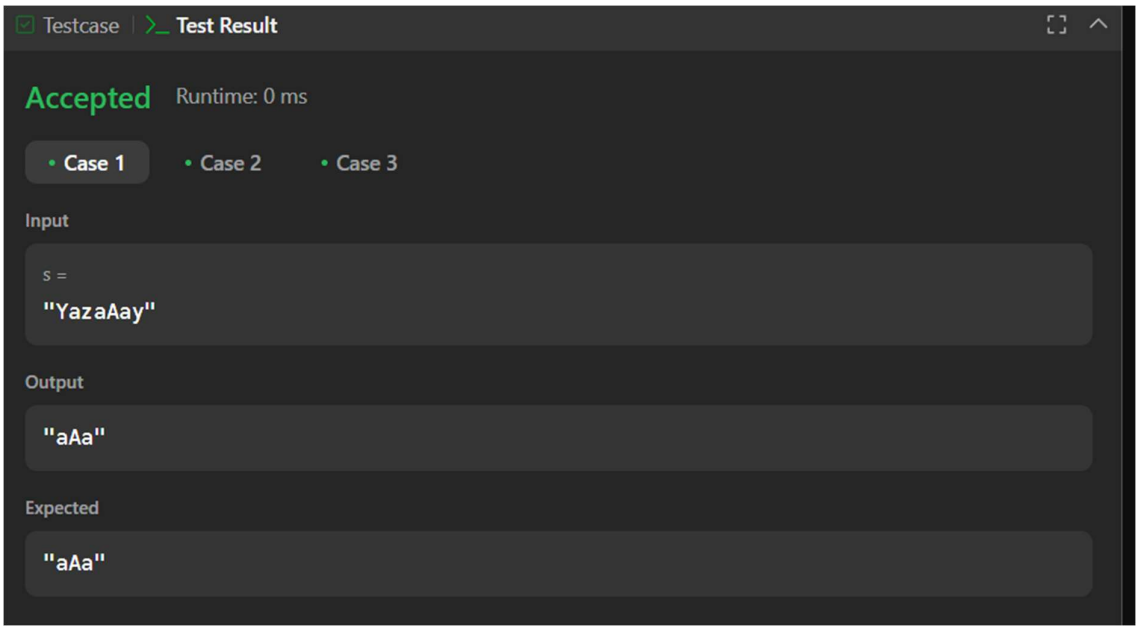
Ques no. 1 :- Longest Nice Substring(1763).

Solution:

```
class Solution {
public:
    string longestNiceSubstring(string s) {
        string output = "";
        int count = 0;
        for(int i = 0;i<s.length();i++){
            int smallMask=0;
            int largeMask = 0;
            char ch = s[i];
            int chint = 0;
            if(ch>=65 && ch<=90){
                chint = ch-'A';
                largeMask = 1<<chint;
            }
            else{
                chint = ch-'a';
                smallMask = 1<<chint;
            }
            for(int j = i+1;j<s.length();j++){
                ch = s[j];
                if(ch>=65 && ch<=90){
                    chint = ch-'A';
                    largeMask |= 1<<chint;
                }
                else{
                    chint = ch-'a';
```

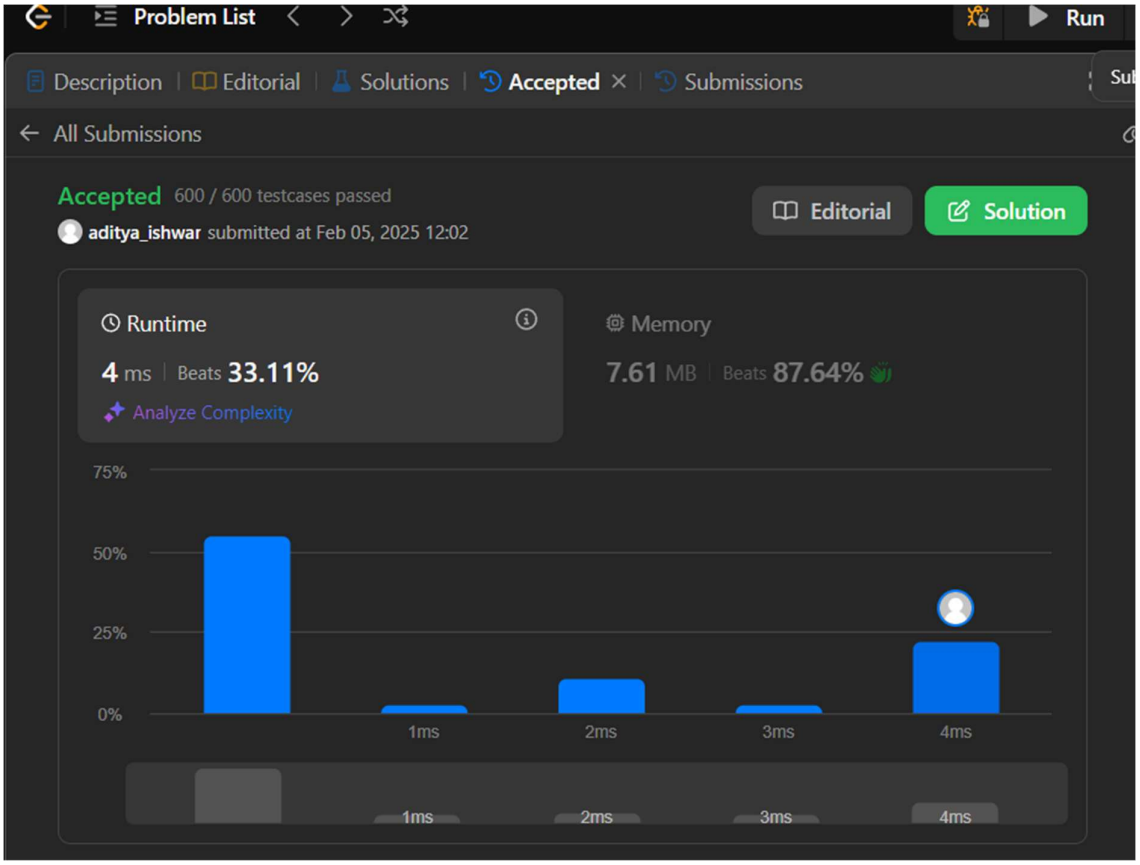
```
                smallMask |= 1<<chint;
            }
            //checking for nice
            if((smallMask^largeMask) == 0){
                if(count<j-i+1){
                    count = j-i+1;
                    string temp(s.begin()+i,s.begin()+j+1);
                    output = temp;
                }
            }
        }
    }
    return output;
  }
};
```



