122-买卖股票的最佳时机

题述

> 122. 买卖股票的最佳时机 II
>
> 难度 中等 1544
>
> 给定一个数组 `prices` ，其中 `prices[i]` 表示股票第 `i` 天的价格。
>
> 在每一天，你可能会决定购买和/或出售股票。你在任何时候 **最多** 只能持有 **一股** 股票。你也可以购买它，然后在 **同一天** 出售。
> 返回 *你能获得的* ***最大*** *利润* 。
>
> **示例 1:**
>
> **输入：** prices = [7,1,5,3,6,4]
> **输出：** 7
> **解释：** 在第 2 天（股票价格 = 1）的时候买入，在第 3 天（股票价格 = 5）的时候卖出，这笔交易所能获得利润 = 5-1 = 4 。
> 随后，在第 4 天（股票价格 = 3）的时候买入，在第 5 天（股票价格 = 6）的时候卖出，这笔交易所能获得利润 = 6-3 = 3 。
>
> **示例 2:**
>
> **输入：** prices = [1,2,3,4,5]
> **输出：** 4
> **解释：** 在第 1 天（股票价格 = 1）的时候买入，在第 5 天 （股票价格 = 5）的时候卖出，这笔交易所能获得利润 = 5-1 = 4 。
> 注意你不能在第 1 天和第 2 天接连购买股票，之后再将它们卖出。因为这样属于同时参与了多笔交易，你必须在再次购买前出售掉之前的股票。
>
> **示例 3:**
>
> **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。

思路

这道题难度定位为中等

乍一看非常麻烦

其实，要先清楚两点：

- 只有一只股票！
- 只有买股票或者卖股票的操作
- 想获得利润至少要以两天为一个交易单元

一开始可能能够想到，选一个低的买入，再选个高的卖出，如此往复

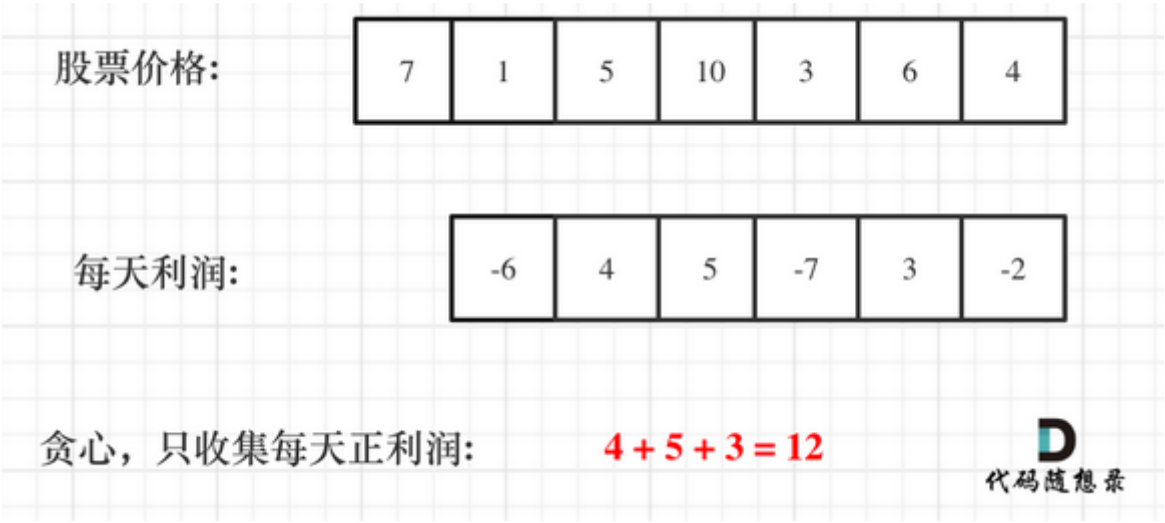
如果想到最终利润是可以分解的，那就好下手了

如何分解呢？

假如第0天买入，第3天卖出，那么利润为：prices[3] - prices[0]。

相当于(prices[3] - prices[2]) + (prices[2] - prices[1]) + (prices[1] - prices[0])。

这时，利润就分解为以天为单位的维度了，就便于计算了！



那么，第一天为什么没有利润呢？

第一天当然没有利润，至少要第二天才会有利润，所以利润的序列比股票序列少一天！

从图中可以发现，其实我们需要收集每天的正利润就可以，**收集正利润的区间，就是股票买卖的区间，而我们只需要关注最终利润，不需要记录区间。**

那么只收集正利润就是贪心所贪的地方！

局部最优：收集每天的正利润，全局最优：求得最大利润。

局部最优可以推出全局最优，找不出反例，试一试贪心！

题解

C++

```
class Solution {
public:
    int maxProfit(vector<int>& prices)
    {
        //贪心
        //局部最优：收集每天的正利润
        //全局最优：求得最大利润
        int result=0;
        for(int i = 1; i < prices.size(); i++)
        {
```

```
            result += max(prices[i]-prices[i-1],0);
        }
        return result;
        //是不是想不到这个这么简单？
    }
};
```

Python

```python
class Solution:
    def maxProfit(self, prices: List[int]) -> int:
        result = 0
        for i in range(1,len(prices)):
            result += max(prices[i]-prices[i-1],0)  #只收集正利润
        return result
```

思考

股票问题其实是一个系列的，属于动态规划的范畴，因为目前在讲解贪心系列，所以股票问题会在之后的动态规划系列中详细讲解。

可以看出有时候，贪心往往比动态规划更巧妙，更好用，所以别小看了贪心算法。

本题中理解利润拆分是关键点！ 不要整块的去看，而是把整体利润拆为每天的利润。

一旦想到这里了，很自然就会想到贪心了，即：只收集每天的正利润，最后稳稳的就是最大利润了