```
/**
 * // This is the interface that allows for creating nested lists.
 * // You should not implement it, or speculate about its implementation
 * public interface NestedInteger {
 *
 *     // @return true if this NestedInteger holds a single integer, rather than a nested list.
 *     public boolean isInteger();
 *
 *     // @return the single integer that this NestedInteger holds, if it holds a single integer
 *     // Return null if this NestedInteger holds a nested list
 *     public Integer getInteger();
 *
 *     // @return the nested list that this NestedInteger holds, if it holds a nested list
 *     // Return null if this NestedInteger holds a single integer
 *     public List<NestedInteger> getList();
 * }
 */
public class NestedIterator implements Iterator<Integer> {

    Stack<ListIterator<NestedInteger>> stack;
    public NestedIterator(List<NestedInteger> nestedList) {
        stack = new Stack<>();
        stack.push(nestedList.listIterator());
    }

    @Override
    public Integer next() {
        hasNext();
        return stack.peek().next().getInteger();
    }

    @Override
    public boolean hasNext() {
        while (!stack.isEmpty()) {
            if (!stack.peek().hasNext()) {
                stack.pop();
                continue;
            }
            NestedInteger ni = stack.peek().next();
            if (ni.isInteger()) {
                stack.peek().previous();
                return true;
            } else {
                stack.push(ni.getList().listIterator());
            }
        }
        return false;
    }
}

/**
 * Your NestedIterator object will be instantiated and called as such:
 * NestedIterator i = new NestedIterator(nestedList);
 * while (i.hasNext()) v[f()] = i.next();
```

*/

Given a nested list of integers, implement an iterator to flatten it.

Each element is either an integer, or a list -- whose elements may also be integers or other lists.

Example 1:

```
Input: [[1,1],2,[1,1]]
Output: [1,1,2,1,1]
Explanation: By calling next repeatedly until hasNext returns false,
             the order of elements returned by next should be: [1,1,2,1,1].
```

Example 2:

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