Assignment-2

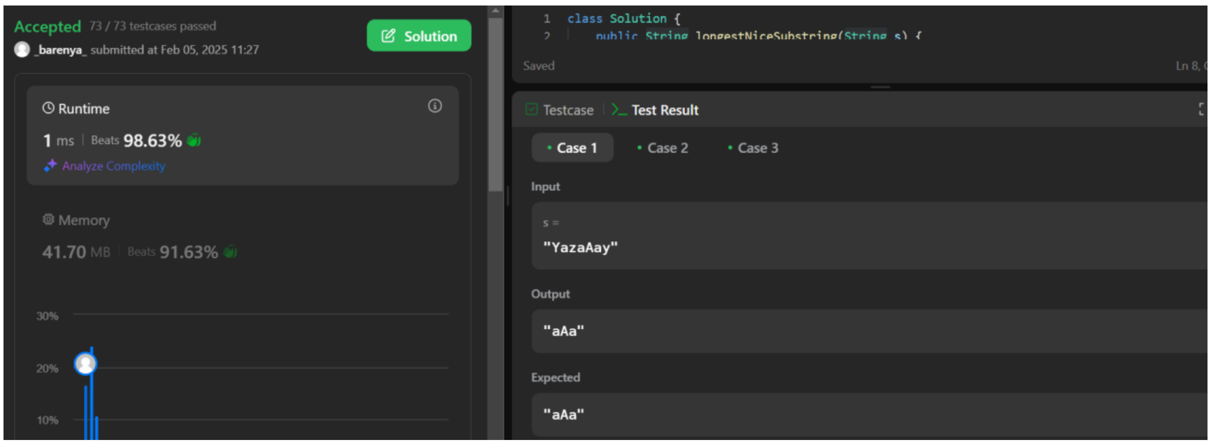
Barenya Behera

22BCS_FL_602/A

UID- 22BCS15121

1. Longest Nice Substring

```
class Solution {
public String longestNiceSubstring(String s) {
if (s.length() < 2) return "";
for (int i = 0; i < s.length(); i++) {
char c = s.charAt(i);
if (s.indexOf(Character.toLowerCase(c)) == -1 || s.indexOf(Character.toUpperCase(c)) == -1) {
String left = longestNiceSubstring(s.substring(0, i));
String right = longestNiceSubstring(s.substring(i + 1));
return left.length() >= right.length() ? left : right;
}
}
return s;
}
}
```


Accepted 73 / 73 testcases passed
barenya submitted at Feb 05, 2025 11:27

Runtime
1 ms | Beats 98.63%

Memory
41.70 MB | Beats 91.63%

```
class Solution {
    public String longestNiceSubstring(String s) {
```

Testcase | Test Result
Case 1 · Case 2 · Case 3

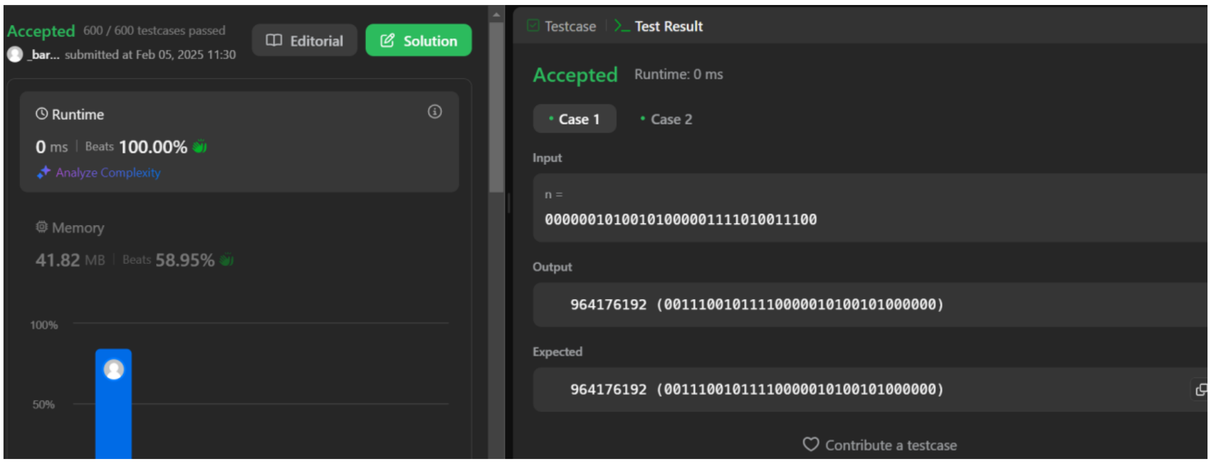
Input
s =
"YazaAay"

