<https://leetcode.com/problems/best-time-to-buy-and-sell-stock>

**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 ith 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.**

Example 1:

Input: prices = [7,1,5,3,6,4]

Output: 5

Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.

Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.

Example 2:

Input: prices = [7,6,4,3,1]

Output: 0

Explanation: In this case, no transactions are done and the max profit = 0.

Constraints:

1 <= prices.length <= 105

0 <= prices[i] <= 104

**Method 1: (Brute Force)**

For

Time Complexity: O()

Space Complexity: O()

**Method 2: (Sliding window)**

Two pointers minL points to minimum left value, maxR points to maximum right value.

For each element if it is minL or maxR

Max profit =maxR- minL

Time Complexity: O(n) *[]*

Space Complexity: O(1) *[]*

int maxProfit(vector<int>& prices) {

        int minL=10001 , maxR=-1 , i=0, n=prices.size(), profit=0;

        while(i<n){

            int x = prices[i];

            if( x<minL ) { minL=x; maxR=x; }

            else if(x>maxR){

                maxR=x;

                profit = max(profit, (maxR-minL));

            }

            i++;

        }

        return profit;

    }