Ques no. 2 : Reverse Bits (190).

Solution :

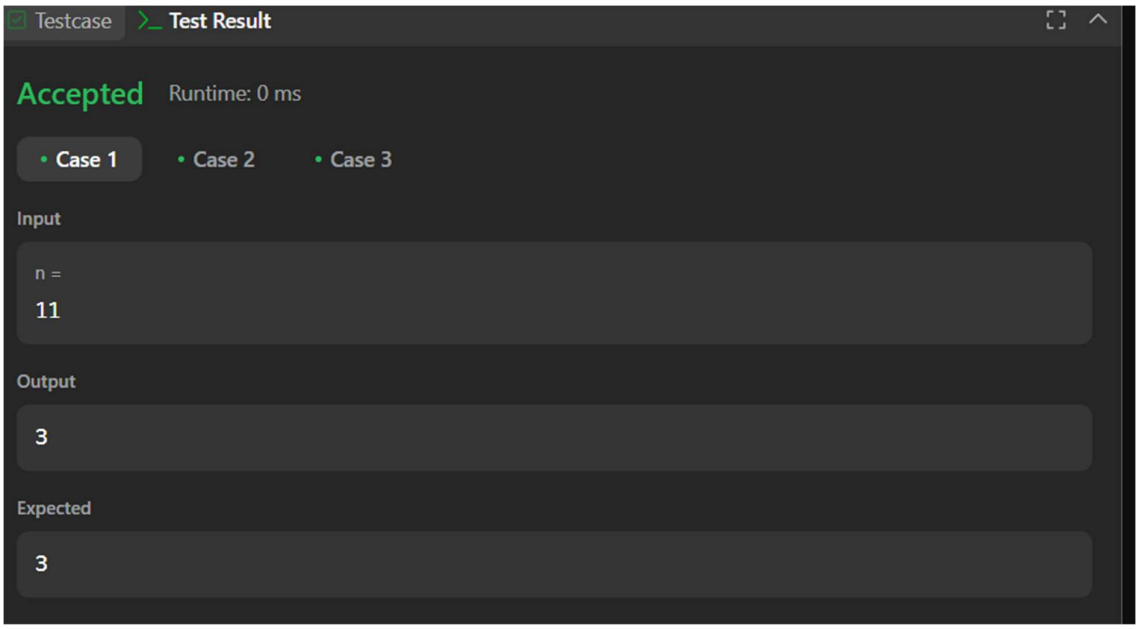
```
class Solution {
public:
    uint32_t reverseBits(uint32_t n) {
        uint32_t result = 0;
        for (int i = 0; i < 32; i++) {
            int bit = n & 1;       // Extract the least significant bit
            result = (result << 1) | bit; // Append the bit to the result
            n = n >> 1;          // Right-shift n to process the next bit
        }
        return result;
    }
};
```


Problem List
Run
Description
Editorial
Solutions
Accepted
Submissions
All Submissions
Accepted 600 / 600 testcases passed
aditya_ishwar submitted at Feb 05, 2025 12:02
Editorial
Solution
Runtime
4 ms | Beats 33.11%
Analyze Complexity
Memory
7.61 MB | Beats 87.64%
75%
50%
25%
0%
1ms
2ms
3ms
4ms
1ms
2ms
3ms
4ms

Ques no. 3 : Number of 1 Bits (191).

Solution :

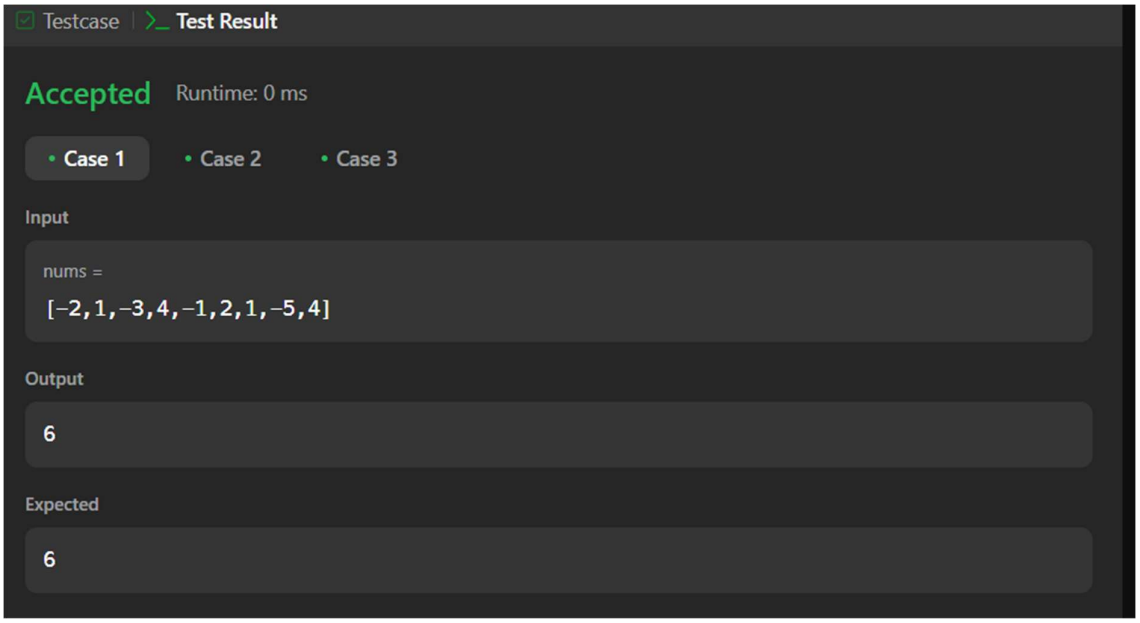
```
class Solution {
public:
    int hammingWeight(uint32_t n) {
        int res = 0;
        for (int i = 0; i < 32; i++) {
            if ((n >> i) & 1) {
                res += 1;
            }
        }
        return res;
    }
};
```



