## 03. TROJKE

**Giới hạn bộ nhớ: 32MB**

Mirko và Slavko đang chơi một trò chơi có tên gọi là “Trojke” (Triplets). Đầu tiên họ sử dụng phấn để vẽ một lưới ô vuông N×N. Sau đó, họ viết lên mỗi ô vuông một chữ cái. Mỗi chữ cái chỉ xuất hiện trên lưới duy nhất một lần.

Trò chơi yêu cầu tìm 3 chữ cái trên một đường thẳng càng nhanh càng tốt. 3 chữ cái cùng nằm trên một đường thẳng nếu có một đường thẳng đi qua trung tâm 3 ô vuông chứa chữ cái.

Càng về sau càng khó khăn để tìm được 3 ô vuông như vậy. Mirko và Slavko cần một chương trình đếm tất cả 3 chữ cái thẳng hàng để họ biết trò chơi kết thúc hay chưa hoặc họ cần phải tìm kiếm thêm.

**INPUT**

Dòng đầu chứa một số nguyên 3 ≤ N ≤ 100 là kích thức của lưới ô vuông. Sau đó là N dòng, mỗi dòng chứa N kí tự mô tả lưới bởi gồm các chữ cái in hoa và dấu ‘.’ đại diện cho một ô vuông trống.

**OUTPUT**

In ra số lượng của bộ 3 ô vuông chữ cái thẳng hàng.

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| 4  ...D  ..C.  .B..  A... | 4 |
| 5  ..T..  A....  .FE.R  ....X  S.... | 3 |
| 10  ....AB....  ..C....D..  .E......F.  ...G..H...  I........J  K........L  ...M..N...  .O......P.  ..Q....R..  ....ST.... | 0 |

## 04. TENKICI

**Giới hạn bộ nhớ: 32MB**

Mirko tìm thấy bộ sưu tập N chiếc xe tăng đồ chơi từ Chiến tranh thế giới thứ hai trên gác xép của ông nội mình. Anh ta nhanh chóng gọi bạn mình Slavko đến chơi cùng. Họ tạo ra một chiến trường trên một bảng gỗ - là một lưới ô vuông N dòng và N cột.

Mỗi chiếc xe tăng có thể di chuyển đến một trong 4 ô xung quanh trong mỗi bước. Và chiếc xe tăng có thể bắn đến bất kì ô nào ở trên cùng hàng hoặc cùng cột. Chiếc xe tăng được giao nhiệm vụ bảo vệ hàng và cột mà nó đang đứng.

Ngoài ra, không có hai chiếc xe tăng ở trong cùng một ô vuông vào bất kì thời điểm nào.

Sau nhiều giờ chơi, mẹ của Mirko gọi họ xuống ăn trưa, và họ quyết định sắp xếp lại các xe tăng để mà mỗi chiếc bảo vệ bảo vệ một dòng và một cột (cũng có nghĩa là mỗi hàng và cột chỉ chứa duy nhất một xe tăng).

Tuy nhiên, họ muốn thực hiện điều này bằng ít bước di chuyển nhất.

Viết chương trình tìm số bước di chuyển ít nhất để sắp xếp các xe tăng sao cho mỗi dòng và mỗi cột chứa một chiếc xe tăng, thỏa mãn số bước di chuyển ít nhất.

**INPUT**

Dòng đầu tiên chứa số nguyên N (5 ≤ N ≤ 500). Mỗi dòng trong N dòng tiếp theo chứa 2 số nguyên R và C (1 ≤ R, S ≤ N) là hàng và cột của mỗi chiếc xe tăng tại thời điểm người mẹ gọi. Không có 2 chiếc xe tăng trong cùng một ô vuông. Các hàng và các cột được đánh dấu từ 1 đến N, từ trên xuống dưới và từ trái qua phải.

**OUTPUT**

Dòng đầu tiên chứa số nguyên K là số bước di chuyển tối thiểu. Mỗi dòng trong K dòng tiếp theo chứa số thứ tự của chiếc xe tăng (các xe tăng được đánh số từ 1 đến N theo danh sách trong đầu vào) và hướng di chuyển của chiếc xe tăng (U : up, D : down, L : left, R : right).

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| 5  1 1  1 2  1 3  1 4  1 5 | 10  1 D  2 D  3 D  4 D  1 D  2 D  3 D  1 D  2 D  1 D |
| 5  2 3  3 2  3 3  3 4  4 3 | 8  1 R  1 R  2 U  2 U  4 D  4 D  5 L  5 L |
| 6  1 1  1 2  2 1  5 6  6 5  6 6 | 8  2 R  2 D  3 D  3 R  4 U  4 L  5 L  5 U |

## 05. BICIKLI

**Giới hạn bộ nhớ: 32MB**

Một cuộc đua xe đạp đang được tổ chức ở một vùng đất ở rất xa. Có N thị trấn trên vùng đất đó, được đánh số từ 1 đến N. Có M đường một chiều nối giữa 2 thị trấn. Cuộc đua xuất phát từ thị trấn 1 và kết thúc tại thị trấn 2. Có bao nhiêu cách khác nhau để thiết lập tuyến đường đua ? 2 tuyến đường được cho là khác nhau nếu chúng không sử dụng cùng một đường.

