121. Best Time to Buy and Sell Stock

Solved

Easy | Topics | Companies

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 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 <= 10^5`
- `0 <= prices[i] <= 10^4`

Python:

```python
from typing import List

class Solution:
    def maxProfit(self, prices: List[int]) -> int:
        min_price = float('inf')  # Start with a very high value
        max_profit = 0            # Start with zero profit

        for price in prices:
            # Update min_price if current price is lower
            if price < min_price:
                min_price = price
            else:
                # Calculate profit if we sell today
                profit = price - min_price
                if profit > max_profit:
                    max_profit = profit

        return max_profit
```

JavaScript:

```javascript
/**
 * @param {number[]} prices
 * @return {number}
 */
var maxProfit = function(prices) {
    let minPrice = Infinity;   // Store the minimum price so far
    let maxProfit = 0;         // Store the maximum profit

    for (let price of prices) {
        // Update minPrice if we find a smaller price
        if (price < minPrice) {
            minPrice = price;
        }
        // Calculate profit if we sell today
        else if (price - minPrice > maxProfit) {
            maxProfit = price - minPrice;
        }
    }

    return maxProfit;
};
```

Java:

```
class Solution {
    public int maxProfit(int[] prices) {
        int minPrice = Integer.MAX_VALUE;  // Track minimum price
        int maxProfit = 0;  // Track maximum profit

        for (int price : prices) {
            if (price < minPrice) {
                minPrice = price;  // Update min price if lower price found
            } else {
                maxProfit = Math.max(maxProfit, price - minPrice);
            }
        }

        return maxProfit;
    }
}
```