

251. Flatten 2D Vector

Description

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:

- $0 \leq \text{vec.length} \leq 200$
- $0 \leq \text{vec}[i].\text{length} \leq 500$
- $-500 \leq \text{vec}[i][j] \leq 500$
- 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](#).

