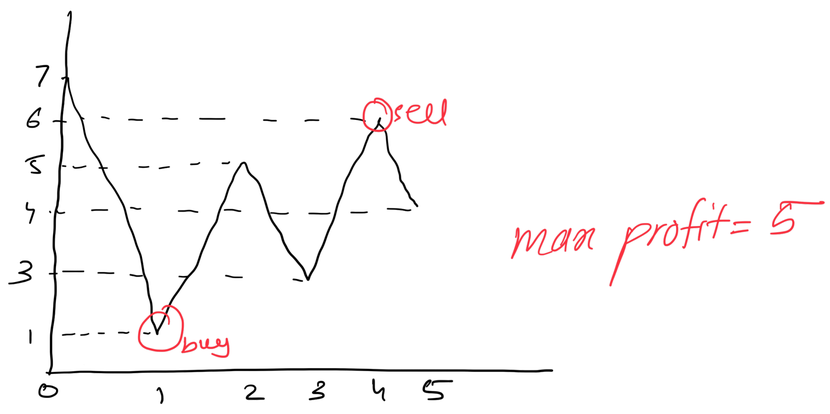
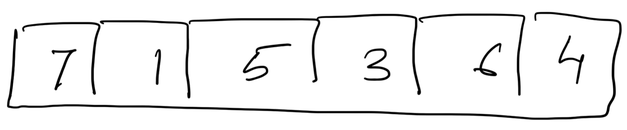
Buy & sell stock k best time

prices = [7, 1, 5, 3, 6, 4] return max profit



max profit = 5

first buy & choose diff day to sell (in future)



sabse min

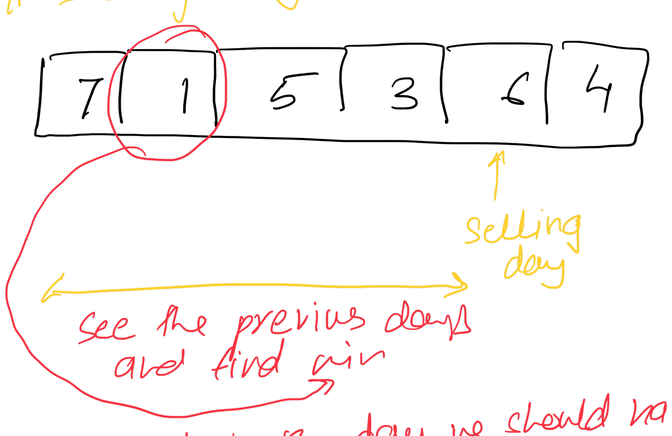
buy

sell

sabse expensive in future

imagine every day as a selling day and see the previous days and find at what you should've bought it for max profit

if selling day is at 6 rn



∴ 1 is the day we should have bought it to get max profit at 6

find max profit like this every single day seeing the previous ones & compare

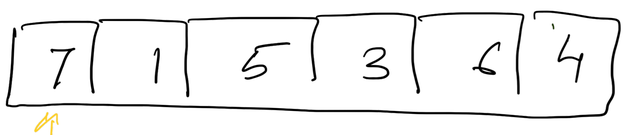
global var's var profit

maxProfit = 0

bestBuy = arr[0] ← stores the min value before the selling day

so in the start bestBuy will be arr[0]

koi bhi idx ke liye uska bestBuy will be 0 to idx-1 ke bich mein



selling day will never stored at idx 0 but idx 1

```
maxProfit = 0
bestBuy = arr[0] = 7
for (i=1; i<n; i++) {
    if (price[i] > bestBuy) {
        maxProfit = max(maxProfit, price[i] - bestBuy);
    }
    bestBuy = min(BB price[i])
}
```

time complexity : O(n)

Dry run

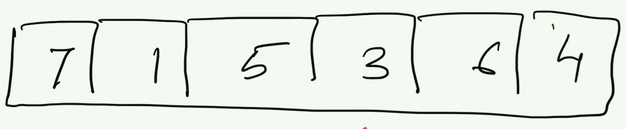
MP = 0 BB = 7

price ≯ bestBuy ⇒ doesn't go in loop

MP = 0 bb = 1

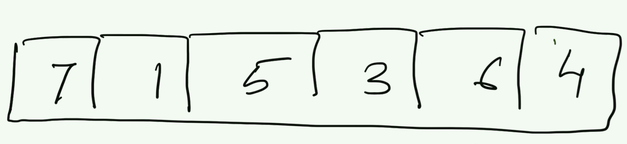
price > bestBuy ⇒ MP = max(0, 5-1) = 4

MP = 4 bb = 1



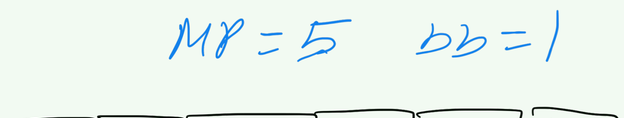
price > bestBuy ⇒ MP = max(4, 3-1) = 4

MP = 4 bb = 1



price > bestBuy ⇒ MP = max(4, 6-1) = 5

MP = 5 bb = 1



price > bestBuy ⇒ MP = max(5, 4-1) = 5

MP = 5 bb = 1