Best time to Buy and Sell Stocks

Brute force Approach :

Iterate through the array find cp,sp and profit maintain a max variable for profit return the maxprofit.

```
#include <bits/stdc++.h>
int maximumProfit(vector<int> &prices){
    // Write your code here.
    int profit=0,maxProfit=0;
    for(int i=0;i<prices.size();i++)
    {
        int buy=prices[i];
        for(int j=i+1;j<prices.size();j++)
        {
            profit=prices[j]-buy;
            maxProfit=max(profit,maxProfit);
        }
    }
    return maxProfit;
}
```

- Time Complexity : O(N^2)
- Space Complexity : O(N)

Approach 2:

We use a sell vector to keep track of what could be the highest selling price of the stock bought on a particular day for this we need elements that has a maximum value on its right so we make use of a vector to store maximum value. Iterating from right the stock bought on last day can be sold at that price only so sell[n-1]=prices[n-1] when we move to n-2 then we compare whether the last element was greater or the current element and store the value in sell that is maximum.

After sell vector is created we subtract the values of sell and price to find the maxProfit.

```
#include <bits/stdc++.h>
int maximumProfit(vector<int> &prices){
    // Write your code here.
    int n=prices.size(),maxProfit=INT_MIN;
    vector<int> sell(n,0);
    sell[n-1]=prices[n-1];
    for(int i=n-2;i>=0;i--)
    {
        sell[i]=max(prices[i],sell[i+1]);
    }
    for(int i=0;i<n;i++)
    {
        maxProfit=max(sell[i]-prices[i],maxProfit);
    }
    return maxProfit;
}
```

- Time Complexity : O(N)
- Space Complexity : O(N)

Approach 3 : Using Kadane’s Algo

We keep a track of the min amount at which we can buy a stock. Initially we buy it at a very huge amt say INT_MAX if we get a less amt for a stack we update the min buy amount and we also keep a track of max amount at which the stock can be sold using the min buy value if the price of the stock - minBuy is greaterthan maxSell of stock then we update the maxSell.

```
#include <bits/stdc++.h>
int maximumProfit(vector<int> &prices){
    // Write your code here.
    int minBuy=INT_MAX,maxSell=0;
    for(int i=0;i<prices.size();i++)
    {
        if(prices[i]<minBuy)
            minBuy = prices[i];
        if(prices[i]-minBuy>maxSell)
            maxSell=prices[i]-minBuy;

    }
    return maxSell;
}
```

- Time Complexity : O(N)
- Space Complexity: O(1)