Output
"aAa"

Expected
"aAa"

2. Reverse Bits

```
 public class Solution {
public int reverseBits(int n) {
int result = 0;
for (int i = 0; i < 32; i++) {
result <<= 1;
result |= (n & 1);
n >>>= 1;
}
return result;
}
}
```


Accepted 600 / 600 testcases passed
_bar... submitted at Feb 05, 2025 11:30

Runtime
0 ms | Beats 100.00%

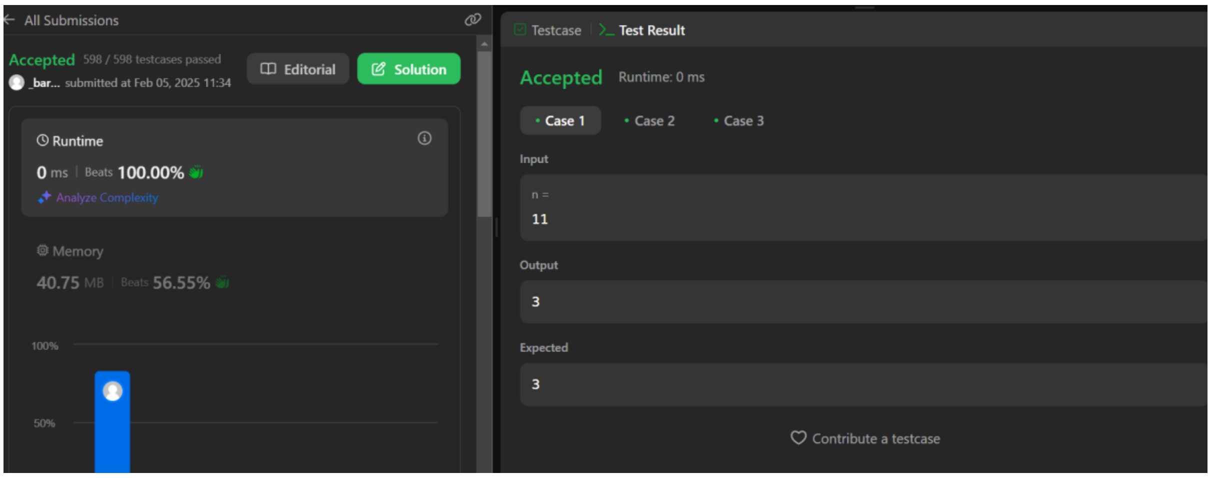
Memory
41.82 MB | Beats 58.95%

Accepted Runtime: 0 ms

Input
```
n =
00000010100101000001111010011100
```
Output
```
964176192 (00111001011110000010100101000000)
```
Expected
```
964176192 (00111001011110000010100101000000)
```

3. Number of 1 Bits

```
 class Solution {
public int hammingWeight(int n) {
int count = 0;
while (n != 0) {
count += (n & 1);
n >>>= 1;
}
return count;
}
}
```


