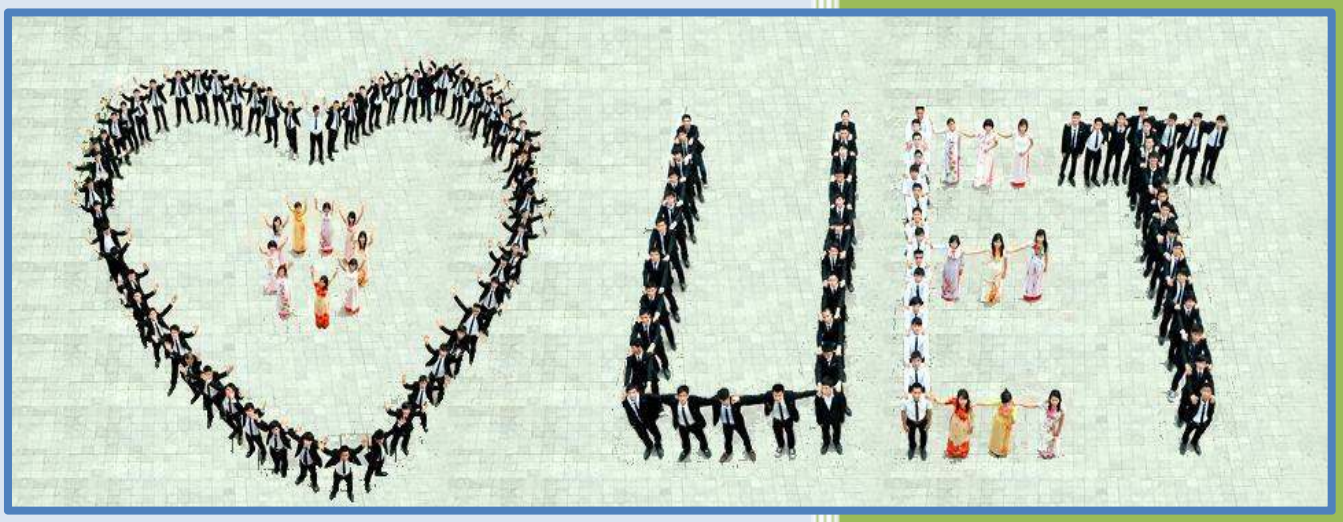
UET

Lớp Ứng dụng Thuật Toán 2018



Bài tập ĐỒ THỊ

KHOA CÔNG NGHỆ THÔNG TIN

UET

1. DUYỆT

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 → j…i → 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).

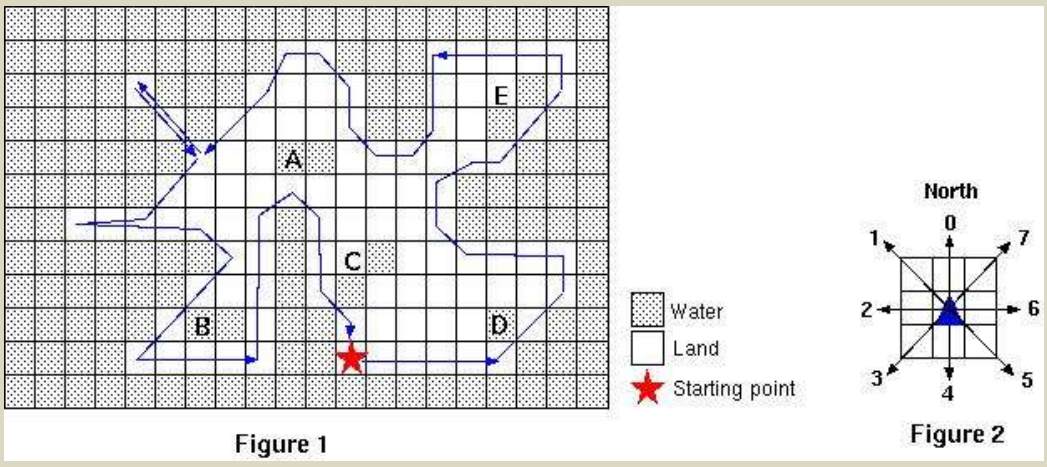


Figure 1 Figure 2

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 dự 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.

