Experiment -9

Student Name:Harinder Singh. **UID: 22BCS10280**
Branch: BE-CSE **Section/Group: 605-A**
Semester: 5th **Date of Performance:**
Subject Name: Advanced Programming Lab **Subject Code: 22CSP-351**

Ques 1: Minimum Operations to Make the Array Increasing

You are given an integer array nums (**0-indexed**). In one operation, you can choose an element of the array and increment it by 1.

For example, if nums = [1,2,3], you can choose to increment nums[1] to make nums = [1,**3**,3].

Return *the **minimum** number of operations needed to make* nums ***strictly increasing***.

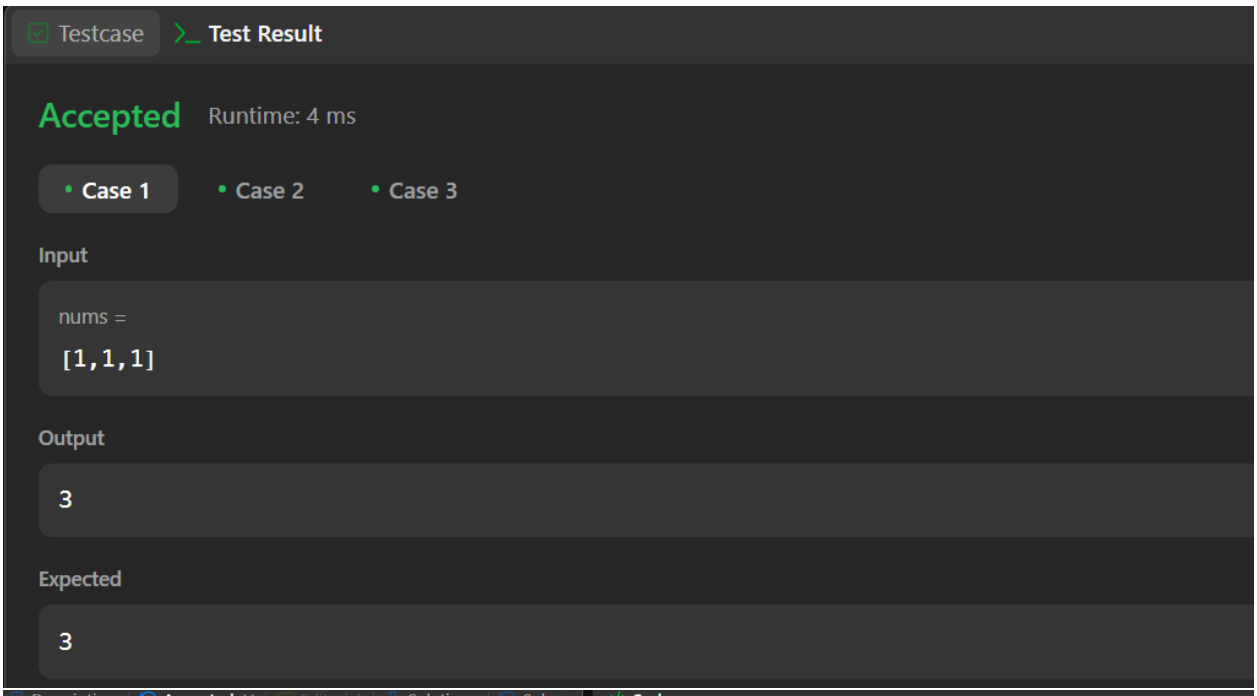
An array nums is **strictly increasing** if nums[i] < nums[i+1] for all 0 <= i < nums.length - 1. An array of length 1 is trivially strictly increasing.

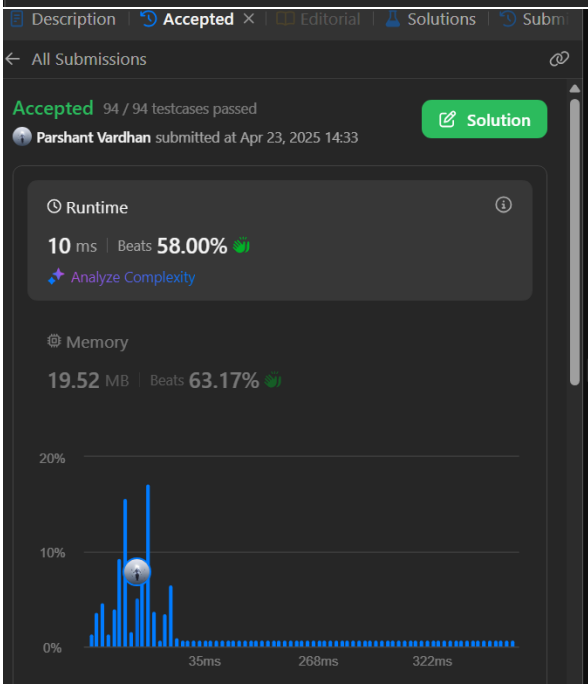
Code:

