**Introduction to Nim Game**

**Nim** is the most famous two-player algorithm game. The basic rules for this game are as follows:

- The game starts with a number of piles of stones. The number of stones in each pile may not be equal.
- The players alternately pick up 1 or more stones from 1 pile
- The player to remove the last stone wins.

For example, there are $n = 3$ piles of stones having $pile = [3, 2, 4]$ stones in them. Play may proceed as follows:

<table>
<thead>
<tr>
<th>Player</th>
<th>Takes</th>
<th>Leaving</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 from pile[1]</td>
<td>pile=[3,2,4]</td>
</tr>
<tr>
<td>2</td>
<td>2 from pile[1]</td>
<td>pile=[3,2]</td>
</tr>
<tr>
<td>1</td>
<td>1 from pile[0]</td>
<td>pile=[2,2]</td>
</tr>
<tr>
<td>2</td>
<td>1 from pile[0]</td>
<td>pile=[1,2]</td>
</tr>
<tr>
<td>1</td>
<td>1 from pile[1]</td>
<td>pile=[1,1]</td>
</tr>
<tr>
<td>2</td>
<td>1 from pile[0]</td>
<td>pile=[0,1]</td>
</tr>
<tr>
<td>1</td>
<td>1 from pile[1]</td>
<td>WIN</td>
</tr>
</tbody>
</table>

Given the value of $n$ and the number of stones in each pile, determine the game's winner if both players play optimally.

**Function Description**

Complete the `nimGame` function in the editor below. It should return a string, either **First** or **Second**.

`nimGame` has the following parameter(s):

- `pile`: an integer array that represents the number of stones in each pile

**Input Format**

The first line contains an integer, $g$, denoting the number of games they play.

Each of the next $g$ pairs of lines is as follows:

1. The first line contains an integer $n$, the number of piles.
2. The next line contains $n$ space-separated integers $pile[i]$, the number of stones in each pile.

**Constraints**

- $1 \leq g \leq 100$
- $1 \leq n \leq 100$
- $0 \leq s_i \leq 100$
- Player 1 always goes first.

**Output Format**
For each game, print the name of the winner on a new line (i.e., either First or Second).

**Sample Input**

```
2
2
1 1
3
2 1 4
```

**Sample Output**

```
Second
First
```

**Explanation**

In the first case, there are \( n = 2 \) piles of \( pile = [1, 1] \) stones. Player 1 has to remove one pile on the first move. Player 2 removes the second for a win.

In the second case, there are \( n = 3 \) piles of \( pile = [2, 1, 4] \) stones. If player 1 removes any one pile, player 2 can remove all but one of another pile and force a win. If player 1 removes less than a pile, in any case, player 2 can force a win as well, given optimal play.