Duyệt Đồ Thị

Duyệt, Vét , Sort, Hai phía.

HĐP

02. CHỈ DUYỆT ĐỒ THỊ

UVA118. THĂM DÒ THẾ GIỚI PHẲNG

Khoa học người máy, nghiên cứu về chuyển động của robot và học máy là những lĩnh vực đã vượt khỏi ranh giới của nhiều ngành trong Khoa học máy tính: Trí tuệ nhân tạo, Thuật toán và độ phức tạp, Kỹ thuật điện và cơ khí. Hơn nữa, những con robot cũng như "những chú rùa" (lấy cảm hứng từ công trình của Papert, Abelson và diSessa) hay giống "beeper-pickers" (lấy cảm hứng từ công trình của Pattis) đã được các sinh viên nghiên cứu và sử dụng trong khóa học về nhập môn lập trình trong nhiều năm. Bài toán này có liên quan đến việc xác định vị trí của một robot đang thám hiểm thế giới phẳng thời tiền Columbus. Cho trước một lưới chữ nhật và một dãy các vị trí của robot cùng những chỉ dẫn, hãy viết một chương trình xác định từng dẫy vị trí robot và chỉ dẫn đến vị trí cuối cùng của robot. Vị trí của một robot bao gồm toạ độ lưới (một cặp số nguyên: toạ độ x và y) và hướng (N, S, E, W chỉ các hướng Bắc, Nam, Đông và Tây). Một chỉ dẫn robot là một chuỗi ký tự 'L', 'R', và 'F' tương ứng như sau:
Left: robot quay trái 90 độ và giữ nguyên vị trí tại điểm lưới hiện tại. Right: robot quay phải 90 độ và giữ nguyên vị trí tại điểm lưới hiện tại. Forward: robot tiến lên một điểm lưới theo phương hướng hiện tại và giữ nguyên hướng.
Phương Bắc tương ứng với phương từ điểm lưới (x,y) đến điểm lưới (x, y+1)
Vì lưới hình chữ nhật và có đường biên, một robot di chuyển ra ngoài rìa của lưới bị coi như mất tích vĩnh viễn. Tuy nhiên, những robot đã mất tích sẽ lưu lại một "mùi" ngăn cản những robot khác lặp lại sai lầm tương tự (nghĩa là bị rơi ra khỏi lưới ở vị trí mà trước đây đã có con bị rơi). Mùi này sẽ lưu lại ở vị trí lưới cuối cùng mà con robot trước đó đã đứng trước khi tiến thêm một bước nữa và lọt ra ngoài lưới. Chỉ dẫn để một con robot bị lọt ra ngoài lưới sẽ bị những con robot sau loại bỏ không nghe nữa.

INPUT

Dòng đầu là toạ độ góc trên bên phải của lưới chữ nhật, toạ độ góc dưới bên trái được giả định đặt ở điểm (0,0).
Còn lại sẽ là dãy các vị trí của robot và các chỉ dẫn (mỗi robot 2 dòng). Vị trí gồm 2 số nguyên cho biết toạ độ xuất phát của robot và hướng (N, S, E, W), tất cả được viết trên cùng một dòng và phân tách nhau bởi dấu cách. Chỉ dẫn robot là một chuỗi các ký tự 'L', 'R', và 'F' được viết trên cùng một dòng. Mỗi robot được xử lý tuần tự, tức là sau khi thực hiện toàn bộ chỉ dẫn của robot này xong mới đến robot khác. File input kết thúc ký tự EOF. Có thể giả thiết rằng tất cả các vị trí khởi đầu của robot đều nằm trong vùng biên của lưới đã cho. Giá trị lớn nhất của bất kỳ một toạ độ nào là 50. Tất cả các chuỗi chỉ dẫn đều có độ dài ít hơn 100 ký tự.

OUTPUT

Với mỗi vị trí/chỉ dẫn của một robot trong file input, output sẽ chỉ ra vị trí lưới cuối cùng và hướng của robot đó. Nếu một robot rơi khỏi cạnh lưới thì máy tính in ra từ "LOST" phía sau vị trí và hướng của nó.

Sample Input	Sample Output
5 3 1 1 E RFRFRFRF 3 2 N FRRFLLFFRRFLL 0 3 W LLFFFLFLFL	1 1 E 3 3 N LOST 2 3 S

UVA280. ĐỈNH

Viết chương trình tìm kiếm trên một đồ thị có hướng các đỉnh không thể đến được từ một đỉnh khởi đầu cho trước.
Một đồ thị có hướng gồm n đỉnh, $1 \le n \le 100$, được đánh số liên tiếp từ 1 đến n và một loạt các cạnh $p \rightarrow q$ nối cặp đỉnh theo hướng từ p đến q. Từ một đỉnh p có thể tới được đỉnh r nếu có một cạnh $p \rightarrow r$ hoặc tồn tại đỉnh q sao cho từ p có thể đi tới q và từ q có thể đi tới r. Đỉnh r được coi là không thể xâm nhập được từ đỉnh p nếu từ p không đi tới được r.

INPUT

Input nhiều đồ thị có hướng và các nút khởi điểm. Với mỗi đồ thị, dòng đầu tiên là một số nguyên n. Đây là số đỉnh của đồ thị. Các dòng tiếp theo mỗi dòng là một tập các số nguyên. Kết thúc là một dòng chứa toàn số 0. Mỗi tập số nguyên là tập hợp các cạnh. Số nguyên đầu tiên trong tập, i, là đỉnh khởi điểm, các số nguyên tiếp theo, j…k, cho biết một loạt các cạnh $i \rightarrow j \ldots i \rightarrow k$, số nguyên cuối cùng của dòng luôn luôn là số 0. Mỗi đỉnh khởi đầu i, $1 \le i \le n$, sẽ xuất hiện một lần hoặc không xuất hiện. Sau mỗi định nghĩa đồ thị, là một dòng mang một danh sách các số nguyên. Số nguyên đầu tiên cho biết có bao nhiêu số nguyên đứng sau nó. Mỗi số nguyên tiếp theo biểu diễn một đỉnh khởi đầu mà chương trình của bạn kiểm tra. Các đồ thị tiếp đó theo đúng quy tắc trên lần lượt được đưa ra. Nếu không còn đồ thị nào nữa, dòng tiếp theo của file sẽ chỉ có số 0.

OUTPUT

Với mỗi điểm khởi đầu đã được kiểm tra, chương trình phải xác định tất cả các đỉnh không thể xâm nhập từ đỉnh khởi đầu đã cho. Mỗi danh sách được đặt trên một dòng, bắt đầu với số lượng các đỉnh không xâm nhập được và tiếp đó là số hiệu của các đỉnh này.

Sample Input	Sample Output
3 1 2 0 2 2 0 3 1 2 0 0 2 1 2 0	2 1 3 2 1 3

UVA824. HỆ THỐNG KIỂM TRA BỜ BIỂN

Năm 2222 sau Công Nguyên, cơ quan Không gian Trái Đất (ESA) đang chuẩn bị cho một sứ mệnh trên hành tinh Atlantis, hành tinh này mới được khám phá ra cách đây 1 thế kỷ. Hành tinh này được đặt theo tên của một huyền thoại cổ xưa về một lục địa đã biến mất vì đã hoàn toàn bị nhấn chìm dưới biển khơi bao la. Không có lục địa nào trên hành tinh này, nhưng bề mặt của nó hầu như được phủ kín bởi hàng triệu hòn đảo nhỏ, chưa được khám phá, những hòn đảo đã từng tồn tại trên khắp lục địa Atlantis. Một trong số những mục tiêu sứ mệnh này đặt ra là xây dựng một bản đồ hoàn chỉnh các hòn đảo của hành tinh. Khi không thể quan sát từ trên không, ESA quyết định đưa một nhóm các robot thăm dò với nhiệm vụ phải thực hiện là quan sát các bờ đảo. Ý tưởng là đặt một con robot ở một điểm trên bờ của một hòn đảo và nó sẽ tự mình khám phá hình trạng của toàn bộ bờ đảo này, sau đó quay lại điểm xuất phát và chuyển đến hòn đảo tiếp theo. Bạn tham gia nhóm lập trình phần mềm rà soát cho con robot này, và đã có một vài quyết định: bề mặt của hành tinh sẽ được phân thành các hình vuông có kích cỡ bằng nhau; ý định thật đơn giản, mỗi khu vực sẽ được xem như chỉ toàn là đất hoặc nước. Bởi thế, một robot sẽ luôn bắt đầu công việc của mình từ một hình vuông có biển ở bên tay phải; bởi vậy bãi biển sẽ được rà soát theo chiều kim đồng hồ. (xem Hình 1).

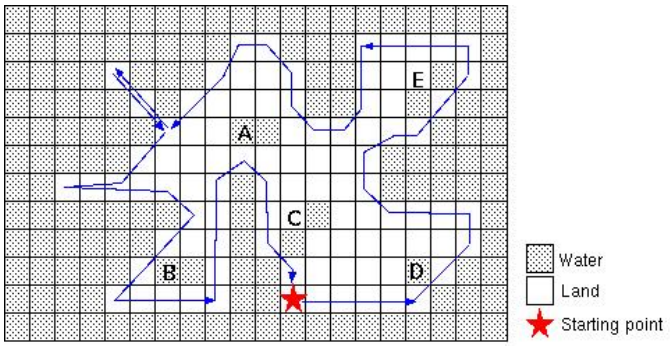


Figure 1

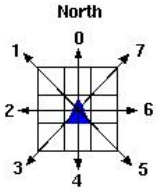


Figure 2

Con robot có hệ thống cảm giác bị giới hạn, hệ thống này có thể xác định kiểu bề mặt của 8 khu vực lân cận quanh khu vực nó đang đứng. Khả năng dịch chuyển cũng bị hạn chế: con robot chỉ có thể có khả năng quay đến 1 trong 8 hướng cố định (được mã hoá từ 0 đến 7, trong đó 0 là phía Bắc, xem hình 2) và chuyển đến vị trí ô hàng xóm ở trước nó. Mỗi lần robot di chuyển đến 1 vị trí mới, hệ thống cảm giác lại tự động thu lại tình hình vị trí mới. Có các hồ (ví dụ như A, B, C, D và E trong hình 1), nhưng robot không phí thời gian với chúng: mục tiêu chỉ là theo dõi bờ đảo mà thôi. Cho trước vị trí hiện tại và hướng của robot, và một bức tranh về thế giới bao quanh, chọn hướng cho bước đi tiếp theo. Lưu ý rằng người ta không yêu cầu bạn phải lập trình toàn bộ phần mềm theo dõi bờ biển; chỉ cần lập trình một phần thôi. Phải tính toán hướng bằng cách gọi lặp chương trình, giả sử thế giới bao quanh không thay đổi gì, việc gọi lặp chương trình sẽ cho phép con robot rà soát bờ biển.
Chương trình phải có khả năng xử lý nhiều tình huống. Mỗi tình huống gồm vị trí hiện tại của robot và hướng cũng như khung cảnh. Hình 3 minh hoạ 3 tình huống giả thiết và kết quả dự định: hướng của lần di chuyển tiếp theo.

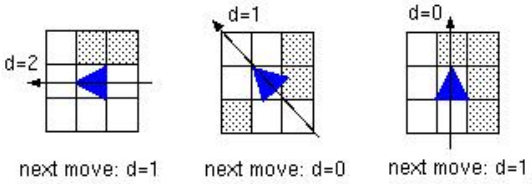


Figure 3

INPUT

File input cung cấp nhiều tình huống. Mỗi tình huống gồm 9 dòng như sau: Dòng 1: x y d, trong đó x và y là toạ độ (luôn dương) vị trí hiện tại của robot, và d là hướng hiện tại của robot, d là số nguyên thoả mãn điều kiện $0 \le d \le 7$ (xem hình 2); 8 dòng tiếp theo: xi yi si, trong đó si là số biểu diễn kiểu bề mặt của vị trí lân cận (xi, yi); nếu bề mặt là đất thì biểu diễn bằng 1, nước thì biểu diễn bằng 0. Các giá trị tiếp sau in trên cùng một dòng, được phân tách nhau bởi một hoặc nhiều dấu cách. Ở tình huống cuối cùng, sau dữ liệu in số -1.

