

3.
1
4

Best Time to Buy and Sell Stock

EASY

■ What You Will Learn

- ✓ Track minimum buy price as you scan right
- ✓ Calculate profit at each day as price - minSoFar
- ✓ One pass — same idea as Kadane's variant

■ Intuition First

You can only buy once and sell once. As you walk through prices left to right, you want to buy at the cheapest day you've seen so far, and sell at the best price after that day. Track the minimum price seen so far, and at each day compute: what's my profit if I sell today?

■ The Core Idea

- minPrice = lowest price seen so far (potential buy day)
- At each price: profit = price - minPrice
- maxProfit = max of all such profits
- Update minPrice if current price is lower

■ Visual Trace on [7,1,5,3,6,4]

Buy at 1 (day 2), sell at 6 (day 5) = profit 5

7	1	5	3	6	4
---	---	---	---	---	---

```
price=7: min=7, profit=0, max=0
price=1: min=1, profit=0, max=0
price=5: profit=5-1=4, max=4
price=3: profit=3-1=2, max=4
price=6: profit=6-1=5, max=5 ✓
price=4: profit=4-1=3, max=5
```

■ Java Code

```
public static int maxProfit(int[] prices) {
    int minPrice = prices[0];
    int maxProfit = 0;
    for (int i = 1; i < prices.length; i++) {
```

```
// Profit if we sell today
int profit = prices[i] - minPrice;
maxProfit = Math.max(maxProfit, profit);
// Update cheapest buy day
minPrice = Math.min(minPrice, prices[i]);
}
return maxProfit;
}
```

■ Complexity

Case	Time	Space	What happens?
All Cases	O(n)	O(1)	Single pass, two variables

■ Common Mistakes

- You MUST buy before sell — left to right only
- If prices always go down, return 0 (no profitable trade) — maxProfit starts at 0
- Don't use nested loops — $O(n^2)$ is too slow

■ Extension

- Multiple transactions allowed: sum of all upward slopes — $O(n)$ greedy
- At most 2 transactions: use DP with states
- At most K transactions: harder DP

■ Practice Problems

[Easy] What if prices are [7,6,4,3,1]? What's max profit?

[Med] Best time with multiple unlimited transactions

[Hard] Best time with at most 2 transactions

■ Quick Revision

- Track minPrice and maxProfit as you scan
- profit = price - minPrice at each step
- Update minPrice = min(minPrice, price)

- maxProfit starts at 0 (no trade = 0 profit)
- $O(n)$ time, $O(1)$ space