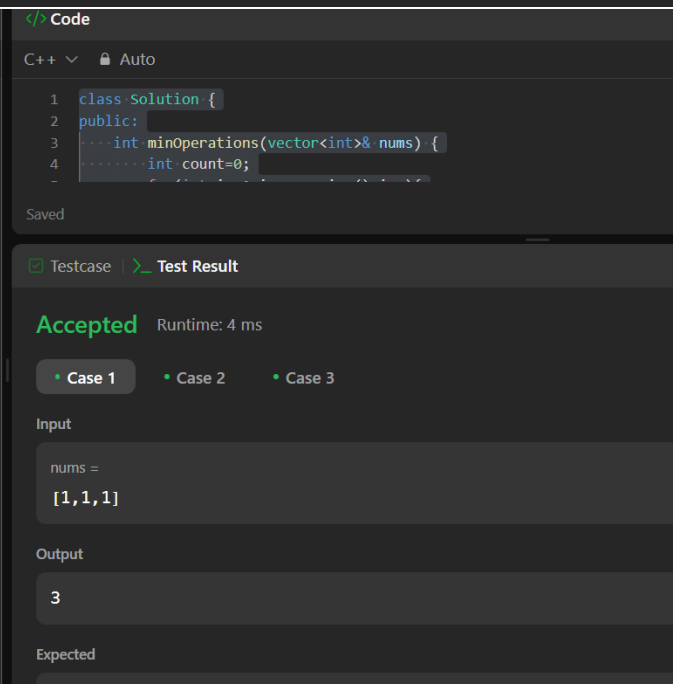
```
class Solution {
public:
    int minOperations(vector<int>& nums) {
        int count=0;
        for(int i =1;i<nums.size();i++){
            if(nums[i]<=nums[i-1]){
                count+=(nums[i-1]-nums[i]+1);
                nums[i]=nums[i-1]+1;

            }
        }
        return count;
    }
};
```

OUTPUT:


Testcase
Test Result
Accepted
Runtime: 4 ms
Case 1
Case 2
Case 3
Input
nums =
[1,1,1]
Output
3
Expected
3


Description
Accepted
Editorial
Solutions
All Submissions
Accepted 94 / 94 testcases passed
Parshant Vardhan submitted at Apr 23, 2025 14:33
Solution
Runtime
10 ms | Beats 58.00%
Analyze Complexity
Memory
19.52 MB | Beats 63.17%
20%
10%
0%
35ms
268ms
322ms


Code
C++
Auto
class Solution {
public:
int minOperations(vector<int>& nums) {
int count=0;
Saved
Testcase
Test Result
Accepted
Runtime: 4 ms
Case 1
Case 2
Case 3
Input
nums =
[1,1,1]
Output
3
Expected

Ques 2 Remove Stones to Minimize the Total:

You are given a **0-indexed** integer array piles, where piles[i] represents the number of stones in the i^{th} pile, and an integer k. You should apply the following operation **exactly** k times:

Choose any piles[i] and **remove** floor(piles[i] / 2) stones from it.

Notice that you can apply the operation on the **same** pile more than once.

