10000 Cards

Suppose there are N cards on the desk, from top to bottom. They are numbered from 1 to N and they are shuffled. Let's do this to the cards: first, take the first card on the top, destroy it. Now that the second card is on the top, take this card and put it at the bottom. Then take the third card which is now on the top and destroy it. Then put the forth card at the bottom. This goes on until all cards are destroyed.

Here's the problem, the order they are destroyed is known, calculate the initial order when they are all on the desktop, from top to bottom.

Input

Each input contains an integer N $(1 \le N \le 10000)$ indicate the number of cards. Then follow N lines, each line contains an integer c_i $(1 \le c_i \le N)$, where c_1 is the number of the first burnt card, c_2 is the number of the second burnt card.

Output

For each input, output N lines, the initial order of the cards, from top to bottom.

Sample Input 1

```
4
1
3
2
4
```

Sample Output 1

```
1
2
3
4
```

Sample Input 2

```
4
1
2
3
4
```

Sample Output 2

```
1
3
2
4
```

Hint

For the first sample, the initial cards are ordered as: {1, 2, 3, 4}. First destroy 1 and put 2 at the bottom. The cards become {3, 4, 2}. Then destroy 3 and put 4 at the bottom. The cards become {2, 4}. Then destroy 2 and 4 is already at the bottom. Then destroy 4. So the order to destory cards is {1, 3, 2, 4}. In our problem, the order they are destroyed, which is {1, 3, 2, 4}, is given, while the initial order {1, 2, 3, 4} is what you need to calculate.

Requirement as homework

1. Complete the Cards.java to solve the problem. Finish the method "int[] calculateInitCards(int[] destroyOrder)". After that your can run "run.sh" to see the result. However, if your computer cannot run linux shells, you may need to modify the "main" method to read files and do comparason.

2. Generate random new test data to test your program and proves its correctness. Please refer to lab 2 to know how to do that. It is recommanded to upload java code for generating random data together with Cards.java that solves the problem. Measuring running time is not required. However, an exhaustive search may be too slow for this problem, try to find a more efficient way.

3. Include some text file to explain your code if you think it is necessary.