OUTPUT

Với mỗi tình huống cho trước, chương trình phải xuất ra hướng chuyển động tiếp theo của robot. Kết quả của các tình huống kế tiếp sẽ được in ra trên các dòng kế tiếp.

Sample Input	Sample Output
22 25 2 22 26 0 21 26 1 21 25 1	1 0 1

21 24 1 22 24 1 23 24 1 23 25 1 23 26 0 21 26 1 21 27 1 20 27 1 20 26 1 20 25 0 21 25 1 22 25 1 22 26 0 22 27 0 21 27 0 21 28 0 20 28 1 20 27 1 20 26 1 21 26 1 22 26 0 22 27 0 22 28 0 -1	

UVA1205. TÔ MÀU CÂY

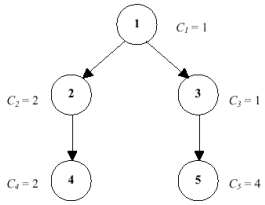


Figure-1. A tree with five nodes

Bob rất thích cấu trúc dữ liệu kiểu cây. Cây là đồ thị có hướng, có một nút đặc biệt được lựa chọn làm gốc của cây, và chỉ tồn tại một đường đi duy nhất từ gốc đến mọi nút khác trong cây.

Bob định tô màu tất cả các nút trong cây. Cây này có ẩ nút, các nút được đánh số từ 1 đến ẩ . Giả sử việc tô màu cho một nút cần 1 đơn vị thời gian, và sau khi tô xong, Bob được phép chuyển màu khác. Thêm vào đó, Bob chỉ được phép tô màu cho một nút sau khi đã tô màu xong cho nút cha của nó. Rõ ràng là Bob chỉ được phép tô màu cho nút gốc trong lần tô đầu tiên. Mỗi nút có một hệ số giá trị màu là Ci. Giá trị màu của từng nút phụ thuộc vào cả Ci và thời gian Bob cần để tô xong màu cho nút đó. Khởi đầu, thời gian được đặt là 0. ẩ ếu thời gian hoàn thành việc tô màu cho một nút là Fi, thì giá trị màu của một nút sẽ là Ci * Fi. Ví dụ, một cây có 5 nút như trong Hình 1. Hệ số giá trị màu của từng nút là 1, 2, 1, 2 và 4. Bob có thể tô màu cây theo thứ tự 1, 3, 5, 2, 4, với tổng giá trị màu tối thiểu là 33. Cho trước một cây và hệ số giá trị màu của từng nút, hãy giúp Bob tìm tổng giá trị màu tối thiểu có thể được khi tô màu tất cả các nút.

INPUT

Input gồm nhiều test. Dòng đầu tiên của mỗi test có 2 số nguyên ẩ và R ($1 \le$ ẩ ≤ 1000, $1 \le R \le$ ẩ), trong đó ẩ là số nút của cây và R là số hiệu của nút gốc. Dòng thứ 2 có ẩ số nguyên, số nguyên thứ i là hệ số giá trị màu của nút I, tức là giá trị Ci ($1 \le Ci \le 500$). Trên ẩ -1 dòng tiếp theo, mỗi dòng có 2 giá trị V1 và V2, viết cách nhau bởi dấu cách, chính là số hiệu nút của 2 nút đánh dấu một cạnh trong cây, có nghĩa là V1 là cha của V2. Không một cạnh nào được liệt kê 2 lần, và tất cả các cạnh đều được liệt kê rõ. Trường hợp test ẩ = 0 và R = 0 sẽ không được xử lý, và là trường hợp kết thúc của input.

OUTPUT

Với mỗi test, xuất ra tổng giá trị tối thiểu

Sample Input	Sample Output
5 1 1 2 1 2 4 1 2 1 3 2 4 3 5 0 0	33

UVA10113. TỶ GIÁ GIAO DỊCH

Dùng tiền để chi trả hàng hoá và dịch vụ thường khiến cho cuộc sống dễ dàng hơn, nhưng đôi khi người ta lại thích trao đổi hàng hoá trực tiếp mà không dùng tiền. Để đảm bảo giá cả phù hợp, người giao dịch đặt tỷ giá giao dịch giữa các món đồ. Tỷ giá giao dịch giữa hai món đồ A và B được biểu diễn bằng 2 số nguyên dương m và n, và ta nói m món đồ A đáng giá n món đồ B. Ví dụ, 2 cái lò gốm có giá bằng 3 cái tủ lạnh. (Theo toán học, 1 cái lò gốm có giá bằng 1,5 cái tủ lạnh nhưng vì quá khó để mua nửa cái tủ lạnh, nên tỷ giá giao dịch luôn dùng số nguyên để biểu diễn). Hãy viết một chương trình để tính toán tỷ giá giao dịch giữa hai món đồ bất kỳ với một dãy các tỷ giá giao dịch cho trước. File input có một hoặc nhiều lệnh, cuối cùng là một dòng bắt đầu bằng một dấu chấm câu báo hiệu hết file. Mỗi lệnh nằm trên một dòng và có thể là lệnh xác nhận hoặc lệnh truy vấn. Một lệnh xác nhận bắt đầu bằng dấu chấm than và có khuôn dạng như sau: **! m itema = n itemb** trong đó itema và itemb là tên của 2 món đồ khác nhau, m và n là hai số nguyên dương nhỏ hơn 100. Câu lệnh này có nghĩa là m itema bằng với n itemb. Một lệnh truy vấn bắt đầu bằng một dấu hỏi chấm, có khuôn dạng như sau: **? itema = itemb** và hỏi tỷ giá giao dịch giữa itema và itemb, trong đó itema và itemb là 2 món đồ khác nhau, cả hai đều đã được nêu ra trong các câu lệnh xác nhận trước đó (mặc dù không nhất thiết phải có mặt trong cùng một lệnh xác nhận). Với mỗi câu truy vấn, kết quả của tỷ giá giao dịch giữa itema và itemb dựa trên tất cả các câu lệnh xác nhận đã có cho đến thời điểm đó. Tỷ giá giao dịch phải là các số nguyên và phải được tối giản. ấ ếu không có tỷ giá giao dịch nào được xác định cho đến thời điểm này thì dùng dấu hỏi chấm thay cho các số nguyên. Khuôn dạng các kết quả sẽ được hiển thị như trong ví dụ dưới đây.
Lưu ý:
Tên của các món đồ sẽ có độ dài tối đa là 20 và chỉ gồm các ký tự thường.
Tên của món đồ chỉ ở dạng số ít (không dùng số nhiều)
Có tối đa 60 món đồ khác nhau.
Có tối đa 1 câu lệnh xác nhận cho hai món đồ bất kỳ nào.
Không có những câu lệnh xác nhận trái ngược nhau. Ví dụ, "2 pig = 1 cow", "2 cow = 1 horse", "2 horse = 3 pig" là các câu xác nhận trái ngược nhau.
Các câu lệnh xác nhận không cần thiết phải tối giản, nhưng kết quả thì phải tối giản.
Mặc dù các câu lệnh xác nhận sử dụng các số < 100, các câu lệnh truy vấn có thể nhận kết quả là những số lớn hơn nhưng không vượt quá 10.000 khi đã tối giản.

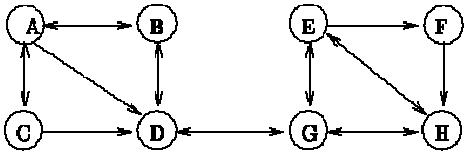
Example input:	Example output:
! 6 shirt = 15 sock ! 47 underwear = 9 pant ? sock = shirt ? shirt = pant ! 2 sock = 1 underwear ? pant = shirt .	5 sock = 2 shirt ? shirt = ? pant 45 pant = 188 shirt

UVA168. THESEUS AND MINOTAUR

Các bạn ai đã từng nghiên cứu về truyền thuyết cổ đại chắc hẳn đều biết đến huyền thoại về Theseus và Minotaur. Đây là câu chuyện hư cấu về một con quái vật đầu bò, một mê cung dưới lòng đất với toàn những lối đi quanh co khúc khuỷu, một thiếu nữ thất tình và những quả bóng bằng lụa. Chúng ta sẽ cùng nhau khám phá sự thực của câu chuyện. Mê cung thực ra gồm một loạt hang động, được nối với nhau bởi những lối đi thẳng, mà trong số đó có một

số lối chỉ có thể đi theo một hướng. Để đặt bẫy Minotaur, Theseus lén mang vào mê cung một số cây nến, bởi chàng biết rằng Minotaur rất sợ ánh sáng. Theseus thơ thẩn đi lòng vòng cho đến khi chàng nghe thấy tiếng Minotaur đang lọ mọ dọc theo đường hầm. Chàng bèn thắp một cây nến và đuổi bắt Minotaur. Minotaur rút lui vào hang động và chạy trốn theo một lối khác. Theseus đuổi theo, cho đến khi chàng tới hang động thứ k tính từ lúc chàng thắp nến. Ở đây, chàng có đủ thời gian để đặt cây nến đã thắp trước đó vào hang, thắp một cây nến khác, và tiếp tục cuộc truy đuổi. Tiếp tục cuộc truy đuổi khi cây nến đã được đặt tại mỗi cái hang thứ k khi đi qua, nhằm hạn chế đường chạy của Minotaur. Mỗi khi vào một hang động, Minotaur sẽ kiểm tra tất cả các lối thoát và chạy đến cái hang đầu tiên không có nến. (Hãy nhớ rằng Theseus mang theo một cây nến được thắp sáng, và Minotaur sẽ không quay lại đường hầm mà dẫn nó đến hang động đang đứng). Cuối cùng, Minotaur bị mắc bẫy, Theseus đã đánh bại nó.

Hãy xem một mê cung ví dụ dưới đây, trong trường hợp này, Minotaur sẽ kiểm tra lối thoát từ một hang động theo thứ tự từ điển:



Giả định Theseus trong hang C, chàng nghe tiếng Minotaur đang ở trong hang A, trong trường hợp này k = 3. Ông thắp một cây nến rồi đuổi theo Minotaur, qua hang A, B, D (đặt một cây nến), G, E, F (đặt một cây khác), H, E, G (một cây khác), H, E (Minotaur mắc bẫy). Viết một chương trình mô phỏng cuộc truy đuổi của Theseus. Cách mô tả mê cung sẽ định rõ mỗi hang động bằng một chữ cái in hoa, sau đó là các hang động Minotaur có thể chạy đến. Tiếp đó là các hang động mà Minotaur và Theseus đang đứng, cuối cùng là giá trị k.

INPUT

Input có nhiều dòng, mỗi dòng sẽ giống như mô tả. Không có dòng nào chứa nhiều hơn 255 kí tự. Input kết thúc bằng một dòng chứa dấu #.

OUTPUT

Output sẽ chứa một dòng với mỗi mê cung trong input, biểu thị các hang động được thắp nến, theo thứ tự những cây nến được đặt, và hang động Minotaur sập bẫy. Dưới đây là ví dụ.

