### Trapping Rain Water

**Company Tags**

[**Flipkar t**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Flipkart)[**Amazon**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Amazon) [**Microsoft**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Microsoft) Google

### Statement….

Given an array **arr[]** of **N** non-negative integers representing the height of blocks. If width of each block is 1, compute how much water can be trapped between the blocks during the rainy season. 

**Example 1:**

**Input:**

N = 6

arr[] = {3,0,0,2,0,4}

**Output:**

10

**Explanation:**

**Example 2:**

**Input:**

N = 4

arr[] = {7,4,0,9}

**Output:**

10

**Explanation:**

Water trapped by above

block of height 4 is 3 units and above

block of height 0 is 7 units. So, the

total unit of water trapped is 10 units.

**Example 3:**

**Input:**

N = 3

arr[] = {6,9,9}

**Output:**

0

**Explanation:**

No water will be trapped.

### Java Code

//{ Driver Code Starts

import java.io.\*;

import java.util.\*;

import java.lang.\*;

class Array {

public static void main (String[] args) throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int t = Integer.parseInt(br.readLine().trim()); //Inputting the testcases

while(t-->0){

//size of array

int n = Integer.parseInt(br.readLine().trim());

int arr[] = new int[n];

String inputLine[] = br.readLine().trim().split(" ");

//adding elements to the array

for(int i=0; i<n; i++){

arr[i] = Integer.parseInt(inputLine[i]);

}

Solution obj = new Solution();

//calling trappingWater() function

System.out.println(obj.trappingWater(arr, n));

}

}

}

// } Driver Code Ends

class Solution{

// arr: input array

// n: size of array

// Function to find the trapped water between the blocks.

static long trappingWater(int arr[], int n) {

long leftmax=0;

long rightmax=0;

int left=0,right=n-1;

long ans=0;

while(left<right){

if(arr[left]<arr[right]){

if(arr[left]>=leftmax){

leftmax=arr[left];

}

else{

ans+=leftmax-arr[left];

}

left++;

}

else{

if(arr[right]>=rightmax){

rightmax=arr[right];

}

else{

ans+=rightmax-arr[right];

}

right--;

}

}

return ans;

}

}