Ques no. 4: Maximum Subarray (53).

Solution:

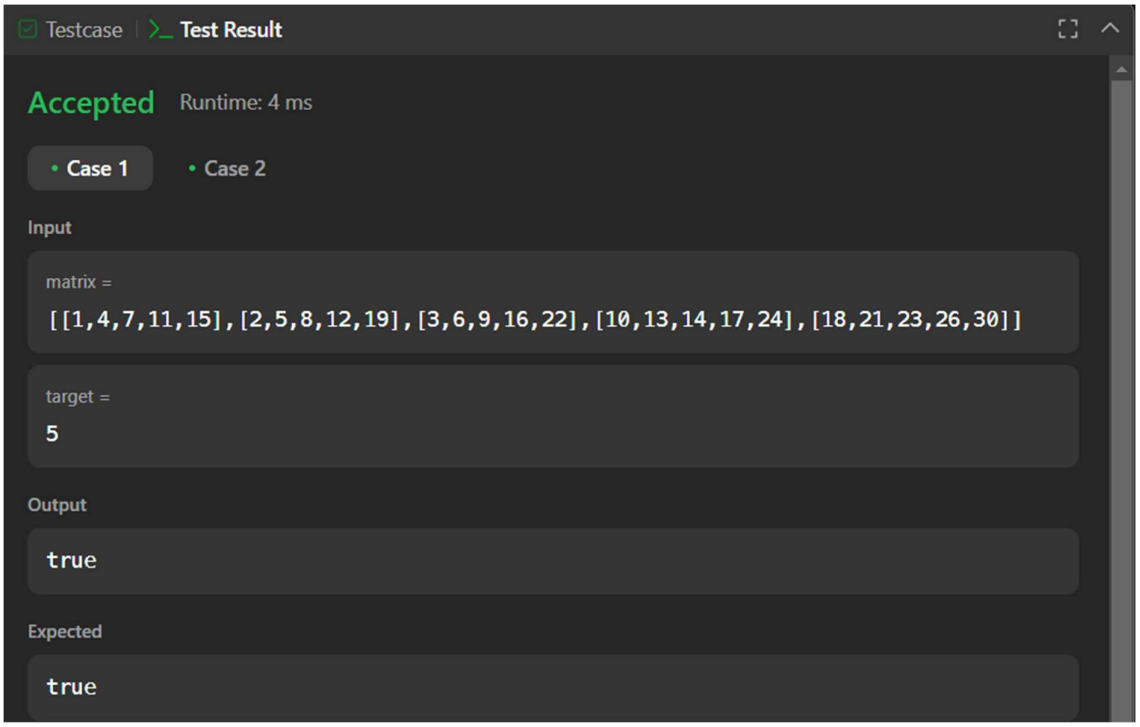
```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int res = nums[0];
        int total = 0;

        for (int n : nums) {
            if (total < 0) {
                total = 0;
            }

            total += n;
            res = max(res, total);
        }

        return res;
    }
};
```


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
nums =
[-2,1,-3,4,-1,2,1,-5,4]
Output
6
Expected
6

Ques no. 5 : Search a 2D Matrix II (240).

Solution:

```
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {
        int n = matrix.size(), m = matrix[0].size();
        int row = 0, col = m - 1;

        while (row < n && col >= 0) {
            if (matrix[row][col] == target) return true;
            else if (matrix[row][col] < target) row++;
            else col--;
        }
        return false;
    }
};
```


Testcase
Test Result
Accepted
Runtime: 4 ms
Case 1
Case 2
Input
matrix =
[[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]]
target =
5
Output
true
Expected
true

Ques no. 6: Super Pow (372).

