Instructor: Jun Yi
Due 09/22/21 11:00 pm

Responses will be graded on correctness, code design, and code style. Most of your grade is determined by how many tests you pass. If you test 0 passes, you will get a 0. Bonus assignment 1 works as the bonus assignment. If you get more than 90, then I will add 1 points directly to your final grade.

30 points) In this problem, you will write a function to reverse an array of integers. Your algorithm should use O(1) space beyond the input; any algorithms that use space not in O(1) will not receive credit. You may not use any library functions in this question; any solutions that do will receive zero credit.

Your program should accept an integer array and reverse it. For example, the input array of 3 4 7 6 1 should result in returning the array 1 6 7 4 3. Complete the method. Note you should think about what is the best way to name your class/method.

(2) (30 points) Suppose you are given a string containing only the characters (and). In this problem, you will write a function to determine whether the string has balanced parentheses. Your algorithm should use no more than O(1) space beyond the input; any algorithms that use space not in O(1) will receive half credit at most. Any solutions that always return true (or always return false) or otherwise try to game the distribution of test cases will receive zero credit.

Balanced parentheses means that every open parenthesis has exactly one close parenthesis corresponding to it and vice versa, and that for each pair the opening parenthesis precedes the closed parenthesis. The following strings have balanced parentheses:

() (()())

((())())

The following strings do not have balanced parentheses:

)((()

())((()))())

We consider the empty string to have balanced parentheses, as there is no imbalance. The input to XXX (xxx is your method) will always be a string
only containing ”(” or ”)”. Your task is to complete the XX X method.

Hint: There’s a pattern for how to determine if parentheses are balanced. Try to see if you can find that pattern first before coding this method up.

3) Review the slides in lecture03 starting from page 37 and then finish the question below,

i) (20 points) implement the finding square roots with binary search and output your Big-O in the terminal.

ii) (20 points) implement finding Cube root. For example, the cube root is 3 for number 27. Also output the big-o.

Note that you should think about how to build your coding style now. Basically it includes how to name those class/methods, variables. Because of the braces and brackets and the tab space.