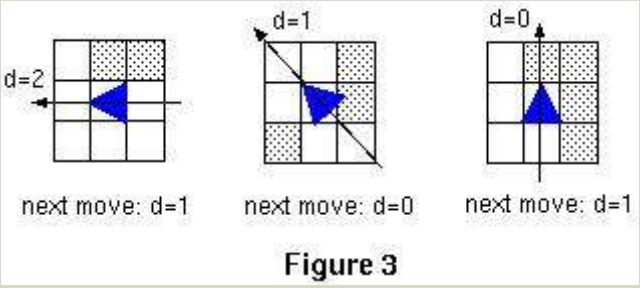


Figure 3

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 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	1 0 1

22 26 0 22 27 0 22 28 0 -1	

UVA1205. TÔ MÀU CÂY

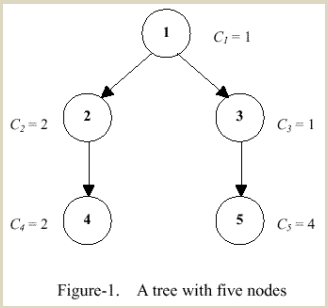


Figure-1. A tree with five nodes

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 N nút, các nút được đánh số từ 1 đến N. 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. Nếu 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 N và R ($1 \le N \le 1000$, $1 \le R \le N$), trong đó N là số nút của cây và R là số hiệu của nút gốc. Dòng thứ 2 có N 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 N-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 N = 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 item a = 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. Nếu 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.

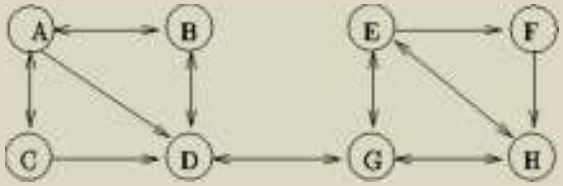
Sample Input	Sample Output
! 6 shirt = 15 sock ! 47 underwear = 9 pant ? sock = shirt ? shirt = pant ! 2 sock = 1 underwear ? pant = shirt .	5 sock = 2 shirt ? shirt = ? pant 45 pant = 188 shirt

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 N 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 N, M và S ($1 \le N, M \le 100$, $1 \le S \le 5$ 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 N 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;
- `N', `S', `L', `O' – ô nơi robot bắt đầu cuộc đua (chỉ có một robot trong trường đua). Các kí tự `N', `S', `L', `O' biểu thị hướng của robot (lần lượt là Bắc, Nam, 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.

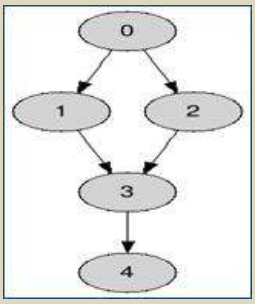
OUTPUT

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	0 1 3

...# *#O. *.*. *.#. FFEFF 10 10 20 ....*..... ....*.. .....*.... ..*.#..... ...#N.*..* ...*..... .... FDFFFFFFEEFFFF FFEFDF 0 0 0	

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. Nếu 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 N (0 < N < 100) biểu thị số nút trong đồ thị. N dòng tiếp theo, mỗi dòng chứa N số nguyên. Nếu 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 2N+1 dòng tóm tắt quan hệ chi phối nhau giữa mỗi cặp 2 nút. Nếu nút A chi phối nút B, in 'Y' tại ô (A, B), còn nếu không thì in 'N'. 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 N, 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, N-1 cặp tiếp theo chỉ toạ độ của các trạm khác. Các điều kiện ràng buộc sẽ là $-20 \le Xi, Yi \le 20$, và $1 \le N \le 1000$. Kết thúc với N = 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 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	All stations are reachable. All stations are reachable. There are stations that are unreachable.

UVA11906. HIỆP SĨ TRONG MẠNG LƯỚI CHIẾN TRẬN

Ngày 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, N), tức là chàng có thể di chuyển M ô vuông theo chiều ngang và N ô vuông theo chiều dọc hoặc chàng có thể di chuyển M ô vuông theo chiều dọc và N ô vuông theo chiều ngang trong một nước đi. Nói cách khác, chàng có thể nhảy từ ô (a,b) đến ô (c,d) khi và chỉ khi ($|a - c| = M$ và $|b - d| = N$) hoặc ($|a - c| = N$ 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. Những 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, N ($1 < R, C \leq 100, 0 \leq M, N \leq 50, M + N > 0$). 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 ($0 \leq x_i < R, 0 \leq y_i < C, x_i + y_i > 0$).

