🎯 Question (LeetCode 121: Best Time to Buy and Sell Stock)

You are given an array `prices` where `prices[i]` is the price of a given stock on the `i` th day. You want to maximize your profit by choosing **one day to buy** and **one day to sell** the stock. Return the maximum profit you can achieve. If you cannot achieve any profit, return 0.

Example:

- Input: prices = [7,1,5,3,6,4]
- Output: 5
- Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.
- Input: prices = [7,6,4,3,1]
- Output: 0
- Explanation: No transaction is done, profit = 0.

🏷️ Best Time to Buy and Sell Stock

📂 1. Definition and Purpose

- Find the maximum profit achievable from a single buy-sell transaction.
- Useful in stock trading analysis and profit prediction.

📂 2. Syntax and Structure (Python)

```
# prices: list of integers representing stock prices per day
```

📂 3. Two Approaches

🧾 Approach 1: Brute Force

- Check profit for every pair of buy and sell days.

```
def max_profit_bruteforce(prices):
    max_profit = 0
    n = len(prices)
    for i in range(n):
        for j in range(i+1, n):
            max_profit = max(max_profit, prices[j] - prices[i])
    return max_profit
```

- **Time Complexity:** O(n^2)
- **Space Complexity:** O(1)

🧾 Approach 2: Optimized (Single Pass)

- Track minimum price so far and calculate potential profit.
- O(n) time, O(1) space.

📂 4. Optimized Pseudocode

```
min_price = infinity
max_profit = 0
for price in prices:
    if price < min_price:
        min_price = price
    else:
        profit = price - min_price
        max_profit = max(max_profit, profit)
return max_profit
```

📂 5. Python Implementation with Detailed Comments

```python
def max_profit(prices: list[int]) -> int:
    """
    Calculate maximum profit from a single buy-sell transaction.
    """
    min_price = float('inf')  # Initialize minimum price
    max_profit = 0            # Initialize maximum profit

    for price in prices:
        if price < min_price:
            min_price = price  # Update minimum price if lower
        else:
            profit = price - min_price  # Potential profit if sold today
            max_profit = max(max_profit, profit)  # Update max profit

    return max_profit

# Example Usage
prices = [7,1,5,3,6,4]
print(max_profit(prices))  # Output: 5
```

📂 6. Internal Working

- Maintain the lowest purchase price encountered.
- Calculate profit if sold at current price.
- Update maximum profit throughout traversal.
- Single pass ensures linear time complexity.

📂 7. Best Practices

- Avoid nested loops for large datasets.
- Handle edge cases where no profit is possible.
- Use float('inf') for initial min_price for clarity.

📂 8. Related Concepts

- Array traversal
- Tracking minimum/maximum
- Dynamic programming for stock problems

📂 9. Complexity Analysis

- **Optimized Approach:**
  - Time: O(n)
  - Space: O(1)
- **Brute Force Approach:**
  - Time: O(n^2)
  - Space: O(1)

📂 10. Practice and Application

- LeetCode: 121 Best Time to Buy and Sell Stock
- Applicable in trading bots, financial analytics, and profit optimization.