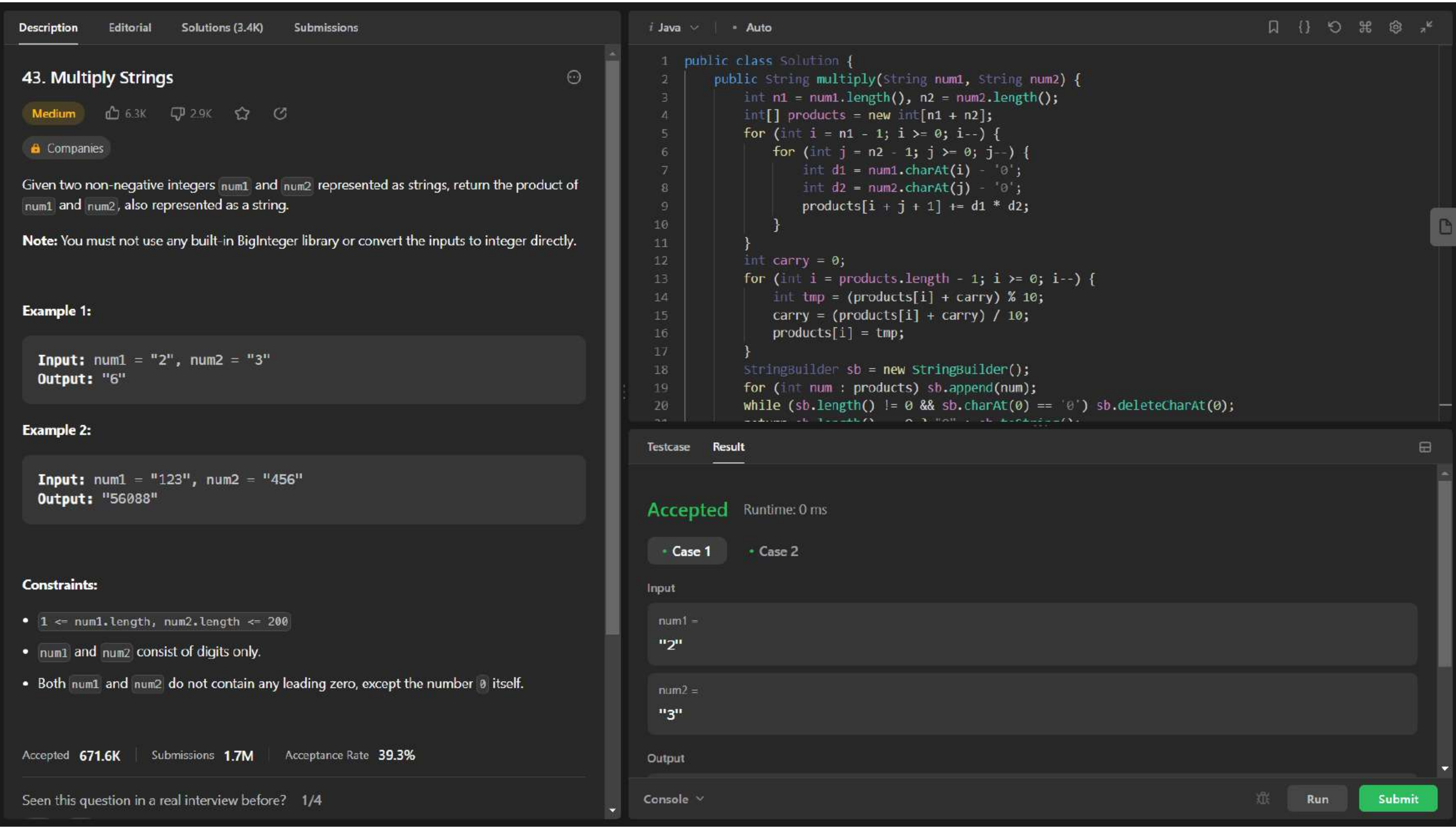


43. Multiply Strings

Given two non-negative integers `num1` and `num2` represented as strings, return the product of `num1` and `num2`, also represented as a string.

Note: You must not use any built-in BigInteger library or convert the inputs to integer directly.

Example 1:

```
Input: num1 = "2", num2 = "3"
Output: "6"
```

Example 2:

```
Input: num1 = "123", num2 = "456"
Output: "56088"
```

Constraints:

- `1 <= num1.length, num2.length <= 200`
- `num1` and `num2` consist of digits only.
- Both `num1` and `num2` do not contain any leading zero, except the number `0` itself.