Solution :

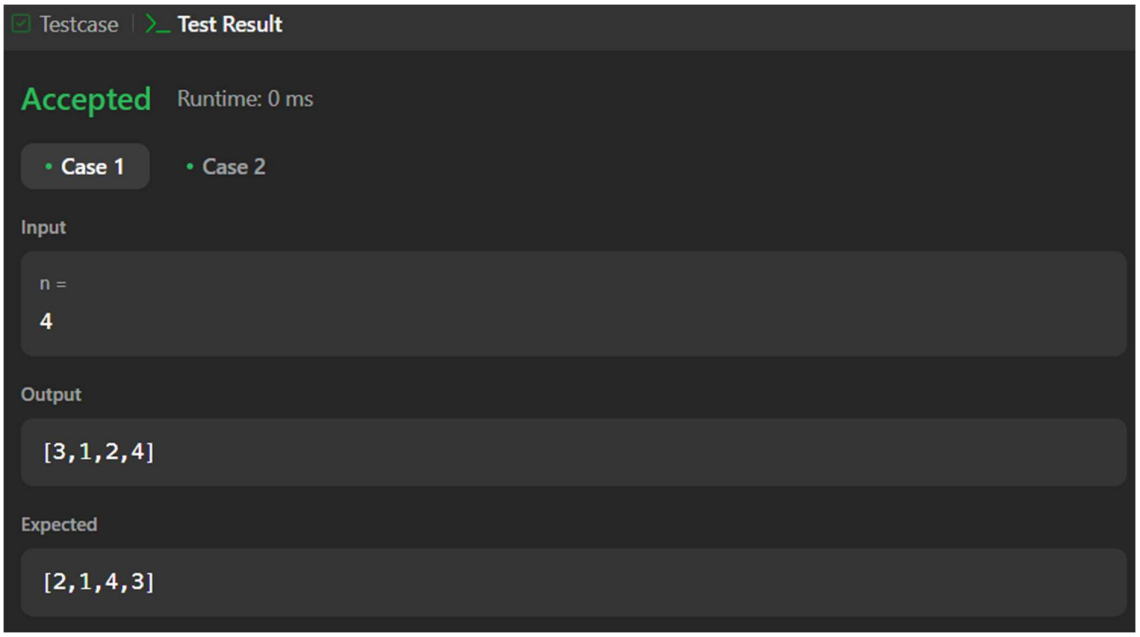
```
class Solution {
    const int base = 1337;
    int powmod(int a, int k) //a^k mod 1337 where 0 <= k <= 10
    {
        a %= base;
        int result = 1;
        for (int i = 0; i < k; ++i)
            result = (result * a) % base;
        return result;
    }
public:
    int superPow(int a, vector<int>& b) {
        if (b.empty()) return 1;
        int last_digit = b.back();
        b.pop_back();
        return powmod(superPow(a, b), 10) * powmod(a, last_digit) % base;
    }
};
```


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Case 3
Input
a =
2
b =
[3]
Output
8
Expected
8

Ques no. 7 : Beautiful Array(932).

Solution :

```
class Solution {
public:
  static bool comp(const int &a, const int &b){
    int mask = 1;
    while(true)
      if((a&mask) == (b&mask)) mask = mask<<1;
      else return (a&mask) > (b&mask);
  }

  vector<int> beautifulArray(int n) {
    vector<int> answer;
    while(n) answer.push_back(n--);

    sort(answer.begin(), answer.end(), comp);

    return answer;
  }
};
```

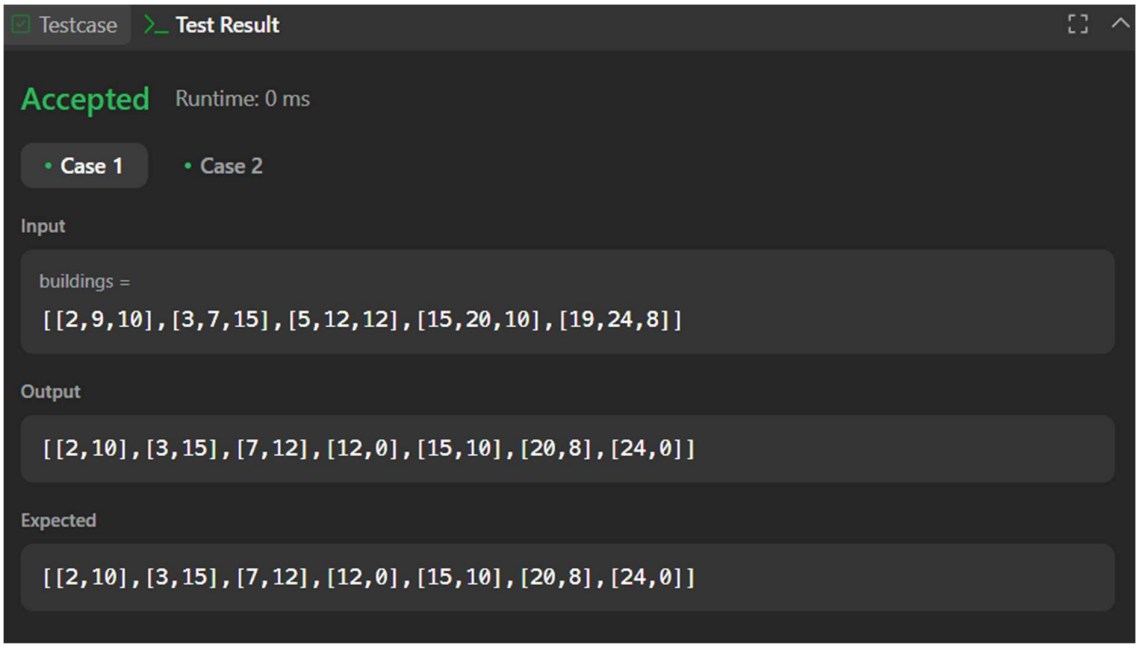

Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
n =
4
Output
[3,1,2,4]
Expected
[2,1,4,3]

Ques no. 8 : The Skyline Problem(218).

