Online on Array

Section: C1+C2 Time: 60 minutes

Problem 1: Given an array of non-negative integers representing an elevation map where the width of each bar is 1, compute how much water can be trapped after it rains.

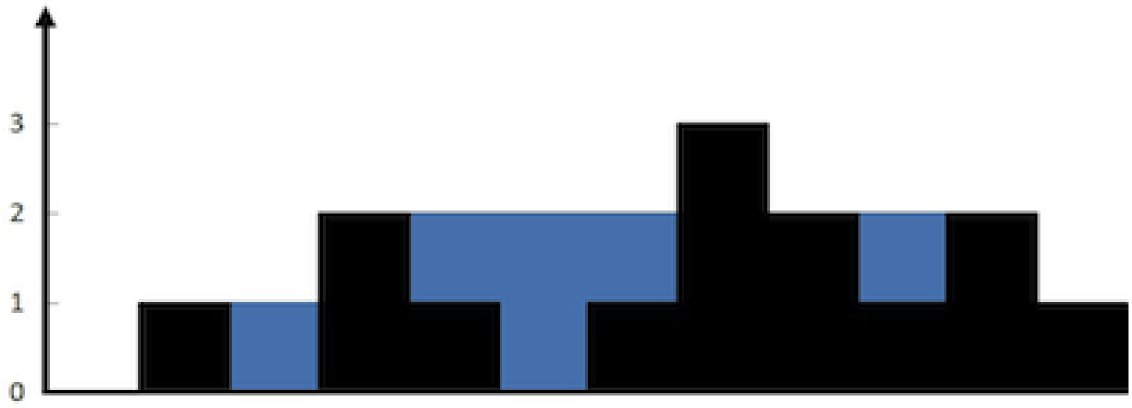
Input:

- First line: An integer n ($1 \le n \le 10^5$) — the number of bars.
- Second line: n non-negative integers separated by space — representing the height of each bar.

Output:

- Print a single integer — the total amount of water that can be trapped.

Examples:



Input:
12
0 1 0 2 1 0 1 3 2 1 2 1
Output: 6
Explanation: Water is trapped in the dips between taller bars (See above figure).

Input:

6
1 2 3 4 5 6
Output: 0

Input:
6
6 5 4 3 2 1
Output: 0

Problem 2:

You are given an array representing the prices of a stock over n days. You can perform **at most one transaction** (i.e., buy once and sell once). Your task is to determine the maximum profit you can make.

You **must buy** the stock before you **sell** it.
If no profit can be made, output 0.

Input:

- First line: An integer n ($1 \leq n \leq 10^5$) — the number of days.
- Second line: n space-separated integers — the price of the stock on each day.

Output:

- Print a single integer — the maximum profit that can be achieved. Print 0 if no profit is possible.

Hint:

- Iterate through the array while maintaining the minimum price seen so far.
- Calculate potential profit at each step by subtracting the minimum from the current price.
- Keep track of the maximum profit.

Examples:
Input:
6
7 1 5 3 6 4
Output: 5
Explanation:
Buy at price 1 (on day 2)
Sell at price 6 (on day 5)

Input:
6
7 6 5 4 3 2
Output: 0
Explanation: Prices are strictly decreasing, so no profitable transaction is possible. Best to do nothing.