Sample input	Sample output
A:BCD;B:AD;D:BG;F:H;G:DEH;E:FGH;H:EG;C:AD. A C 3 #	D F G /E

UVA11831. ĐUA ROBOT

Một trong những môn thể thao được ưa thích tại RoboLand là Robots Rally - cuộc đua được diễn ra trên một trường đua hình chữ nhật gồm các ô vuông với ả hàng và M cột. Một số ô trống, một số ô chứa sticker cho World Footbal Cup Album và một số dùng để đặt cột chống. Trong cuộc đua, robot có thể chiếm bất cứ ô nào trong trường đua, ngoại trừ những ô được dùng để đặt cột chống. Đường đi của robot trong trường đua được xác định bởi một chuỗi các lệnh. Mỗi lệnh được biểu thị bởi một trong các kí tự: `D', `E' và `F', theo thứ tự, là "rẽ phải", "rẽ trái" và "đi thẳng". Robots bắt đầu cuộc đua ở một số vị trí trong trường đua, và chúng sẽ theo sát các lệnh đã được đưa ra (vì dù sao, chúng cũng là robot!). Mỗi khi robot chiếm một ô chứa Cup sticker, robot sẽ nhặt nó lên. Sticker không bị thay thế, mỗi sticker chỉ có thể được nhặt một lần. Khi cố di chuyển vào ô chứa cột chống hoặc rời khỏi trường đua, robot sẽ ngừng đi và sẽ đứng yên ở ô trước đó, với hướng như trước. Cho bản đồ trường đua, mô tả vị trí các cột chống và các sctickers, và chuỗi các lệnh cho robot, bạn hãy viết một chương trình xác định số lượng stickers mà robot nhặt được.

INPUT

Input chứa nhiều test cases. Dòng đầu mỗi test case chứa 3 số nguyên ả , M và S (1≤ả , M≤100, 1≤S≤5 x104), cách nhau bởi khoảng trống, lần lượt biểu thị số hàng, số cột của trường đua và số lệnh cho robot. Mỗi dòng trong ả dòng kế tiếp mô tả một hàng của trường đua bằng một xâu M kí tự. Dòng đầu trong mô tả trường đua là hướng Bắc, cột đầu là hướng Tây. Mỗi cell trong trường đua được mô tả bởi một trong các kí tự sau:

- `.' – cell bình thường;
- `*' – cell chứa sticker;
- `#' -- cell dùng để đặt cột chống;
- `ả ', `S', `L', `O' – ô nơi robot bắt đầu cuộc đua (chỉ có một robot trong trường đua). Các kí tự `ả ', `S', `L', `O' biểu thị hướng của robot (lần lượt là Bắc, ả am, Đông, Tây).

Dòng cuối input chứa chuỗi S kí tự `D', `E' và `F', biểu thị các lệnh cho robot. Test case cuối chứa 3 số 0, ngăn cách nhau bởi khoảng trống.

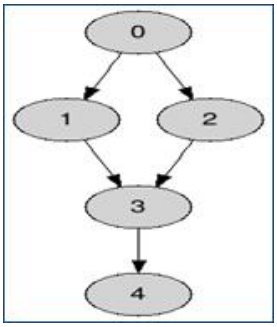
OUTPUT

Với mỗi cuộc đua được mô tả trong input, chương trình của bạn sẽ in một dòng chứa một số nguyên biểu thị số stickers mà robot nhặt được trong cuộc đua.

Sample Input	Sample Output
3 3 2 *** *N* *** DE 4 4 5 ...# *#O. *.*. *.#. FFEFF 10 10 20 ....*..... ....*.. .....*.... ..*.#..... ...#N.*..* ...*..... .... FDFFFFFFEEFFFFFFEFDF 0 0 0	0 1 3

UVA11902. NÚT CHI PHỐI

Trong lý thuyết đồ thị, nút X chi phối nút Y nếu mọi đường dẫn từ nút xuất phát đến Y đều đi qua X. ả ếu không có đường đi nào từ nút xuất phát đến Y thì Y không bị nút nào chi phối. Theo định nghĩa, mọi nút mà có thể đi tới từ nút xuất phát tự chi phối mình. Trong bài toán này, bạn có một đồ thị có hướng, và với mỗi nút trong đồ thị, bạn phải tìm tất cả các nút chi phối nó, với nút thứ 0 là nút xuất phát. Ví dụ trong đồ thị bên , 3 chi phối 4 vì tất cả các đường đi từ 0 đến 4 đều phải qua 3. 1 không chi phối 3 vì có đường đi 0-2-3 không chứa 1.



INPUT

Dòng đầu input chứa T (≤ 100) xác định số test. Mỗi test bắt đầu với một số nguyên ả (0 < ả < 100) biểu thị số nút trong đồ thị. ả dòng tiếp theo, mỗi dòng chứa ả số nguyên. ả ếu số nguyên thứ j (bắt đầu từ 0) của dòng thứ i (bắt đầu từ 0) là 1 thì có một cạnh nối 2 nút i và j, là 0 thì không có cạnh nào.

OUTPUT

Với mỗi case, in ra số thứ tự case trước. Sau đó in ra 2ả +1 dòng tóm tắt quan hệ chi phối nhau giữa mỗi cặp 2 nút. ả ếu nút A chi phối nút B, in 'Y' tại ô (A, B), còn nếu không thì in 'ả '. Ô (A, B) là ô nằm tại hàng thứ Ath và cột thứ Bth . Quanh output là các dấu |, + và – để làm cho output thêm dễ hiểu. Xem thử ví dụ để biết mẫu output chuẩn.

Sample Input	Sample Output
2 5 0 1 1 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 1	Case 1: +---------+ |Y|Y|Y|Y|Y| +---------+ |N|Y|N|N|N| +---------+ |N|N|Y|N|N| +---------+ |N|N|N|Y|Y| +---------+ |N|N|N|N|Y| +---------+ Case 2: +-+ |Y| +-+

UVA10678. GIÁM SÁT AMAZON

Một mạng lưới tự quản, thu nhận dữ liệu, chạy bằng năng lượng pin được lắp đặt để giám sát thời tiết vùng Amazon. Một trạm phát lệnh có thể bắt đầu truyền các chỉ dẫn đến các trạm điều khiển để chúng thay đổi các tham số. Để tránh làm pin chóng hết, mỗi trạm (gồm cả trạm phát lệnh chỉ có thể truyền đến 2 trạm khác. Đích gửi đến của một trạm là hai trạm gần nó nhất. Trong trường hợp có nhiều trạm gần nhất thì tiêu chí đầu tiên là chọn hướng tây nhất (bên trái nhất của bản đồ) và tiêu chí thứ hai là chọn hướng nam nhất (thấp nhất trên bản đồ).
Chính quyền bang Amazon đặt bạn viết một chương trình quyết định xem liệu, cho trước vị trí của từng trạm, các thông điệp có thể đến tất cả các trạm không.

INPUT

Đầu vào bao gồm một số nguyên ả , sau đó là cặp số nguyên Xi, Yi, chỉ toạ độ vị trí của từng trạm. Cặp toạ độ đầu tiên xác định vị trí của trạm phát lệnh, ả -1 cặp tiếp theo chỉ toạ độ của các trạm khác. Các điều kiện ràng buộc sẽ là -20 ≤ Xi, Yi ≤ 20, và 1 ≤ ả ≤ 1000. Kết thúc với ả = 0.

OUTPUT

Với mỗi biểu thức đã cho, output sẽ hiển thị một dòng thông báo tất cả các trạm có nhận được thông điệp truyền đến hay không (xem output mẫu để có khuôn dạng đúng).

Sample input	Sample output
4 1 0 0 1 -1 0 0 -1 8	All stations are reachable. All stations are reachable. There are stations that are unreachable.

1 0 1 1 0 1 -1 1 -1 0 -1 -1 0 -1 1 -1 6 0 3 0 4 1 3 -1 3 -1 -4 -2 -5 0	

UVA11906. HIỆP SĨ TRONG MẠNG LƯỚI CHIẾN TRẬN

ầ gày xửa ngày xưa, có một hiệp sĩ đang chuẩn bị cho trận đánh lớn ở GridLand. GridLand là lưới các ô vuông có chiều ngang R và chiều dọc C. Chàng hiệp sĩ của chúng ta luôn có thể thực hiện bước di chuyển (M, ầ), tức là chàng có thể di chuyển M ô vuông theo chiều ngang và ầ ô vuông theo chiều dọc hoặc chàng có thể di chuyển M ô vuông theo chiều dọc và ầ ô vuông theo chiều ngang trong một nước đi. ầ ói cách khác, chàng có thể nhảy từ ô (a,b) đến ô (c,d) khi và chỉ khi (|a - c| = M và |b - d| = ầ) hoặc (|a - c| = ầ và |b - d| = M). Tuy nhiên, một số ô vuông trong vùng chiến trận lại bị ngập nước. Để nhảy thành công từ một ô này sang một ô khác, thì điều kiện đặt ra là không có ô nào bị ngập nước. Chàng hiệp sĩ muốn đi duyệt trận địa để kiểm tra xem mọi thứ đã sẵn sàng chưa. Chàng sẽ làm như sau:

a) Chàng bắt đầu và kết thúc chuyến kiểm tra ở ô (0,0) nhưng cố gắng thăm nhiều ô khác nhất có thể được.

b) Với mỗi ô s_i, chàng đếm k_i là số các ô vuông mà từ đó chàng có thể nhảy một bước tới s_i (thoả mãn điều kiện nhảy). Sau đó chàng đánh dấu ô vuông này là ô chẵn nếu k_i là chẵn hoặc lẻ nếu k_i là lẻ. ầ ững ô nào mà chàng không thể thăm được thì để nguyên không đánh dấu.

c) Sau khi quay lại ô (0,0) chàng đếm số ô lẻ và chẵn đã được đánh dấu. Chàng có thể thăm 1 ô hơn 1 lần.

Bạn, với vai trò là cố vấn của chàng hiệp sĩ, khuyên chàng nên viết một chương trình thay vì phải trực tiếp đi đến tất cả các ô vuông. Vậy là chàng hiệp sĩ cầu cứu bạn hãy giúp chàng. Chàng sẽ kiểm tra kết quả của bạn vào cuối cuộc viếng thăm của mình.

INPUT

Dòng đầu tiên là một số T (≤ 50) chỉ số lượng test.

Với mỗi trường hợp bắt đầu bằng 4 số nguyên R, C, M, ầ ($1 < R, C \leq 100$, $0 \leq M$, ầ ≤ 50, $M +$ ầ > 0). Dòng tiếp theo là số nguyên W ($0 \leq W < R * C$), đây là số lượng các ô bị ngập nước. Trên W dòng tiếp theo, mỗi dòng sẽ có một cặp bốn số nguyên x_i, y_i ($0 \leq x_i < R$, $0 \leq y_i < C$, $x_i + y_i > 0$).

OUTPUT

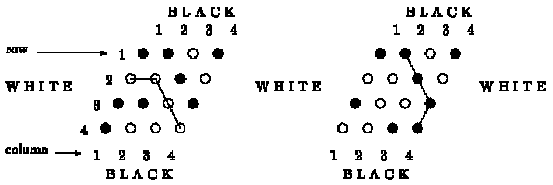
Với từng trường hợp, in ra số hiệu của trường hợp và số các ô lẻ và chẵn đã được đánh dấu.

