



Practice > Algorithms > Implementation > Organizing Containers of Balls

Organizing Containers of Balls ☆

Problem

Submissions

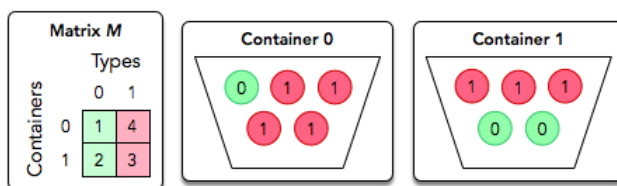
Leaderboard

Discussions

Editorial

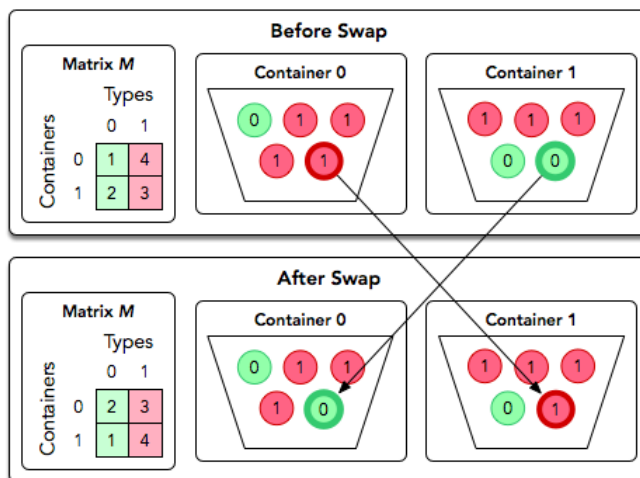
David has several containers, each with a number of balls in it. He has just enough containers to sort each type of ball he has into its own container. David wants to sort the balls using his sort method.

As an example, David has $n = 2$ containers and 2 different types of balls, both of which are numbered from 0 to $n - 1 = 1$. The distribution of ball types per container are described by an $n \times n$ matrix of integers, $M[\text{container}][\text{type}]$. For example, consider the following diagram for $M = [[1, 4], [2, 3]]$:



In a single operation, David can *swap* two balls located in different containers.

The diagram below depicts a single swap operation:



David wants to perform some number of swap operations such that:

- Each container contains only balls of the same type.
- No two balls of the same type are located in different containers.

You must perform q queries where each query is in the form of a matrix, M . For each query, print `Possible` on a new line if David can satisfy the conditions above for the given matrix. Otherwise, print `Impossible`.

Input Format

The first line contains an integer q , the number of queries.

Each of the next q sets of lines is as follows:

- The first line contains an integer n , the number of containers (rows) and ball types (columns).
- Each of the next n lines contains n space-separated integers describing row $M[i]$.

Author

ma5termind

Difficulty

Medium

Max Score

30

Submitted By

8573

NEED HELP?

[View discussions](#)[View editorial](#)[View top submissions](#)

RATE THIS CHALLENGE



MORE DETAILS

[Download problem statement](#)[Download sample test cases](#)[Suggest Edits](#)

Constraints

- $1 \leq q \leq 10$
- $1 \leq n \leq 100$
- $0 \leq M[\textit{container}][\textit{type}] \leq 10^9$

Scoring

- For 33% of score, $1 \leq n \leq 10$.
- For 100% of score, $1 \leq n \leq 100$.

Output Format

For each query, print `Possible` on a new line if David can satisfy the conditions above for the given matrix. Otherwise, print `Impossible`.

Sample Input 0

```
2
2
1 1
1 1
2
0 2
1 1
```

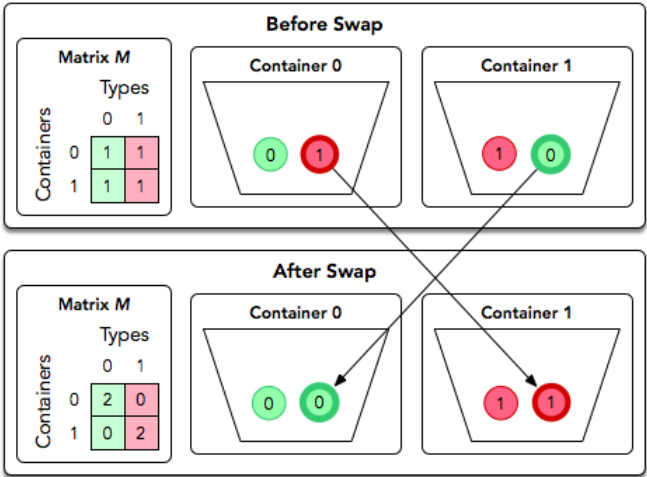
Sample Output 0

```
Possible
Impossible
```

Explanation 0

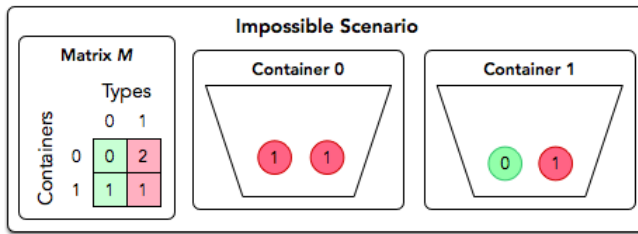
We perform the following $q = 2$ queries:

1. The diagram below depicts one possible way to satisfy David's requirements for the first query:



Thus, we print `Possible` on a new line.

2. The diagram below depicts the matrix for the second query:



No matter how many times we swap balls of type t_0 and t_1 between the two containers, we'll never end up with one container only containing type t_0 and the other container only containing type t_1 . Thus, we print `Impossible` on a new line.

Current Buffer (saved locally, editable) Java 7

```

1  import java.io.*;
2  import java.util.*;
3  import java.text.*;
4  import java.math.*;
5  import java.util.regex.*;
6
7  public class Solution {
8
9      static String organizingContainers(int[][] container) {
10         // Complete this function
11     }
12
13     public static void main(String[] args) {
14         Scanner in = new Scanner(System.in);
15         int q = in.nextInt();
16         for(int a0 = 0; a0 < q; a0++){
17             int n = in.nextInt();
18             int[][] container = new int[n][n];
19             for(int container_i = 0; container_i < n; container_i++){
20                 for(int container_j = 0; container_j < n;
21 container_j++){
22                     container[container_i][container_j] = in.nextInt();
23                 }
24             }
25             String result = organizingContainers(container);
26             System.out.println(result);
27         }
28         in.close();
29     }
30 }

```

Line: 1 Col: 1

[Upload Code as File](#) ☐ [Test against custom input](#)

[Run Code](#)

[Submit Code](#)