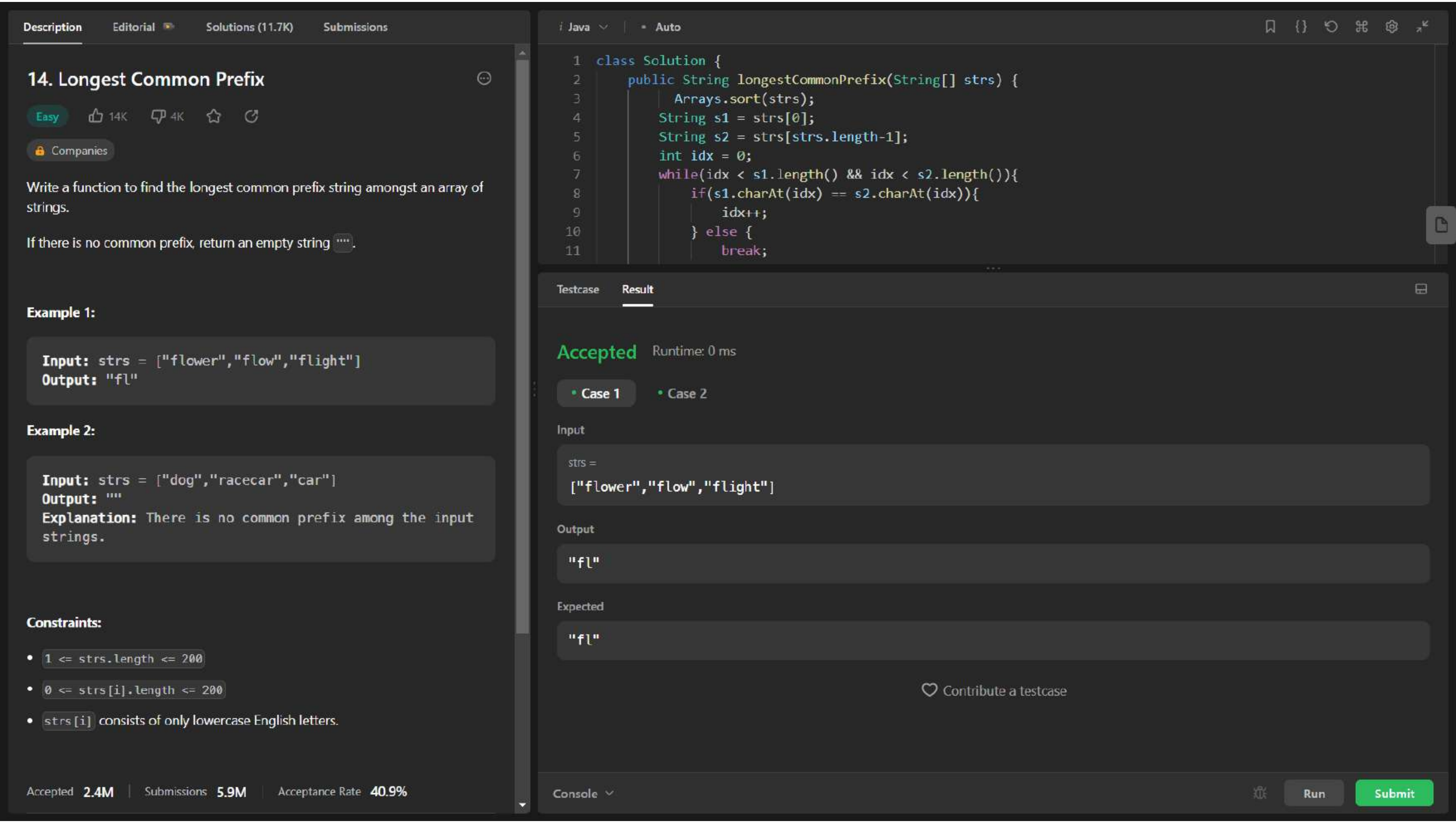


14. Longest Common Prefix

Write a function to find the longest common prefix string amongst an array of strings.

