Description | Solution | Discuss (440) | Submissions

251. Flatten 2D Vector

Medium 525 288 Add to List Share

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.

Accepted 96,102 | Submissions 203,373

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months | 6 months ~ 1 year | 1 year ~ 2 years

Airbnb | 8

Related Topics

Array | Two Pointers | Design | Iterator

Similar Questions

Binary Search Tree Iterator	Medium
Zigzag Iterator	Medium
Peeking Iterator	Medium
Flatten Nested List Iterator	Medium

Hide Hint 1

How many variables do you need to keep track?

Hide Hint 2

Two variables is all you need. Try with x and y.

Hide Hint 3

Beware of empty rows. It could be the first few rows.

Hide Hint 4

To write correct code, think about the invariant to maintain. What is it?

Hide Hint 5

The invariant is x and y must always point to a valid point in the 2d vector. Should you maintain your invariant *ahead of time* or *right when you need it*?

Hide Hint 6

Not sure? Think about how you would implement `hasNext()`. Which is more complex?

Hide Hint 7

Common logic in two different places should be refactored into a common method.

Java | Autocomplete

```java
import java.util.NoSuchElementException;

class Vector2D {

    private int[][] vector;
    private int inner = 0;
    private int outer = 0;

    public Vector2D(int[][] v) {
        // We need to store a *reference* to the input vector.
        vector = v;
    }

    // If the current outer and inner point to an integer, this method does nothing.
    // Otherwise, inner and outer are advanced until they point to an integer.
    // If there are no more integers, then outer will be equal to vector.length
    // when this method terminates.
    private void advanceToNext() {
        // While outer is still within the vector, but inner is over the
        // end of the inner list pointed to by outer, we want to move
        // forward to the start of the next inner vector.
        while (outer < vector.length && inner == vector[outer].length) {
            inner = 0;
            outer++;
        }
    }

    public int next() {
        // As per Java specs, throw an exception if there's no next.
        // This will also ensure the pointers point to an integer otherwise.
        if (!hasNext()) throw new NoSuchElementException();
        // Return current element and move inner so that is after the current
        // element.
        return vector[outer][inner++];
    }

    public boolean hasNext() {
        // Ensure the position pointers are moved such they point to an integer,
        // or put outer = vector.length.
        advanceToNext();
        // If outer = vector.length then there are no integers left, otherwise
        // we've stopped at an integer and so there's an integer left.
        return outer < vector.length;
    }
}
```

Problems | Pick One | Prev | /99 | Next | Console | Contribute | Run Code | Submit