Design

Run Submit

Description | Accepted | Editorial | Submissions | Solutions

281. Zigzag Iterator Premium

Solved

Medium Topics Companies

Given two vectors of integers `v1` and `v2`, implement an iterator to return their elements alternately.

Implement the `ZigzagIterator` class:

- `ZigzagIterator(List<int> v1, List<int> v2)` initializes the object with the two vectors `v1` and `v2`.
- `boolean hasNext()` returns `true` if the iterator still has elements, and `false` otherwise.
- `int next()` returns the current element of the iterator and moves the iterator to the next element.

Example 1:

```
Input: v1 = [1,2], v2 = [3,4,5,6]
Output: [1,3,2,4,5,6]
Explanation: By calling next repeatedly until hasNext returns false, the order of elements
returned by next should be: [1,3,2,4,5,6].
```

Example 2:

```
Input: v1 = [1], v2 = []
Output: [1]
```

Example 3:

```
Input: v1 = [], v2 = [1]
Output: [1]
```

Constraints:

- `0 <= v1.length, v2.length <= 1000`
- `1 <= v1.length + v2.length <= 2000`
- `-2^31 <= v1[i], v2[i] <= 2^31 - 1`

Follow up: What if you are given `k` vectors? How well can your code be extended to such cases?

Clarification for the follow-up question:

The "Zigzag" order is not clearly defined and is ambiguous for `k > 2` cases. If "Zigzag" does not look right to you, replace "Zigzag" with "Cyclic".

Follow-up Example:

```
Input: v1 = [1,2,3], v2 = [4,5,6,7], v3 = [8,9]
Output: [1,4,8,2,5,9,3,6,7]
```

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

Yes No

Accepted 93.3K | Submissions 146.1K | Acceptance Rate 63.9%

Topics

Companies

Similar Questions

Discussion (3)

Copyright © 2024 LeetCode All rights reserved

673 3

Code

Python3 • Auto

```python
class ZigzagIterator:
    def __init__(self, v1: List[int], v2: List[int]):
        self.v1 = v1[::-1]
        self.v2 = v2[::-1]
        self.chance = True


    def next(self) -> int:
        if self.chance and self.v1:
            self.chance = False
            return self.v1.pop()
        elif not self.chance and self.v2:
            self.chance = True
            return self.v2.pop()
        elif self.v1:
            return self.v1.pop()
        else:
            return self.v2.pop()


    def hasNext(self) -> bool:
        if self.v1 or self.v2:
            return True
        else:
            return False
```

Saved Ln 15, Col 22

Testcase | Test Result

Accepted Runtime: 38 ms

Case 1 Case 2 Case 3

Input

v1 =

[1,2]