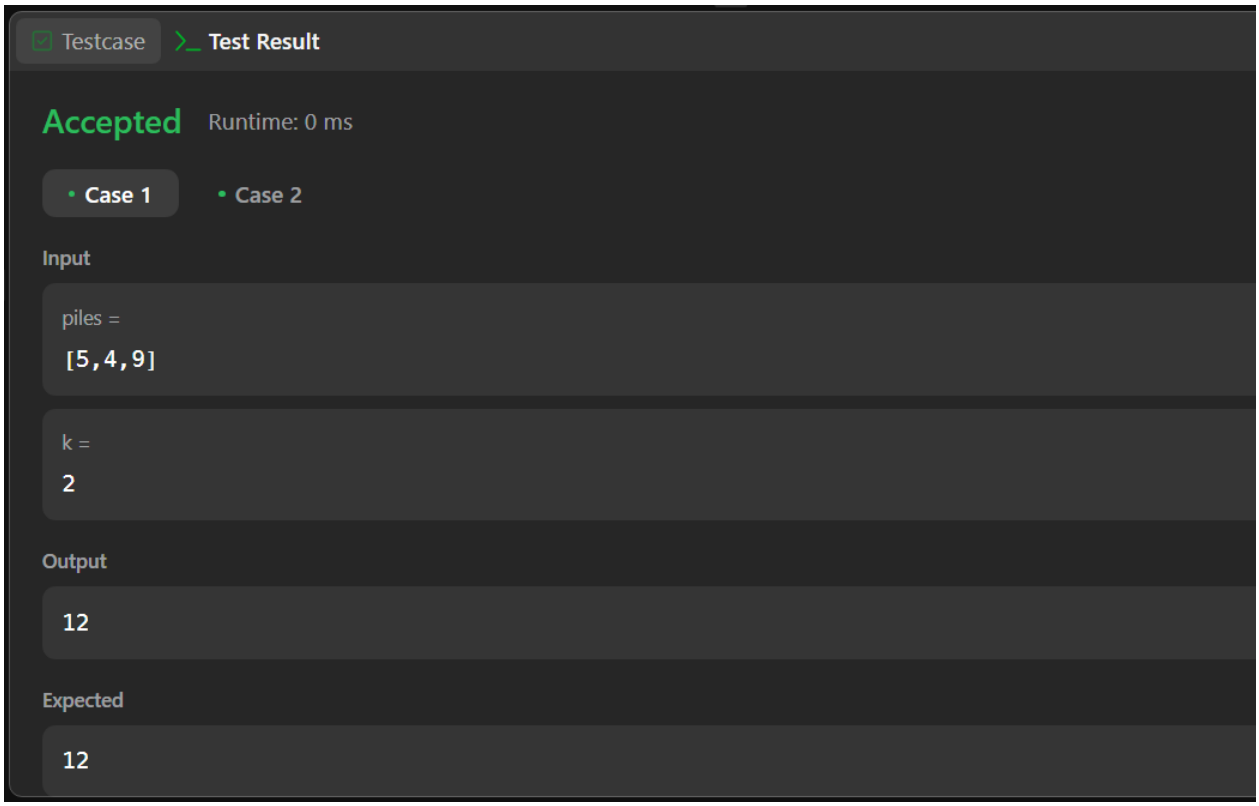
Return *the **minimum** possible total number of stones remaining after applying the* k *operations*.

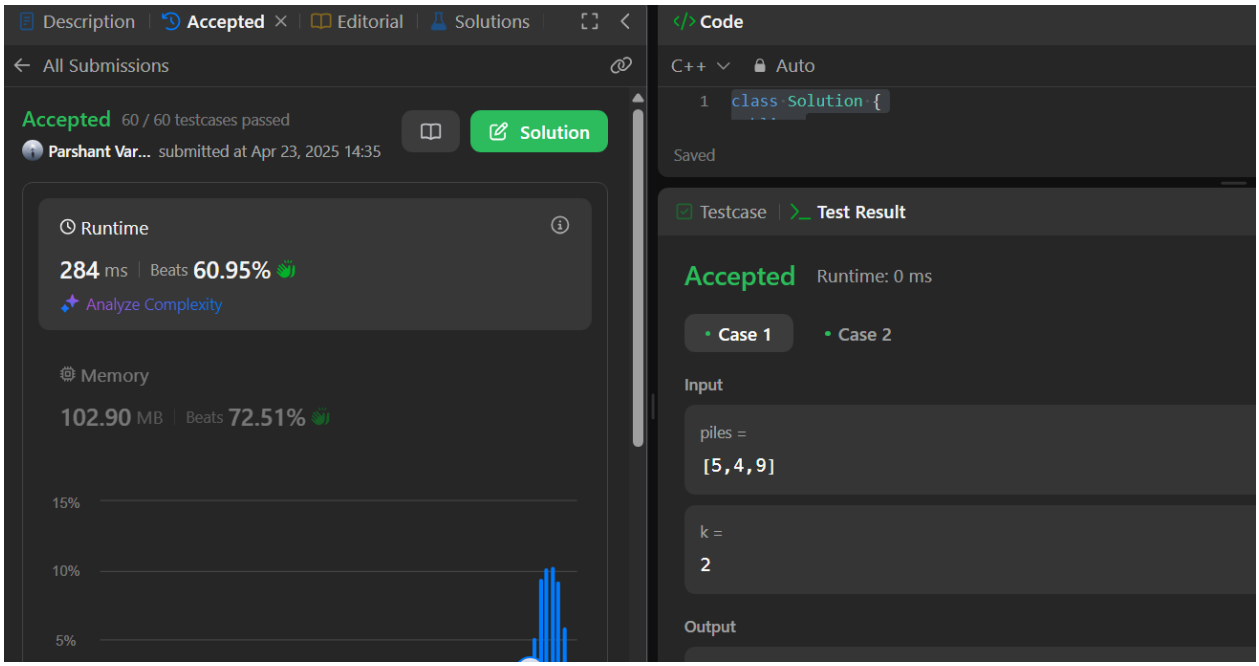
floor(x) is the **greatest** integer that is **smaller** than or **equal** to x (i.e., rounds x down).

