CMPS 4143

Programming Assignment-5 (Due November 22nd 11:59 PM)

[*To get the full credit of your programming assignment, find the problem-solving steps and documentation guidelines listed in 'Programming Assignments' section in D2L. You are supposed to submit a zip file (don't use winrar to zip) that should contain a documentation file like a docx file and your codes with proper programming extensions like .java or .py. Make sure all of your code is compliable or no runtime error. TA will not grade if your code is not runnable and you will get a zero; Sometimes your grade determines based on how good you can describe/explain the code to TA and Instructor***]**

1. (35 points) A stack data structure has following functionalities like empty(), size(), top(), push() and pop(). See the lecture slides for the details (how it works, complexity, etc.) I have shown you in the class how to implement a stack using a *list* data structure. You need to implement the stack with *Linkedlist* data structure in python. Implement the stack means, all of the stack functionalities including the construction of stacks should present on your code.

2. (30 points) Given the expression as string str, find the duplicate parenthesis from the expression. Your program will output whether or not finding the duplicates, that is true of false

   Input type: An expression in string; **Output type**: A boolean value True or False

   Example 1:

   Input: str = '((x+y))+z'

   Output: True

   Example 2:

   Input: strs = (x+y)

   Output: False

   Example 3:

   Input: strs = '((x+y)+((z)))'

Output: True

3. (35 points) Write a python code to find the average from a stream. The input of this program will receive a stream of numbers and a window size to find the moving average of all the numbers in the sliding window. Write your code in OOP style and solve the program with queue and or stack data structure.

   Input type: Window Size as integer and a stream contains numbers as list;

   Output type: A list of float values denoting the moving averages

   Example 1:

   Input: Window Size = 3; Stream = [1, 10, 3, 5]

   Output: [1.0, 5.5, 4.67, 6.0]

   Explanation:

   1st iteration -> 1/1 -> 1.0

   2nd iteration -> (1+10)/2 -> 5.5

   3rd iteration -> (1+10+3)/3 -> 4.67

   4th iteration -> (10+3+5)/3 -> 6.0

   Example 2:

   Input: Window Size = 2; Stream = [1, 10, 3, 5]

   Output: [1.0, 5.5, 6.5, 4.0]

   Explanation:

   1st iteration -> 1/1 -> 1.0

   2nd iteration -> (1+10)/2 -> 5.5

   3rd iteration -> (10+3)/2 -> 6.5

   4th iteration -> (3+5)/2 -> 4.0