A screen shot of a computer program

Description automatically generated

The isPalindrome() function works by taking in two parameters first. The string s and taking in and index that should always be called first with 0 as all strings start at 0. The base case of this is if the string is only 1 character or if there is no characters in the string at all. Say for ex, the input is “a”, the idx would be 0, s.length() would be 1, but I subtract 1, so then idx == s.length() – idx – 1 for the base case. For any general cases where the s.length() > 1, it uses the next 2 lines (26 and 27), where line 26 only runs if it the string isn’t a palindrome at all (since it is checking for if I were to reverse the string, would it still be a palindrome (beginning – last character set -> last – beginning characters). Line 27 is where the recursive call happens. It keeps the String s the same, however, it keeps incrementing the idx by 1. This is to make it to where the base case will be hit eventually where idx >= (s.length() – idx – 1) and if it does hit that part, it will be true.

Note: Calls OnlyAlpha() function to convert the string into only alphabetical characters with no whitespaces and all lowercase before the isPalindrome() function call.

Two cases one when false and one when true:

Case 1: True

isPalindrome(“racecar”, 0) -> isPalindrome(“racecar”, 1) -> isPalindrome(“racecar”, 2) -> isPalindrome(“racecar”, 3) (true)

stops at isPalindrome(3) since “racecar”.length() = 7 and 7 – 3 – 1 = 3 which is the idx. So this will return true. And backpedal all the other calls as true.

Case 2: false

isPalindrome(“yesterday”, 0) -> isPalindrome(“yesterday”,1) (false)

Stops at isPalindrome(“yesterday”, 1) since ‘e’ != ‘a’. Thus, this will return false.