Accepted 671.6K | Submissions 1.7M | Acceptance Rate 39.3%

Seen this question in a real interview before? 1/4

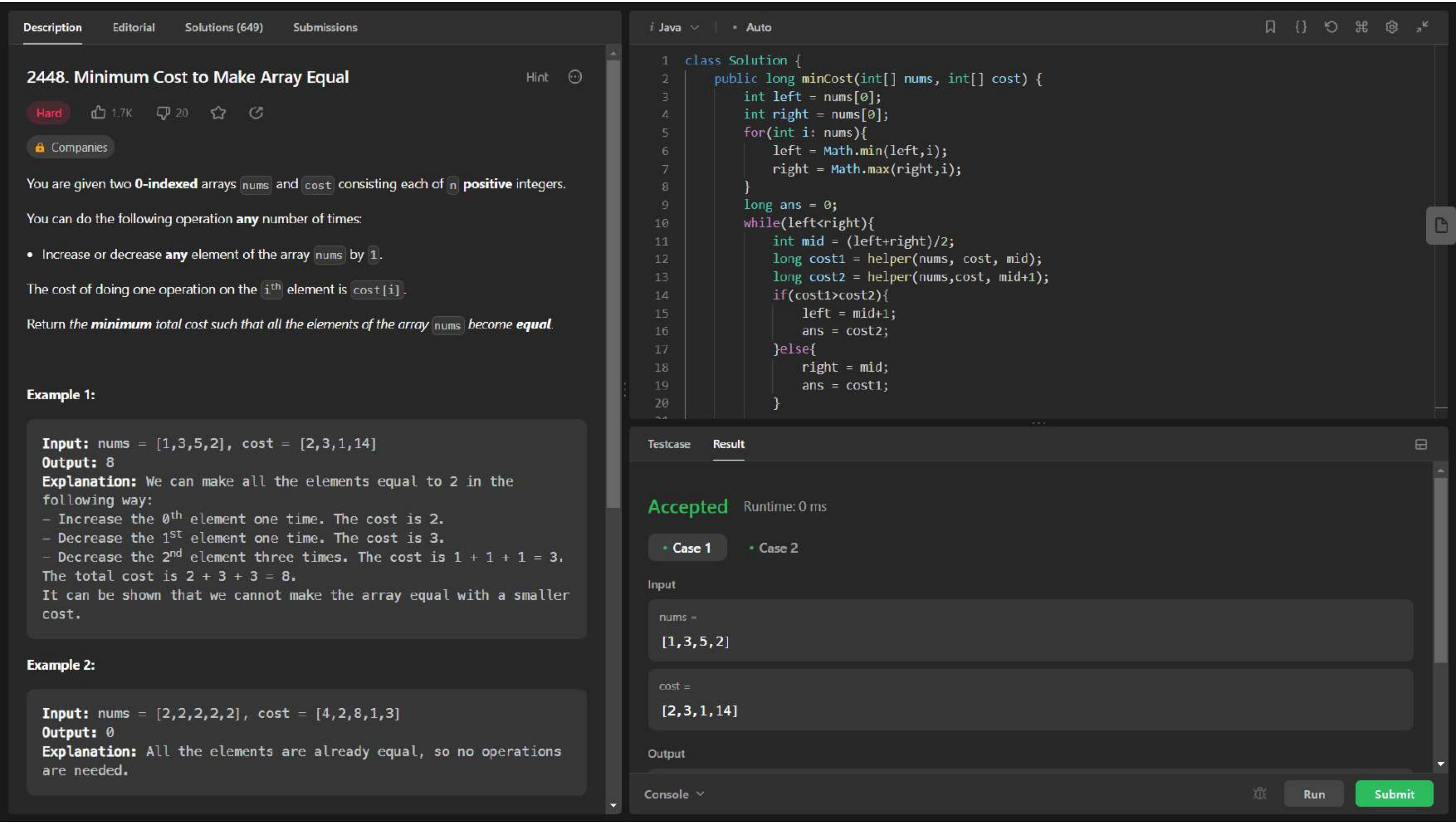
```java
public class Solution {
    public String multiply(String num1, String num2) {
        int n1 = num1.length(), n2 = num2.length();
        int[] products = new int[n1 + n2];
        for (int i = n1 - 1; i >= 0; i--) {
            for (int j = n2 - 1; j >= 0; j--) {
                int d1 = num1.charAt(i) - '0';
                int d2 = num2.charAt(j) - '0';
                products[i + j + 1] += d1 * d2;
            }
        }
        int carry = 0;
        for (int i = products.length - 1; i >= 0; i--) {
            int tmp = (products[i] + carry) % 10;
            carry = (products[i] + carry) / 10;
            products[i] = tmp;
        }
        StringBuilder sb = new StringBuilder();
        for (int num : products) sb.append(num);
        while (sb.length() != 0 && sb.charAt(0) == '0') sb.deleteCharAt(0);
```

Accepted Runtime: 0 ms

Input

```
num1 =
"2"
num2 =
"3"
```

Output



2448. Minimum Cost to Make Array Equal

You are given two **0-indexed** arrays `nums` and `cost` consisting each of `n` **positive** integers.

You can do the following operation **any** number of times:

- Increase or decrease **any** element of the array `nums` by `1`.

The cost of doing one operation on the `i^th` element is `cost[i]`.

Return *the **minimum** total cost such that all the elements of the array* `nums` *become **equal**.*

Example 1:

```
Input: nums = [1,3,5,2], cost = [2,3,1,14]
Output: 8
Explanation: We can make all the elements equal to 2 in the
following way:
- Increase the 0th element one time. The cost is 2.
- Decrease the 1st element one time. The cost is 3.
- Decrease the 2nd element three times. The cost is 1 + 1 + 1 = 3.
The total cost is 2 + 3 + 3 = 8.
It can be shown that we cannot make the array equal with a smaller
cost.
```

Example 2:

```
Input: nums = [2,2,2,2,2], cost = [4,2,8,1,3]
Output: 0
Explanation: All the elements are already equal, so no operations
are needed.
```

```java
class Solution {
    public long minCost(int[] nums, int[] cost) {
        int left = nums[0];
        int right = nums[0];
        for(int i: nums){
            left = Math.min(left,i);
            right = Math.max(right,i);
        }
        long ans = 0;
        while(left<right){
            int mid = (left+right)/2;
            long cost1 = helper(nums, cost, mid);
            long cost2 = helper(nums,cost, mid+1);
            if(cost1>cost2){
                left = mid+1;
                ans = cost2;
            }else{
                right = mid;
                ans = cost1;
            }
```

Accepted Runtime: 0 ms

• Case 1 • Case 2

Input

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

cost =
```
[2,3,1,14]
```

Output