Recursion Lab 1 Option A

Introduction

The goal of this lab is to improve in implementing recursion, using Reddits nested commenting structure the algorithm I will create will transverse through each comment and determine if it is neutral, positive or negative. Comments are structured in a list format, for example if user X comments and user Y comments both comments would be stored and accessed in a list. However, if a user wishes to reply to user X's comment then a new comment list would be created. As each comment is accessed, the program will take the string given by these comments use an outside library to determine if this string is positive, negative, or neutral and add this string to a negative, neutral, or positive list.

Solution

The high level structure of the algorithm is to use two functions, "process_coments()", which would make the actual recursive call while "process_string()" actually determine where the comment string is placed.

Process comments function

The parameters for this function will be the list of comments, an index, and a list of lists. The list contains three sub lists, lists[0] = negative list, lists[1] = positive list, and lists[2] = neutral list. The reasoning behind this data structure is to keep the function from passing multiple parameters or having to return a list. That or using a global variables which I personally am not a fan of. This algorithm will only edit data.

I have four base cases for this function as it is the function that will make the recursive call. First two base cases make sure the correct data types are passed. These base cases are not required to solve the problem, they do keep the function from crashing upon a bad input. The next two base cases are what actually stops the recursive call, one case checks if the comment data type is NULL, and the other checks if the index parameter is greater than or equal to the length of the comment list.

The actual recursive call is simple, the reply lists will be passed first once it gets to then of the stack for the original comment then the next comment in the list is passed. The next comment is determined by the index parameter and the process repeats. The actual proccess_string is done first then the recursive call.

Process comment

This function adds the given string to the respected list, two parameters are passed a string being the comment from Reddit and the list of lists. It builds a list of three values, the first value is the probability of the given string being negative, the same is done for the second being positive and third being negative. It then runs a for loop to determine the largest by setting a variable to largest index in the list, this variable index is then used to add to the negative, positive, or neutral list. It is much more clear if you see the actual code.

Results

In Big O notation the runtime for all experiments is N. Upon running the code you will be given the same experiments ran.

- Test Case 1: Tests case is a smaller version of the given example. This Reddit page is populated by my own comments. Which means there is a controlled amount of negative, positive, and neutral comments.
    - Results: Passed

        Expected Negative: 3, Actual: 3

        Expected Positive: 2, Actual: 2

        Expected Neutral: 2, Actual: 2

- Test Case 2: Tests that each nested comment is evaluated, I wanted to have a page where the nested comments get pretty deep.
    - Results: Passed

        Expected Negative: 0, Actual: 0

        Expected Positive: 6, Actual: 6

        Expected Neutral: 0, Actual: 0

- Test Case 3: Tests that the program will not crash if a given list is empty.
    - Results: Passed

- Test Case 4: Tests that the program will not crash with bad inputs, all bad inputs are tests.
    - Results: Passed

Conclusion

I didn’t learn anything particularly new other than how to use reddit. However, this was a good review and good practice because it did challenge me. Granted much the challenge came from using external libraries and learning a new language. Good review though, I would say I still need to practice my recursion with other harder problems.