Josh Barber   Assignment 10

Using a dynamic programming approach.
Steps:

1. Create a matrix by string length by string length.
2. The matrix diagonal should all equal 1 because every single character by itself is a palindrome
3. Start iterating from most right bottom to top right
   - start iterating backward for the outer loop "i" and for the inner loop "j" iterate forward starting from the index of outer loop.
4. Pick character from the input string based on the at "i" and "j" position. If the characters match two conditions apply
   - If the length of the substring is just 1 (a letter matching is good to be a palindrome)
   - But if the length of the substring is greater than 1, need to check if the inner substring is also a palindrome.
     - Go to left bottom corner and check if it is 1
     - left bottom corner represents the inner sub string of the current substring Eg. If matrix [i][j] = "ababa", this would result in a 1 because matrix [i+1][j-1] which equal's "bab", which is also 1
   - If matrix [i+1][j-1] = 1 that means the substring grows and continues to be a palindrome So compare previous palindrome substring with current and take max

Pseudo code:

```
Palindrome (string)
    palidrome indexing array = [0,1]
    create matrix string length by string length with 0's
    diagonally assign 1 values to matrix
    for i in string length to 0
        for j in i to string length
            if sub string string[i] = substring string[j]
                if j - i = 1 or matrix[i+1][j-1] = 1
                    matrix[i][j] = 1
                    if (palindrome array[1] - palindrome array[0])
                      < j - i + 1 :
                        palindrome indexing array = [i, j+1]

    return string[palindrome indexing array[0] to palindrome indexing
                                                  array[1]]
```

If we take the string "character" and use the Palidrome algorithm, it would work as so:

It would iterate over all characters in the word ~~back~~wards, while ~~also~~ on each iteration, iterate each character forward to do a comparison to see if a palindrome exists. For the word "character", it would compare the letters "a" and "a" ~~and~~ add the indexes of the string as the spread between "i" and "j" is only 1. The inner loop with "j" will keep iterating forward on "j" position to compare strings and if they match, add a 1 to the matrix.

In this case, the word "character" only has a palindrome of "ara", so just will return the indexes of the string.