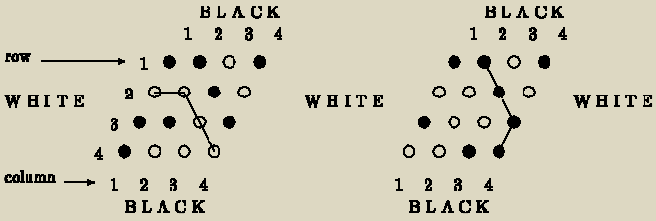
OUTPUT

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

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

2. FLOOD FILL

UVA260. Il Gioco dell'X



Trò chơi "Il Gioco dell' X" được chơi trên bàn cờ NxN (N ≥ 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 NxN 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 ≤ N). Hàng xóm của (i, j) là: (i-1, j-1), (i-1, j), (i, j-1), (i, j+1), (i+1, j) , (i+l , 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 N trong khi bên Trắng cố gắng nối liền từ cột 1 đến cột N. 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.

INPUT

Đầu vào là một file text cho biết số ván chơi. Mỗi ván chơi được cho như sau: một dòng chứa số nguyên N, là số cột. (2 ≤ N ≤ 200). Tiếp theo là N dòng nữa, mỗi dòng gồm N 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. Những 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.

UVA469. VÙNG ĐẤT ẨM FLORIDA

Một công ty xây dựng sở hữu một khu đất lớn ở Florida. Gần đây công ty này quyết định xây dựng hạ tầng trên khu đất này. Khi thăm dò vùng đất, họ phát hiện ra rất nhiều khu vực bị ngập nước. Những khu vực trũng ngập nước cạnh nhau có thể kết nối với nhau tạo thành hồ nước. Họ thuê bạn xác định việc đó.

Bạn chia khu đất thành lưới ô vuông, mỗi ô vuông hoặc là một vùng đất hoặc là một vùng trũng nước. Giờ bạn phải trả lời câu hỏi sau: " với tọa độ (vị trí hàng và cột) của một ô là vùng trũng, cái hồ chứa ô đó có bao nhiêu ô ?". Chú ý các ô chung cạnh hoặc chung đỉnh có thể được xem là kề nhau.

INPUT

Dòng đầu tiên là số test. Sau đó là một dòng trống. Các test cách nhau một dòng trống. Trong mỗi test, đầu tiên là 0<n<100 dòng, mỗi dòng có 0<m<100 ký tự. Ký tự là các chữ cái L và W. Sau đó là k dòng, mỗi dòng chứa cặp số nguyên (i, j). n dòng đầu tiên mô tả lưới có kích thước nxm, trong đó chữ cái thứ c ở hàng r tương ứng với ô ở hàng c, cột r trong lưới. k dòng sau, mỗi dòng là tọa độ 1 ô trũng (W).

OUTPUT

Kết quả mỗi test cách nhau một dòng trống, ở mỗi test, xuất ra k dòng, ở mỗi dòng là diện tích hồ (tính theo đơn vị ô) chứa ô ở hàng i, cột j.

Sample Input	Sample Output
1	12 4

LLLLLLLLL LLWWLLWLL LWWLLLLLL LWWWLWWLL LLLWWWLLL LLLLLLLLL LLLWWLLWL LLWLWLLLL LLLLLLLLL 3 2 7 5	

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. Nếu hai túi ở cạnh nhau thì chúng cùng thuộc một mỏ dầu. Các mỏ dầu có thể rất lớn và bao gồm nhiều túi. Nhiệm 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. Nếu 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

	B		
	A	C	

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.

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 tọa 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. Những 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 '%'.

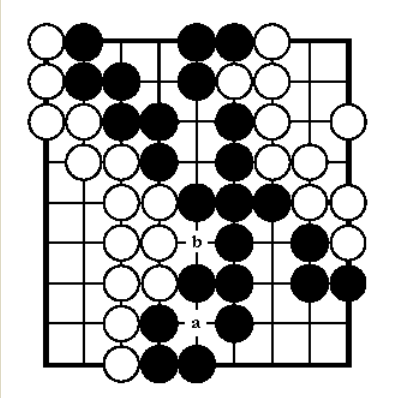
OUTPUT

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 %

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. Nếu 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. Người 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. Như 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.

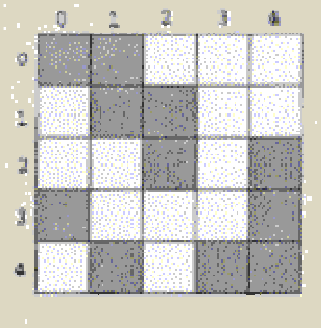
Sample Input	Sample Output
1 OX..XXO..	Black 39 White 41

OXX.XOO.. OOXX.XO.O .OOX.XOO. ..OOXXXOO ..OO.X.XO ..OOXX.XX ..OX.X... ..OXX....	

