Question: [https://leetcode.com/problems/best-time-to-buy-and-sell-stock/](https://leetcode.com/problems/two-sum/)

Constraints: Choose a ****single day**** to buy one stock and choosing a ****different day in the future**** to sell that stock i.e., we can perform only one transactions.

We take prices array as [5, 6, 2, 4, 8, 9, 5, 1, 5]  
In the given problem, we assume the first element as the stock we initially buy.  
Now we will traverse the array from left to right. And if a element is cheaper than the present buying price then set buy = ar[i] and set sell as 0, as we are buying so the selling price should be reset as well, else set sell a ar[i] and then calculate the profit , and check if its max profit or not.

So in the given array 5 is the stock we bought. So next element is 6. If we sell the stock at that price we will earn profit of $1.

Prices: [5, 6, 2, 4, 8, 9, 5, 1, 5]

Profit: Bought:5 Sell:6 Profit:$1 max profit=$1

Now the next element is 2 which have lower price than the stock we bought previously which was 5.

Profit: Bought:2 Sell:- Profit:- max profit=$1

Next element is 4 which has higher price than the stock we bought. So if we sell the stock at this price.

Profit: Bought:2 Sell:4 Profit:$2 max profit=$2

Profit: Bought:2 Sell:8 Profit:$6 max profit=$6

Profit: Bought:2 Sell:9 Profit:$7 max profit=$7

Profit: Bought:2 Sell:5 Profit:$3 max profit=$7

Now next stock price is $1 which is less than the stock we bought of $2.

Profit: Bought:1 Sell:- Profit:- max profit=$7

Now next stock is of $5. So this price is higher than the stock we bought.

Profit: Bought:1 Sell:5 Profit:$4 max profit=$7

But our maximum profit will be $7.

Solution:

class Solution {

public int maxProfit(int[] ar) {

int buy=ar[0], sell=0, maxProfit=0;

for(int i=1; i<ar.length; i++){

if(ar[i]<buy){

buy=ar[i];

sell=0;

}else{

sell=ar[i];

}

maxProfit=Math.max(sell-buy, maxProfit);

}

return maxProfit;

}

}