Solution :

```
class Solution {
public:
    vector<vector<int>> getSkyline(vector<vector<int>>& buildings) {
        int edge_idx = 0;
        vector<pair<int, int>> edges;
        priority_queue<pair<int, int>> pq;
        vector<vector<int>> skyline;

        for (int i = 0; i < buildings.size(); ++i) {
            const auto &b = buildings[i];
            edges.emplace_back(b[0], i);
            edges.emplace_back(b[1], i);
        }

        std::sort(edges.begin(), edges.end());

        while (edge_idx < edges.size()) {
            int curr_height;
            const auto &[curr_x, _] = edges[edge_idx];
            while (edge_idx < edges.size() &&
                   curr_x == edges[edge_idx].first) {
                const auto &[_, building_idx] = edges[edge_idx];
                const auto &b = buildings[building_idx];
                if (b[0] == curr_x)
                    pq.emplace(b[2], b[1]);
                ++edge_idx;
```

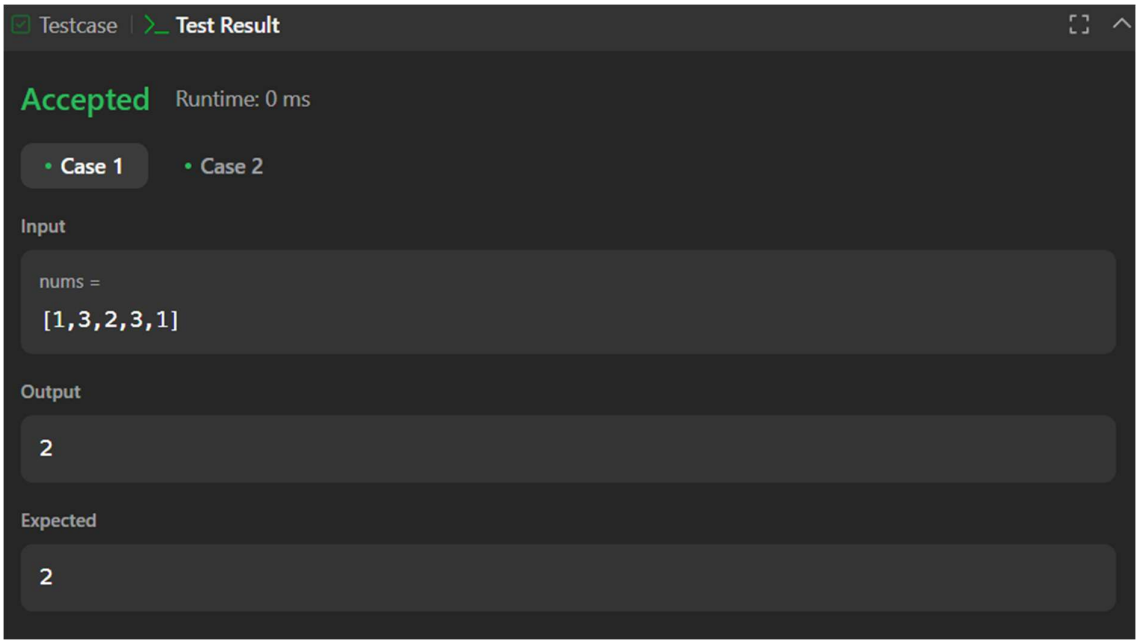
```
        }
        while (!pq.empty() && pq.top().second <= curr_x)
            pq.pop();
        curr_height = pq.empty() ? 0 : pq.top().first;
        if (skyline.empty() || skyline.back()[1] != curr_height)
            skyline.push_back({curr_x, curr_height});
      }
      return skyline;
    }
};
```



Ques no. 9: Reverse Pairs(493).

Solution :

```
class Solution {
public:
    int reversePairs(vector<int>& nums) {
        int n = nums.size();
        long long reversePairsCount = 0;
        for(int i=0; i<n-1; i++){
            for(int j=i+1; j<n; j++){
                if(nums[i] > 2*(long long)nums[j]){
                    reversePairsCount++;
                }
            }
        }
        return reversePairsCount;
    }
};
```