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. Nhiệm 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 0 0	4 1 1

UVA10036. XẾP HẠNG CÁC NGÔN NGỮ

Chú ý là tiếng Anh và tiếng Tây Ban Nha 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

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. Như 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 N. 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. 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. Nếu 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	World #1 t: 3 u: 3 d: 1 World #2

uuttuudd 9 9 bbbbbbbbb aaaaaaaab bbbbbbbab baaaaacab bacccccab bacbbbcab bacccccab baaaaaaab bbbbbbbbb	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. Nhưng 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). Những 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 × N 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, N - 1). Vùng đất có toạ độ (x, N - 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 N ≤ 20 là số dòng và số cột của bản đồ. tiếp đó là M dòng, mỗi dòng có N ký tự biểu diễn bản đồ. Cuối cùng, ở dòng cuối sẽ có 2 ký tự 0 ≤ X < M và 0 ≤ Y < N, 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 wllww wwwww 1 3	2

UVA11244. ĐẾM SAO

Trên bầu trời đêm, mọi người thường chỉ chú ý đến Mặt Trăng chứ không chú ý đến Ngôi 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. Nếu 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. N 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. Nội 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. Người chơi có thể di chuyển đi lên, đi xuống, sang trái, sang phải, nhưng không được đi chéo. Người

chơi sẽ nhặt được vàng nếu đi vào ô chứa vàng. Nếu 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). Nếu 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). Người 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 NxN. 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× N/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 ($T \leq 100$) được ghi ở dòng đầu tiên. Mỗi bộ test bao gồm một số nguyên N ($N \leq 100$) là kích thước của tờ giấy. Sau đó là N dòng, mỗi dòng có N 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: N", với T là số bộ test và N là số con tàu còn nổi.

Sample Input	Sample Output
2 4 x... ..x. @.@.	Case 1: 2 Case 2: 1

2 .. x.	

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. Nhưng 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.

INPUT

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.

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 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	3 5

3. 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. Nhà 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

Nhặt 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. Nếu như có nhiều cách lấy que, bạn có thể in một trường hợp bất kỳ. Nếu 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 N ($1 \le N \le 100$), là số đồ uống có sẵn. Sau đó là N dòng, mỗi dòng là tên của một loại đồ uống. Kế đó là một dòng ghi số nguyên M ($0 \le M \le 200$). 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 ... BN 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 apple-juice martini rum gin wine whiskey apple-juice beer beer rum wine vodka beer 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.

4. 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 NOT 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. Nhưng 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**. Nếu *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.** Nếu *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 N ≤ 200 là số người. Sau đó là N 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. Nhưng 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 ≤ v ≤ 200) và e (0 ≤ e ≤ 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 -1

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

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 ($4 \le V \le 300$). 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. ($1 \le a,b \le V$). 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 NO 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 5 6 0 0 0	NO NO

5. 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 N. 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 N<100. Sau đó là tối đa N 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.

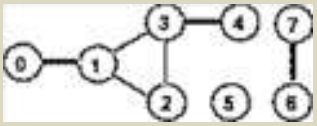
OUTPUT

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. Nếu 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	1 1 2 2 4 3 1 3 6

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	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. Người ta biết rằng:

1. Các liên kết là hai chiều.

2. Một server không tự liên kết với chính mình.

3. Hai server liên thông nếu chúng trực tiếp kết nối với nhau hoặc chúng liên thông cùng với một server thứ ba. Hãy viết chương trình xác định các kết nối quan trọng.

INPUT

Có nhiều bộ test. Với mỗi bộ test, dòng đầu tiên là số N – số lượng các server trên mạng. Các server đánh số từ 0 đến N-1. N dòng tiếp theo mô tả về kết nối của các server, ở dòng thứ i có dạng i (m) a_1 a_2 .... a_m nghĩa là server thứ i có kết nối với m server là a_1, a_2, ..., a_m

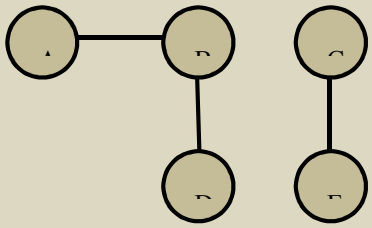
OUTPUT

In ra số lượng sau đó liệt kê các đường quan trọng.

