Problem Name: Best Time to Buy and Sell Stock

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

Solution Language(s): Java

Approach

In this problem we are tasked with determining the best day to buy and then sell a stock for profit based on an array which lists the stock price each day. We are returning the profit only, so we do not need to store the purchasing and selling days as we go. In this problem we are essentially starting with the first day as the buy day, and the next day as the sell. If the sell day price is lower than the buy day price, we change the buy day price to the sell day. If it is greater, we check if the profit would be greater than the current greatest profit we have found. If it is, we update the greatest possible profit. If not, we simply increment the sell day until we go through the entire array. At this point we return the profit.

Solution

```java
class Solution {
    public int maxProfit(int[] prices) {
        int profit = 0;
        int buy = 0;
        for (int sell = 1; sell < prices.length; sell++) {
            if (prices[sell] - prices[buy] > profit) {
                profit = prices[sell] - prices[buy];
            } else if (prices[sell] < prices[buy]) {
                buy = sell;
            }
        }
        return profit;
    }
}
```

Time Complexity: O(n)

The time complexity in this approach is O(n), where *n* is the number of elements in *prices,* as we go through the entire array starting at the 1st element.

Space Complexity: O(1)

The space complexity of this method is O(1) because we are only storing the max profit in a variable, and that is constant complexity.

Conclusion

In conclusion, this problem is relatively simple. We simply need to find the greatest difference between two values in the array, which I will call left and right, and ensure that the left value is less than the right value.