Sample Input	Output for Sample Input
2 3 3 2 1 0 4 4 1 2 2 3 3 1 1	Case 1: 8 0 Case 2: 4 10

02. FLOOD FILL

UVA260. Il Gioco dell'X

Trò chơi "Il Gioco dell' X" được chơi trên bàn cờ ả xả (ả ≥ 2). Mục tiêu của cả 2 bên Đen và Trắng, là thâm nhập được vào bên đối phương bằng cách lần lượt đặt những con cờ của mình lên bàn cờ sao cho những quân cờ cùng màu đặt liên tiếp cạnh nhau sẽ kéo dài thành một đường chạy từ bên nọ sang bên kia. Bàn cờ ả xả không phải là bàn cờ vuông mà giống hình kim cương hơn. Chỉ ra những vùng trên bàn cờ theo hàng i và cột j bằng cặp (i,j) (1 ≤ i, j ≤ ả). Hàng xóm của (i, j) là: (i-1, j-1), (i-1, j), (i, j-1), (i, j+1), (i+1, j) , (i+1 , j+1) miễn là những hàng xóm này không nằm ngoài khu vực bàn cờ. Bên Đen cố gắng nối liền dòng 1 với dòng ả trong khi bên Trắng cố gắng nối liền từ cột 1 đến cột ả . Về mặt Toán học có thể chứng minh trận đấu này không thể hoà (phải có người thắng). Tuy nhiên, rất khó để biết được ai là người chiến thắng, cần có sự trợ giúp của máy tính để đoán định kết quả. Trong ví dụ 1, bên Trắng thắng, và trong ví dụ 2 chiến thắng lại thuộc về bên Đen.



Formatted: Space Before: 0 pt, After: 0 pt

INPUT

Đầu vào là một file text cho biết số ván chơi. Mỗi ván chơi được cho như sau: một dòng chứa số nguyên ả , là số cột. (2 ≤ ả ≤ 200). Tiếp theo là ả dòng nữa, mỗi dòng gồm ả các ký tự 'b' và 'w' chỉ tương ứng các quân Đen và Trắng. Số quân đen và trắng chênh nhau nhiều nhất là 1. Lưu ý rằng mọi vị trí trên bàn cờ đều được đặt kín các quân cờ. Danh sách các ván cờ kết thúc bằng một số 0 nằm trên một dòng riêng.

OUTPUT

Kết quả cũng là một file text, kết quả thắng thua của mỗi ván cờ sẽ nằm trên một dòng riêng. Dòng này sẽ có số hiệu của ván chơi (bắt đầu tính từ ván 1) và sau đó là ký tự 'B' nếu quân Đen thắng, 'W' nếu quân Trắng thắng.

Sample Input	Sample Output
4 bbwb wwbw bbwb bwww 4 bbwb wwbw bwwb wwbb 0	1 W 2 B

UVA352. CUỘC CHIẾN MÙA

Dân cư ở thị trấn Hổ và Voi tham gia vào một cuộc chiến mùa. Tháng trước, thị trấn Voi đưa ra và vận hành thành công kính thiên văn do thám có tên là Bumble Scope. Mục đích của chiếc kính này là đếm số Đại Bàng Chiến Đấu của thị trấn Hổ. Tuy nhiên, nó có hai vấn đề do việc quản lý kém. Ống kính chính bị hỏng do lỗi chắn một phần hình ảnh và cơ chế chỉnh tiêu điểm của kính gặp trục trặc khiến cho các ảnh bị biến đổi kích thước và độ sắc nét.
Các lập trình viên phải khắc phục những hư hỏng của Bumble Scope lại đang bị cầm giữ trong khách sạn Programming Contest ở Alaland bởi những thành viên voi đội lốt Hổ. ả hững hình ảnh hỏng hóc của Bumble Scope được lưu trong một file có tên là Bumble.in. Ảnh có hình vuông và mỗi pixel trong ảnh có thể nhận giá trị 0 hay 1. Chiếc camera độc nhất của Bumble Scope (viết tắt là BSC) báo cáo rằng tại vị trí pixel bằng 1 chính là một phần hay

toàn bộ Đại Bàng Chiến Đấu đang đậu và 0 là bất kỳ đối tượng nào khác, bao gồm cả lỗi. Các lập trình viên có thể giả định những điều sau:

- Một Đại Bàng Chiến Đấu được biểu diễn bởi ít nhất là một con số 1.
- Các pixel kề với cùng một ô – nếu tất cả đều được biểu diễn bởi số số 1, định hình một Đại Bàng Chiến Đấu. Một ảnh rất lớn của một Đại Bàng Chiến Đấu có thể chứa tất cả các pixel.
- Hình ảnh các Đại Bàng Chiến Đấu không bị chồng chéo lên nhau. Giả thiết này có khả năng không chính xác, nhưng các lập trình viên vẫn đánh liều cho như thế là đúng.
- Không có tính năng cong. Các pixel ở đáy không kề với các pixel ở đầu và các pixel bên trái không kề với bên phải (dĩ nhiên, trừ phi chỉ có 2 dòng hay 2 cột)

INPUT AND OUTPUT

Viết một chương trình đọc ảnh từ file input vào (file text), đếm chính xác số Đại Bàng Chiến Đấu trong ảnh và in số thứ tự của ảnh cũng như số Đại Bàng Chiến Đấu đếm được của ảnh đó trên một dòng riêng trong file output (cũng là file text). In như khuôn dạng output mẫu dưới đây. Thực hiện việc đó với từng ảnh trong file input. Mỗi ảnh sẽ có con số đầu tiên chỉ ra kích thước vuông của nó. Kích thước này không vượt quá 25.

Sample Input	Sample Output
6 100100 001010 000000 110000 111000 010100 8 01100101 01000001 00011000 00000010 11000011 10100010 10000001 01100000	Image number 1 contains 3 war eagles. Image number 2 contains 6 war eagles.

Formatted Table

UVA572. MỎ DẦU

Công ty khảo sát địa chất GeoSurvComp chịu trách nhiệm thăm dò những mỏ dầu dưới lòng đất. GeoSurvComp thăm dò trong một khu vực rộng lớn hình chữ nhật và dựng một mạng lưới chia khu vực này ra thành nhiều ô vuông nhỏ. Sau đó công ty sẽ phân tích từng lô riêng biệt, sử dụng thiết bị cảm ứng để xét xem liệu ô đó có chứa dầu không. Ô chứa dầu được gọi là túi. ấ ếu hai túi ở cạnh nhau thì chúng cùng thuộc một mỏ dầu. Các mỏ dầu có thể rất lớn và bao gồm nhiều túi. ấ hiệm vụ của bạn là xét xem có bao nhiêu mỏ dầu khác nhau có trong lưới.

INPUT

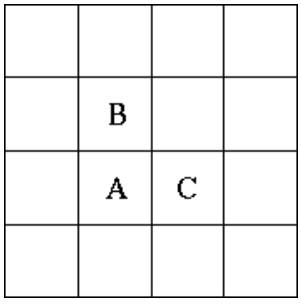
File input có thể có 1 hoặc nhiều hơn một lưới. Mỗi lưới bắt đầu bằng một dòng có 2 giá trị m và n, tương ứng với số dòng và số cột trong lưới, viết cách nhau bởi dấu cách. ấ ếu m = 0 thì ta biết rằng đã hết file, còn lại $1 \le m \le 100$ và $1 \le n \le 100$. Tiếp theo là m dòng, mỗi dòng có n ký tự (không kể ký tự hết dòng). Mỗi ký tự tương ứng với một ô và ký tự '*' biểu thị việc không phát hiện có dầu và '@' là có túi dầu bên dưới.

OUTPUT

Với mỗi lưới, xuất ra số mỏ dầu đã thăm dò được. Hai túi khác nhau được xem là cùng thuộc m mỏ dầu nếu chúng kề nhau theo chiều dọc, ngang hoặc chéo. Một mỏ dầu sẽ có không quá 100 túi dầu.

Sample Input	Sample Output
1 1 * 3 5 *@*@* **@** *@*@* 1 8 @@****@* 5 5 ****@ *@@*@ *@**@ @@@*@ @@**@ 0 0	0 1 2 2

UVA657. BÚT SA GÀ CHẾT



InterGames là một công ty hoạt động trong lĩnh vực công nghệ cao, đặc biệt là phát triển các công nghệ cho phép người dùng chơi game qua Internet. Một báo cáo phân tích thị trường cảnh báo họ về một thực tế là các game cơ hội rất thông dụng trong những khách hàng tiềm năng của họ. Đó có thể là monolopy, ludo hay backgammon, hầu hết các trò này có liên quan đến việc gieo xúc xắc ở một vài pha của trò chơi. Dĩ nhiên, người chơi không được phép gieo xúc xắc và nhập kết quả vào máy tính vì rất dễ gian lận. Bởi thế, thay vào đó, InterGames quyết định cung cấp cho người dùng của họ một máy camera để chụp ảnh con xúc xắc đã gieo, phân tích bức ảnh và sau đó chuyển kết quả gieo một cách tự động. Để làm được điều này, họ cần một chương trình xác định số điểm trên xúc xắc với một ảnh cho trước chứa nhiều xúc xắc. Chúng ta giả thiết về ảnh input. Các ảnh này chỉ chứa 3 giá trị pixel khác nhau: nền, mặt xúc xắc và chấm trên mặt con xúc xắc đó. Ta xét hai pixel là kề nhau nếu giữa chúng có chung một cạnh (chung góc không được tính). Trong hình dưới đây, pixel A và B kề nhau, nhưng B và C thì không. Tập S các pixel là liên thông nếu với mọi cặp pixel (a,b) đều nằm trong S, có một chuỗi $a_1, a_2, ..., a_k$ thuộc S, sao cho $a = a_1$, $b = a_k$ và a_i và a_{i+1} kề nhau, $1 \le i < k$. Chúng ta coi tất cả các tập liên thông tối đa của các pixel không phải là nền là xúc xắc. Khái niệm "liên thông tối đa" có nghĩa là bạn không thể thêm bất cứ một pixel không phải là nền vào tập mà không vi phạm điều kiện "liên thông". Tương tự như vậy, xét tất cả các tập liên thông gồm các chấm dạng điểm là một điểm.

INPUT

Đầu vào bao gồm các ảnh chụp nhiều lần gieo xúc xắc. Mỗi ảnh gồm các thông tin sau: bắt đầu là một dòng chứa 2 con số w và h, là chiều dài và chiều cao của ảnh. Các giá trị này phải thoả mãn điều kiện: $5 \le w, h \le 50$. h dòng tiếp theo mỗi dòng chứa w ký tự. Các ký tự sẽ là '.' nếu là pixel nền, '*' nếu là pixel xúc xắc và 'X' nếu là pixel chấm trên xúc xắc. Xúc xắc có thể có kích thước khác nhau và không hoàn toàn vuông do độ nhòe quang học. Ảnh sẽ gồm ít nhất một xúc xắc và số các chấm trên xúc xắc từ 1 cho đến 6. Kết thúc là ảnh có thông số w = h = 0, ảnh này sẽ không được xử lý.

OUTPUT

Với mỗi lần gieo xúc xắc, đầu tiên là in ra lần gieo thứ bao nhiêu. Sau đó in tiếp số điểm trên mặt của con xúc xắc trong bức ảnh, được sắp xếp theo thứ tự tăng dần. In một dòng trắng sau mỗi test.

Formatted: Subscript
Formatted: Subscript
Formatted: Subscript
Formatted: Subscript
Formatted: Subscript
Formatted: Subscript
Formatted: Subscript
Formatted: Font: 10 pt, Font color: Text 1
Formatted: Font: 10 pt, Font color: Text 1

Sample Input	Sample Output
30 15 ....*..... ...*****.....****..... ...*X***.....**X***..... ...*****....***X**..... ...***X*.....****..... ...*****.....*..... ....***.....******..... .....**X****.....*X**X*..... .....*******.....******..... .....****X**.....*X**X*..... .....***.....******..... ..... 0 0	Throw 1 1 2 2 4

UVA776. NHỮNG CHÚ KHỈ TRONG RỪNG

Hãy tưởng tượng có một khu rừng, cây cối trong rừng mọc trong một mạng lưới chữ nhật hữu hạn hai chiều. Tại mỗi vị trí chỉ có một cây mọc và nó có thể là một trong số n loại. Mỗi loại được gán bởi một ký tự duy nhất (ví dụ {A, B, C,...}). Hai cây cùng loại được xem như kề nhau nếu sự khác biệt lớn nhất giữa toạ độ của chúng chỉ là 1. Các gia đình khỉ (đúng hơn là các loài khỉ) được thả vào khu rừng. Mỗi gia đình sẽ chiếm tất cả các cây kề nhau của cùng một loài cây miễn sao khu vực này chưa bị gia đình khỉ khác chiếm mất. ầ hững gia đình khỉ được thả vào rừng từ trái qua phải, từ trên xuống dưới. Cho trước bản đồ khu rừng, xây dựng một bản đồ định vị các gia đình khỉ, bắt đầu từ 1 và lần lượt đánh số chúng.

