Documentation for "Best Time to Buy and Sell Stock" Problem

Problem Description

- You are given an array of prices where prices[i] is the price of a given stock on the ith day.
- Your goal is to maximize your profit by choosing a single day to buy one stock and choosing a different day in the future to sell that stock.
- Return the maximum profit you can achieve from this transaction. If you cannot achieve any profit, return 0.

Example 1:

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. Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.

Example 2:

Input: prices = [7, 6, 4, 3, 1]

Output: 0

Explanation: In this case, no transactions are done and the max profit = 0.

Constraints

- 1 <= prices.length <= 105
- 0 <= prices[i] <= 104

Solution Explanation

The solution involves iterating through the list of prices while keeping track of the minimum price encountered so far and the maximum profit possible at each step.

1) Initialization:

- min_price is initialized to infinity to ensure that any price in the list will be less than this initial value.
- max_profit is initialized to 0 because the minimum profit we can have is 0 (if no transaction is done).

2) Iteration:

- For each price in the list:
- If the current price is less than min_price, update min_price to the current price.
- Otherwise, calculate the profit if the stock was bought at min_price and sold at the current price. If this profit is greater than max_profit, update max_profit to this new value.

3) **Result:**

- After iterating through all the prices, max_profit will contain the maximum profit that can be achieved from a single buy-sell transaction. If no profit is possible, max_profit will be 0.

Key Points

- The algorithm maintains a running minimum price to determine the best day to buy.
- It cacalculates the maximum possible profit by comparing the current price with the running minimum price.
- It ensure that the buy day is always before the sell day by only updating the profit if the current price is higher than the running minimum price.

This approach ensures an optimal solution with a time complexity of O(n), where n is the number of prices since it processes each price exactly once. The space complexity is O(1) because it uses only a fixed amount of extra space.