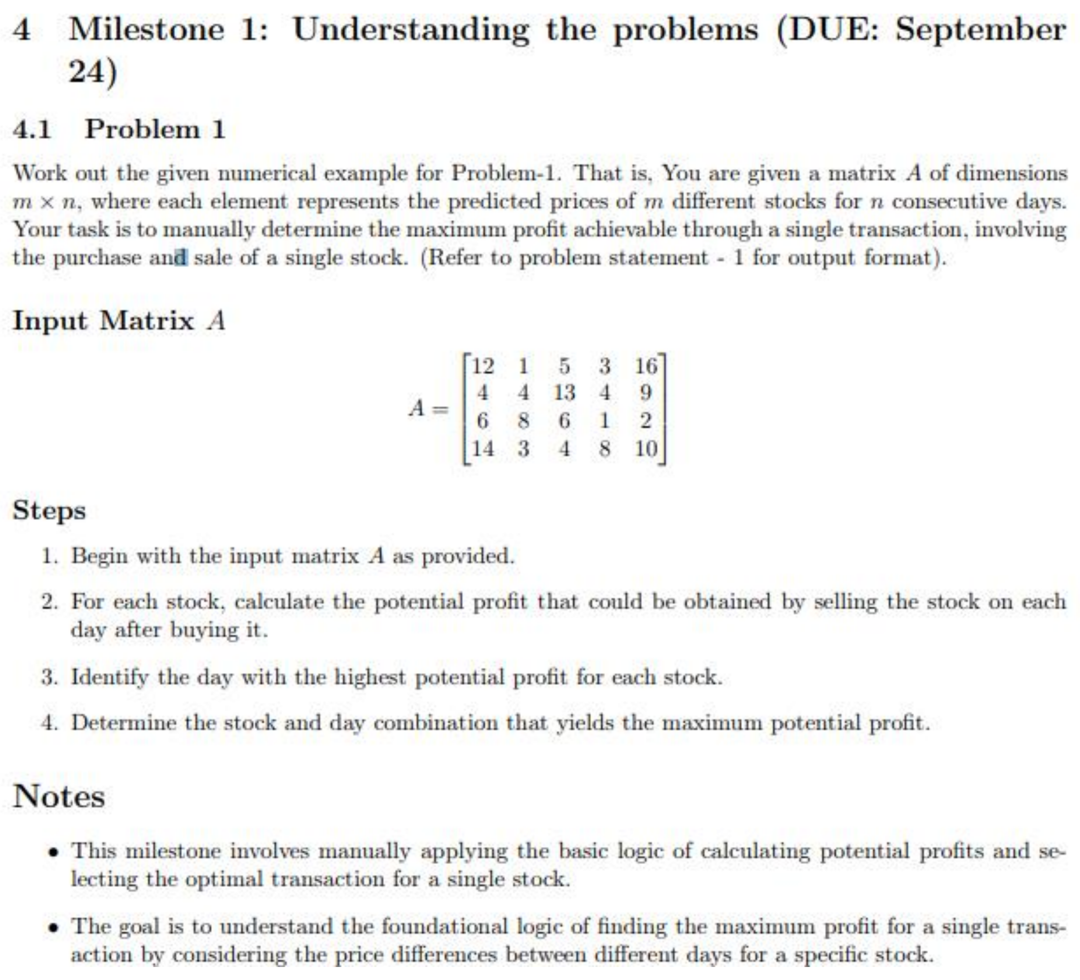
Milestone 1: Understanding the problems.

Group Members: Khai Dao – (I couldn't find a group yet since all the group seems to be full at the moment, I am trying to find a group for milestone 2 and onward)

Member Role: Khai Dao – Coordinator/ leader

GitHub Repository Link: https://github.com/khaidao000/Project---Milestone-1.pdf.git



4 Milestone 1: Understanding the problems (DUE: September 24)

4.1 Problem 1

Work out the given numerical example for Problem-1. That is, You are given a matrix A of dimensions $m \times n$, where each element represents the predicted prices of m different stocks for n consecutive days. Your task is to manually determine the maximum profit achievable through a single transaction, involving the purchase and sale of a single stock. (Refer to problem statement - 1 for output format).