INPUT

File input gồm các dòng của một ma trận chứa các ký tự, phân tách nhau bởi một dấu cách. Các ma trận tiếp theo (mỗi ma trận là một test khác nhau của bài toán) sẽ có cấu trúc tương tự, bắt đầu bằng một dòng với một ký tự đơn '%'.

OUTPUT

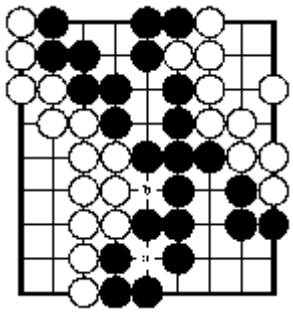
File output phải hiển thị các dòng số nguyên phân cách nhau bởi dấu cách như yêu cầu để căn các theo lề phải.
Đáp án cho mỗi trường hợp phải kết thúc bằng một dòng chứa ký tự '%'.

Sample Input	Sample Output
A B D E C C D F F W D D D D P W E W W W W % a A b B c d E t a a a a a c c t e f g h c a a t	1 2 3 4 5 5 3 6 6 7 3 3 3 3 8 7 9 7 7 7 7 % 1 2 3 4 5 6 7 8 1 1 1 1 1 5 5 8 9 10 11 12 5 1 1 8 %

Formatted: Font: (Default) Times New Roman

Formatted: Font: (Default) Times New Roman

UVA852. QUYẾT ĐỊNH THẮNG LỢI TRONG CỜ VÂY



Lịch sử của Cờ Vây bắt đầu từ 3000 năm trước và kể từ đó, luật chơi vẫn giữ nguyên. Trò chơi này nguyên là xuất phát từ Trung Quốc hoặc chính xác hơn là từ Himalaya. Ở vùng Viễn Đông, nơi khởi nguồn của trò chơi này, Cờ Vây vẫn rất được yêu thích và niềm yêu thích này đang lan nhanh ở Châu Âu và Mỹ. Ván Cờ Vây khởi đầu bằng một bàn cờ vuông trống không, mỗi người chơi có một số viên đá không giới hạn, một bên lấy đá đen, một bên lấy đá trắng làm quân. Mục tiêu của trò chơi là dùng quân của mình chiếm đất bằng cách bao kín quanh các khu vực trên bàn cờ. ả ếu có thể thì bắt quân của đối phương bằng cách bao vây kín chúng. Các bên chơi lần lượt đặt một quân vào vị trí cần vây mỗi lần đi, quân Đen được đi trước. Lưu ý rằng các viên đá được đặt ở giao điểm của các đường kẻ trên bàn cờ (không tính đường chéo). Một khi đã được đặt xuống, không được phép di chuyển viên đá dù có thể bị bắt, trong trường hợp đó các viên đá sẽ bị nhấc khỏi bàn cờ. Cuối ván (khi cả hai người đều hết nước đi) thì người chơi tính điểm trên số điểm trống trong khu đất của mình và mỗi quân tù binh họ có một điểm nữa. ả gười chơi nào có số điểm lớn hơn sẽ là người chiến thắng. Cho trước một vị trí bàn Cờ Vây; xác định điểm cho mỗi người chơi. Ví dụ: Quân Đen có 15 điểm trong đất của mình: 3 ở đầu bàn cờ, 2 ở ngay dưới và 9 ở góc trái thấp nhất cộng với 1 điểm ở vùng đất chỗ giao điểm a. Thêm những quân thực tế có trên bàn cờ (24 quân), quân Đen có tổng cộng 39 điểm. Đất của quân Trắng là 17 điểm: 11 điểm ở góc trái cộng với 6 điểm ở góc phải. Với 24 quân trên bàn cờ, quân Trắng có tổng cộng 41 điểm. ả hư vậy, quân Trắng thắng ván chơi này với 2 điểm trội. Chú ý rằng giao điểm b không thuộc về bên nào cả.

INPUT

File input sẽ có một dòng chứa 1 số nguyên xác định số vị trí của bàn cờ. Trên các dòng tiếp theo sẽ chỉ ra cụ thể các vị trí này. Mỗi vị trí gồm 9 dòng với 9 ký tự: X cho quân Đen, O cho quân Trắng và '.' cho giao điểm rỗng. Không có dòng trống nào phân tách giữa các bài toán.

OUTPUT

Kết quả chính xác gồm một tập các dòng (một dòng cho mỗi đáp án của bài toán) trong đó mỗi dòng gồm: Đen <tổng điểm Đen> Trắng <tổng điểm Trắng>, mỗi kết quả nằm trên một dòng.

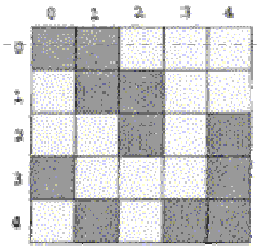
Sample Input	Sample Output
1 OX..XXO.. OXX.XOO.. OOXX.XO.O .OOX.XOO. ..OOXXOO ..OO.X.XO ..OOXX.XX ..OX.X... ..OXX....	Black 39 White 41

UVA871. ĐẾM SỐ Ô TRONG ĐỐM MÀU

Xét một lưới 2 chiều, mỗi ô có thể rỗng hoặc được tô kín. Các ô được tô kín được kết nối với nhau hình thành nên đốm màu lớn hơn. Hai ô được gọi là liên thông với nhau nếu chúng kề nhau theo chiều dọc, ngang hoặc chéo. Có thể có nhiều đốm màu trong lưới. ả hiệm vụ của chúng ta là tìm ra đốm màu lớn nhất (về số lượng ô) trong lưới. Hình dưới đây minh hoạ một lưới có 3 đốm màu (đốm lớn nhất gồm 5 ô). Viết một chương trình xác định kích cỡ của đốm màu lớn nhất trong tập các đốm màu đã cho.

INPUT

Input khởi đầu bằng một dòng chứa số nguyên dương cho biết số test, mỗi test được mô tả chi tiết bên dưới. Tiếp theo là một dòng trống, và giữa các input liên tiếp nhau cũng sẽ có một dòng trống. Lưới được cho dưới dạng một tập chuỗi, gồm các số 0 và 1. 1 cho biết ô đã được tô kín và 0 chỉ ra rằng đó là ô trống. Các chuỗi sẽ được chuyển sang dạng lưới. Lưới lớn nhất có kích thước 25x25.



OUTPUT

Với mỗi test, output phải theo như mô tả dưới đây. Output của hai test liên tiếp nhau sẽ cách nhau bằng một dòng trắng.
Output là kích thước của đốm lớn nhất tìm ra trên lưới.

Sample Input	Sample Output
1 **11000** **01100** **00101** **10001** **01011**	**5**

UVA1197. TÌNH NGHI

Hội chứng hô hấp cấp tính nặng (SARS), một dạng viêm phổi không rõ nguyên nhân đã được công nhận là một dịch bệnh toàn cầu vào giữa tháng Ba năm 2003. Để giảm thiểu việc lây nhiễm, cách tốt nhất là phân tách những người nghi ngờ bị lẫy nhiễm ra. Ở học viện Hogwart, có nhiều nhóm sinh viên. Các sinh viên trong cùng một nhóm thường xuyên trao đổi với nhau, và một sinh viên có thể tham gia vào nhiều nhóm khác. Để ngăn chặn khả năng lây truyền SARS, học viện thu thập danh sách thành viên của tất cả các nhóm sinh viên, và đề ra quy tắc sau đây trong nội quy hoạt động chuẩn mực của họ: Một khi một thành viên trong một nhóm bị nghi ngờ nhiễm SARS, tất cả các thành viên trong nhóm sẽ đều bị xếp vào diện bị nghi ngờ. Tuy nhiên, không dễ để xác định tất cả những người bị nghi ngờ khi một sinh viên bị cho là người bị nhiễm. Hãy viết một chương trình để tìm ra tất cả những người bị nghi ngờ.

INPUT

File input gồm nhiều test. Mỗi test bắt đầu bằng 2 số nguyên n và m trong một dòng, trong đó n là số sinh viên và m là số nhóm. Có thể giả thiết là $0 < n \leq 30000$ và $0 < m \leq 500$. Mỗi sinh viên được đánh số bằng một số nguyên duy nhất từ 0 đến n - 1, khởi đầu sinh viên 0 được cho là người bị tình nghi trong tất cả các test. Tiếp đó là m danh sách thành viên của các nhóm, mỗi nhóm trên một dòng. Mỗi dòng bắt đầu bằng một số nguyên k biểu thị số thành viên trong nhóm. Tiếp đó là số hiệu của các thành viên, có k số nguyên biểu diễn các sinh viên trong nhóm này. Tất cả các số nguyên trên một dòng được phân tách với nhau bằng một hoặc một vài dấu cách.
Trường hợp n = 0, m = 0 chỉ ra file đã hết, không cần xử lý.

OUTPUT

Với mỗi test, xuất ra số người bị tình nghi đã nhiễm bệnh.

Sample Input	Sample Output
100 4 2 1 2 5 10 13 11 12 14 2 0 1 2 99 2 200 2 1 5 5 1 2 3 4 5 1 0	4 1 1

Formatted: Font: (Default) Times New Roman, 10 pt, Font color: Text 1

Formatted: Font: Bold

Formatted: Centered, Tab stops: Not at 8.25 cm + 16.51 cm

0 0	

UVA11036. XẾP HẠNG CÁC NGÔN NGỮ

Chú ý là tiếng Anh và tiếng Tây Ban ả ha được sử dụng ở rất nhiều nơi trên thế giới. Xếp hạng tất cả các ngôn ngữ theo số nước sử dụng chúng là một ý kiến hay. Cho trước một bản đồ chỉ ra các nước và ngôn ngữ mà các nước này sử dụng. Hãy nhìn vào bản đồ sau:

ttuuttdd
ttuuttdd
uuttuudd
uuttuudd

Formatted: Font: Bold, Italic

Bản đồ này được đọc như sau: Mỗi ký tự là viết tắt của một ngôn ngữ và các nước được xác định là các khu vực liên thông với nhau có cùng ký tự. Hai ký tự được xem là liên thông với nhau nếu một ký tự nằm ở bên trái, bên phải, bên trên hoặc bên dưới ký tự kia. ả hư thế, ở bản đồ trên, có 3 nước ở đó nói ngôn ngữ "t", 3 nước nói ngôn ngữ "u" và 1 nước nói ngôn ngữ "d". Xác định mỗi ngôn ngữ được dùng ở bao nhiêu nước và in ra kết quả theo thứ tự nhất định.

INPUT

Dòng đầu tiên là số test ả . Mỗi test gồm 1 dòng chứa 2 số H và W, là chiều dài và chiều rộng của bản đồ. Tiếp theo là H dòng, mỗi dòng chứa một chuỗi gồm W ký tự. ả những ký tự này sẽ chỉ gồm các chữ thường từ 'a' đến 'z'.

