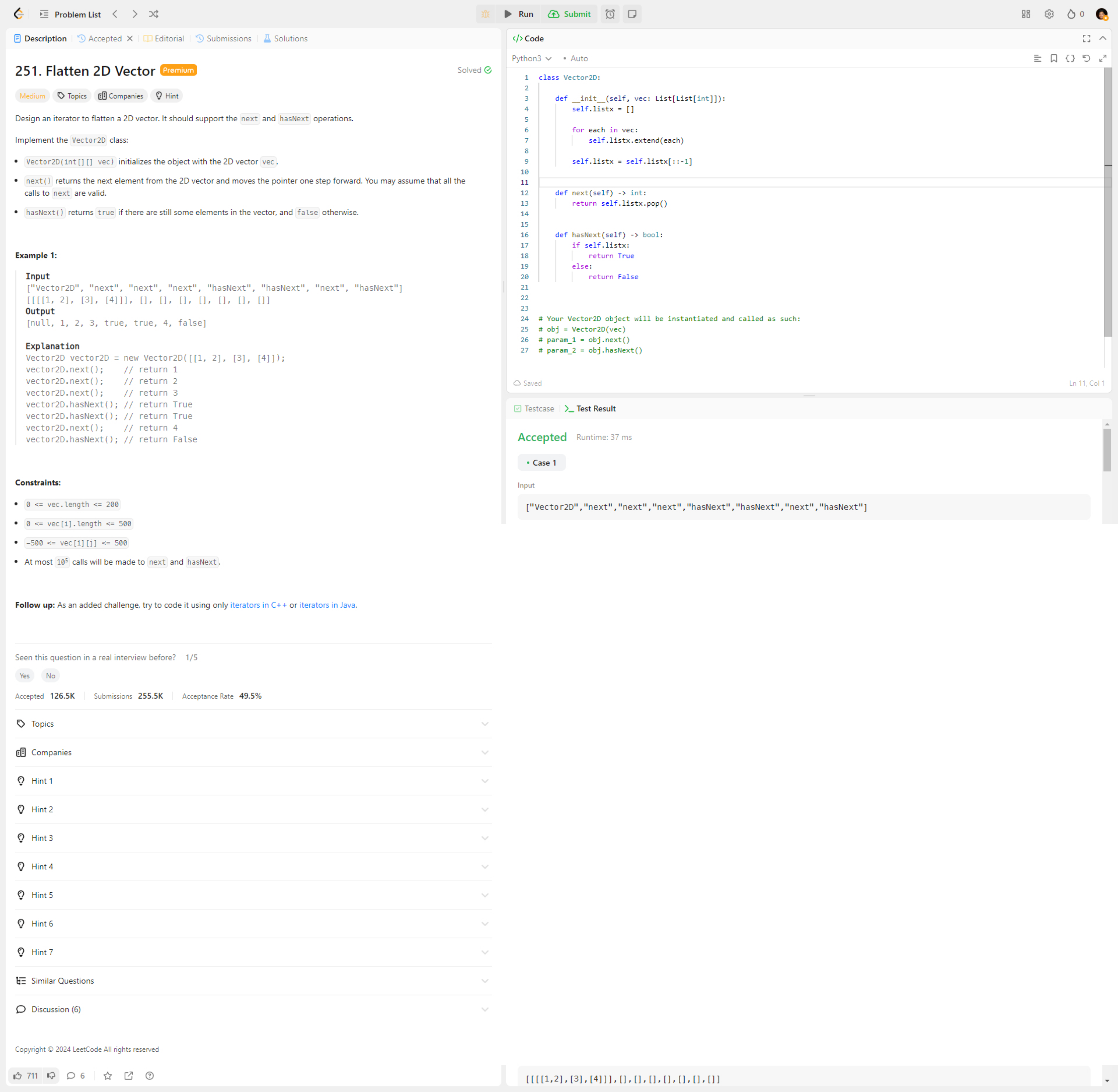

251. Flatten 2D Vector

Design an iterator to flatten a 2D vector. It should support the `next` and `hasNext` operations.

Implement the `Vector2D` class:

- `Vector2D(int[][] vec)` initializes the object with the 2D vector `vec`.
- `next()` returns the next element from the 2D vector and moves the pointer one step forward. You may assume that all the calls to `next` are valid.
- `hasNext()` returns `true` if there are still some elements in the vector, and `false` otherwise.

Example 1:

```
Input
["Vector2D", "next", "next", "next", "hasNext", "hasNext", "next", "hasNext"]
[[[[1, 2], [3], [4]]], [], [], [], [], [], [], []]
Output
[null, 1, 2, 3, true, true, 4, false]

Explanation
Vector2D vector2D = new Vector2D([[1, 2], [3], [4]]);
vector2D.next();    // return 1
vector2D.next();    // return 2
vector2D.next();    // return 3
vector2D.hasNext(); // return True
vector2D.hasNext(); // return True
vector2D.next();    // return 4
vector2D.hasNext(); // return False
```

Constraints:

- `0 <= vec.length <= 200`
- `0 <= vec[i].length <= 500`
- `-500 <= vec[i][j] <= 500`
- At most 10^5 calls will be made to `next` and `hasNext`.

Follow up: As an added challenge, try to code it using only iterators in C++ or iterators in Java.

Seen this question in a real interview before? 1/5

Yes No

Accepted 126.5K | Submissions 255.5K | Acceptance Rate 49.5%

```python
class Vector2D:

    def __init__(self, vec: List[List[int]]):
        self.listx = []

        for each in vec:
            self.listx.extend(each)

        self.listx = self.listx[::-1]


    def next(self) -> int:
        return self.listx.pop()


    def hasNext(self) -> bool:
        if self.listx:
            return True
        else:
            return False



# Your Vector2D object will be instantiated and called as such:
# obj = Vector2D(vec)
# param_1 = obj.next()
# param_2 = obj.hasNext()
```

Accepted Runtime: 37 ms

Input

```
["Vector2D","next","next","next","hasNext","hasNext","next","hasNext"]
```

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