Problem #121: Best Time to Buy and Sell Stock

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

Solution:

1. Initialize max\_profit to 0 and min\_price to the price on the first day.

2. Then traverse the array of prices. If the price on day i is less than the min\_price,

Then min\_price becomes that day’s price.

If the price that day is greater than min\_price, then see what profit can be made.

If the profit is greater than max\_profit, then set max\_profit to the profit that can be made that day.

3. Return max\_profit.

----------------------------------

class Solution:

def maxProfit(self, prices: List[int]) -> int:

n = len(prices)

if n <= 1:

return(0)

max\_profit = 0

min\_price = prices[0] # initialize minimum price so far to the first day's price

# we are traversing the array from index 1 to n - 1 to see what is max profit we can make if we sell on that day

for i in range(1, n):

if prices[i] < min\_price: # price today is less than minimum price we have seen so far, then make it min\_price

min\_price = prices[i]

elif prices[i] > min\_price: # price today is greater than minimum\_price so far, profit = prices[i] - min\_price

if prices[i] - min\_price > max\_profit: # check if profit is greater than max\_profit, then update max\_profit

max\_profit = prices[i] - min\_price

return(max\_profit)

----------------------------------------

Shorter Solution using the same logic:

Shorter solution:

class Solution:  
    def maxProfit(self, prices: List[int]) -> int:  
        n = len(prices)  
        if n <= 1:  
            return(0)  
        max\_profit = 0  
        min\_price = prices[0]    
         
        for price in prices:  
            min\_price = min(min\_price, price)  
            max\_profit = max(max\_profit, price - min\_price) # profit that day = price - min\_price  
         
        return(max\_profit)

-------------