CODE:

```
class Solution {
public:
    int minStoneSum(vector<int>& piles, int k) {
        priority_queue<int> maxHeap(piles.begin(), piles.end());

        while (k--) {
            int largestPile = maxHeap.top();
            maxHeap.pop();
            largestPile -= floor(largestPile / 2);
            maxHeap.push(largestPile);
        }
        int res=0;
        while(!maxHeap.empty()){
            res += maxHeap.top();
            maxHeap.pop();
        }
        return res;
    }
};
```


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
piles =
[5,4,9]
k =
2
Output
12
Expected
12


Description
Accepted
Editorial
Solutions
All Submissions
Accepted 60 / 60 testcases passed
Parshant Var... submitted at Apr 23, 2025 14:35
Solution
Runtime
284 ms | Beats 60.95%
Analyze Complexity
Memory
102.90 MB | Beats 72.51%
15%
10%
5%
Code
C++
Auto
1 class Solution {
Saved
Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
piles =
[5,4,9]
k =
2
Output

Ques3: Number of Islands

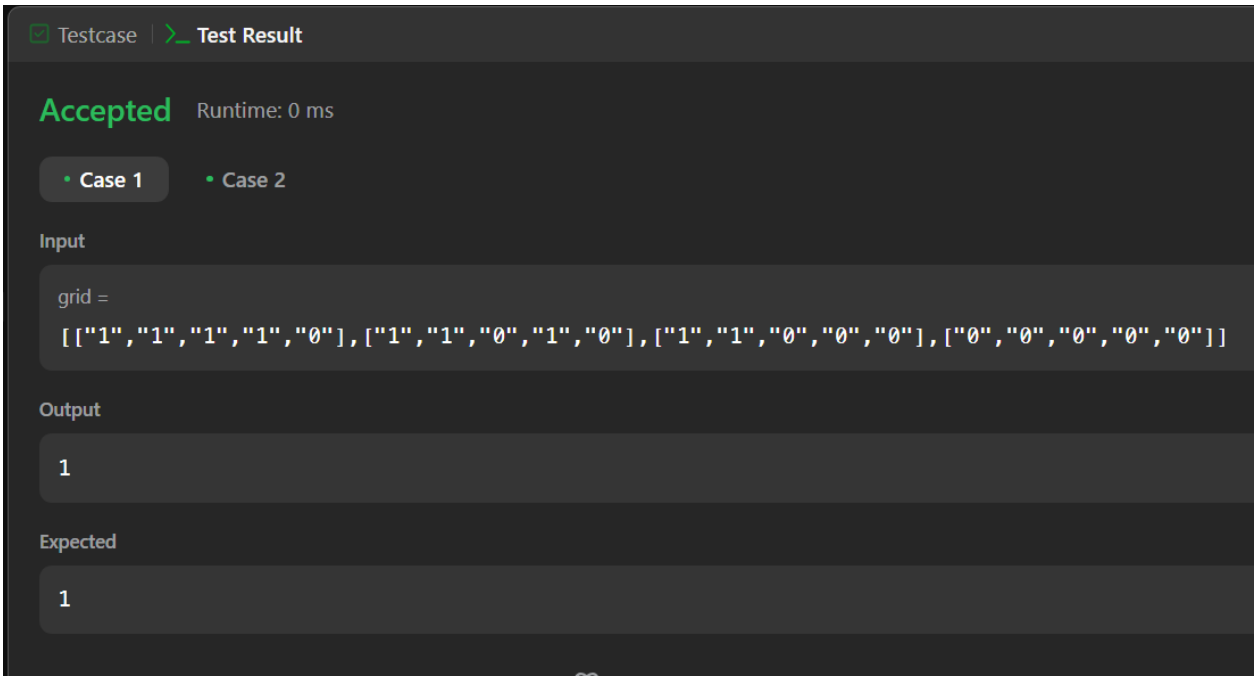
Given an 2D binary grid which represents a map of s (land) and s (water), return .

