Best Time to Buy and Sell Stock II

Problem Statement

- You are given an integer array prices where prices[i] is the price of a given stock on the ith day. On each day, you may decide to buy and/or sell the stock. You can only hold at most one share of the stock at any time. However, you can buy it and then immediately sell it on the same day. Your goal is to find and return the maximum profit you can achieve.

Example 1:

- **Input:** prices = [7, 1, 5, 3, 6, 4]
- **Output:** 7
- **Explanation:**
  - Buy on day 2 (price = 1) and sell on day 3 (price = 5), profit = 5 1 = 4.
  - Then buy on day 4 (price = 3) and sell on day 5 (price = 6), profit = 6 3 = 3.
  - Total profit is 4 + 3 = 7.

Example 2:

- **Input:** prices = [1, 2, 3, 4, 5]
- **Output:** 4
- **Explanation:**
  - Buy on day 1 (price = 1) and sell on day 5 (price = 5), profit = 5 1 = 4.
  - Total profit is 4.

Example 3:

- **Input:** prices = [7, 6, 4, 3, 1]
- **Output:** 0
- **Explanation:** There is no way to make a positive profit, so we never buy the stock to achieve the maximum profit of 0.

Constraints

- 1 <= prices.length <= 30,000
- 0 <= prices[i] <= 10,000

Approach

- The goal is to maximize the profit by taking advantage of every upward trend in the stock prices. The key observation is that every local increase should be added to the total profit. Therefore, for each pair of consecutive days, if the price on the second day is higher than the price on the first day, we should "buy" on the first day and "sell" on the second day, adding the difference to the total profit.
- This approach ensures that we capture all upward movements in the stock prices, which leads to the maximum possible profit.

Solution

- The solution involves iterating through the prices array and accumulating the profit whenever there is an increase from one day to the next.
- The detailed solution is implemented in the provided code snippet.