AMS 516 Project: Pseudo Code v0.1

In general, we divide our tasks into three parts:

- Find a proper data structure (maybe a queue) to meet the following requirements:
    1. It can contain 10 price element [**open_price, close_price**]
    2. We operate it with “FIFO”: First IN, First Out. (Every time, we discard the first price element <oldest>, move the rest forward, then add the latest at the end of queue)
- Pattern Recognition:
    1. We construct a **Weight Matrix Table** following the same one proposed by the author. Idea is to use a **moving point** to **go through the 10 time periods** and **record the according fit value** by referring its **index position** mapped in **Weight Matrix Table**.

        | 0 0 0 0 0 0 0 0 0 0
        | 0 0 0 0 0 0 0 0 0 0
        | 0 0 0 0 0 0 0 0 0 0
        | : : : : : : : : :
        | 5 -1 -5 -5 : : : :
        ---
    2. Initialization:
        a. The moving point starts at the left bottom of matrix, so we set its initial position [cur_row=9, cur_col=0] (if we consider an array’s index in Python starting from 0, so current row number is 9 and current column number is 0)
        b. The **fit value**: **we start the pattern recognition process only when the close_price > open_price in the 1st time period.**
        c. As the **Weight Matrix Table** can be divided into 10 * 10 grids, the **grid size** is extremely important to map the position so we use the size referred from the paper: 5/9000 (by observation, the author set the grid size with Dow index from 9210-9260 covering 10 grid, so each grid is 5). Note that we may need to adjust it later.
    3. Process:

        Basically, if we start the pattern recognition process, we need to calculate the whole 10 time periods to get the sum of fit value, then to decide whether this fit value meets the **threshold fitting value** we set (for example, 2, 3 ,4, 5) we set. If yes, we can conclude that it is a bull pattern.else, it is not a bull pattern.

        For loop: i = 0, 1 .. 9

        1) Mov_length (moving length) = close_price[i] - open_price[i]
        2) Num_grid_mov (number of grids to move) = Mov_length / grid_size
        3) Update the moving point position with: [cur_row - Num_grid_mov, cur_col += 1]

4) Map the closing and opening prices of the asset to the cells in the table (per column) and calculate the difference between indices to get the number of cells the candlestick falls into
5) Take the sum of the weights contained in the cells containing the candlesticks (calculated in the previous part) and add it to the total value (initialized at 0).
6) If the total value in any stage (1 .. 9) is less than the **threshold fitting value** we set, we can exit the "for .. loop". If the for loop is broken, start the same process from the second column to column 11 (include a new price at the end = new window of 10 prices)
7) If the fit value is larger than the threshold fitting value after we finish the 10 time periods, we can conclude that we've found the bull pattern.
8) **Meanwhile, sometimes it will happen that during the 10 time series, the price may jump or drop significantly which crosses the boundary of matrix, we need to figure it out how to properly handle this situation as this will go beyond the Weight Matrix Table index which we can NOT do the mapping.**

4. Trading part:
    1) We put the buy trigger at the beginning of the incoming time period
    2) We closely monitor the price change in each time period. And make a decision when to sell based on following principle:
        1. If the close_price - our_cost > take_profit (TP), we sell
        2. If the close_price - our_cost < stop_loss (SL), we sell
        3. Otherwise, we hold.