0121. 买卖股票的最佳时机

ITCharge 大约 2 分钟

- 标签：数组、动态规划
- 难度：简单

题目链接

- 0121. 买卖股票的最佳时机 - 力扣

题目大意

描述：给定一个数组 `prices` ，它的第 `i` 个元素 `prices[i]` 表示一支给定股票第 `i` 天的价格。只能选择某一天买入这只股票，并选择在未来的某一个不同的日子卖出该股票。

要求：计算出能获取的最大利润。如果你不能获取任何利润，返回 0。

说明：

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

示例：

- 示例 1：

```py
输入：[7,1,5,3,6,4]
输出：5
解释：在第 2 天（股票价格 = 1）的时候买入，在第 5 天（股票价格 = 6）的时候卖出，最大利润 = 6-1 = 5 。
     注意利润不能是 7-1 = 6，因为卖出价格需要大于买入价格；同时，你不能在买入前卖出股票。
```

- 示例 2：

```py
输入：prices = [7,6,4,3,1]
输出：0
解释：在这种情况下，没有交易完成，所以最大利润为 0。
```

解题思路

最简单的思路当然是两重循环暴力枚举，寻找不同天数下的最大利润。但更好的做法是进行一次遍历，递推求解。

思路 1：递推

1. 设置两个变量 `minprice`（用来记录买入的最小值）、 `maxprofit`（用来记录可获取的最大利润）。
2. 从左到右进行遍历数组 `prices`。
3. 如果遇到当前价格比 `minprice` 还要小的，就更新 `minprice`。
4. 如果遇到当前价格大于或者等于 `minprice`，则判断一下以当前价格卖出的话能卖多少，如果比 `maxprofit` 还要大，就更新 `maxprofit`。
5. 最后输出 `maxprofit`。

思路 1：代码

```py
class Solution:
    def maxProfit(self, prices: List[int]) -> int:
        minprice = 10010
        maxprofit = 0
        for price in prices:
            if price < minprice:
                minprice = price
            elif price - minprice > maxprofit:
                maxprofit = price - minprice
        return maxprofit
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$，其中 n 是数组 `prices` 的元素个数。

- **空间复杂度**：$O(1)$。