Problem Description

- Your task will be to calculate number of different assignments of n different topics to n students such that everybody gets exactly one topic he likes.
- First line of input contains number of students n.
- Each of the next n lines contains n integers describing preferences of one student. 1 at the ith position means that this student likes ith topic, 0 means that he definitely doesn't want to take it.
- For each test case output number of different assignments (it will fit in a signed 64-bit integer).

Problem Breakdown:

1. **Objective:**

    - Assign `n` different topics to `n` students such that each student gets exactly one topic they like.

2. **Input:**

    - The first line contains the number of students `n`.
    - The next `n` lines each contain `n` integers (0 or 1). The `i-th` line represents the preferences of the `i-th` student, where `1` means the student likes the topic and `0` means they do not.

3. **Output:**

    - The number of different assignments where each student gets a topic they like, and no two students are assigned the same topic.

Solution Approach:

Graph Theory Perspective:

- This problem can be visualized as finding perfect matchings in a bipartite graph:

    - One set of vertices represents students.
    - The other set represents topics.
    - An edge exists between a student and a topic if the student likes that topic (preference value is `1`).

Method: Backtracking

- **Backtracking** is a systematic way to explore all possible assignments. It tries all potential solutions and backtracks when a conflict is detected (i.e., a topic is already assigned).

Steps in the Backtracking Approach:

1. **Initialization:**

    - Use an array `match` to keep track of which topic is assigned to which student.
    - Use a boolean array `used` to check if a topic is already assigned.

2. **Recursive Function:**

- Create a recursive function backtrack(student) that attempts to assign a topic to the current student.
- If student == n (base case), it means all students have been assigned topics successfully, so increment the count of valid assignments.
- For each student, iterate through all topics.
- If a topic is liked by the student and not yet assigned (preferences[student][topic] == 1 and used[topic] == False), assign it.
- Recursively call backtrack for the next student.
- After exploring all possibilities for the current student, backtrack by unassigning the topic.

3. **Counting Valid Assignments:**

- Start with the first student and attempt to assign each topic.
- Track the count of valid assignments through recursive exploration.