Input Matrix A

$$A = \begin{bmatrix} 12 & 1 & 5 & 3 & 16 \\ 4 & 4 & 13 & 4 & 9 \\ 6 & 8 & 6 & 1 & 2 \\ 14 & 3 & 4 & 8 & 10 \end{bmatrix}$$

Steps

1. Begin with the input matrix A as provided.
2. For each stock, calculate the potential profit that could be obtained by selling the stock on each day after buying it.
3. Identify the day with the highest potential profit for each stock.
4. Determine the stock and day combination that yields the maximum potential profit.

Notes

- This milestone involves manually applying the basic logic of calculating potential profits and selecting the optimal transaction for a single stock.
- The goal is to understand the foundational logic of finding the maximum profit for a single transaction by considering the price differences between different days for a specific stock.

Looking at the given chart, we can manually calculate the maximum profit for each stock by simply calculation: Profit = Selling Price − Buying Price.

1.1.4 Example:

Input:

$$A = \begin{bmatrix} 7 & 1 & 5 & 3 & 6 \\ 2 & 4 & 3 & 7 & 9 \\ 5 & 8 & 9 & 1 & 2 \\ 9 & 3 & 14 & 8 & 7 \end{bmatrix}$$

Output:

$$(4, 2, 3, 11)$$

Explanation: Choosing the 4th stock (1-based indexing). Buying it in day 2nd day and selling it on the 3rd day (1-based indexing), yields the maximum profit of 11 (sell price 14 minus buy price 3).

Input Matrix A

$$A = \begin{bmatrix} 12 & 1 & 5 & 3 & 16 \\ 4 & 4 & 13 & 4 & 9 \\ 6 & 8 & 6 & 1 & 2 \\ 14 & 3 & 4 & 8 & 10 \end{bmatrix}$$

Using the example 1.1.4 , the first stock would have the output: (1, 2, 5, **15)**

Explanation: choose the 1st stock, buying it in day 2nd and selling on 5th day, yields a maximum profit of 15 (sell price 16 minus buy price 1)

Second stock would have the output: (2, 1, 3, **9**)

third stock would have the output: (3, 1, 2, **2**)

4th stock would have the output: (4, 2, 5, **7**)

Among all these calculations, the first stock provides a maximum profit of 15. So, the final output for the given matrix A should be **(1, 2, 5, 15).**

-- Explanation---

Step 1: Identify Maximum Profit for Each Stock

I traversed through each stock and calculated the potential profit for every possible buying and selling day combination. The goal was to identify the buying day and the selling day that yield the maximum profit for each stock.

First stock: I discerned that buying on the 2nd day and selling on the 5th day yields a maximum profit of 15, represented as (1, 2, 5, 15).

Second stock: The combination of buying on the 1st day and selling on the 3rd day provided a profit of 9, denoted as (2, 1, 3, 9).

Third stock: The maximum profit of 7 was achievable by buying on the 1st day and selling on the 2nd day, represented by (3, 1, 2, 7).

Fourth stock: A profit of 7 was identified when buying on the 2nd day and selling on the 5th day, represented as (4, 2, 5, 7).

Step 2: output

With all calculations and comparisons made, the final output was determined to be (1, 2, 5, 15), which signifies the stock index, buying day, selling day, and the resultant maximum profit which is 15 in this case.

4.2 Problem 2

Work out the given numerical example for Problem-2. That is, You are given a matrix A of dimensions $m \times n$, where each element represents the predicted prices of m different stocks for n consecutive days. Additionally, you are given an integer k $(1 \leq k \leq m)$. Your task is to manually find a sequence of at most k transactions, each involving the purchase and sale of a single stock, that yields the maximum profit. (Refer to problem statement - 2 for output format).

Input Matrix A

$$A = \begin{bmatrix} 25 & 30 & 15 & 40 & 50 \\ 10 & 20 & 30 & 25 & 5 \\ 30 & 45 & 35 & 10 & 15 \\ 5 & 50 & 35 & 25 & 45 \end{bmatrix}$$

Steps

1. Begin with the input matrix A as provided.
2. Determine the sequence of at-most K non-overlapping transactions. A valid transaction is a buy-sell of the same stock. Different transactions can have different stocks, but one transaction would deal with only a single stock.
3. Output should be a sequence of atmost K transactions in the format of (i,j,l) that yields the maximum potential profit by selling ith stock on lth day that was bought on jth day.