Accepted 598 / 598 testcases passed
_bar... submitted at Feb 05, 2025 11:34

Runtime
0 ms | Beats 100.00%

Memory
40.75 MB | Beats 56.55%

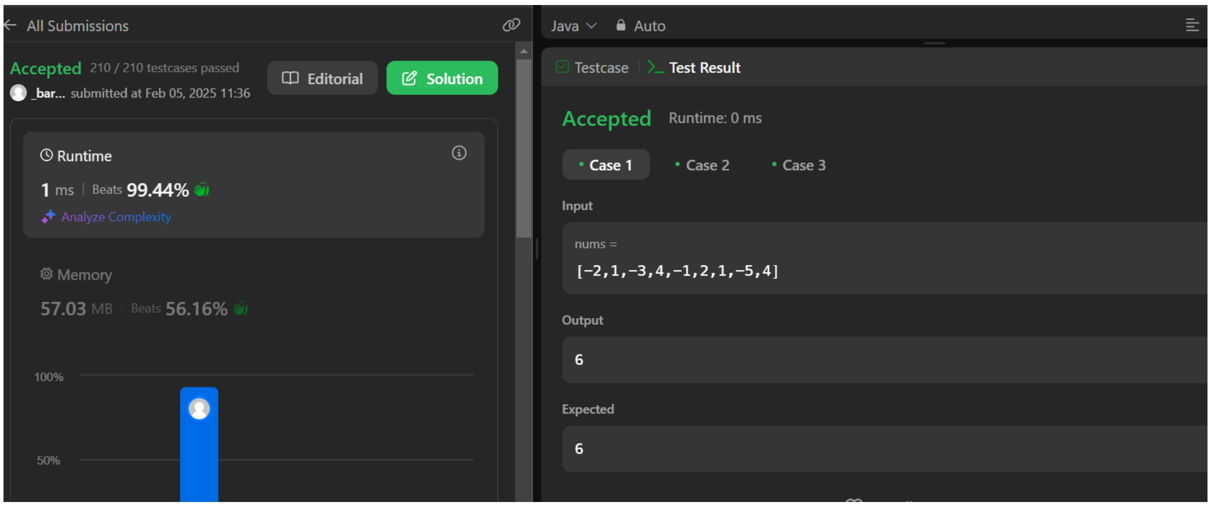
Accepted Runtime: 0 ms

Input
```
n =
11
```
Output
```
3
```
Expected
```
3
```

4. Maximum Subarray

```
 class Solution {
public int maxSubArray(int[] nums) {
int maxSoFar = nums[0];
int currentMax = nums[0];
for (int i = 1; i < nums.length; i++) {
```

```
currentMax = Math.max(nums[i], currentMax + nums[i]);
maxSoFar = Math.max(maxSoFar, currentMax);
}
return maxSoFar;
}
}
```



All Submissions

Accepted 210 / 210 testcases passed

_bar... submitted at Feb 05, 2025 11:36

Runtime

1 ms | Beats 99.44%

Memory

57.03 MB | Beats 56.16%

Java Auto

Testcase | Test Result

Accepted Runtime: 0 ms

Input

```
nums =
[-2,1,-3,4,-1,2,1,-5,4]
```

Output

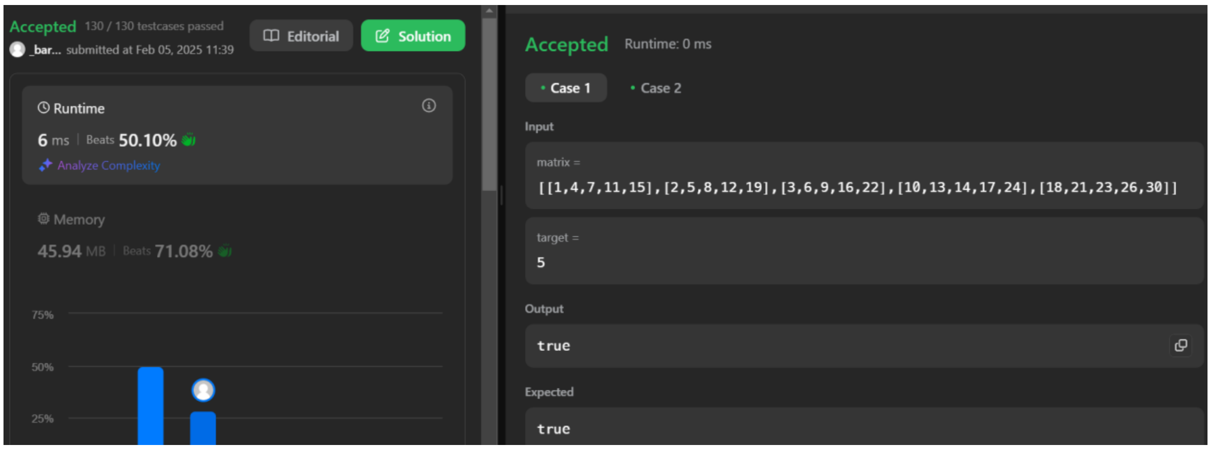
```
6
```

Expected

```
6
```

5. Search a 2D Matrix II

```
 class Solution {
public boolean searchMatrix(int[][] matrix, int target) {
if (matrix == null || matrix.length == 0 || matrix[0].length == 0) {
return false;
}
int m = matrix.length;
int n = matrix[0].length;
int row = 0;
int col = n - 1;
while (row < m && col >= 0) {
if (matrix[row][col] == target) {
return true;
} else if (matrix[row][col] > target) {
col--;
} else {
row++;
}
}
return false;
}
}
```