OUTPUT

Mỗi test in ra dòng chữ "World#n", trong đó n là số test. Tiếp đó mỗi ngôn ngữ xuất hiện trong test được in trên một dòng, mỗi dòng này gồm các thông tin sau: ngôn ngữ, dấu hai chấm, dấu cách, và số nước sử dụng ngôn ngữ đó. Các dòng này phải được sắp xếp giảm dần theo số nước. ả ếu hai ngôn ngữ có số nước sử dụng ngang nhau thì sắp xếp theo bảng chữ cái theo tên ngôn ngữ, ví dụ ngôn ngữ "i" phải đứng trước ngôn ngữ "q".

Sample Input	Sample Output
2 4 8 ttuuttdd ttuuttdd uuttuudd uuttuudd 9 9 bbbbbbbbb aaaaaaaab bbbbbbbab baaaaacab baccccab bacbbbcab baccccab baaaaaab bbbbbbbbb	World #1 t: 3 u: 3 d: 1 World #2 b: 2 a: 1 c: 1

UVA10946. LẤP GÌ ĐÂY?

Ryan và Larry đang tức điên lên vì một việc khó khăn, họ bị buộc phải làm công việc hạ đẳng để kiếm miếng ăn, một trong số đó là lấp các vũng trên hòn đảo hoang vắng này. ả hưng dĩ nhiên là việc này không hề dễ chút nào, họ bị buộc phải lấp cái vũng lớn nhất trước tiên. Vì Ryan và Larry vẫn uể oải (rất khó thay đổi, bạn biết đấy), chỉ họ thứ tự để lấp các vũng nhé.

INPUT

Dòng đầu tiên có hai số x và y, sau đó là x dòng, mỗi dòng có y ký tự (x và y ít hơn 50). ả hững cái vũng này sẽ được biểu diễn bằng các ký tự viết hoa từ A đến Z và những vùng đất thường sẽ được biểu diễn bằng ký tự ".". Không có ký tự nào khác trong bản đồ. Kết thúc là dòng ghi 0 0.

OUTPUT

Với mỗi bản đồ, xuất ra số bài toán (như đã chỉ ra trước đây), sau đó xuất ra cái vũng được biểu diễn bằng ký tự, và số dấu cách mà cái vũng này chiếm giữ, được sắp theo kích thước của vũng. Các ký tự được sắp xếp theo thứ tự của bảng chữ cái, như chỉ ra trong output mẫu trên một dòng riêng như đã chỉ ra dưới đây:

Sample Input	Sample Output
5 5 ..AAA E.BBB ..AA. CC.DD CC.D. 5 5 ..AAA E.BBB ..AA. CC.DD CC.D. 0 0	Problem 1: C 4 A 3 B 3 D 3 A 2 E 1 Problem 2: C 4 A 3 B 3 D 3 A 2 E 1

UVA11094. CÁC LỤC ĐỊA

Đại đế Mydas trị vì đế quốc Phrygia. Ông thích du hành giữa các thành phố trong vương quốc của mình và thực tế là không thể nhìn thấy ông ở 1 thành phố trong 2 ngày liền. Bởi vậy, ông cai quản tất cả các vùng đất trong châu lục của mình. Ông đã đi thăm tất cả các thành phố trong vương quốc của mình và muốn cai quản châu lục khác để lại được đi thăm những thành phố mới. Với bản đồ thế giới trong tay, ông cần sự giúp đỡ để tìm ra châu lục lớn nhất ngoại trừ châu lục mà ông đang ở. Các bản đồ cũng giống như các bảng M × ả được cho trước, chỉ dùng tối đa 2 ký tự khác nhau để đánh dấu chỗ nào là đất, chỗ nào là nước. Một châu lục là một tập các vùng đất kết nối với nhau, được hoàn toàn bao quanh bởi nước hoặc biên bản đồ. Hai vùng đất được xem như kết nối với nhau nếu chúng có chung cạnh. Toạ độ của vùng bên trái trên cùng là (0,0) và vùng bên phải dưới cùng là (M - 1, ả - 1). Vùng đất có toạ độ (x, ả - 1) được coi như có cạnh chung với vùng đất (x, 0) với mọi x nằm trong khoảng từ 0 đến M - 1.

INPUT

Sẽ có nhiều test. Dòng đầu tiên của mỗi test là 2 số nguyên M ≤ 20 và ả ≤ 20 là số dòng và số cột của bản đồ. tiếp đó là M dòng, mỗi dòng có ả ký tự biểu diễn bản đồ. Cuối cùng, ở dòng cuối sẽ có 2 ký tự 0 ≤ X < M và 0 ≤ Y < ả , là toạ độ của khu vực và vua Midas đang ở. Sau mỗi test có một dòng trắng.

OUTPUT

Với mỗi test, xuất ra một dòng chứa một số nguyên là số vùng trong châu lục lớn nhất mà vua Midas có thể cai quản.

Sample Input	Sample Output
5 5 wwwww wwllw wwwww	2

wllww wwwww 1 3	

UVA11244. ĐẾM SAO

Trên bầu trời đêm, mọi người thường chỉ chú ý đến Mặt Trăng chứ không chú ý đến ả gôi Sao. Lần này, bạn phải viết chương trình đếm sao trên màn hình. Bầu trời là một lưới hai chiều. Ô trống (không chứa gì) là '.' (mã ASCII 46) và ô không trống là '*' (mã ASCII 42). Một ngôi sao rất nhỏ và chỉ nằm trong một ô. Và trên bầu trời của chúng ta, hai ngôi sao không thể đứng cạnh nhau. ả ếu nhiều điểm "* " (số điểm lớn hơn 2) kề nhau có thể coi là một thiên thể lớn (mặt trăng, mặt trời,...). Ở hình vẽ dưới, * ở giữa có 8 hàng xóm, trong đó có 3 * và 5 ".".

*..
.**
..*

INPUT

Không có quá 1000 test, mỗi test được mô tả như sau: Dòng đầu của mỗi test là 2 số nguyên r và c (0< r, c<101), mô tả số hàng và cột. r dòng kế tiếp, mỗi dòng có c ký tự . và * thể hiện bầu trời. Kết thúc khi r=c=0.

OUTPUT

Với mỗi test, in ra số ngôi sao trong bầu trời đêm.

Sample Input	Sample Output
5 5 ....* ....* ...*. *.... 4 3 ... .*. ... *.* 0 0	1 3

UVA11470. TỔNG HÌNH VUÔNG

Bạn biết là trong mỗi một hình vuông lại có một hình vuông. Cái này có vẻ dễ gây nhầm lẫn, nhưng hãy nhìn thử xem hình dưới đây. Giả sử bạn có 1 bảng ô vuông 5 × 5, mỗi ô có 1 ghi một số.

5	3	2	7	9
3	*7*	***4***	***2***	1
6	***5***	**2**	***3***	4
5	***4***	***3***	***4***	5

4	1	1	1	6

Bạn có thể tạo ra 3 hình vuông từ hình vuông ban đầu… thật ra chúng ta có thể tạo được nhiều hơn nhưng ta chỉ lấy những hình vuông đồng tâm (tâm của hình vuông chính là giao điểm của 2 đường chéo). Các hình vuông được lựa chọn được biểu thị bằng các font khác nhau. Bạn phải tìm tổng của các số trong mỗi hình vuông.
Trong trường hợp này. Tổng của các hình vuông là:
5 + 3 + 2 + 7 + 9 + 1 + 4 + 5 + 6 + 1 + 1 + 1 + 4 + 5 + 6 + 3 = 63
7 + 4 + 2 + 3 + 4 + 3 + 4 + 5 = 32
2 = 2

INPUT

Mỗi test bắt bằng số n ($n \leq 10$) là số cạnh của bảng vuông. ẩ dòng tiếp theo, mỗi dòng sẽ chứa n số để biểu thị bảng vuông. Input sẽ kết thúc khi n = 0.

OUTPUT

Mỗi dòng input sẽ bắt đầu với "Case #:" dấu # được thay thế bởi số thứ tự của test đó sau đó bạn phải in ra ceil(n/2) tổng các số trên mỗi hình vuông ra. Các hình vuông được in ra theo thứ tự từ vòng ngoài vào.

Sample Input	Sample Output
5 5 3 2 7 9 1 7 4 2 4 5 3 2 4 6 1 3 4 5 1 1 4 5 6 3 1 1 0	Case 1: 63 32 2 Case 2: 1

UVA11561. TÌM VÀNG

Chúng ta sẽ xây dựng một trò chơi cổ điển. ẩ ội dung của nó rất đơn giản – dựa trên một chuyến phiêu lưu đi tìm kho báu trong một mê cung và phải tránh những chiếc bẫy. Mê cung là một hình chữ nhật và bạn sẽ có rất ít thông tin về những gì đang có xung quanh bạn.

Trong trò chơi, số bước người chơi di chuyển không bị giới hạn. ẩ gười chơi có thể di chuyển đi lên, đi xuống, sang trái, sang phải, nhưng không được đi chéo. ẩ gười chơi sẽ nhặt được vàng nếu đi vào ô chứa vàng. ẩ ếu người chơi đứng cạnh ô có bẫy (trên, dưới, trái , phải) thì người chơi sẽ cảm nhận được chung quanh có bẫy (nhưng sẽ không biết bẫy nằm ở đâu và chung quanh có bao nhiêu bẫy). ẩ ếu ô đang đứng chứa bức tường thì người chơi sẽ đứng yên ở đó mà không di chuyển tiếp. Tìm số vàng nhiều nhất người chơi có thể lấy được nếu có một chiến lược phù hợp và ô kế tiếp mà người chơi di chuyển vào là ô an toàn (không có bẫy). ẩ gười chơi không biết trước bản đồ.

INPUT

Có nhiều test, mỗi test được mô tả như sau: Dòng đầu tiên là 2 số $3 \leq W,H \leq 50$ là chiều cao và chiều rộng của bản đồ. H dòng tiếp theo bao gồm W ký tự để mô tả bản đồ. Các ký tự có thể xuất hiện trong bản đồ: P: nơi bạn đang đứng. G: ô chứa vàng. T: ô chứa bẫy. #: ô chứa tường. -: ô bình thường. Có duy nhất một chữ P trong bản đồ và tường sẽ bao bọc quanh bản đồ.

OUTPUT

Số vàng nhiều nhất có thể lấy.

Sample Input	Sample Output
7 4 ####### #P.GTG# #..TGG# ####### 8 6 ######## #...GTG# #..PG.G# #...G#G# #..TG.G# ########	1 4

UVA11953. BẮN TÀU CHIẾN

Game tàu chiến là một trò chơi bằng giấy và bút chì được Clifford Von Wickler nghĩ ra khoảng đầu thế kỉ XX. Trong trò chơi này, mỗi người chơi sử dụng 2 tờ giấy kẻ ô ả xả . Một người đặt các con tàu và ghi lại các phát bắn của địch thủ. Trên tờ giấy khác, người chơi kia ghi lại những phát bắn của mình. Kích thước tàu chiến trong trò chơi này trong khoảng từ 1×1 đến 1× ả /2 và có thể đặt ngang hoặc dọc. Khi tất cả các ô của tàu chiến bị bắn thì tàu chìm, nếu không tàu vẫn còn nổi. Bên cạnh đó, có thể có nhiều tàu chiến có cùng kích cỡ, và không có 2 tàu chiến nào có thể đè hoặc chạm vào nhau. Trong bài toán này, bạn biết được vị trí của các con tàu. Bạn sẽ phải tính số tàu mà người chơi có.

