Organizing Containers of Balls

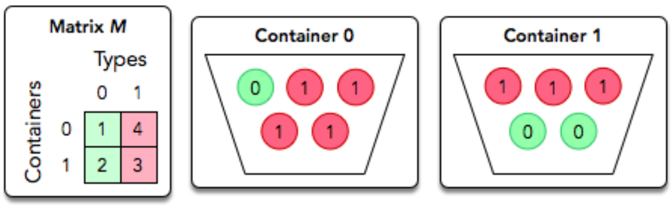
locked

by ma5termind

Problem Submissions Leaderboard Discussions Editorial

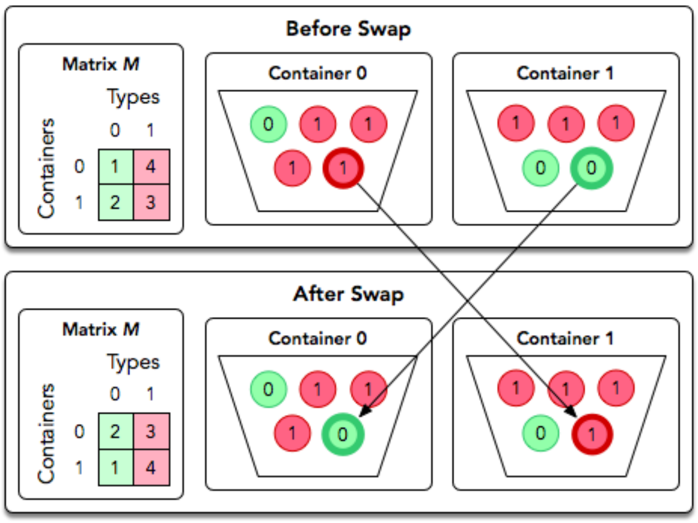
David has several containers, each with a number of balls in it. He has just enough containers to sort each type of ball he has into its ow
container. David wants to sort the balls using his sort method.

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 distri
ball types per container are described by an $n \times n$ matrix of integers, $M[container][type]$. For example, consider the following diagr
$M = [[1, 4], [2, 3]]$:



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 car
the conditions above for the given matrix. Otherwise, print `Impossible`.

Function Description

Complete the *organizingContainers* function in the editor below. It should return a string, either `Possible` or `Impossible`.

organizingContainers has the following parameter(s):

- *containter*: a two dimensional array of integers that represent the number of balls of each color in each container

Input Format

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

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

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 `Impossi`

Sample Input 0

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

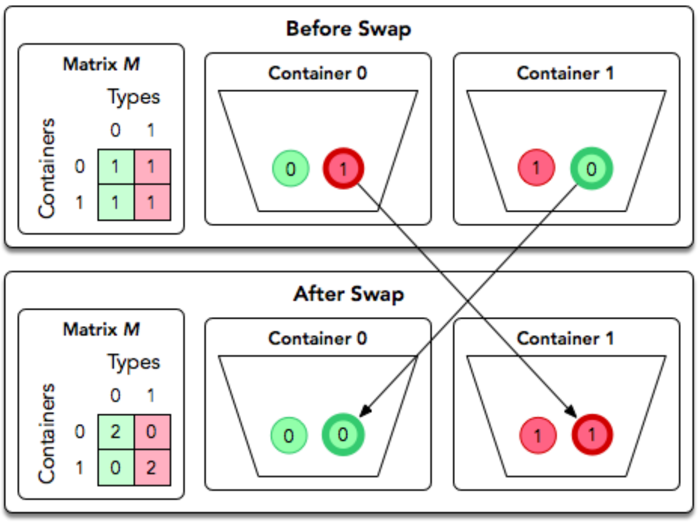
Sample Output 0

```
Possible
Impossible
```

Explanation 0

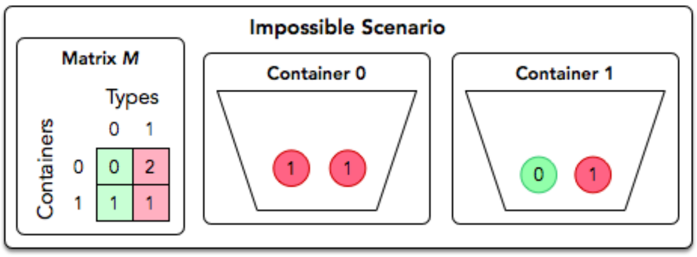
We perform the following $q = 2$ queries:

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:



```
3
1 3 1
2 1 2
3 3 3
3
0 2 1
1 1 1
2 0 0
```

Sample Output 1

```
Impossible
Possible
```

Submissions: 62
Max Score: 15

Rate This Challeng

☆ ☆ ☆ ☆ ☆

More

Current Buffer (saved locally, editable)

C

```c
#include <assert.h>
#include <limits.h>
#include <math.h>
#include <stdbool.h>
#include <stddef.h>
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char* readline();
char** split_string(char*);

// Complete the organizingContainers function below.

// Please either make the string static or allocate on the heap. For example,
// static char str[] = "hello world";
// return str;
//
// OR
//
// char* str = "hello world";
// return str;
//
char* organizingContainers(int container_rows, int container_columns, int** container) {


}

int main()
{
    FILE* fptr = fopen(getenv("OUTPUT_PATH"), "w");

    char* q_endptr;
    char* q_str = readline();
    int q = strtol(q_str, &q_endptr, 10);

    if (q_endptr == q_str || *q_endptr != '\0') { exit(EXIT_FAILURE); }

    for (int q_itr = 0; q_itr < q; q_itr++) {
        char* n_endptr;
        char* n_str = readline();
        int n = strtol(n_str, &n_endptr, 10);

        if (n_endptr == n_str || *n_endptr != '\0') { exit(EXIT_FAILURE); }

        int** container = malloc(n * sizeof(int*));

        for (int i = 0; i < n; i++) {
            *(container + i) = malloc(n * (sizeof(int)));

            char** container_item_temp = split_string(readline());

```