Accepted 130 / 130 testcases passed
_bar... submitted at Feb 05, 2025 11:39

Runtime
6 ms | Beats 50.10%

Memory
45.94 MB | Beats 71.08%

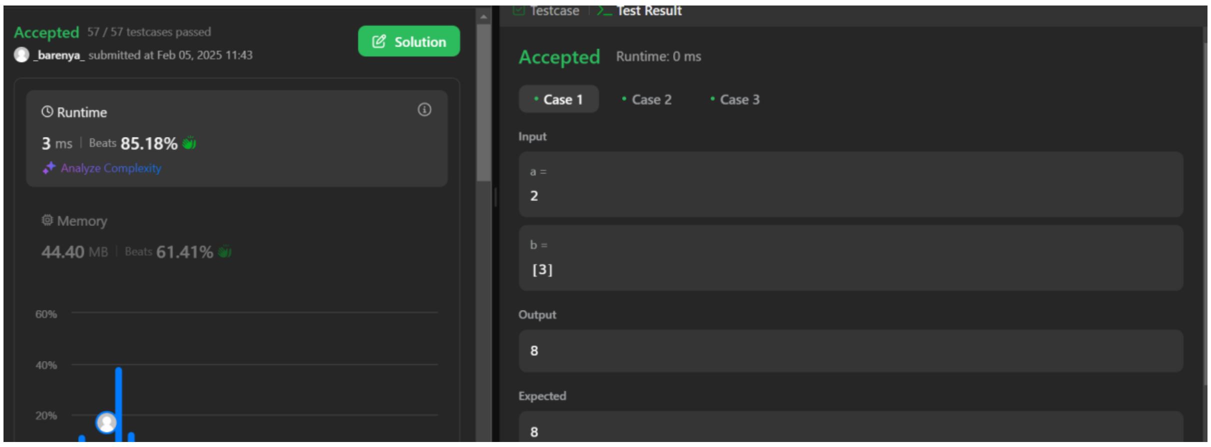
Accepted Runtime: 0 ms

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
```

6. Super Pow

```
 public class Solution {
public int superPow(int a, int[] b) {
return superPowHelper(a, b, b.length - 1);
}
private int superPowHelper(int a, int[] b, int index) {
if (index < 0) {
return 1;
}
int mod = 1337;
int part1 = modPow(a, b[index], mod);
int part2 = modPow(superPowHelper(a, b, index - 1), 10, mod);
return (part1 * part2) % mod;
}
private int modPow(int a, int k, int mod) {
int result = 1;
a %= mod;
for (int i = 0; i < k; i++) {
result = (result * a) % mod;
}
return result;
}
}
```


Accepted 57 / 57 testcases passed
barenya submitted at Feb 05, 2025 11:43

Runtime
3 ms | Beats 85.18%

Memory
44.40 MB | Beats 61.41%

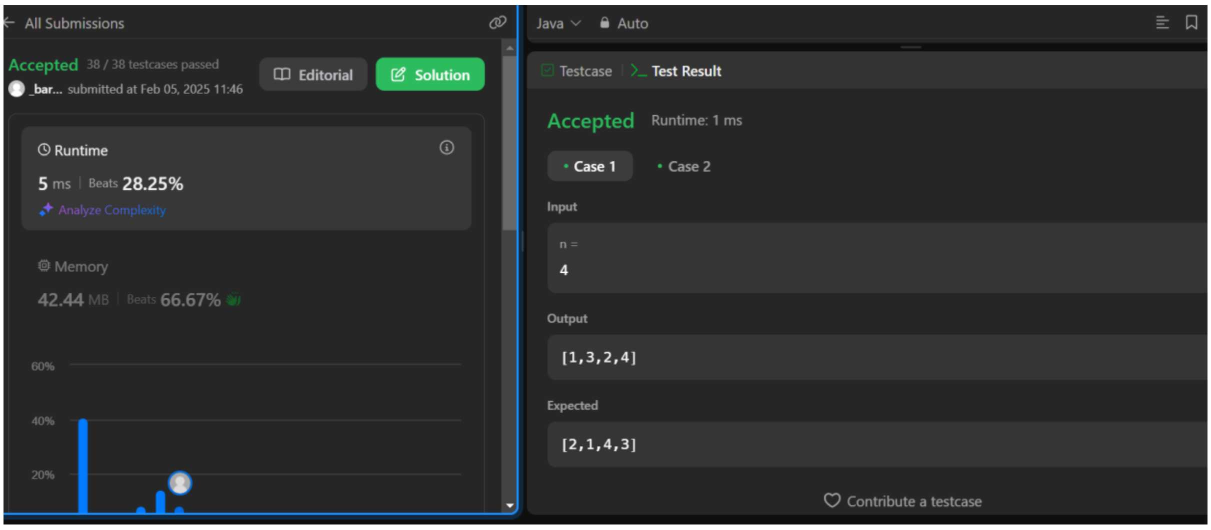
Accepted Runtime: 0 ms

Input

a =
```
2
```
b =
```
[3]
```
Output
```
8
```
Expected
```
8
```

7. Beautiful Array

