**122. Best Time to Buy and Sell Stock II**

Easy

Say you have an array for which the *i*th element is the price of a given stock on day *i*.

Design an algorithm to find the maximum profit. You may complete as many transactions as you like (i.e., buy one and sell one share of the stock multiple times).

**Note:** You may not engage in multiple transactions at the same time (i.e., you must sell the stock before you buy again).

**Example 1:**

**Input:** [7,1,5,3,6,4]

**Output:** 7

**Explanation:** Buy on day 2 (price = 1) and sell on day 3 (price = 5), profit = 5-1 = 4.

  Then buy on day 4 (price = 3) and sell on day 5 (price = 6), profit = 6-3 = 3.

**Example 2:**

**Input:** [1,2,3,4,5]

**Output:** 4

**Explanation:** Buy on day 1 (price = 1) and sell on day 5 (price = 5), profit = 5-1 = 4.

  Note that you cannot buy on day 1, buy on day 2 and sell them later, as you are

  engaging multiple transactions at the same time. You must sell before buying again.

**Example 3:**

**Input:** [7,6,4,3,1]

**Output:** 0

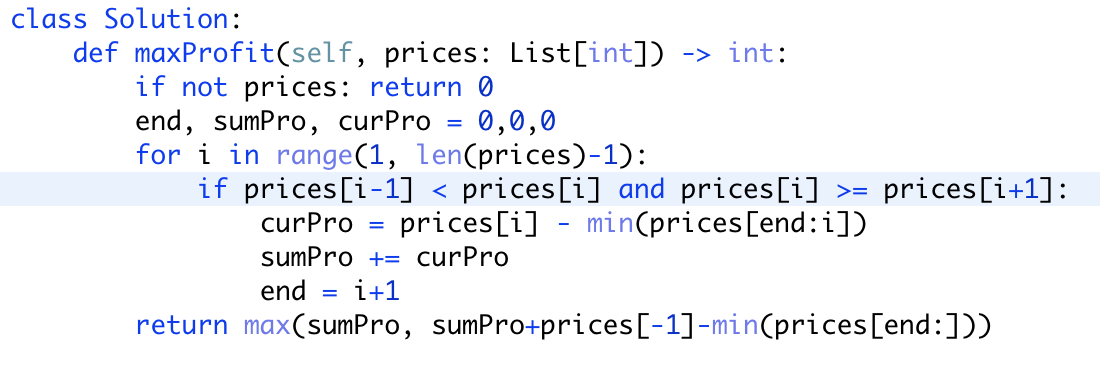
**Explanation:** In this case, no transaction is done, i.e. max profit = 0.

是121的变形题，差别就在与可以进行**多次交易**。所以大致思路也延续121的两个思路：

1. 寻找拐点法：

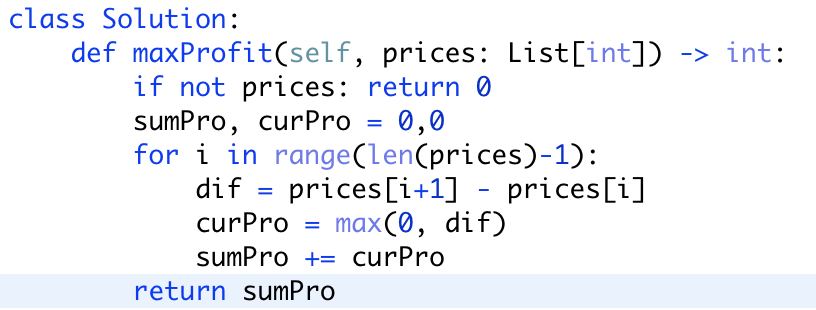
主要思想：当 prices[i-1] < prices[i] and prices[i] >= prices[i+1] 时，prices[i]一定是一个峰值，即可以卖出的时刻。将其与上一次出售后数组中最小值做差，能得到此次利润。记录利润和。注意：数列最后一个数并不能记录进去，所以最后要计算一下是否有利润。

End 用来记录上一次出售的位置，sumPro是和利润, curPro是当前利润。



1. Kadane's Algorithm

十分不错的算法，用在这一题甚至比121还要简单。还不用记录上次交易结束的位置。



改良版：

