CS 3510 – Assignment 2

Due `Friday, June 03, 2022` at `11:59pm` on Canvas

- Please type your answers (LaTeX is highly recommended) and upload a single PDF file named `<Your-GT-Account>.pdf`, e.g., GBurdell3.pdf, including all your answers. You can submit multiple times. Canvas keeps track of the submissions and append a version number when you re-submit. We always grade your most recent submissions.
- Please read the policies, and do not forget to acknowledge your collaborators and cite your references.
- If you do not understand a question, please ask on Piazza or come to office hours well ahead of the due date.

Problem 1 — Divide and Conquer, Dynamic Programming (50 pts)

Assume there exists a cryptocurrency called *GTCoin*, where its rise and fall rates are known in advance for the next n days into the future. George Burdell wants to use this invaluable information to buy and sell this cryptocurrency and hopefully earn some extra money. However, there is an important rule for using this information: **"One can only buy and sell GTCoin once in the next n days"**! That means, George can buy GTCoin only once and sell it entirely sometime after that.

For instance, let `rates` $= [5.4, 2.3, -1.4, -0.6, 8.1, -4.2, \ldots]$ and George wants to buy \$100 of GTCoin on day 2, where the rate is +2.3. Then, if he sells his GTCoins on day 5, he will receive $\$100 \times (2.3 - 1.4 + 0.6 + 8.1)$. In other words, the total earning rate will be the sum of the rates between the buying and selling dates (inclusive). Therefore, he is trying to calculate the maximum total earning that he can get from investing in this cryptocurrency using the information given by the rates array. Let's help George by designing "efficient" algorithms that can solve this problem, which can be defined as following:

Given an array, `rates`, of size n, including positive and negative numbers representing the changes in the GTCoin prices over the next n days, design algorithms to find the sub-array which has the largest sum, then return this largest sum value.

Example 1:
Input: rates = [-1.2, 5.4, 2.5, -1.4, -0.6, 8.1, -4.0, -1.5, 3.9]
Output: largest rate sum = 14.0
Explanation: The maximum rate sum can be obtained by buying on day 2 and selling on day 6: 5.4 + 2.5 -1.6 -0.6 + 8.1 = 14.0

Example 2:
Input: rates = [-1.7, -4.4, -2.5, 3.9, -2.7, 0.0]
Output: largest rate sum = 3.9

Explanation: The maximum rate sum can be obtained by buying on day 4 and selling at the end of the same day.

Example 3:
Input: rates = [1.8, 2.4, 3.5, 0.0, 2.1]
Output: largest rate sum = 0.8
Explanation: The maximum rate sum can be obtained by buying on day 1 and selling on the last day (Because all rate values are non-negative)

Notes:

- A sub-array is defined as a contiguous subset of the given array
- It is not possible to sell before buying anything! But it is possible to buy and sell on the same day (see Example 2).
- You can assume that the problem always has a solution.
- The numbers data type does not matter. In other words, you can assume the rate values are either integer or float (decimal) numbers.

Deliverables:

1. (25 pts) Design an algorithm using the **divide-and-conquer** approach to return the largest sub-array sum. (Hint: You can start by dividing the rates array into two halves. Then, note the sub-array that gives the largest sum can be either in one of these two halves or can start in the first half and end in the second half.)
    - **(a) (Design and Correctness)** Explain your divide-and-conquer algorithm in words and provide the corresponding pseudocode.
    - **(b) (Analysis)** Discuss the time and space complexity of your algorithm.

2. (25 pts) Design another algorithm using the **dynamic programming** approach to return the largest sub-array sum.
    - **(a) (Design and Correctness)** Explain your dynamic programming algorithm in words and provide the corresponding pseudocode.
    - **(b) (Analysis)** Discuss the time and space complexity of your algorithm.