1.2.4 Example:

Input:

$$A = \begin{bmatrix} 7 & 1 & 5 & 3 & 6 \\ 2 & 9 & 3 & 7 & 9 \\ 5 & 8 & 9 & 1 & 6 \\ 9 & 3 & 4 & 8 & 7 \end{bmatrix}$$

$$k = 3$$

Output:

$$[(2,1,2),(1,2,3),(3,4,5)]$$

Explanation: Performing at most 3 transactions, selling the 2nd stock on the 2nd day after buying on 1st day, 1st stock selling on the 3rd day buying on 2nd day, 3rd stock selling on 5th day buying on 4th day yields the maximum profit of 16 (transaction 1: $9-2=7$, transaction 2: $5-1=4$), transaction 3: $6-1=5$).

K = 3

--Solution---

Input Matrix A

$$A = \begin{bmatrix} 25 & 30 & 15 & 40 & 50 \\ 10 & 20 & 30 & 25 & 5 \\ 30 & 45 & 35 & 10 & 15 \\ 5 & 50 & 35 & 25 & 45 \end{bmatrix}$$

The output is a list of transactions, represented as tuples (I, J, L) where:

- I is the stock index
- J is the buying day index
- L is the selling day index

Using example 1.2.4, the first transaction would be: (4, 1, 2).

This indicates we would buy 1st stock on row 4th for 5, and subsequently sell for 50. yielding **45 profits.**

The second transaction would be: (2, 2, 3)

This indicates we would buy stock index 2, on the second day, and sell on the third day. In other words, buy for 20, sell for 30. Yielding **10 profit**

The 3rd transaction would be: (1, 3, 5)

This indicated that we would buy stock index 1, on the 3 day, and sell on the 5th day. In other words, buy for 15, sell for 50. Yielding **35 profit.**

Therefore, the total maximum possible profit is 45 + 10 + 35 **= 90 profit**

The output should be: **[(4, 1, 2), (2, 2, 3), (1, 3, 5)]**

Using the same logic, another possible permutation would be: **[(4, 1, 2), (1, 3, 4), (4, 4, 5)]**

---Explanation--

Step 1: Identify the First Transaction

I started by identifying the first transaction that would yield the highest profit, which is buying the stock in the 4th row on the first day and selling it on the second day. This transaction is represented as the tuple (4, 1, 2) and yields a profit of 50 – 5 = 45

Step 2: Identify the Second Transaction

Next, I chose to buy the stock in the 2nd row on the second day and sell it on the third day because it would yield the most profit. This transaction is represented as the tuple (2, 2, 3) and yields a profit of 30 – 20 = 10.

Step 3: Identify the Third Transaction

Afterwards, I maximized the profit by buying the stock in the 1st row on the third day and sell it on the fifth day. This transaction is represented as the tuple (1, 3, 5) and yields a profit of 50 – 15 = 35.

Step 4: Calculate Total Profit

Finally, the total maximum possible profit is calculated by summing up the profits from each transaction: 45 + 10 + 35 = 90. Which represents the tuples: **[(4, 1, 2), (2, 2, 3), (1, 3, 5)] and [(4, 1, 2), (1, 3, 4), (4, 4, 5)]**

4.3 Problem 3

Work out the given numerical example for Problem-3. Work out the given test-case for Problem-3 manually. That is, You are given a matrix A of dimensions $m \times n$, where each element represents the predicted prices of m different stocks for n consecutive days. Additionally, you are given an integer c $(1 \leq c \leq n-2)$. Your task is to manually determine the maximum profit achievable under the given trading restrictions, where you cannot buy any stock for c days after selling any stock. If you sell a stock on day i, you are not allowed to buy any stock until day $i+c+1$. (Refer to problem statement - 3 for output format).

Input Matrix A

$$A = \begin{bmatrix} 7 & 1 & 5 & 3 & 6 & 8 & 9 \\ 2 & 4 & 3 & 7 & 9 & 1 & 8 \\ 5 & 8 & 9 & 1 & 2 & 3 & 10 \\ 9 & 3 & 4 & 8 & 7 & 4 & 1 \\ 3 & 1 & 5 & 8 & 9 & 6 & 4 \end{bmatrix}$$

$c = 2$

Steps

1. Begin with the input matrix A and integer c as provided.
2. For each day j, identify the maximum price after $c+1$ days (i.e., on day $j+c+1$ or later) to sell a stock that was bought on day j.
3. Determine the sequence (i,j,l) that yields the maximum potential profit by selling ith stock on lth day that was bought on jth day.

