Quant Researcher Intern
Technical Assessment Supplement

Submission Deadline:

Submit this along with the main Deep Learning Time-Series Prediction assessment.

Submission Format:

- **Coding Question:** Submit your solution as a **Jupyter Notebook (`.ipynb`)** or a **Python script (`.py`)**.
  Ensure your code is well-commented, and include brief explanations of your approach and any assumptions made.

- **Statistics Question:** Provide a clearly written answer in one of the following formats:

    - **Markdown** (within a Jupyter Notebook)

    - **PDF document**

    - **Text cell** within the same Jupyter Notebook as the coding question

Section 1: Data Structures and Algorithms

Problem: Sliding Window Stock Span

Objective:
To assess your problem-solving ability and understanding of efficient data structures used in financial time-series analysis.

Problem Statement:
You are given an array `prices[]` where each element represents the **daily closing price** of a stock for `n` consecutive days. For a given integer `k`, your task is to calculate the **price span** for each day, considering only the past `k` days (including the current day).

> ### What is Price Span?
>
> For a given day, the **price span** is the number of **consecutive days (going backwards) up to `k` days** where the stock price was **less than or equal to the current day's price**.

Function Signature:

```
def calculate_price_span(prices: List[float], k: int) -> List[int]:
    pass


Input:
     prices = [100, 80, 60, 70, 60, 75, 85]
     k = 4

Output:
     [1, 1, 1, 2, 1, 4, 4]
```

Explanation:

- **Day 0 (price = 100):** No previous days, span = 1
- **Day 1 (price = 80):** Only day 0 is within 4-day window. 80 < 100 → span = 1
- **Day 2 (price = 60):** 60 < 80 → span = 1
- **Day 3 (price = 70):** Previous 3 days: [60 (yes), 80 (no)] → span = 2
- **Day 4 (price = 60):** 60 < 70 → span = 1
- **Day 5 (price = 75):** Check last 4 days [60, 70, 60, 80] → span = 4 (60, 70, 60 all ≤ 75)
- **Day 6 (price = 85):** Check last 4 days [75, 60, 70, 60] → span = 4 (all ≤ 85)

Constraints:

- 1 ≤ n ≤ 10^5
- 1 ≤ k ≤ n
- 0 ≤ prices[i] ≤ 10^5

Hint:

Use a **monotonic stack** or **deque** to efficiently compute the span in **O(n)** time.

Section 2: Probability and Statistics

Problem: Analyzing Correlation Between Stock Returns

Objective:
Evaluate your understanding of statistical measures like **covariance** and **correlation**, commonly used to analyze relationships between financial instruments.

Problem Statement:

You are analyzing the **daily return percentages** of two NASDAQ stocks, Stock A and Stock B, over a 10-day period:

Day	Return A (%)	Return B (%)
1	0.5	0.4
2	-0.2	-0.1
3	0.3	0.2
4	0.7	0.8
5	-0.3	-0.4
6	0.1	0.0
7	0.4	0.3
8	0.2	0.2
9	-0.1	-0.2
10	0.6	0.5

Tasks:

1. **Compute the sample covariance** between Return A and Return B.
2. **Compute the Pearson correlation coefficient** between Return A and Return B.
3. **Interpret the correlation value**:
    - Is the relationship positive or negative?
    - Is it weak, moderate, or strong?
    - What does it imply about the movement of these two stocks?

Notes:

- You may compute the values manually, or use a short Python/NumPy snippet to show your working.
- Clearly **show formulas** and **intermediate steps** used in your calculation.

Submission Checklist:

- Well-commented Python code for the stock span problem
- Written answer (with formulas/code) for the statistics question