121 买卖股票的最佳时机

Lable：滑动窗口、动态规划

给定一个数组，它的第 i 个元素是一支给定股票第 i 天的价格。

如果你最多只允许完成一笔交易（即买入和卖出一支股票一次），设计一个算法来计算你所能获取的最大利润。

注意：你不能在买入股票前卖出股票。

示例1：

```
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：

```
输入: [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。
```

- 遍历一次 相当于找到minPrice，逐步向后移动，更新maxProfit

```
class Solution {
    public int maxProfit(int[] prices) {
       int minPrice = Integer.MAX_VALUE;
       int maxProfit = 0;
       for(int i : prices){
           minPrice = Math.min(minPrice,i);
           maxProfit = Math.max(maxProfit, i - minPrice);
       }
       return maxProfit;
    }
}
// 理解为滑动窗口
class Solution {
    public int maxProfit(int[] prices) {
      //滑动窗口
    int begin = 0,maxProfit = 0,minPrice = Integer.MAX_VALUE;
    for(int end = 0; end < prices.length; end++){
        if(prices[end] < prices[begin]){
            begin = end;
        }
        maxProfit = Math.max(maxProfit, prices[end] - prices[begin]);
    }
    return maxProfit;
    }
}
```

- 遍历求差值

```
class Solution {
    public int maxProfit(int[] prices) {
        int max = Integer.MIN_VALUE;  // 记忆一下
        for (int i = 0 ; i < prices.length; i++) {
            for (int j = i + 1 ; j < prices.length; j++) {
                int diff = prices[j] - prices[i];
                max = Math.max(diff, max);
            }
        }
        return Math.max(max,0);
    }
}
```