INPUT

Input có T bộ test (T ≤ 100) được ghi ở dòng đầu tiên. Mỗi bộ test bao gồm một số nguyên ả (ả ≤ 100) là kích thước của tờ giấy. Sau đó là ả dòng, mỗi dòng có ả chữ, mô tả trận đang chơi. Ký tự "."là ô trống, "x" là ô chứa 1 phần của con tàu và "@" chỗ đã bị bắn của con tàu.

OUTPUT

Mỗi bộ test abo gồm một dòng "Case T: ả ", với T là số bộ test và ả là số con tàu còn nổi.

SAMPLE INPUT	SAMPLE OUTPUT
2 4 x... ..x. @.@. .... 2 .. x.	Case 1: 2 Case 2: 1

UVA11518. DOMINOS 2

Dominos thật thú vị.Trẻ em thích ngồi xếp chúng trên một đường thẳng dài. Khi bị đổ, một domino sẽ làm đổ domino bên cạnh,và cứ tiếp tục như thế cho đến khi làm đổ cả đường thẳng. ả hưng đôi lúc chiếc domino không làm đổ cái phía trước nó. Trong trường hợp này chúng ta phải làm đổ chiếc domino bằng tay. Cho một tập những chiếc domino đổ bằng tay, bạn phải tính tổng số domino bị đổ.

OUTPUT

Dòng đầu tiên của input là số nguyên là số bộ test. Mỗi test bắt đầu bằng 3 số nguyên n,m,l<10000, và sau đó là m + l dòng. Số n là số quân domino . Các quân domino đánh số từ 1 đến n. Tiếp theo m dòng, mỗi dòng chứa 2 số nguyên x và y chỉ ra rằng nếu domino x đổ sẽ kéo theo domino y đổ. Tiếp theo là L dòng, mỗi dòng chứa 1 số nguyên z là số quân domino được làm đổ bằng tay.

Formatted: Font: 10 pt, Font color: Text 1

OUTPUT

Với mỗi test, xuất ra số nguyên là tổng số domino bị đổ

Sample Input	Sample Output
1 3 2 1 1 2 2 3 2	2

UVA11749. NGƯỜI CỐ VẤN TỘI NGHIỆP

Mấy hôm trước, bạn đang tìm hiểu cách hoạt động của một cỗ máy lạ tìm thấy ở một nhà kho cũ (Về sau bạn sẽ biết đó là nhà kho của *Dr. Emmett Brown*), bạn tình cờ ấn vào một nút và hậu quả là bạn bị gửi quay ngược thời gian về năm 48 trước công nguyên với chiếc laptop trên tay. May mắn thay, bạn có biết một chút tiếng Latin để trò chuyện với cố vấn thương mại của Hoàng đế Caesar. Ông ta mời bạn về nhà và mời bạn ăn. Đổi lại, bạn phải giúp ông ta. Gần đây Hoàng đế Caesar quyết định mở thêm một tỉnh trong Đế quốc. Đế quốc là một mạng lưới các thành phố nối với nhau bằng các con đường 2 chiều, và mỗi con đường có giá trị PPA riêng. Giá trị PPA của mỗi đường được tính bằng hiệu của lãi hàng năm trừ đi chi phí bảo dưỡng hàng năm.

Tỉnh mới gồm một mạng lưới đường có kết nối với nhau và đương nhiên là các thành phố kèm theo. Đồng thời Tỉnh mới phải có PPA trung bình cao nhất có thể. Tỉnh mới có số thành phố nhiều nhất có thể, nếu không làm được thì ông ta sẽ không giữ được cái đầu của mình. Ông cố vấn muốn bạn giúp. Với chiếc laptop trên tay, hãy viết một chương trình để giúp ông ta.

INPUT

Mỗi test bao gồm 2 số nguyên $1<n\leq 500$ và $1\leq m\leq 1000000$ là số thành phố và số con đường. m dòng sau mỗi dòng có 3 số nguyên với 2 số đầu là tên hai thành phố (các thành phố đánh số từ 1 đến n), và số thứ 3 là PPA của quãng đường đó. Input sẽ kết thúc khi 2 số m,n bằng 0

OUTPUT

Với mỗi test in ra số thành phố trong tỉnh có PPA trung bình cao nhất.

Sample Input	Sample Output
4 5 1 2 100 1 3 100 1 4 1 2 3 100 3 4 1 9 14 1 2 9 6 9 8 2 4 9 2 3 9 4 5 1 4 3 9	3 5

5 9 2 9 8 9 7 8 9 7 9 5 6 7 9 5 6 4 5 8 7 7 5 9 0 0	

03. TOPOSORT

UVA124/872. TUÂN THỦ RÀNG BUỘC

Trật tự là khái niệm quan trọng trong Toán học và Tin học. Ví dụ Bổ đề Zorn phát biểu như sau: "Trong một tập hợp có trật tự bộ phận – và mỗi chuỗi thứ tự có cực đại – thì sẽ tồn tại phần tử lớn nhất".Trong bài toán này, giả sử có một danh sách các điều kiện ràng buộc trên các biến có dạng x<y, bạn phải xác định tất cả các trật tự của các biến mà thỏa mãn mọi điều kiện ràng buộc. Ví dụ, cho hai điều kiện x < y và x < z thì tồn tại hai trật tự của các biến x, y, và z thỏa mãn hai điều kiện trên: x y z và x z y.

INPUT

Input gồm nhiều test. Mỗi test có 2 dòng. Dòng đầu chứa danh sách các biến, dòng thứ hai là danh sách các ràng buộc, trong đó môi ràng buộc là một cặp hai biến có dạng x y (nghĩa là x<y). Mỗi biến là 1 chữ cái Latin thường. Mỗi test không có quá 20 biến. Số ràng buộc trong mỗi test lớn hơn 1 và nhỏ hơn 50.

OUTPUT

Với mỗi test, in ra theo thứ tự từ điển tất cả các trật tự có thể có. Mỗi trật tự nằm trên một dòng.

Sample Input	Sample Output
a b f g a b b f v w x y z v y x v z v w v	abfg abgf agbf gabf wxzvy wzxvy xwzvy xzwvy zwxvy zxwvy

UVA200. TRẬT TỰ HIỂM ÁC

Một nhà sưu tầm sách cổ mới tìm ra một quyển sách viết bằng một ngôn ngữ kỳ lạ - vẫn sử dụng các chữ cái Tiếng Anh nhưng theo một trật tự khác. ấ hà sưu tầm tìm thấy một danh sách mục lục – một danh sách các từ nhưng thứ tự xuất hiện các từ trong danh sách (theo ngôn ngữ này) không giống như thứ tự nếu xét theo bảng chữ cái Tiếng Anh. Từ danh sách các từ này (danh sách này đã được sắp xếp theo trật tự bảng chữ cái của ngôn ngữ lạ), bạn có nhiệm vụ tìm thứ tự bảng chữ cái trong ngôn ngữ lạ.

INPUT

Input là một danh sách các từ, mỗi từ nằm trên một dòng. Mỗi từ là một xâu chứa tối đa 20 chữ cái Latin viết hoa. Danh sách kết thúc khi gặp ký tự `#'. Không phải mọi chữ cái đều được sử dụng, nhưng thứ tự các từ tuân theo thứ tự từ điển theo thứ tự bảng chữ cái của ngôn ngữ lạ.

OUTPUT

Xuất ra bảng chữ cái của ngôn ngữ lạ, sắp xếp theo thứ tự từ điển (trong ngôn ngữ ấy).

Sample Input	Sample Output
XWY ZX ZXY ZXW YWWX	XZYW

#	

UVA10305. THỨ TỰ THỰC HIỆN CÔNG VIỆC

John có n công việc phải làm. Thật không may, những công việc này không độc lập với nhau, một công việc chỉ được thực hiện khi một số công việc khác đã hoàn thành.

INPUT

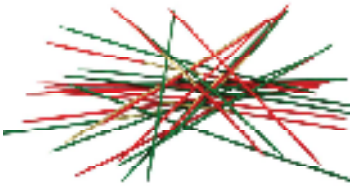
Input gồm nhiều test. Mỗi test bắt đầu bằng một dòng gồm 2 số nguyên, 1≤n≤100 và m. n là số công việc cần hoàn thành (các công việc đánh số từ 1 đến n) và m là số mối quan hệ phụ thuộc giữa các công việc. Sau đó là m dòng, mỗi dòng gồm 2 số i và j, thể hiện rằng công việc thứ i phải được làm trước công việc j. Test có n=m=0 sẽ kết thúc input.

OUTPUT

Với mỗi test, in ra 1 dòng chứa thứ tự hoàn thành các công việc.

Sample Input	Sample Output
5 4 1 2 2 3 1 3 1 5 0 0	1 4 2 5 3

UVA 11686. NHẶT QUE



ẩ hặt Que là một trò chơi thú vị. Có một đống que nằm trên bàn (có thể gối lên nhau). Từng người chơi theo lượt sẽ lấy một que ra sao cho không di chuyển các que khác. Sẽ rất khó để lấy một chiếc que nếu như có một cái que khác ở trên nó. Vì vậy mọi người chơi đề tìm cách để lấy các que theo thứ tự nào đó mà không bao giờ lấy một cái que nằm bên dưới một que khác.

INPUT

Input chứa nhiều test. Dòng đầu của mỗi test chứa 2 số nguyên n và m (1<=n,m<=1000000). Số nguyên n là số que, và m là số dòng sau đó. Các que được đánh số từ 1 đến n. Mỗi dòng tiếp theo chứa 2 số nguyên a và b, thể hiện rằng que a nằm trên que b. Test cuối cùng sẽ là 0 0. Test này không cần xử lí.

OUTPUT

Với mỗi test, output sẽ chứa n dòng chứa n số nguyên, là thứ tự lấy các que ở test đó. ẩ ếu như có nhiều cách lấy que, bạn có thể in một trường hợp bất kỳ. ẩ ếu như không có cách lấy que, in ra một dòng duy nhất chứa từ IMPOSSIBLE.

Sample Input	Sample Output
3 2 1 2 2 3 0 0	1 2 3

UVA11060. UỐNG BIA

Kết thúc năm học, Chàng Khờ nghỉ hè và quyết định đi Bar nhậu nhẹt với đám bạn bè của mình. Khi uống đồ uống có cồn, Chàng Khờ có sở thích sau: đầu tiên uống đồ uống có nồng độ cồn thấp (bia), rồi uống đồ uống có nồng độ cao dần (rượu nhẹ, rồi rượu mạnh). Một khi đã uống uống rượu, Mr Khờ sẽ không uống lại bia (nồng độ cồn trong đồ uống không bao giờ giảm). Hãy giúp Chàng Khờ xác định thứ tự các loại đồ uống theo đúng sở thích.

INPUT

Mỗi test bắt đầu bằng một số nguyên ả (1≤ả ≤100), là số đồ uống có sẵn. Sau đó là ả dòng, mỗi dòng là tên của một loại đồ uống. Kế đó là một dòng ghi số nguyên M (0≤M≤200). M dòng tiếp theo có dạng B1 B2, có nghĩa rằng đồ uống B2 có nhiều cồn hơn đồ uống B1 (Chàng Khờ phải uống B1 trước B2) Giữa mỗi test có một dòng trống.

OUTPUT

Với mỗi test, in ra dòng: *Case #C: Dilbert should drink beverages in this order: B1 B2 ... BN.*, trong đó C là số thứ tự của test, bắt đầu đánh số từ 1, và B1 B2 ... Bả là danh sách các đồ uống. Sau mỗi test là một dòng trống.