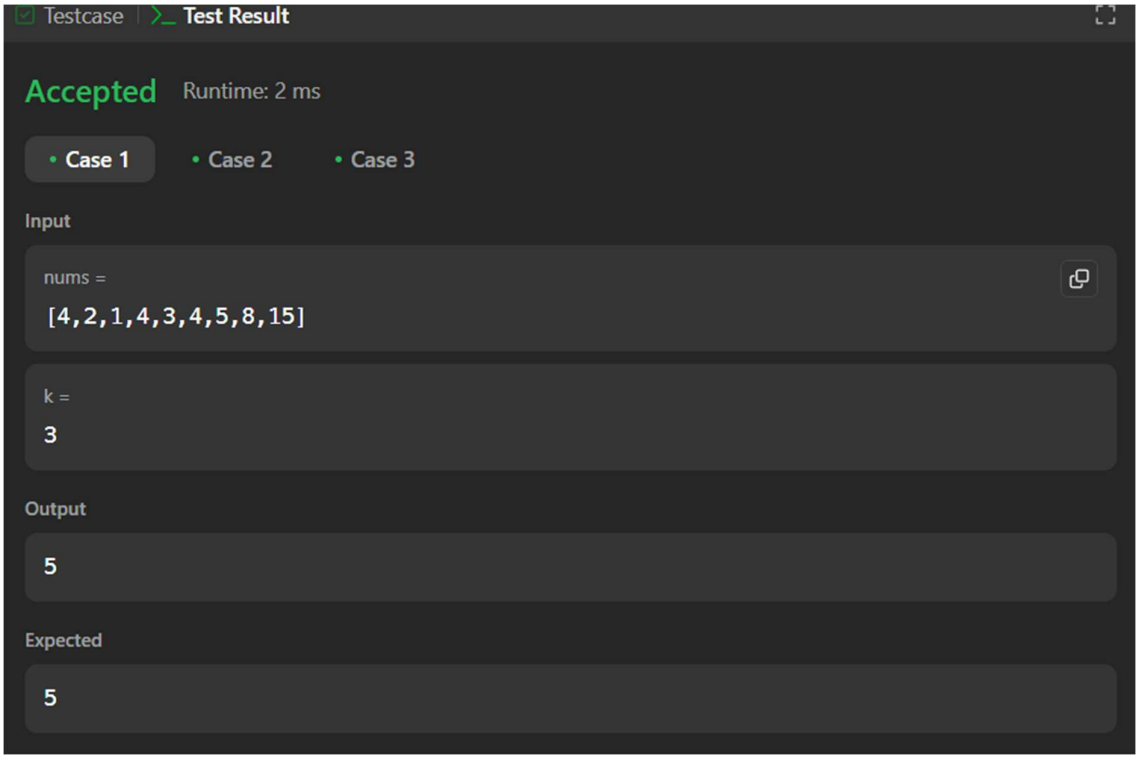

Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
nums =
[1,3,2,3,1]
Output
2
Expected
2

Ques no. 10: Longest Increasing Subsequence II (2407).

Solution :

```
class MaxSegmentTree {
 public:
  int n;
  vector<int> tree;
  MaxSegmentTree(int n_) : n(n_) {
    int size = (int)(ceil(log2(n)));
    size = (2 * pow(2, size)) - 1;
    tree = vector<int>(size);
  }

  int max_value() { return tree[0]; }

  int query(int l, int r) { return query_util(0, l, r, 0, n - 1); }

  int query_util(int i, int qL, int qR, int l, int r) {
    if (l >= qL && r <= qR) return tree[i];
    if (l > qR || r < qL) return INT_MIN;

    int m = (l + r) / 2;
    return max(query_util(2 * i + 1, qL, qR, l, m), query_util(2 * i + 2, qL, qR, m + 1, r));
  }

  void update(int i, int val) { update_util(0, 0, n - 1, i, val); }
  void update_util(int i, int l, int r, int pos, int val) {
    if (pos < l || pos > r) return;
    if (l == r) {
```

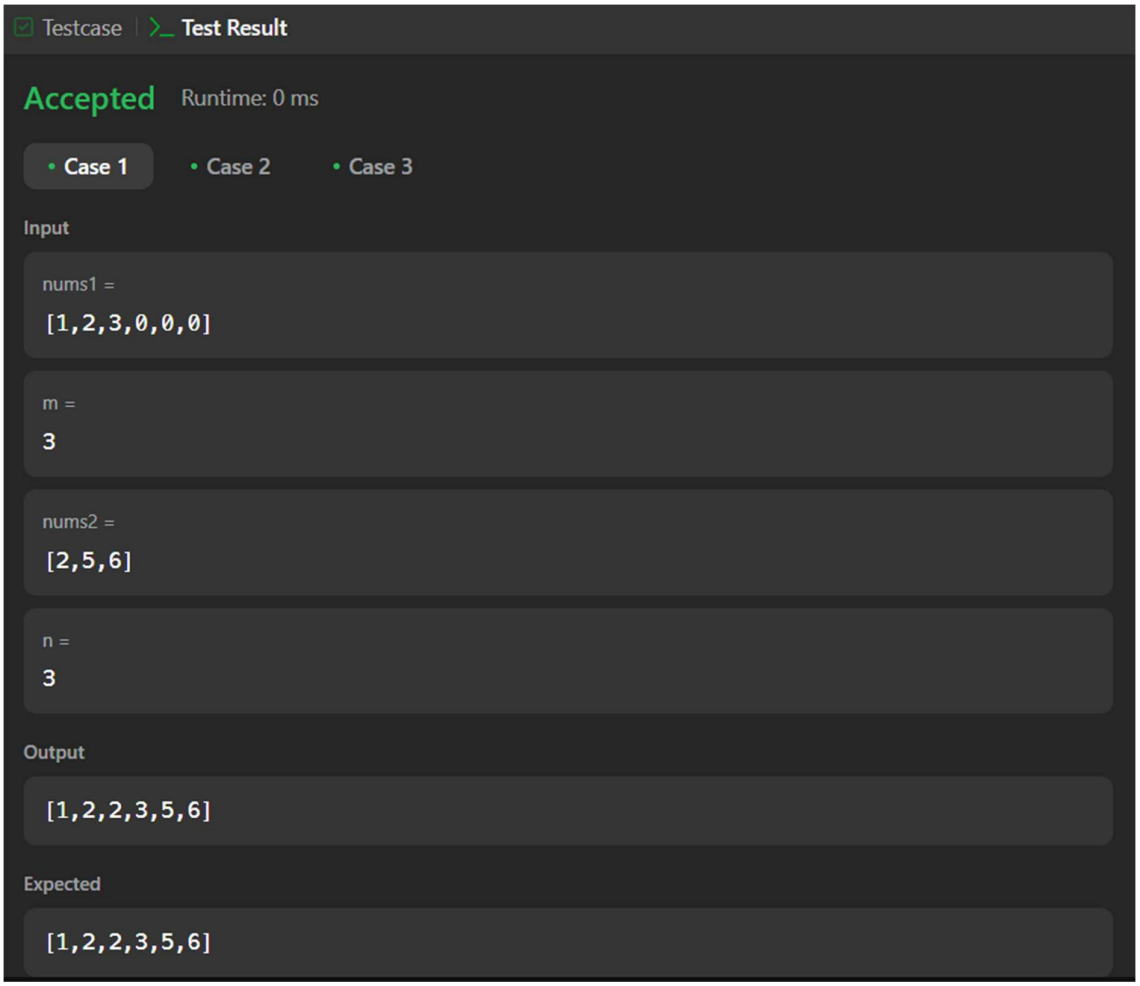
```
      tree[i] = max(val, tree[i]);
      return;
    }

    int m = (l + r) / 2;
    update_util(2 * i + 1, l, m, pos, val);
    update_util(2 * i + 2, m + 1, r, pos, val);
    tree[i] = max(tree[2 * i + 1], tree[2 * i + 2]);
  }
};

class Solution {
 public:
  int lengthOfLIS(vector<int>& nums, int k) {
    MaxSegmentTree tree(1e5 + 1);
    for (int i : nums) {
      int lower = max(0, i - k);
      int cur = 1 + tree.query(lower, i - 1);
      tree.update(i, cur);
    }

    return tree.max_value();
  }
};
```