```
 class Solution {
public int[] beautifulArray(int n) {
List<Integer> result = new ArrayList<>();
result.add(1);
while (result.size() < n) {
List<Integer> temp = new ArrayList<>();
for (int num : result) {
if (2 * num - 1 <= n) {
temp.add(2 * num - 1);
}
}
for (int num : result) {
if (2 * num <= n) {
temp.add(2 * num);
}
}
result = temp;
}
return result.stream().mapToInt(i -> i).toArray();
}
}
```



Accepted 38 / 38 testcases passed
_bar... submitted at Feb 05, 2025 11:46

Runtime
5 ms | Beats 28.25%

Memory
42.44 MB | Beats 66.67%

Accepted Runtime: 1 ms

Case 1 Case 2

Input

```
n =
4
```

Output

```
[1,3,2,4]
```

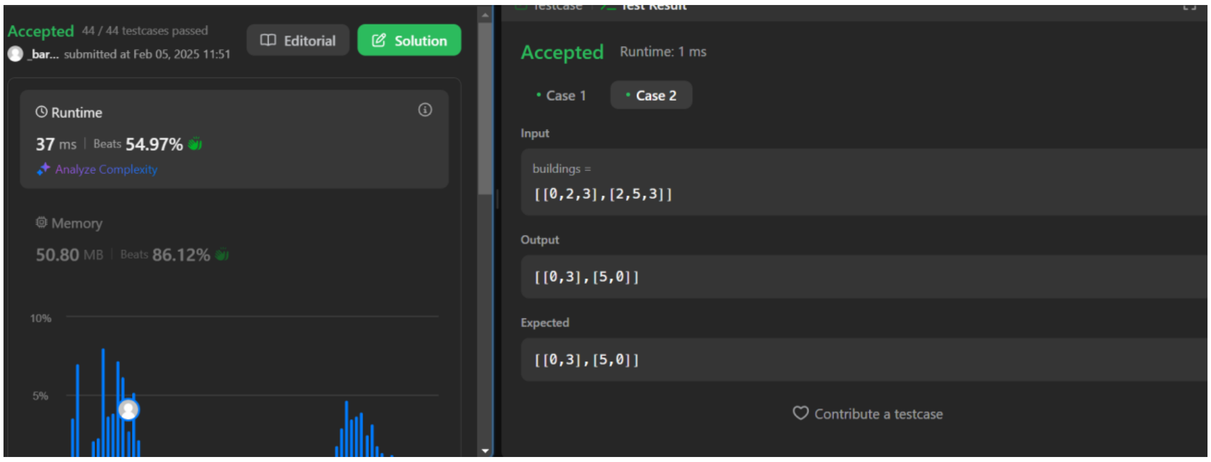
Expected

```
[2,1,4,3]
```

8. The Skyline Problem

```
 class Solution {
public List<List<Integer>> getSkyline(int[][] buildings) {
List<List<Integer>> result = new ArrayList<>();
List<int[]> heightChanges = new ArrayList<>();
for (int[] building : buildings) {
heightChanges.add(new int[]{building[0], building[2]});
heightChanges.add(new int[]{building[1], -building[2]});
}
heightChanges.sort((a, b) -> {
```