**INPUT**

Dòng đầu tiên chứa 2 số nguyên N và M (1 ≤ N ≤ 10000, 1 ≤ M ≤ 100000) là số lượng thị trấn và số lượng đường đi. Mỗi dòng trong M dòng tiếp theo chứa 2 số nguyên A và B, đại diện cho đường đi giữa 2 thị trấn A và B. Các thị trấn có thể nối với nhau bằng nhiều hơn một con đường.

**OUTPUT**

In ra số tuyến đường khác nhau c. Nếu số đó có nhiều hơn 9 chữ số thì chỉ ghi ra 9 chữ số tận cùng. Nếu có nhiều vô kể tuyến đường, in ra “inf”.

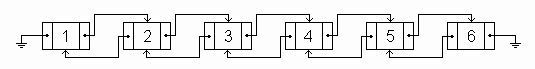
|  |  |
| --- | --- |
| Sample Input | Sample Output |
| 6 7  1 3  1 4  3 2  4 2  5 6  6 5  3 4 | 3 |
| 6 8  1 3  1 4  3 2  4 2  5 6  6 5  3 4  4 3 | Inf |
| 31 60  1 3  1 3  3 4  3 4  4 5  4 5  5 6  5 6  6 7  6 7  …  …  …  28 29  28 29  29 30  29 30  30 31  30 31  31 2  31 2 | 073741824 |

## 06. LISTA

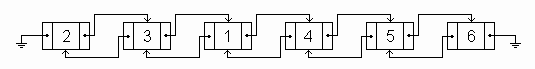
**Giới hạn bộ nhớ: 32MB**

Trong ngày sinh nhật, Mirko nhận được một món quà từ người cô đang ở Mỹ - một danh sách móc nối đôi (một ví dụ ở hình bên dưới). Danh sách này chứa N nút được đánh số từ 1 đến N. 2 cách di chuyển có thể thực hiện trong danh sách là:

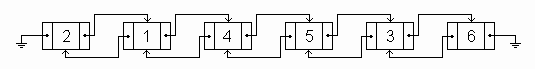
1. Di chuyển nút X về trước nút Y
2. Di chuyển nút X về sau nút Y.



Danh sách với 6 nút.



Danh sách trên sau bước di chuyển “A 1 4”



Danh sách trên sau một bước khác “B 3 5”

Mirko chơi món đồ chơi mới này trong nhiều giờ liên, ghi lại mỗi bước di chuyển một một mảnh giấy để anh ấy có thể dựng lại trạng thái ban đầu của danh sách (các nút từ 1 đến N được đánh số từ trái sang phải).

Khi quyết định dựng lại danh sách, Mirko đã rất ngạc nhiên khi nhận ra rằng không có cách nào dễ dàng để đảo ngược các bước di chuyển và dựng lại trạng thái ban đầu của danh sách. Anh ta không biết được vị trí nút X trước mỗi lần di chuyển mà chỉ biết vị trí sau di chuyển.

Thấy bạn Mirko vẫn đang cố gắng sửa lại sau cú sock đó, bạn hãy viết một chương trình tìm số bước tối thiểu để phục hồi lại danh sách về trạng thái ban đầu từ các bản ghi của Mirko.

**INPUT**

Dòng đầu tiên chứa 2 số nguyên N và K (2 ≤ N ≤ 500000, 0 ≤ M ≤ 100000) là số nút và số bước di chuyển được thực hiện bởi Mirko. Mỗi dòng trong M dòng tiếp theo chứa mô tả của một bước di chuyển – cách thức di chuyển và hai số nguyên X và Y.

**OUTPUT**

Trên dòng đầu tiên in ra số bước tối thiểu (gọi số này là K). Mỗi dòng trong K dòng tiếp theo chứa mô tả của một bước chuyển động với định dạng tương tự như dữ liệu đầu vào.

**GHI ĐIỂM**

Trong các trường hợp đưa ra để kiểm tra, nếu cả số K và trình tự di chuyển các bước đều đúng, chương trình của bạn sẽ ghi được đủ điểm.

Nếu chương trình của bạn ghi ra đúng số K và không ghi ra trình tự các bước, hoặc trình tự các bước bị sai, bạn chỉ được 60% số điểm trong trường hợp đó.

|  |  |
| --- | --- |
| Sample Input | Sample Output |
| 2 1  A 2 1 | 1  A 1 2 |
| 4 3  B 1 2  A 4 3  B 1 4 | 2  A 1 2  B 4 3 |
| 6 5  A 1 4  B 2 5  B 4 2  B 6 3  A 3 5 | 3  A 4 5  B 6 5  A 2 3 |