

Best time to buy & sell stock

You are given an array/list 'prices' where the elements of the array represent the prices of the stock as they were yesterday and indices of the array represent minutes. Your task is to find and return the maximum profit you can make by buying and selling the stock. You can buy and sell the stock only once.

Note:

You can't sell without buying first.

For Example:

For the given array [2, 100, 150, 120],

The maximum profit can be achieved by buying the stock at minute 0 when its price is Rs. 2 and selling it at minute 2 when its price is Rs. 150.

So, the output will be 148.

Detailed explanation (input/output format, Notes, Images)

Constraints:

$1 \leq T \leq 10$

$2 \leq N \leq 10^4$

$1 \leq \text{ARR}[i] \leq 10^9$

Time Limit: 1 sec.

Sample Input 1:

2

4

1 2 3 4

4

2 2 2 2

Sample Output 1:

3

0

Explanation For Sample Output 1:

For the first test case, it's optimal to buy the stock at minute 0 and sell it at minute 3 to get a maximum profit of 3.

For the second test case, the maximum profit will be 0 for all possible ways of buying and selling stock.



Above might be the curve of the elements present in the array.

Our answer would be the difference b/w the green dots.

```
#include <bits/stdc++.h>
int maximumProfit(vector<int> &prices){

    int minPrice = INT_MAX, maxProfit = 0;

    for(int i = 0; i < prices.size(); i++){
        if(minPrice > prices[i]) minPrice = prices[i];
        maxProfit = max(maxProfit, prices[i] - minPrice);
    }
    return maxProfit;
}
```

Time Complexity : $O(n)$

Space Complexity : $O(1)$

Best time to buy & sell stock II

Problem statement

[Send feedback](#)

You have been given stock values/prices for N number of days. Every i-th day signifies the price of a stock on that day. Your task is to find the maximum profit which you can make by buying and selling the stocks.

Note :

You may make as many transactions as you want but can not have more than one transaction at a time i.e, if you have the stock, you need to sell it first, and then only you can buy it again.

Detailed explanation (Input/output format, Notes, Images)

Constraints :

$1 \leq t \leq 10^2$

$0 \leq N \leq 10^5$

Time Limit: 1 sec

Sample Input 1 :

1
7
1 2 3 4 5 6 7

Sample Output 1 :

6

Explanation :

We can make the maximum profit by buying the stock on the first day and selling it on the last day.

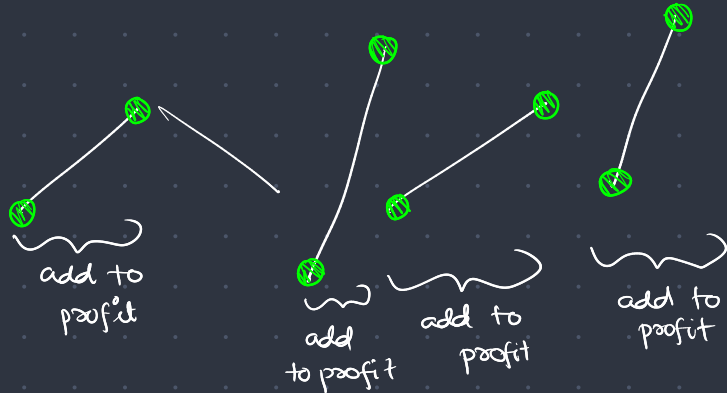
Sample Input 2 :

1
7
7 6 5 4 3 2 1

Sample Output 2 :

0

Explanation :



→ We will add each increasing graph to our profit to get the required profit.

```
long getMaximumProfit(long *values, int n)
{
    long smaller = -1, larger = -1;
    long profit = 0;
    larger = values[0];
    smaller = values[0];
    for(int i = 1; i < n; i++) {
        if(values[i] > larger) larger = values[i];

        if(values[i] < larger) {
            profit += (larger - smaller);
            smaller = values[i];
            larger = smaller;
        }
    }
    profit += (larger - smaller);
    return profit;
}
```

Time Complexity : $O(n)$

Space Complexity : $O(1)$