1.3.1 Example:

Input:

$$A = \begin{bmatrix} 2 & 9 & 8 & 4 & 5 & 0 & 7 \\ 6 & 7 & 3 & 9 & 1 & 0 & 8 \\ 1 & 7 & 9 & 6 & 4 & 9 & 11 \\ 7 & 8 & 3 & 1 & 8 & 5 & 2 \\ 1 & 8 & 4 & 0 & 9 & 2 & 1 \end{bmatrix}$$

$c = 2$

Output:

$$[(3,1,3),(2,6,7)]$$

Explanation: To achieve the maximum profit, buy 3rd stock on day 1, sell it on day 3. buy 2nd stock on day 6 and sell it on day 7 adhering to 2 days waiting period.

The output is a list of transactions, represented as tuples (I, J, L) where:

- I is the stock index
- J is the day when the stock is bought.
- L is the day when the stock is sold to maximize profit in this transaction.

Given an integer c representing the waiting period, where 1 ≤ c ≤ n – 2

The objective is the maximize the profit with the following constraint:

After selling a stock on day I, we are not allowed to buy any stock until day I + c + 1.

Input Matrix A

$$A = \begin{bmatrix} 7 & 1 & 5 & 3 & 6 & 8 & 9 \\ 2 & 4 & 3 & 7 & 9 & 1 & 8 \\ 5 & 8 & 9 & 1 & 2 & 3 & 10 \\ 9 & 3 & 4 & 8 & 7 & 4 & 1 \\ 3 & 1 & 5 & 8 & 9 & 6 & 4 \end{bmatrix}$$

$c = 2$

Looking at the matrix, we can see that buying the 3rd stock on day 1 and sell it on day 3 would yield the most profit. Therefore, we'd have the tuple **(3, 1, 3) this would yield 4 profits.**

The second most profitable trade would be, buying on day buy on day 6, adhering to the 2 days waiting period. We'd have the tuple **(3, 6, 7) this would yield 7 profits.**

When combining two tuples**, the maximum total profit would be 11.**

The final tuples would be: [(3, 1, 3), (3, 6, 7)] Following this logic, other possible solution would be: [(3, 1, 3), (2, 6, 7)], [(3,1,2),(3,5,7)], [(1,2,3),(2,6,7)], [(1,2,3),(3,6,7)], [(5,2,3),(2,6,7)], [(5,2,3),(3,6,7)]. which would yield a total of 11 profits.

--Explanation--

Step 1: Identify the First Transaction

I first identified the transaction that would yield the most profit initially. This was buying the 3rd stock on day 1 and selling it on day 3. This transaction is represented by the tuple (3, 1, 3) and yields a profit of 9 – 5 = 4

Step 2: Apply the Waiting Period Constraint

After the first transaction, I have to consider the waiting period constraint, which is 2 days in this case because c = 2. So, the next transaction could only be made from day 6 onwards.

Step 3: Identify Subsequent Transactions

Next, I looked at the possible transactions from day 6 onwards and identified 7 potential transactions that yield the highest additional profit:

Buying the 3rd stock on day 6 and selling it on day 7, represented by the tuple (3, 6, 7), with a profit of 10 – 3 = 7.

Alternatively, buying the 2nd stock on day 6 and selling it on day 7, represented by the tuple (2, 6, 7), with the same profit of 8 -1 = 7.

Using the same logic, I've tested all other possible permutations and found that there are a total of 7 tuples that would yield a profit of 11 which is the maximum profit.

Step 4: Combine Transactions and Calculate Total Profit

Finally, I combined the initial transaction with each of the subsequent transactions and calculated the total profit. This leads to 7 sequences of transactions.

1. [(3, 1, 3), (3, 6, 7)]
2. [(3, 1, 3), (2, 6, 7)]
3. [(3, 1, 2), (3, 5, 7)]
4. [(1, 2, 3), (2, 6, 7)]
5. [(1, 2, 3), (3, 6, 7)]
6. [(5, 2, 3), (2, 6, 7)]
7. [(5, 2, 3), (3, 6, 7)]