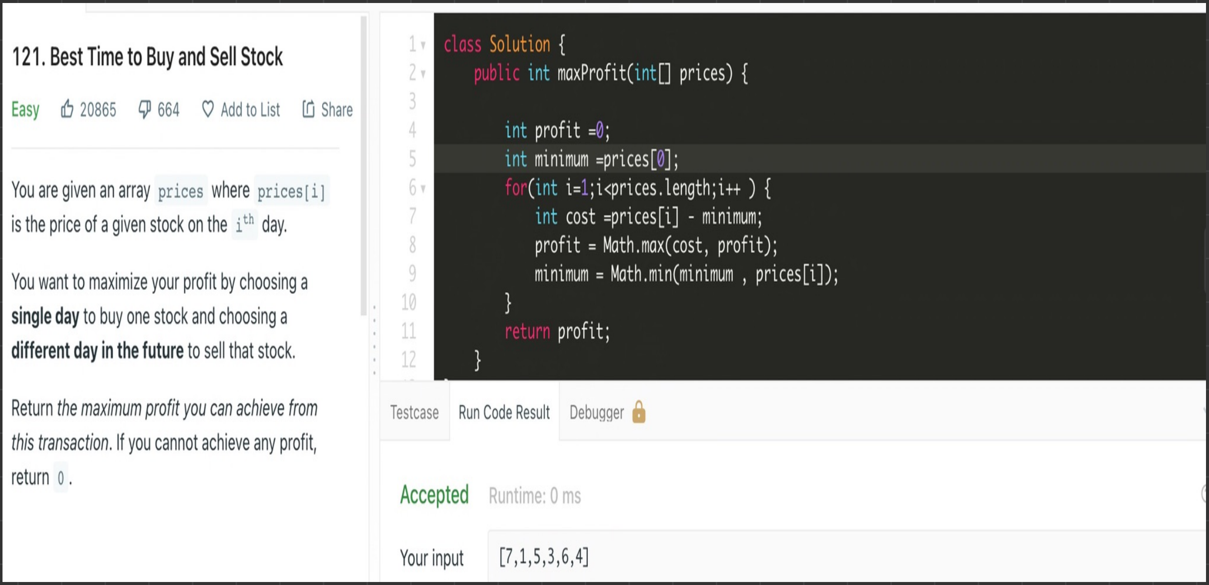
# Best time to buy and sell stock



## 121. Best Time to Buy and Sell Stock

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.

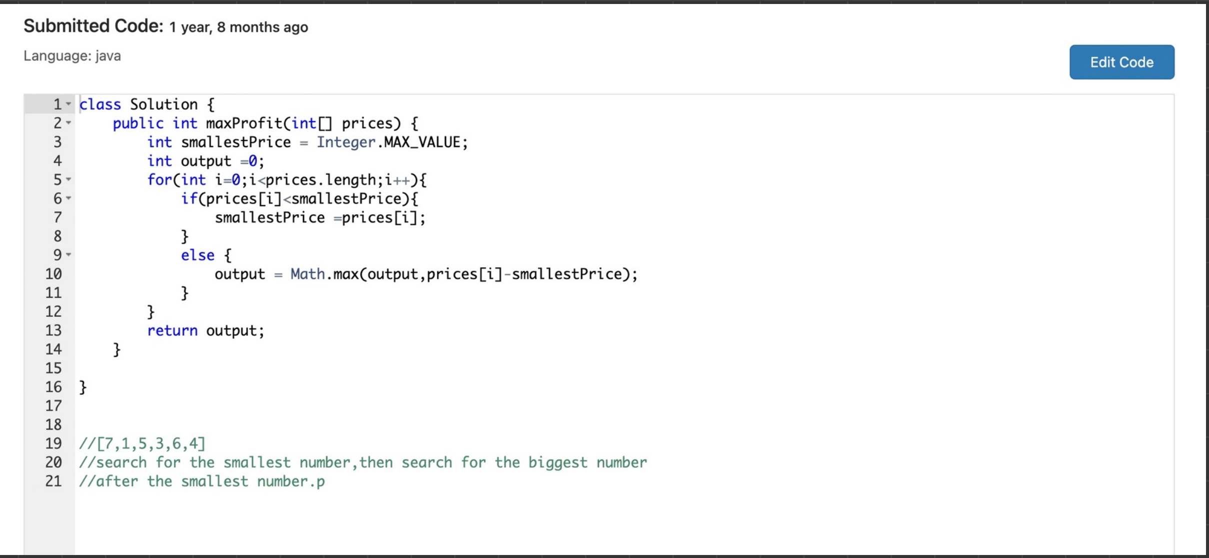
Return *the maximum profit you can achieve from this transaction*. If you cannot achieve any profit, return `0`.

```java
class Solution {
    public int maxProfit(int[] prices) {

        int profit =0;
        int minimum =prices[0];
        for(int i=1;i<prices.length;i++ ) {
            int cost =prices[i] - minimum;
            profit = Math.max(cost, profit);
            minimum = Math.min(minimum , prices[i]);
        }
        return profit;
    }
```

```
Accepted   Runtime: 0 ms

Your input   [7,1,5,3,6,4]
```