```
if (a[0] != b[0]) return Integer.compare(a[0], b[0]);
return Integer.compare(b[1], a[1]);
});
TreeMap<Integer, Integer> heightMap = new TreeMap<>(Collections.reverseOrder());
heightMap.put(0, 1);
int prevMaxHeight = 0;
for (int[] change : heightChanges) {
if (change[1] > 0) {
heightMap.put(change[1], heightMap.getOrDefault(change[1], 0) + 1);
} else {
int count = heightMap.get(-change[1]);
if (count == 1) heightMap.remove(-change[1]);
else heightMap.put(-change[1], count - 1);
}
int currMaxHeight = heightMap.firstKey();
if (currMaxHeight != prevMaxHeight) {
result.add(Arrays.asList(change[0], currMaxHeight));
prevMaxHeight = currMaxHeight;
}
}
return result;
}
}
```



Accepted 44 / 44 testcases passed
_bar... submitted at Feb 05, 2025 11:51

Runtime
37 ms | Beats 54.97%

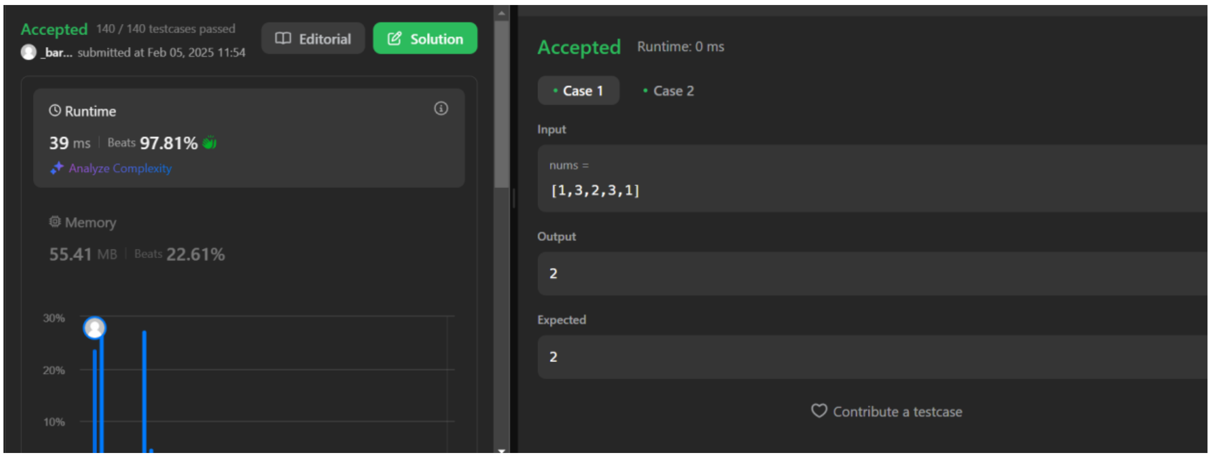
Memory
50.80 MB | Beats 86.12%

Accepted Runtime: 1 ms

Input
buildings =
```
[[0,2,3],[2,5,3]]
```
Output
```
[[0,3],[5,0]]
```
Expected
```
[[0,3],[5,0]]
```

9. Reverse Pairs