If there is no common prefix, return an empty string `""`.

Example 1:

```
Input: strs = ["flower","flow","flight"]
Output: "fl"
```

Example 2:

```
Input: strs = ["dog","racecar","car"]
Output: ""
Explanation: There is no common prefix among the input
strings.
```

Constraints:

- `1 <= strs.length <= 200`
- `0 <= strs[i].length <= 200`
- `strs[i]` consists of only lowercase English letters.

Accepted 2.4M | Submissions 5.9M | Acceptance Rate 40.9%

```java
class Solution {
    public String longestCommonPrefix(String[] strs) {
          Arrays.sort(strs);
        String s1 = strs[0];
        String s2 = strs[strs.length-1];
        int idx = 0;
        while(idx < s1.length() && idx < s2.length()){
            if(s1.charAt(idx) == s2.charAt(idx)){
                idx++;
            } else {
                break;
```

Accepted Runtime: 0 ms

• Case 1 • Case 2

Input

strs =

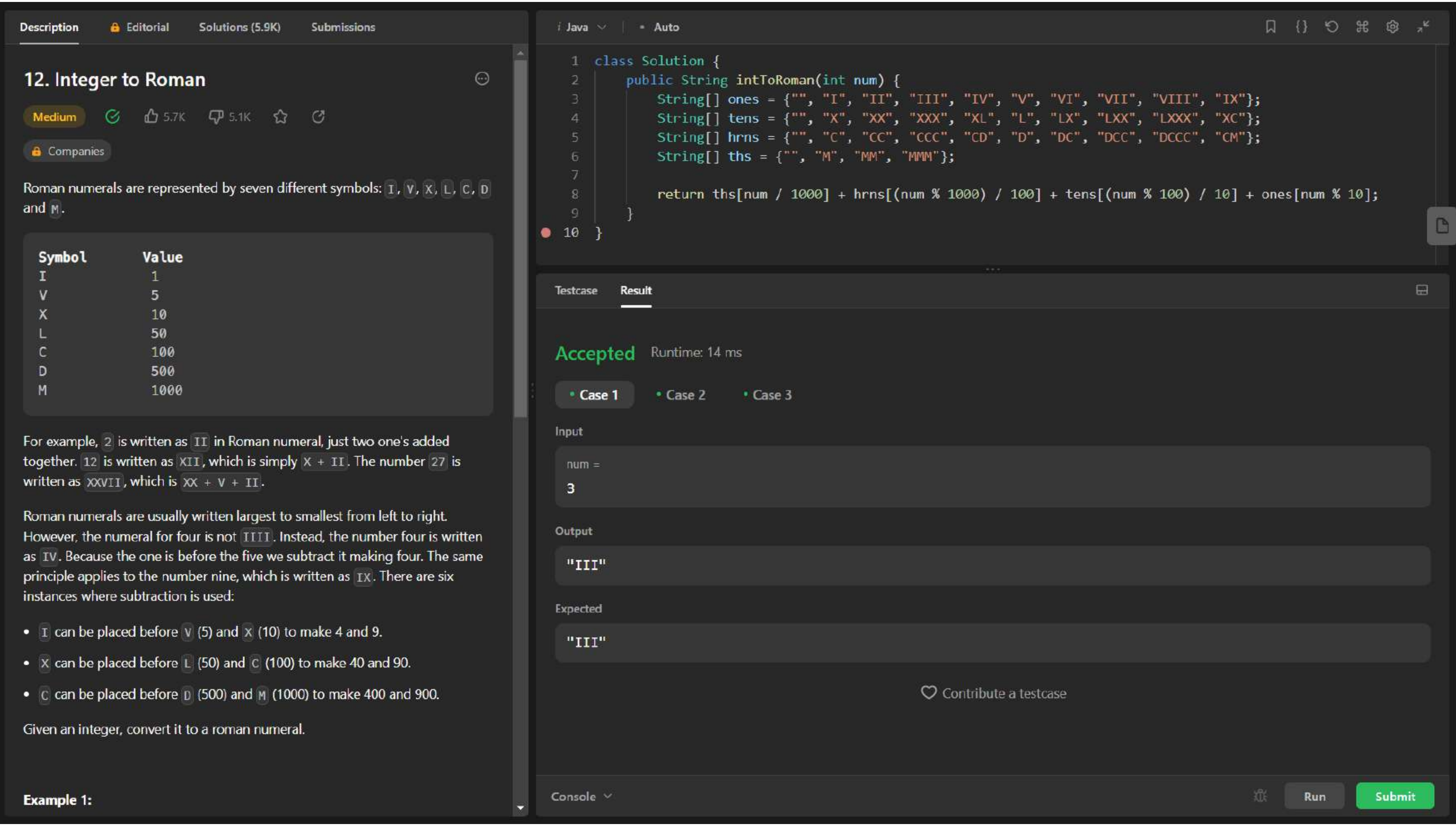
```
["flower","flow","flight"]
```

Output

```
"fl"
```

Expected

```
"fl"
```



12. Integer to Roman

Roman numerals are represented by seven different symbols: `I`, `V`, `X`, `L`, `C`, `D` and `M`.

```
Symbol       Value
I             1
V             5
X             10
L             50
C             100
D             500
M             1000
```

For example, `2` is written as `II` in Roman numeral, just two one's added together. `12` is written as `XII`, which is simply `X + II`. The number `27` is written as `XXVII`, which is `XX + V + II`.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not `IIII`. Instead, the number four is written as `IV`. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as `IX`. There are six instances where subtraction is used:

- `I` can be placed before `V` (5) and `X` (10) to make 4 and 9.
- `X` can be placed before `L` (50) and `C` (100) to make 40 and 90.
- `C` can be placed before `D` (500) and `M` (1000) to make 400 and 900.

Given an integer, convert it to a roman numeral.

Example 1:

```java
class Solution {
    public String intToRoman(int num) {
        String[] ones = {"", "I", "II", "III", "IV", "V", "VI", "VII", "VIII", "IX"};
        String[] tens = {"", "X", "XX", "XXX", "XL", "L", "LX", "LXX", "LXXX", "XC"};
        String[] hrns = {"", "C", "CC", "CCC", "CD", "D", "DC", "DCC", "DCCC", "CM"};
        String[] ths = {"", "M", "MM", "MMM"};

        return ths[num / 1000] + hrns[(num % 1000) / 100] + tens[(num % 100) / 10] + ones[num % 10];
    }
}
```