**Submitted Code:** 1 year, 8 months ago

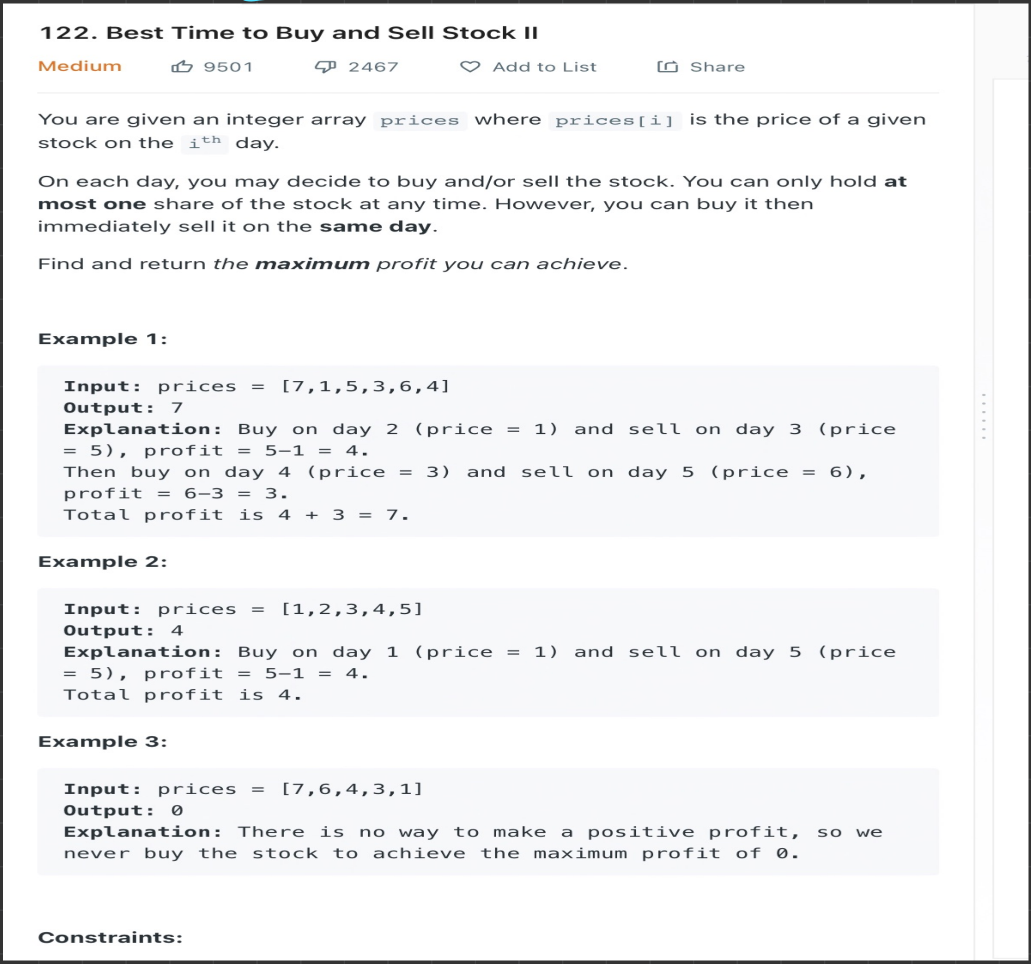
Language: java

```java
class Solution {
    public int maxProfit(int[] prices) {
        int smallestPrice = Integer.MAX_VALUE;
        int output =0;
        for(int i=0;i<prices.length;i++){
            if(prices[i]<smallestPrice){
                smallestPrice =prices[i];
            }
            else {
                output = Math.max(output,prices[i]-smallestPrice);
            }
        }
        return output;
    }

}


//[7,1,5,3,6,4]
//search for the smallest number,then search for the biggest number
//after the smallest number.p
```

# Best time to buy and sell stock 2



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

You are given an integer array `prices` where `prices[i]` is the price of a given stock on the $i^{th}$ day.