```
 class Solution {
public int reversePairs(int[] nums) {
return mergeSort(nums, 0, nums.length - 1);
}
private int mergeSort(int[] nums, int left, int right) {
if (left >= right) return 0;
int mid = left + (right - left) / 2;
int count = mergeSort(nums, left, mid) + mergeSort(nums, mid + 1, right);
int j = mid + 1;
for (int i = left; i <= mid; i++) {
while (j <= right && nums[i] > 2L * nums[j]) j++;
count += (j - (mid + 1));
}
merge(nums, left, mid, right);
return count;
}
private void merge(int[] nums, int left, int mid, int right) {
int[] temp = new int[right - left + 1];
int i = left, j = mid + 1, k = 0;
while (i <= mid && j <= right) {
if (nums[i] <= nums[j]) temp[k++] = nums[i++];
else temp[k++] = nums[j++];
}
while (i <= mid) temp[k++] = nums[i++];
while (j <= right) temp[k++] = nums[j++];
System.arraycopy(temp, 0, nums, left, temp.length);
}
}
```



Accepted 140 / 140 testcases passed
_bar... submitted at Feb 05, 2025 11:54

Runtime
39 ms | Beats 97.81%

Memory
55.41 MB | Beats 22.61%

Accepted Runtime: 0 ms

Case 1 Case 2

Input
nums =
[1,3,2,3,1]

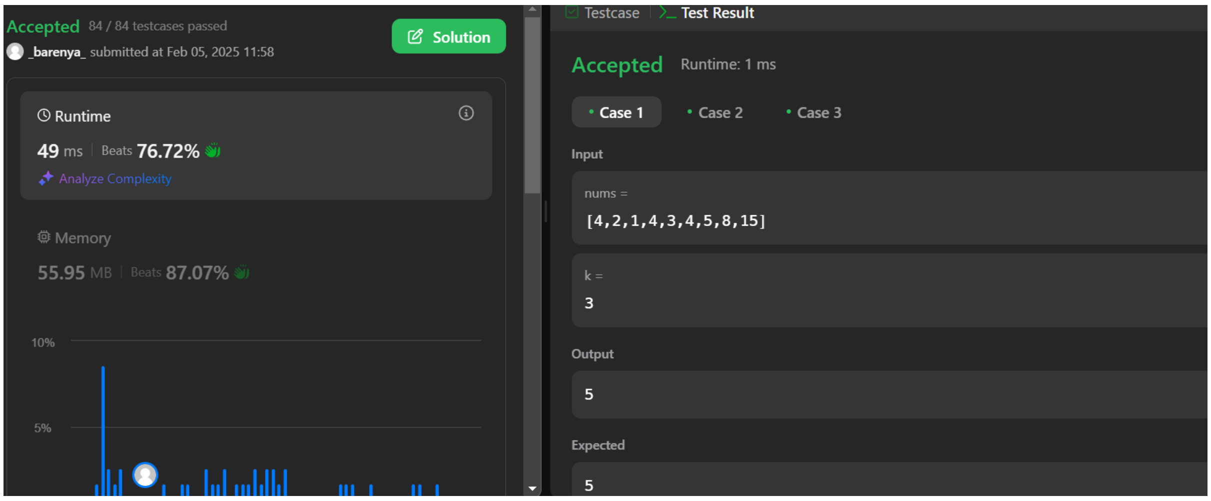
Output
2

Expected
2

10. Longest Increasing Subsequence II

```
 class Solution {
public int lengthOfLIS(int[] nums, int k) {
int maxVal = Arrays.stream(nums).max().getAsInt();
SegmentTree segTree = new SegmentTree(maxVal);
int maxLength = 0;
for (int num : nums) {
int left = Math.max(1, num - k);
int right = num - 1;
int bestPrev = segTree.query(left, right);
int newLength = bestPrev + 1;
segTree.update(num, newLength);
maxLength = Math.max(maxLength, newLength);
}
return maxLength;
}
static class SegmentTree {
```

```
int[] tree;
int size;
SegmentTree(int maxVal) {
size = maxVal + 1;
tree = new int[2 * size];
}
void update(int index, int value) {
index += size;
tree[index] = value;
while (index > 1) {
index /= 2;
tree[index] = Math.max(tree[2 * index], tree[2 * index + 1]);
}
}
int query(int left, int right) {
if (left > right) return 0;
int result = 0;
left += size;
right += size;
while (left <= right) {
if (left % 2 == 1) result = Math.max(result, tree[left++]);
if (right % 2 == 0) result = Math.max(result, tree[right--]);
left /= 2;
right /= 2;
}
return result;
}
}
}
```



Accepted 84 / 84 testcases passed
barenya submitted at Feb 05, 2025 11:58

Runtime
49 ms | Beats 76.72%

Memory
55.95 MB | Beats 87.07%

Accepted Runtime: 1 ms

Input

nums =
```
[4,2,1,4,3,4,5,8,15]
```

k =
3

Output
5

Expected
5