplicates ([“Tyler”, “josh”, “Tyler”, “Eli”], 4)

Two sets of duplicates ([“abc”, “ABC”, “cdef”, “abc”, “ABC”], 5)

Three of the same duplicate ([“bye”, “sam”, “sam”, “Joe”, “sam”], 5)  
Set of three snd set of two ([“yes”, “yes”, “yes”, “maybe”, “maybe”], 5)

Two sets of duplicates with empty string ([“D”, “D”, “”, “”], 4)

shiftLeft:

Negative shift size ([“d”, “E”, “F”, “G”], 4, -5, “Tyler”)

Shift size of zero ([“d”, “E”, “F”, “G”], 4, 0, “Tyler”)

Negative array size ([“d”, “E”, “F”, “G”], -10, 6, “Tyler”)

Array size of zero ([“d”, “E”, “F”, “G”], 0, 3, “Tyler”)

Shift size less than array size ([“Jill”, “mike”, “14”, “d”], 4, 2, “Tyler”)

Shift size equal to array size ([“Jill”, “mike”, “14”, “d”], 4, 4, “Tyler”)

Shift size greater than array size ([“Jill”, “mike”, “14”, “d”], 4, 8, “Tyler”)

Placeholder string inside array ([“Jill”, “Tyler”, “14”, “d”], 4, 3, “Tyler”)

Empty string inside array ([“”, “mike”, “14”, “d”], 4, 3, “Tyler”)

Empty string as placeholder ([“Jill”, “mike”, “14”, “d”], 4, 8, “”)

replaceFirstAndLastOccurrences:

Negative array size ([“oreo”, “VaNilla”, “milk”, “valentine”], -203, ‘e’, ‘$’)

Array size of zero ([“oreo”, “VaNilla”, “milk”, “valentine”], 0, ‘e’, ‘$’)

Character to find not in the array ([“oreo”, “VaNilla”, “milk”, “valentine”], 4, ‘2’, ‘$’)

Character to find seen twice in a string ([“jopjo”, “elk”, “bet”], 3, ‘o’, ‘e’)

Character to find seen once in a string ([“jopjo”, “elk”, “bet”], 3, ‘e’, ’t’)

Character to find seen more than twice in a string ([“mom”, “gmopmmme”, “bet”], 3, ‘m’, ’T’)

Replace the same character [“oreo”, “VaNilla”, “milk”, “valentine”], 4, ‘e’, ‘e’)

Empty string in array [“oreo”, “VaNilla”, “”, “valentine”], 4, ‘e’, ‘4’)