On each day, you may decide to buy and/or sell the stock. You can only hold **at most one** share of the stock at any time. However, you can buy it then immediately sell it on the **same day**.

Find and return *the **maximum** profit you can achieve*.

**Example 1:**

```
Input: prices = [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.
Total profit is 4 + 3 = 7.
```

**Example 2:**

```
Input: prices = [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.
Total profit is 4.
```

**Example 3:**

```
Input: prices = [7,6,4,3,1]
Output: 0
Explanation: There is no way to make a positive profit, so we
never buy the stock to achieve the maximum profit of 0.
```

**Constraints:**

→ Index is definitely changing

→ By Buy / Sell option

like Buy / sell

if Buy = 1

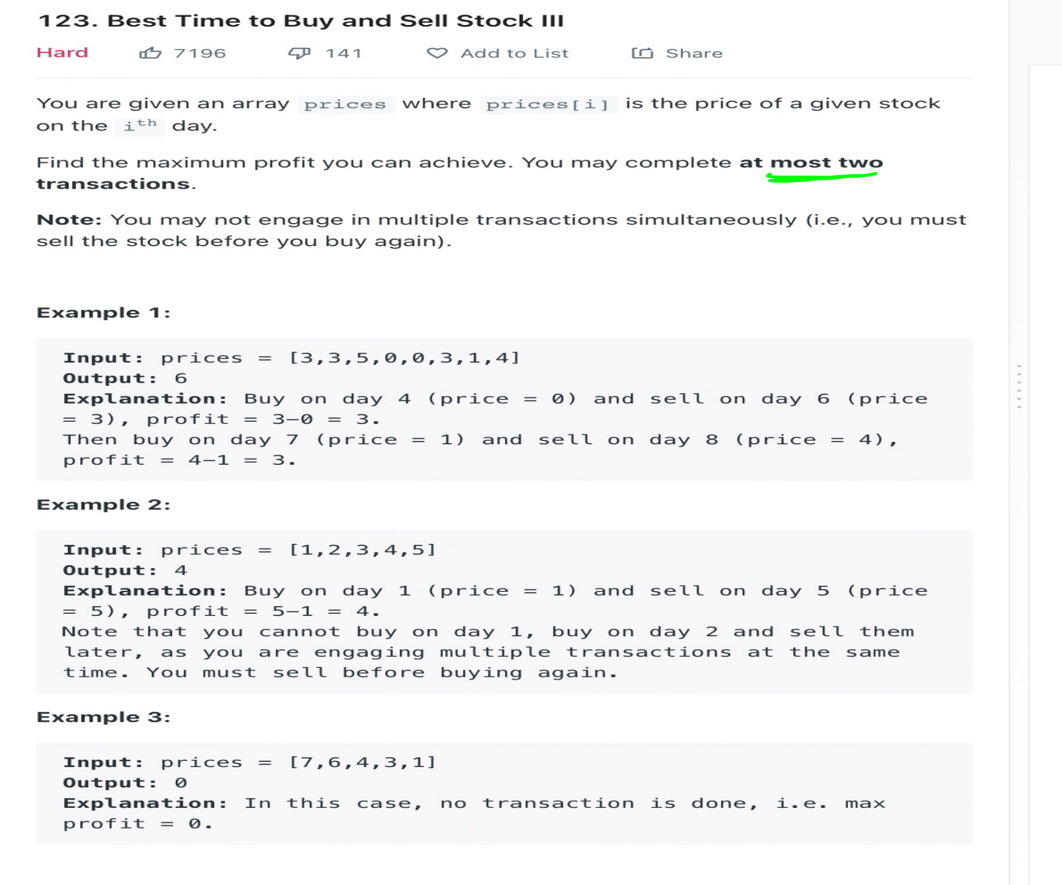
Buy = True ✓ Buy

Buy = False ⇒ sell

7, 1, 5, 3, 6, 4

```
f(ind, Buy) {
    // Base case
        ↳ if ind == n
              return 0
    if Buy :
        Max { -price[ind] + f(ind+1, F)
              0 + f(ind+1, T) }
    else :
        Max { +price [ind] + f(ind+1, T)
              + 0 + f(ind+1, F) }
```

# Best time to buy and sell stock 3



## 123. Best Time to Buy and Sell Stock III

Hard 7196 141 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.

Find the maximum profit you can achieve. You may complete **at most two transactions**.