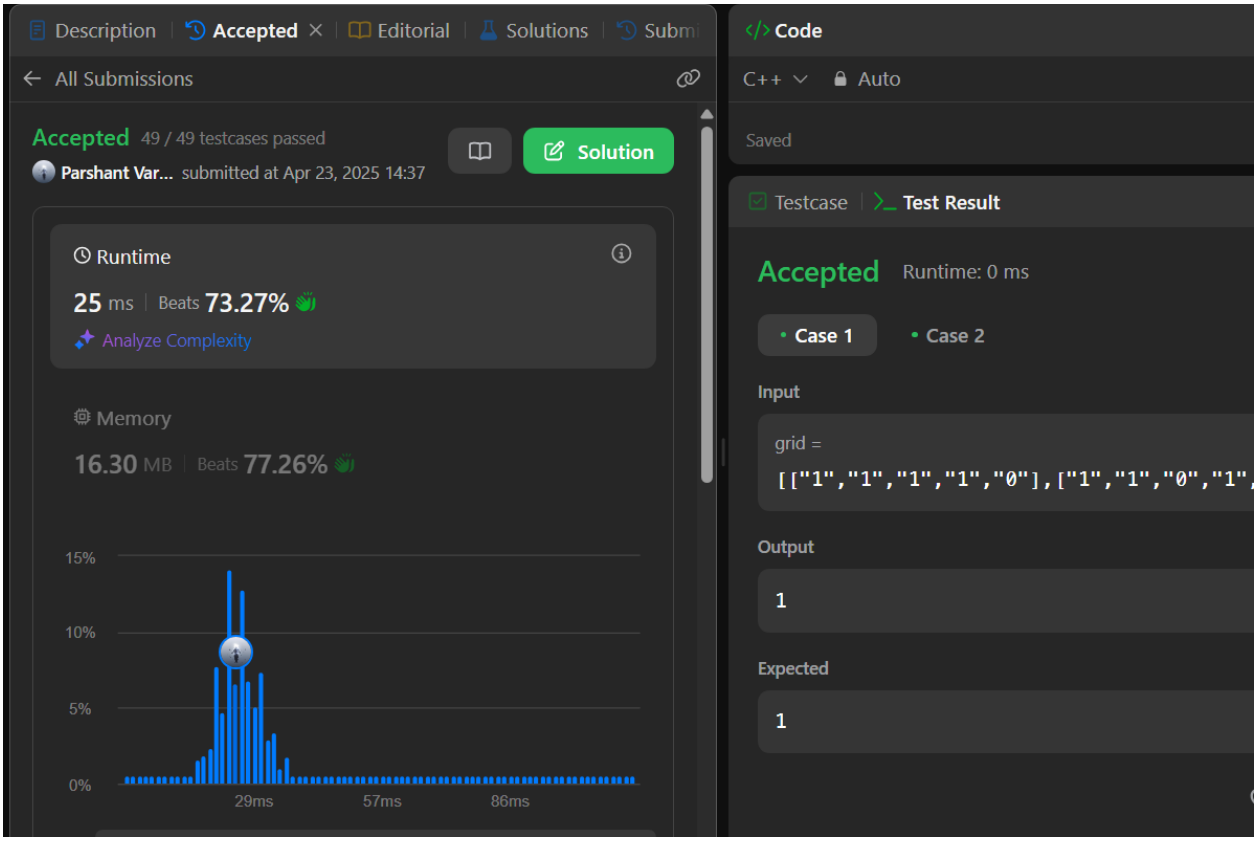
An is surrounded by water and is formed by connecting adjacent lands horizontally or vertically. You may assume all four edges of the grid are all surrounded by water.

CODE:

```
class Solution {
public:
    void dfs(vector<vector<char>>& grid, int i , int j){
        if(i < 0 || i >= grid.size() ||
            j < 0 || j >= grid[0].size() ||grid[i][j]=='0') return;
        grid[i][j]='0';
        dfs(grid,i-1,j);
        dfs(grid,i+1,j);
        dfs(grid,i,j-1);
        dfs(grid,i,j+1);
    }
    int numIslands(vector<vector<char>>& grid) {
        int count=0;
        for(int i=0;i<grid.size();i++){
            for(int j =0;j<grid[0].size();j++){
                if(grid[i][j]=='1'){
                    dfs(grid, i, j);
                    count++;
                }
            }
        }
        return count;
    }

};
```


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
grid =
[["1","1","1","1","0"],["1","1","0","1","0"],["1","1","0","0","0"],["0","0","0","0","0"]]
Output
1
Expected
1


Description
Accepted
Editorial
Solutions
All Submissions
Accepted 49 / 49 testcases passed
Parshant Var... submitted at Apr 23, 2025 14:37
Solution
Runtime
25 ms | Beats 73.27%
Analyze Complexity
Memory
16.30 MB | Beats 77.26%
15%
10%
5%
0%
29ms
57ms
86ms
Code
C++
Auto
Saved
Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
grid =
Output
1
Expected
1