Sample Input	Sample Output
8 0 (1) 1 1 (3) 2 0 3 2 (2) 1 3 3 (3) 1 2 4 4 (1) 3 7 (1) 6 6 (1) 7 5 (0) 0	3 critical links 0 - 1 3 - 4 6 - 7 0 critical links

5. Union- Find

459. ĐỒ THỊ LIÊN THÔNG



Đồ thị G được gọi là liên thông nếu có giữa hai đỉnh bất kỳ có một tuyến đường. Đồ thị dưới đây không liên thông vì không tồn tại đường đi giữa A và C.

Tuy nhiên Đồ thị này có chứa nhiều đồ thị con là thành phần liên thông, chẳng hạn các tập con sau đây {A}, {B}, {C}, {D}, {E}, {A,B}, {B,D}, {C,E}, {A,B,D}

Thành phần liên thông được gọi là cực đại nếu như không thể thêm bất kỳ đỉnh và cạnh nào của đồ thị ban đầu vào mà không mất đi tính liên thông. Trong ví dụ trên{A, B, D} và {C, E} là hai thành phần liên thông cực đại.

Viết chương trình xác định số lượng thành phần liên thông cực đại của một đồ thị cho trước

Input and Output

Dòng đầu tiên chứa một số nguyên là số test. Sau đó là một dòng trống và các bộ test. Mỗi bộ test cách nhau bằng một dòng trống. Dòng đầu tiên trong mỗi bộ test là một chữ cái in hoa biểu diễn tên gọi lớn nhất (theo thứ tự từ điển) của các đỉnh đồ thị. Mỗi dòng kế tiếp là một cặp hai chữ cái in hoa biểu diễn một cạnh đồ thị. Kết thúc input là một dòng trống.

Với mỗi test, in ra số lượng thành phần liên thông cực đại. Mỗi test nằm trên một dòng.

Ví dụ

Sample Input	Sample Output
1 E AB CE DB EC	2

793. KẾT NỐI MẠNG

Khuê phải quản trị một mạng máy tính. Khuê có trong tay nhật ký mạng, ghi lại kết nối giữa các cặp máy tính trong mạng. Mỗi kết nối là hai chiều. Hai máy tính có kết nối nếu hoặc chúng có kết nối trực tiếp với nhau, hoặc nếu chúng có kết nối tới cùng một máy tính thứ ba. Đôi khi Khuê sẽ phải dựa vào nhật ký để trả lời ngay lập tức – hai máy tính nào đó có kết nối (trực tiếp hay gián tiếp) với nhau không. Dựa trên file text nhật ký, hãy viết chương trình để đếm số lượng câu trả lời đúng hoặc sai cho dạng câu hỏi sau: liệu máy tính i có kết nối với máy tính j hay không ?

Input and Output

Dòng đầu tiên chứa số lượng test. Sau đó là một dòng trống. Mỗi test bao gồm::

1. Số N là số lượng các máy tính có trong mạng;

2. Sau đó là các dòng, mỗi dòng là một cặp có dạng sau:

(a) c $computer_i$ $computer_j$, trong đó $computer_i$ và $computer_j$ là các số nguyên từ 1 đến N. thể hiện $computer_i$ và $computer_j$ **có kết nối**.

(b) q computeri computerj, trong đó computeri và computerj là các số nguyên từ 1 đến N. đây là câu hỏi mà bạn phải trả lời $computer_i$ có kết nối với $computer_j$ không ?

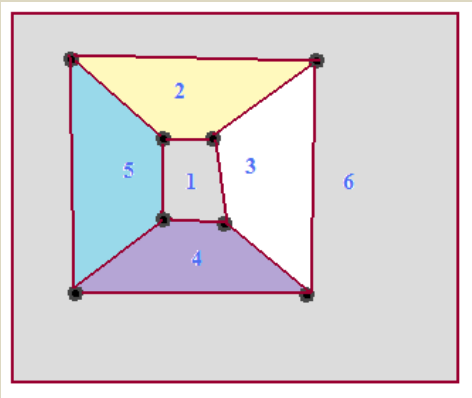
Giữa các test có một dòng trống. Nhật ký được cập nhật liên tục.

Hãy xuất ra N1 và N2, trong đó N1 là số câu trả lời CÓ và N2 là số câu trả lời KHÔNG.

Ví dụ

Sample Input	Sample Output
1 10 c 1 5 c 2 7 q 7 1 c 3 9 q 9 6 c 2 5 q 7 5	1,2

10178. ĐỒ THỊ PHẲNG



Đồ thị phẳng là đồ thị có thể vẽ trên mặt phẳng mà không có hai cạnh nào giao nhau (không tính việc giao nhau tại đỉnh).