Sample Input	Sample Output
3 vodka wine beer 2 wine vodka beer wine 5 wine beer rum apple-juice cachaca 6 beer cachaca apple-juice beer apple-juice rum beer rum beer wine wine cachaca 10 cachaca rum apple-juice tequila whiskey wine vodka beer martini gin 11 beer whiskey apple-juice gin rum cachaca vodka tequila	Case #1: Dilbert should drink beverages in this order: beer wine vodka. Case #2: Dilbert should drink beverages in this order: apple-juice beer wine rum cachaca. Case #3: Dilbert should drink beverages in this order: apple-juice wine vodka beer rum cachaca tequila whiskey martini gin.

apple-juice martini rum gin wine whiskey apple-juice beer beer rum wine vodka beer tequila	

04. KIỂM TRA ĐỒ THỊ HAI PHÍA

UVA10004. TÔ ĐỒ THỊ BẰNG HAI MÀU

Trong năm 1976, định lý *Tô Bản Đồ Bằng Bốn Màu* đã được chứng minh nhờ vào việc sử dụng máy tính. Định lý này nói rằng, chỉ cần dùng 4 màu có thể tô màu bản đồ sao cho hai vùng cạnh nhau có màu khác nhau. Ở đây, bạn được yêu cầu xác định xem liệu một đồ thị liên thông, vô hướng tùy ý có thể được tô bằng hai màu sao cho hai đỉnh kề bất kỳ được tô bằng hai màu khác nhau hay không.

INPUT

Input có nhiều test. Trong mỗi test, dòng đầu ghi một số nguyên n (1 < n < 200) – là số đỉnh đồ thị (các đỉnh đồ thị được đánh số từ 0 đến n-1). Dòng thứ hai ghi số cạnh m. m dòng tiếp theo, mỗi dòng ghi hai số là 2 đỉnh của một cạnh. Input kết thúc khi n=0.

OUTPUT

Xuất ra BICOLORABLE hoặc ả OT BICOLORABLE tùy trường hợp.

Sample Input	Sample Output
3 3 0 1 1 2 2 0 9 8 0 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0	NOT BICOLORABLE. BICOLORABLE.

UVA10505. MỜI BẠN

Cuối cùng thì Romeo và Juliet cũng quyết định tổ chức đám cưới. ả hưng vấn đề tổ chức đám cưới trở nên phức tạp – hai gia tộc Montesco và Capuleto là hai kẻ tử thù. Trong bài toán này, bạn phải xác định ai sẽ được mời, ai không được mời đến đám cưới để bữa tiệc cưới có thể diễn ra vui vẻ.

Chúng ta có danh sách *N* người. Với mỗi người *i*, chúng ta có danh sách các kẻ thù *E1*, *E2*, ..., *Ep*. Quan hệ "kẻ thù" có những đặc tính sau:

- **Kẻ thù của kẻ thù là bạn**. ả ếu *a* là kẻ thù của *b*, và *b* là kẻ thù của *c*, thì *a* là bạn của *c*. Bên cạnh đó, kẻ thù của bạn cũng là kẻ thù và bạn của bạn cũng là bạn.
- **Đối xứng.** ả ếu *a* là kẻ thù của *b*, thì *b* cũng là kẻ thù của *a* (mặc dù có thể b không nằm trong danh sách kẻ thù của a).

Một người chỉ nhận lời tham gia bữa tiệc khi và chỉ khi nếu anh ta được mời thì phải mời toàn bộ bạn bè của anh ta và không được mời bất kỳ kẻ thù nào. Bạn phải đi tìm số lượng người tối đa có thể mời được. Ví dụ, với *N*=5, và chúng ta biết: 1 là kẻ thù của 3, 2 là kẻ thù của 1, và 4 là kẻ thù của 5, thì chúng ta có thể mời tối đa ba người (2, 3 và 4).

INPUT

Dòng đầu tiên là số test (*M)*. Sau đó là một dòng trống. Sau đó là các bộ test, mỗi bộ test cách nhau một dòng trống. Trong mỗi test, dòng đầu là số nguyên â ≤ 200 là số người. Sau đó là â dòng, ở dòng thứ i là danh sách kẻ thù của i. Khuôn dạng ở dòng i như sau: đầu tiên là số nguyên p – là số lượng kẻ thù của người i, kế đó là p số nguyên tương ứng với danh sách các kẻ thù của người i.

OUTPUT

Với mỗi test, in ra số lượng khách mời cực đại.

Sample Input	Sample Output
3 5 1 3 1 1 0 1 5 0 8 2 4 5 2 1 3 0 0 0 1 3 0 1 5 3 2 2 3 1 3 1 1	3 5 0

UVA11080. ĐẶT LÍNH GÁC.

Ở xứ Mễ Trì có phố và ngã tư. Mỗi con phố nối đúng hai ngã tư. Vua xứ Mễ Trì muốn đặt một số lính gác để bảo vệ mọi ngã tư và con phố. Đứng ở một ngã tư, lính gác có thể bảo vệ mọi con phố và ngã tư liền kề. â hưng lính gác có tật xấu: nếu một con phố được nhiều lính gác bảo vệ thì những lính gác này sẽ đánh nhau. Do đó nhà vua sẽ không để điều này xảy ra. Với dữ liệu về con phố và ngã tư, hãy giúp nhà vua xác định số lính gác tối thiểu thực hiện việc này.

INPUT

Dòng đầu chứa số lượng test T (T<80). Mỗi test bắt đầu bằng số nguyên v ($1 \le v \le 200$) và e ($0 \le e \le 10000$.). v là số ngã tư và e là số con phố. Kế đó là e dòng, mỗi dòng chứa hai số nguyên f và t thể hiện có con phố nối giữa f và t. Các ngã tư đánh số từ 0 đến v-1.

OUTPUT

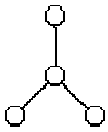
Với mỗi test, in ra số lính gác tối thiểu. Trong trường hợp không thể đặt được lính gác thỏa mãn, in ra -1.

Sample Input	Sample Output

2 4 2 0 1 2 3 5 5 0 1 1 2 2 3 0 4 3 4	2 -1

UVA11396 – VẼ MÓNG

Dưới nhãn quan đồ thị, có thể coi vuốt của động vật (bò sát hay có vú) là dạng đồ thị đặc biệt như hình vẽ dưới đây (đồ thị $K_{1,3}$)



Bạn có một đồ thị vô hướng, bậc mỗi đỉnh là 3. Bạn phải xác định xem, liệu có thể phân rã đồ thị này thành các vuốt được không ? Ở đây, phân rã là tạo ra một danh sách các đồ thị con, trong đó mỗi cạnh chỉ xuất hiện trong đúng một đồ thị con.

INPUT

Input có nhiều test. Mỗi test bắt đầu bằng số nguyên V (4≤V≤300). Sau đó là các dòng mô tả cạnh. Mỗi cạnh được mô tả bằng 2 số nguyên a và b là hai đỉnh của cạnh. (1≤a,b≤V). Danh sách cạnh kết thúc bằng dòng chứa hai số 0. Input kết thúc khi V=0.

OUTPUT

In YES nếu có thể phân rã đồ thị, in ầ O nếu không.

Sample Input	Sample Output
4 1 2 1 3 1 4 2 3 2 4 3 4 0 0 6 1 2 1 3 1 6 2 3 2 5 3 4 4 5 4 6	NO NO

5 6 0 0 0	

05. KHỚP VÀ CẦU

UVA315. MẠNG ĐIỆN THOẠI

Công ty TLC thiết lập một mạng điện thoại mới. Mạng điện thoại kết nối một số trong các địa điểm đánh số từ 1 đến ả . Các vị trí có số thứ tự khác nhau. Đường kết nối nối trực tiếp hai địa điểm với nhau và có tính chất truyền theo cả hai hướng. Từ bất kỳ một địa điểm nào cũng có thể liên lạc được với bất kỳ địa điểm nào khác – có thể là đường kết nối trực tiếp giữa hai địa điểm hoặc gián tiếp qua nhiều địa điểm. Một địa điểm được gọi là thiết yếu nếu như xóa bỏ nó – tính chất liên thông giữa các địa điểm sẽ bị mất đi. Hãy đếm số lượng trạm thiết yếu trên mạng điện thoại mới của công ty TLC

INPUT

Có nhiều test. Ở mỗi test, dòng đầu là số lượng các địa điểm ả <100. Sau đó là tối đa ả dòng, mỗi dòng là một dãy số nguyên có dạng x y z t …. thể hiện giữa x và các địa điểm y z t có đường nối trực tiếp. Dòng cuối mỗi test là một số 0. Dòng cuối input là một số 0.

OUTPUT

Xuất ra kết quả cần tìm

Sample Input	Sample Output
5 5 1 2 3 4 0 6 2 1 3 5 4 6 2 0 0	1 2

UVA610. HƯỚNG PHỐ

Theo tổ chức ACM (Automobile Collision Monitor), các tai nạn thảm khốc thường xảy ra trên đường hai chiều. Để hạn chế điều này, thị trưởng quyết định chuyển các đường hai chiều thành đường một chiều. Tuy nhiên sau khi chuyển, vẫn phải đảm bảo có thể đi từ bất kỳ ngã tư nào đến bất kỳ ngã tư nào. Bạn hãy giúp thị trưởng xác định số lượng tối đa đường có thể chuyển từ hai chiều thành một chiều.

INPUT

Có nhiều test. Dòng đầu của mỗi test là hai số nguyên n (2<n<1000) và m tương ứng là số ngã tư và số đường nối các ngã tư. Các ngã tư đánh số từ 1 đến n, m dòng tiếp theo, mỗi dòng có dạng i j – có con đường nối từ ngã tư i đến j.

OUTPUT

Với mỗi test, in ra số thứ tự của test. Các test được đánh số từ 1. Sau đó là 1 dòng trống. Sau đó in ra danh sách các các cặp i j thể hiện có đường một chiều từ i đến j. Mỗi cặp in trên một dòng. ấ ếu đường i đến j không thể đổi thành đường một chiều, thì kết quả sẽ có cả cặp i j lẫn cặp j i. Kết thúc mỗi test là dòng có in ký tự #.

Sample Input	Sample Output
7 10 1 2 1 3 2 4 3 4 4 5 4 6 5 7 6 7 2 5 3 6 7 9 1 2 1 3 1 4 2 4 3 4 4 5 5 6 5 7 7 6 0 0	1 1 2 2 4 3 1 3 6 4 3 5 2 5 4 6 4 6 7 7 5 # 2 1 2 2 4 3 1 4 1 4 3 4 5 5 4 5 6 6 7 7 5 #

UVA796. ĐƯỜNG QUAN TRỌNG

Trong mạng máy tính, một liên kết L kết nối giữa hai server được coi là quan trọng nếu tồn tại 2 server A và B sao cho mọi tuyến đường kết nối A với B đều chứa L. Xóa bỏ một liên kết quan trọng sẽ tạo ra 2 mạng con không giao nhau, mỗi mạng con là một mạng liên thông. Ví dụ trong hình vẽ, các kết nối 0-1, 3-4 và 6-7 là kết nối quan trọng.

In a computer network a link L, which interconnects two servers, is considered critical if there are at least two servers A and B such that all network interconnection paths between A and B pass through L. Removing a critical link generates two disjoint sub-networks such that any two servers of a sub-network are interconnected. For example, the network shown in figure 1 has three critical links that are marked bold: `0 -1`, `3 - 4` and `6 - 7`.

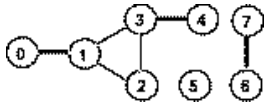


Figure 1: Critical links

It is known that:

1. the connection links are bi-directional;
2. a server is not directly connected to itself;
3. two servers are interconnected if they are directly connected or if they are interconnected with the same server;
4. the network can have stand-alone sub-networks.

Write a program that finds all critical links of a given computer network

INPUT

OUTPUT

Sample Input	Sample Output

UVA10199. HƯỚNG DẪN VIÊN DU LỊCH

INPUT

OUTPUT

Sample Input	Sample Output