Testcase
Test Result
Accepted
Runtime: 2 ms
Case 1
Case 2
Case 3
Input
nums =
[4,2,1,4,3,4,5,8,15]
k =
3
Output
5
Expected
5

Ques no. 11: Merge Sorted Array (88).

Solution :

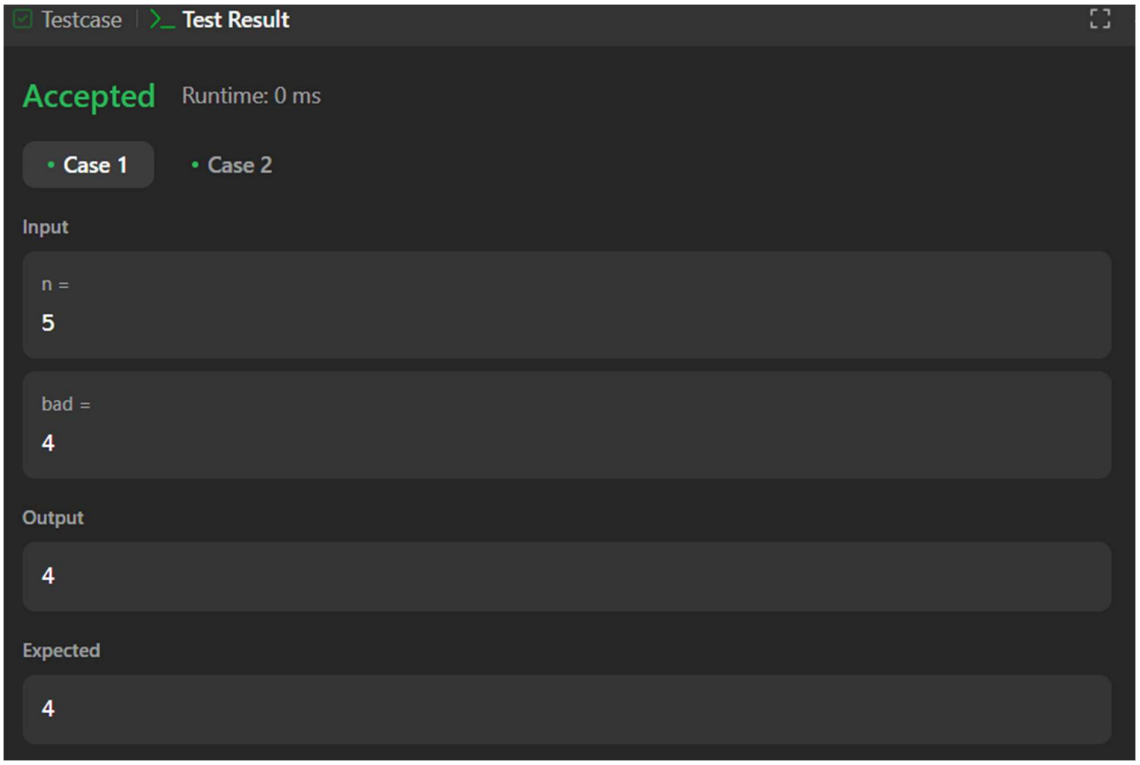
```
class Solution {
public:
    void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {
        for (int j = 0, i = m; j<n; j++){
            nums1[i] = nums2[j];
            i++;
        }
        sort(nums1.begin(),nums1.end());
    }
};
```



Ques no. 12: First Bad Version (278).

Solution :

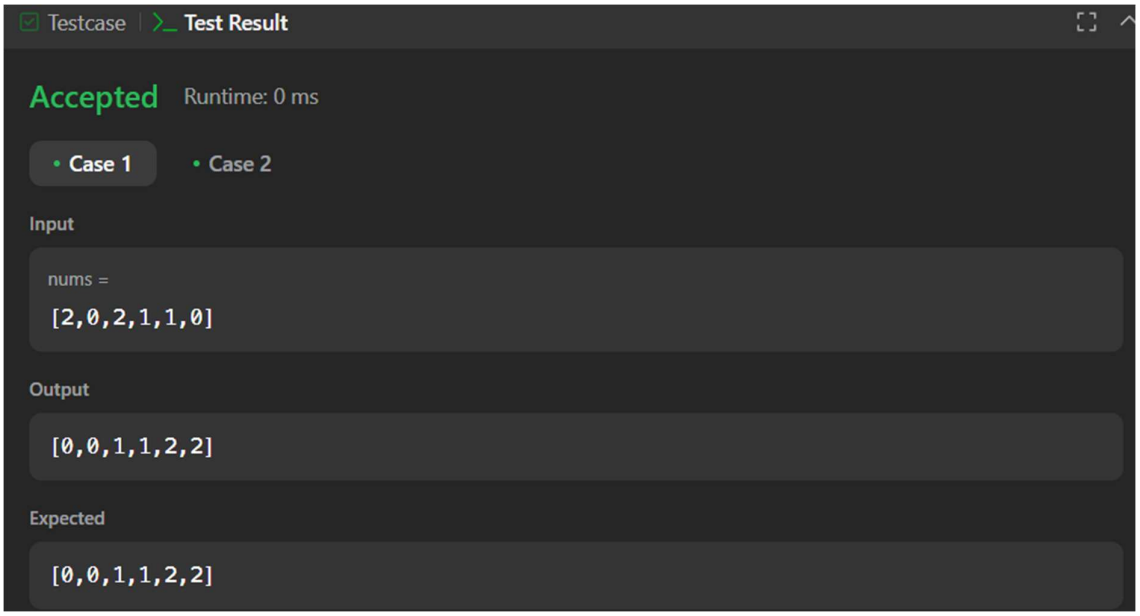
```
/*
By   :: savetrees
Used :: Binary Search
*/
class Solution {
public:
   int firstBadVersion(int n) {
       int low=1;
       int high=n;
       while(low<=high)
       {
          int mid=low+(high-low)/2;
          if(isBadVersion(mid))high=mid-1;
          else low=mid+1;
       }
       return low;
   }
};
```


