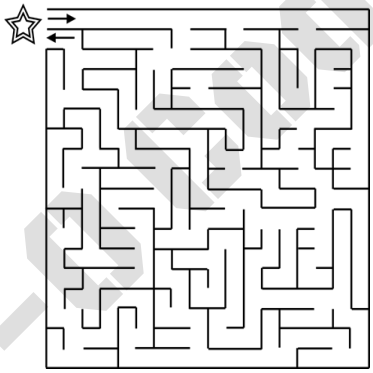
# MAKEMAZE - VALIDATE THE MAZE

Link: http://www.spoj.com/problems/MAKEMAZE/

Solution:

| | |
|---|---|
| C++ | http://ideone.com/437zME |
| Java | https://ideone.com/g45xDd |
| Python | https://ideone.com/cfhE2v |

**Tóm tắt đề:** Cho bạn một mê cung, bạn xác định xem mê cung đó có hợp lệ hay không. Một mê cung hợp lệ hay không nghĩa là: có chính xác một điểm vào và một điểm thoát ra và phải có ít nhất một đường đi đến từ điểm vào đến điểm thoát ra.



**Input**

Dòng đầu tiên chứa một số nguyên t là số lượng test 1≤ t≤10000. Sau đó mỗi bộ test chứa các thông tin:

- Dòng đầu tiên chứa 2 số nguyên m (1≤ m ≤ 20) và n (1≤ n ≤ 20), số dòng và số cột trong mê cung.
- Các dòng tiếp theo, chứa toàn bộ mô tả mê cung M với mxn. M[i][j] = “#” đại diện cho một bức tường và M[i][j] = “.” đại diện cho khoảng trống.

**Output**

Mỗi dòng test tìm xem mê cung có hợp lệ hay không hợp lệ. Nếu hợp lệ in ra “valid” ngược lại sẽ in ra “invalid”.

Big-O Coding

| | |
|---|---|
| 2<br>4 4<br>####<br>#...<br>#..#<br>#.##<br>3 4<br>#..#<br>#.##<br>#.## | valid<br>invalid |

**Giải thích:** Bộ dữ liệu đầu tiên có đầy đủ 1 điểm vào, một điểm ra và đường đi từ điểm vào đến điểm ra nên mê cung này là hợp lệ → valid.

Bộ dữ liệu tiếp theo có hơn 1 điểm vào và điểm ra nên không hợp lệ → invalid.

**Hướng dẫn giải:**

Bài toán này bạn sẽ có 2 cách giải:

- **Cách 1:** Bạn sẽ chuyển toàn bộ đồ thị đã cho về ma trận kề hoặc danh sách kề, bằng cách duyệt qua toàn bộ ma trận, 2 đỉnh được nối với nhau khi vào chỉ khi có thể đi từ "." này đến "." kia.

  Bạn có thể đặt đỉnh ở tọa độ (0, 0) là đỉnh 0, (0, 1) là đỉnh 1… lần lượt như vậy nếu đồ thị có 4x4 thì bạn sẽ có 15 đỉnh.

  Bước tiếp theo bạn sẽ xem đỉnh nào ở rìa ma trận thì sẽ đặt đỉnh đó là đỉnh vào hoặc đỉnh ra, bạn cần tìm đủ 2 đỉnh như vậy rồi chạy BFS. Nếu có đường đi thì bạn sẽ in ra "valid" ngược lại bạn sẽ in ra "invalid".

  ⇨ **Nhận xét:** với cách giải này bạn sẽ tốn thời gian chuẩn bị lại đồ thị cho đúng định dạng.
- **Cách 2:** Cách này bạn sẽ chạy BFS trên mê cung đã cho mà không cần phải chuyển lại thành dạng ma trận kề hay danh sách kề. Cách này bạn phải thêm và chỉnh lại một số dòng code để chạy phù hợp.

  Ban đầu bạn cũng sẽ tìm 2 điểm đầu vào và đầu ra. Từ điểm đầu ra bạn sẽ xác định 4 hướng đi (lên, xuống, trái, phải). Nếu có đường đi, nghĩa là gặp dấu "." và nằm trong giới hạn của mê cung thì bạn sẽ dịch chuyển bước đi của mình xuống điểm mới. Lần lượt đi

Big-O Coding

đến khi nào gặp đỉnh đầu ra thì dừng. Lúc này sẽ in ra là “valid”, ngược lại nếu đi mà không thấy đường ra sẽ in ra “invalid”.

**Độ phức tạp:** **O(T * R * C)** với T là số lượng test, R và C lần lượt là số dòng và số cột của ma trận.

Big-O Coding