

121. Best Time to Buy and Sell Stock

Easy 9351 389 Add to List Share

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:

- $1 \leq \text{prices.length} \leq 10^5$
- $0 \leq \text{prices}[i] \leq 10^4$

Accepted 1,398,427 Submissions 2,678,963

Medium in logic
Easy in code

Main Idea:

if New lower point,

calculate possible

higher max-profit based

on the new lower point

for further days.

update max-profit if found.

A. [5, 8, 2, 3]

买入的最低点, lowest

loop iteration for each day after.

对于往后的每一天,

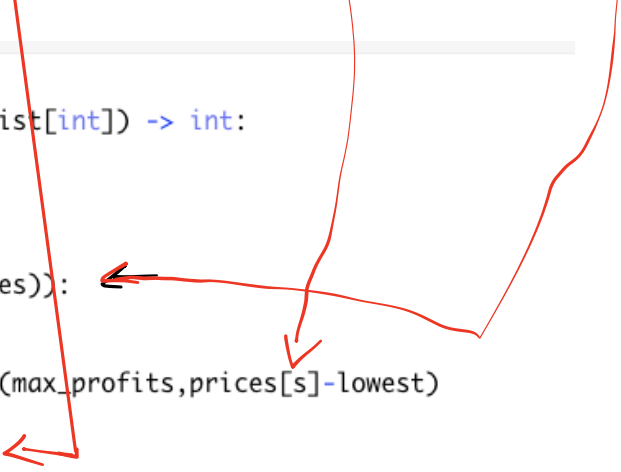
if $p > \text{lowest}$: # 有利润

update max-profit

else: # 新的低点

lowest = p

```
1 class Solution:
2     def maxProfit(self, prices: List[int]) -> int:
3         lowest = prices[0]
4
5         max_profits=0
6
7         for s in range(1,len(prices)):
8
9             if prices[s]> lowest:
10                 max_profits = max(max_profits,prices[s]-lowest)
11             else:
12                 lowest=prices[s]
13
14         return max_profits
```

A red line starts at the top, goes down the right side, then branches into two arrows. One arrow points left to the 'for' loop on line 7, and the other points down to the 'lowest=prices[s]' assignment on line 12.