Accepted Runtime: 14 ms

Input

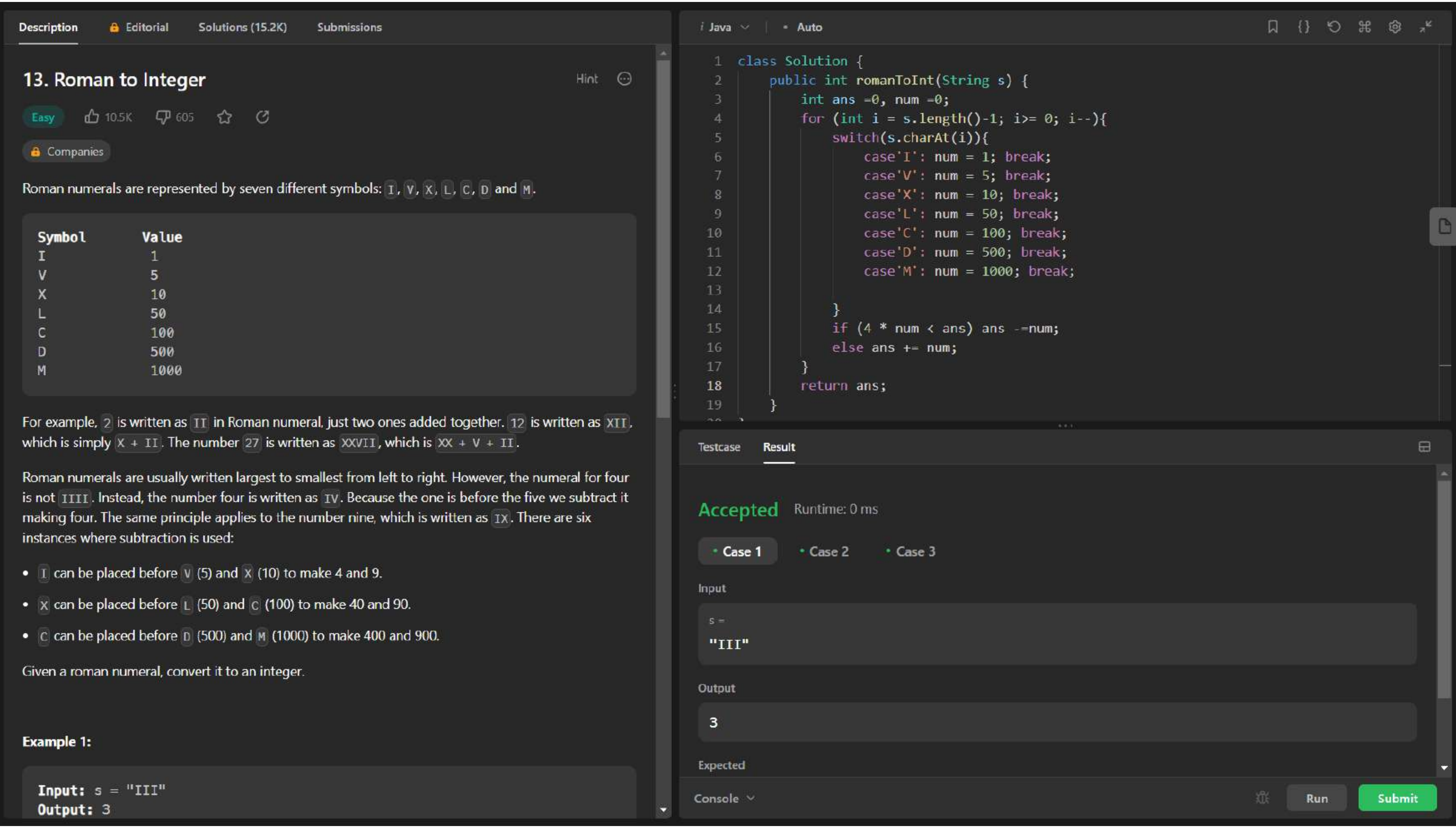
```
num =
3
```

Output

```
"III"
```

Expected

```
"III"
```



13. Roman to Integer

Roman numerals are represented by seven different symbols: `I`, `V`, `X`, `L`, `C`, `D` and `M`.

```
Symbol       Value
I             1
V             5
X             10
L             50
C             100
D             500
M             1000
```

For example, `2` is written as `II` in Roman numeral, just two ones added together. `12` is written as `XII`, which is simply `X + II`. The number `27` is written as `XXVII`, which is `XX + V + II`.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not `IIII`. Instead, the number four is written as `IV`. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as `IX`. There are six instances where subtraction is used:

- `I` can be placed before `V` (5) and `X` (10) to make 4 and 9.
- `X` can be placed before `L` (50) and `C` (100) to make 40 and 90.
- `C` can be placed before `D` (500) and `M` (1000) to make 400 and 900.

Given a roman numeral, convert it to an integer.

Example 1:

```
Input: s = "III"
Output: 3
```

```java
class Solution {
    public int romanToInt(String s) {
        int ans =0, num =0;
        for (int i = s.length()-1; i>= 0; i--){
            switch(s.charAt(i)){
                case'I': num = 1; break;
                case'V': num = 5; break;
                case'X': num = 10; break;
                case'L': num = 50; break;
                case'C': num = 100; break;
                case'D': num = 500; break;
                case'M': num = 1000; break;

            }
            if (4 * num < ans) ans -=num;
            else ans += num;
        }
        return ans;
    }
```

Accepted Runtime: 0 ms

Input

```
s =
"III"
```

Output

```
3
```

Expected