Lab 2: Stacks and Queues with AlgoStreet

You are working for a promising new stock trading startup called "AlgoStreet."

You have been tasked with developing a new trading signal that will be incorporated into the automatic trading strategy. A new metric has been introduced called

positive stock pressure, which measures how many consecutive days before today (not including today) have a higher price.

You will be given daily stock prices for the last *N* days and must return the list of daily positive stock pressures for each day.

Below is an example of 1 week of data.

Day	Price	Stock Pressure
1	100	1
2	90	2
3	95	1
4	100	1
5	105	1
6	110	1
7	80	7

Implement the function to **compute stock pressure**.

To compute positive stock pressure for each stock price, you should find the last day with the lower (or equal) price. In other words, the positive stock pressure is the number of days since the last day with a lower or equal price.

You must solve this problem by using a stack as the primary data structure.

Here is the same table with some useful intermediary values:

Day	Price	Last Day w/ Smaller Price	PosPressure = Day - (Last Day w/ Smaller or Equal Price)
1	100	0	1 - 0 = 1
2	90	0	2 - 0 = 2
3	95	2	3 - 2 = 1
4	100	3	4 - 3 = 1
5	105	4	5 - 4 = 1
6	110	5	6 - 5 = 1
7	80	0	7 - 0 = 7

Deliverable Explanation

You must efficiently solve this problem in either Python or Java, using one of the attached starter code files. Several unit tests are provided to help you test your code.

Having the optimal runtime will grant you extra credit. On the other hand, you may also lose points if your code is substantially inefficient. You must also provide an explanation of the running time of your code in the provided placeholder in the starter code.

Your code will be evaluated against a set of visible and hidden test cases (the visible ones are also provided in the starter code for your convenience) to test correctness. The hidden test cases are intended to verify that you have a robust solution that considers possible edge cases and various inputs.

If you are using Java, your file's package must be set to "student".

Submission Instructions

Submit one of the following files on Gradescope:
AlgoStreet.java to Lab2 (Java)
OR
algo_street.py to Lab2 (Python)

Grading Methodology

The lab is worth 100 points and is broken down into the following categories:

- Test Cases Passing (80 points)
- Overall code performance and style (10 points)
- Explanation of running time (10 points)
- Having Optimal Running Time (5 points of Extra Credit)