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

**Example 1:**

```
Input: prices = [3,3,5,0,0,3,1,4]
Output: 6
Explanation: Buy on day 4 (price = 0) and sell on day 6 (price = 3), profit = 3-0 = 3.
Then buy on day 7 (price = 1) and sell on day 8 (price = 4), profit = 4-1 = 3.
```

**Example 2:**

```
Input: prices = [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: prices = [7,6,4,3,1]
Output: 0
Explanation: In this case, no transaction is done, i.e. max profit = 0.
```

f (ind, buy, cap)

// Base case

```
If (ind==n)
    return 0
If cap==0
    return 0;
```

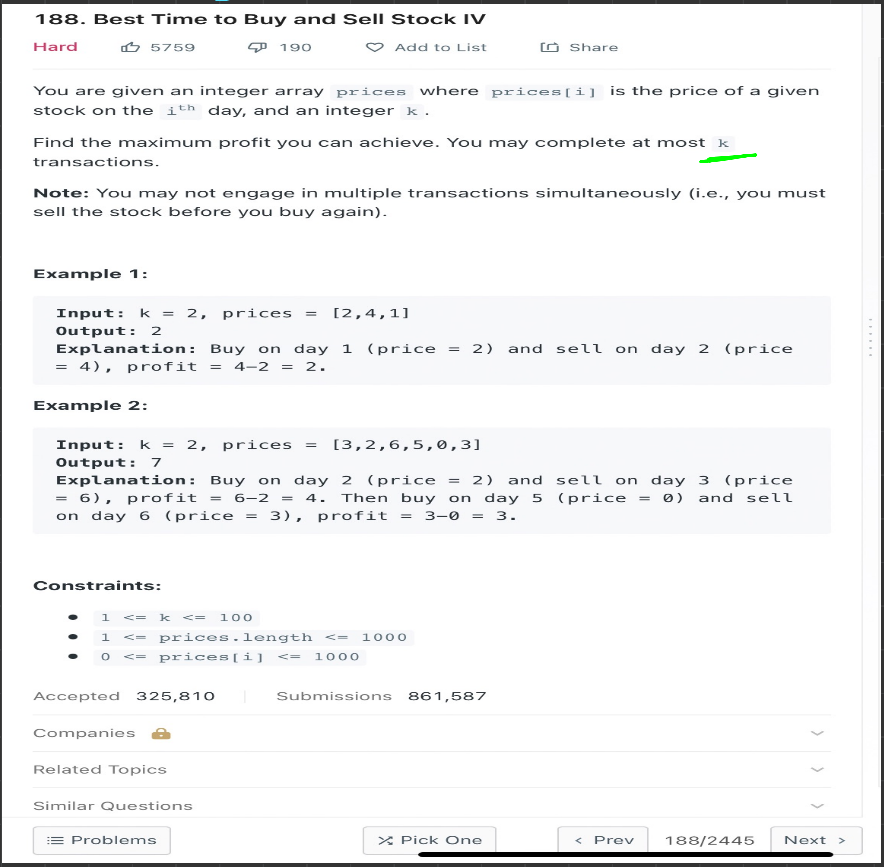
```
If (buy)
    Max { -price[ind] + f(ind+1, 0, cap)
          0 + f(ind+1, 1, cap) }
```

else

sell

$$\max\left\{\begin{array}{l} price[ind] + f(ind+1, 1, Cap-1 \\ 0 + f(ind+1, 0, cap) \end{array}\right]$$

Best time to buy and sell stock 4



### 188. Best Time to Buy and Sell Stock IV

You are given an integer array `prices` where `prices[i]` is the price of a given stock on the $i^{th}$ day, and an integer `k`.

Find the maximum profit you can achieve. You may complete at most `k` transactions.

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

**Example 1:**

```
Input: k = 2, prices = [2,4,1]
Output: 2
Explanation: Buy on day 1 (price = 2) and sell on day 2 (price
= 4), profit = 4-2 = 2.
```

**Example 2:**

```
Input: k = 2, prices = [3,2,6,5,0,3]
Output: 7
Explanation: Buy on day 2 (price = 2) and sell on day 3 (price
= 6), profit = 6-2 = 4. Then buy on day 5 (price = 0) and sell
on day 6 (price = 3), profit = 3-0 = 3.
```

**Constraints:**

- `1 <= k <= 100`
- `1 <= prices.length <= 1000`
- `0 <= prices[i] <= 1000`

Accepted 325,810 | Submissions 861,587