B - Collecting Water

**Trapping Rain Water**

Given n non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it can trap after raining.

**Example 1:**



**Input:** height = [0,1,0,2,1,0,1,3,2,1,2,1]

**Output:** 6

**Explanation:** The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.

**Example 2:**

**Input:** height = [4,2,0,3,2,5]

**Output:** 9

**Constraints:**

* n == height.length
* 1 <= n <= 2 \* 104
* 0 <= height[i] <= 105

#include <iostream>

#include <vector>

#include<algorithm>

using namespace *std*;

int main()

{

int t; *cin* >> t;

while (t--)

{

int n; *cin* >> n;

*vector*<int> height(n);

for (auto i = 0; i < n; i++)

*cin* >> height[i];

*vector*<int> pMax(n);

int ph = pMax[0] = height[0];

for (auto i = 1; i < n; i++)

{

if (ph < height[i])

ph = height[i];

pMax[i] = ph;

}

*vector*<int> sMax(n);

int sh = sMax[n - 1] = height[n - 1];

for (auto i = n - 1; i >= 0; i--)

{

if (sh < height[i])

sh = height[i];

sMax[i] = sh;

}

int totalStoredWater = 0;

for (auto i = 0; i < n; i++)

{

totalStoredWater+= *min*(pMax[i], sMax[i]) - height[i];

}

*cout* << totalStoredWater << *endl*;

}

return 0;

}