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
n =
5
bad =
4
Output
4
Expected
4

Ques no. 13 : Sort Colors (75).

Solution :

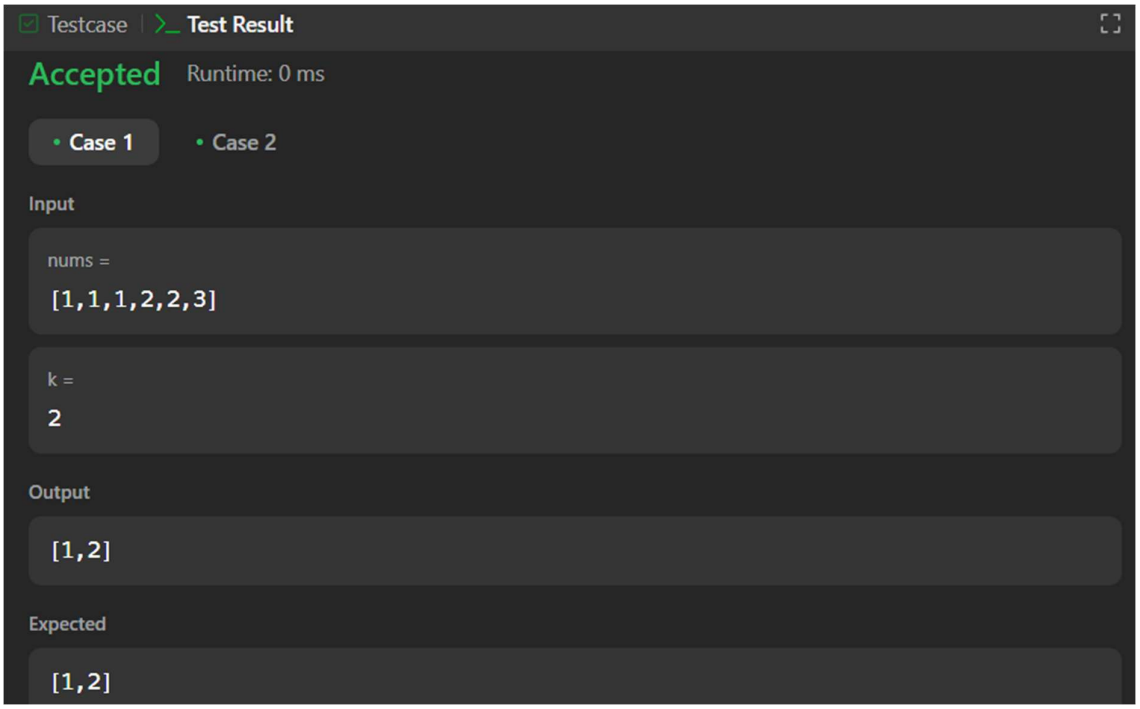
```
class Solution {
public:
    void sortColors(vector<int>& nums) {
        int n = nums.size();
       int low =0, mid =0, high = n-1;
       while(mid <= high){
        if(nums[mid]==0){
            swap(nums[low] , nums[mid]);
            low++;
            mid++;
        }
        else if(nums[mid] ==1){
            mid++;
        }else{
            swap(nums[mid] , nums[high]);
            high--;
        }
       }
    }
};
```


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

Ques no. 14 : Top K Frequent Elements (347).

Solution :

```
class Solution {
public:
    vector<int> topKFrequent(vector<int>& nums, int k) {
        unordered_map<int, int> counter;
        for (int n : nums) {
            counter[n]++;
        }

        auto comp = [](pair<int, int>& a, pair<int, int>& b) {
            return a.second < b.second;
        };
        priority_queue<pair<int, int>, vector<pair<int, int>>, decltype(comp)> heap(comp);

        for (auto& entry : counter) {
            heap.push({entry.first, entry.second});
        }

        vector<int> res;
        while (k-- > 0) {
            res.push_back(heap.top().first);
            heap.pop();
        }

        return res;
    }
};
```


Testcase
Test Result
Accepted
Runtime: 0 ms
Case 1
Case 2
Input
nums =
[1,1,1,2,2,3]
k =
2
Output
[1,2]
Expected
[1,2]

Ques no. 15 : Kth Largest Element in an Array(215).

Solution :

```
class Solution {
public:
    int findKthLargest(std::vector<int>& nums, int k) {
        std::sort(nums.begin(), nums.end(), std::greater<int>());
        return nums[k-1];
    }
};
```