Đồ thị trên là một độ thị phẳng. Đồ thị có 6 khu vực tô với 6 màu khác nhau, được đánh số lần lượt từ 1 đến 6. Bạn có một đồ thị phẳng và phải đếm số khu vực của đồ thị đó

INPUT

Input có nhiều test, mỗi test bắt đầu bằng một dòng ghi hai số nguyên N và E, tương ứng là số đỉnh và số cạnh của đồ thị. E dòng tiếp theo mô tả các cạnh của đồ thị, có khuôn dạng n1 n2 trong đó n1 n2 là các chữ cái Latinh thường, thể hiện giữa đỉnh n1 và đỉnh n2 có một cung.

OUTPUT

In ra số lượng khu vực của đồ thị

Sample Input	Sample Output
1 0 3 3 A B B C A C	1 2

10227. RỪNG

Triết gia Berkeley đặt ra một câu hỏi thú vị : Nếu không có ai nghe, một cây đổ trong rừng có tạo ra âm thanh không ? Trong rừng có T cây đánh số từ 1 đến T và P người đánh số từ 1 đến P.

INPUT

Số đầu tiên là số lượng test, sau đó là một dòng trống. Mỗi test cách nhau một dòng trống. Mỗi test bắt đầu bằng 2 số nguyên P và T (0<P, T<100), sau đó là các dòng dạng *i j*, trong đó người i nghe thấy cây j đổ. Mọi người có thể có những ý kiến khác biệt về cây nào đổ.

OUTPUT

Xuất ra số lượng ý kiến khác biệt ? Hai người có cùng ý kiến khi và chỉ khi tập cây đổ mà họ nghe là giống hệt nhau

Sample Input	Sample Output

1 3 4 1 2 3 3 1 3 2 2 3 2 2 4	2

10507. TỈNH THỨC

Các nghiên cứu mới nhất trong ngành sinh lý học thần kinh chỉ ra rằng bên trong não bộ, có những tuyến truyền dẫn rất lớn kết nối những khu vực khác nhau của não bộ. Hơn thế nữa, các nhà khoa học còn khám phá: nếu trong vòng một năm, một vùng não trong trạng thái ngủ có kết nối trực tiếp với ít nhất 3 vùng não trong trạng thái thức thì vùng não ngủ sẽ chuyển sang trạng thái thức (và sẽ ở lại trạng thái này). Giả sử các khu vực não bộ có tên gọi A, B, C,… và ban đầu, các khu vực đều ở trạng thái ngủ, một số khu vực có kết nối với nhau. Nếu kích thích trực tiếp để làm thức ba khu vực não nào đó, phải mất bao nhiêu năm thì toàn bộ các khu vực não chuyển sang trạng thái thức ?

INPUT

Có nhiều test, các test cách nhau bằng một dòng trống. Khuôn dạng các test như sau: Dòng đầu tiên là số nguyên $3 \le N \le 26$, là số lượng các khu vực trong trạng thái ngủ ban đầu. Dòng thứ hai là số nguyên dương M – số lượng kết nối giữa các khu vực (nếu A có kết nối với B thì B có kết nối với A). Dòng thứ 3 là ba ký tự thể hiện ba khu vực của bộ não bị kích thích trực tiếp. M dòng kế tiếp, mỗi dòng chứa 2 ký tự, thể hiện những đường kết nối trực tiếp giữa hai khu vực não.

OUTPUT

Xuất ra một trong hai dòng:

"THIS BRAIN NEVER WAKES UP"

"WAKE UP IN, " n ", YEARS", trong đó n là số năm sau đó tất cả các khu vực bộ não đều chuyển sang trạng thái thức.

Sample Input	Sample Output
6 11 HAB AB AC AH BD BC BF CD CF CH DF FH	WAKE UP IN, 3, YEARS

10583. ĐẾM TÔN GIÁO

Bạn có nhiệm vụ đếm trong trường học của mình, có bao nhiêu tôn giáo có tín đồ. Trường của bạn có n sinh viên ($0 < n \le 50000$). Bạn không thể đến từng người để hỏi xem họ theo tôn giáo nào. Hơn thế nữa, nhiều sinh viên coi tôn giáo là vấn đề riêng tư và không muốn trả lời trực tiếp. Một cách giải quyết là hỏi m ($0 \le m \le n(n-$

1)/2) cặp sinh viên xem họ có cùng chung một tôn giáo hay không. Từ lượng dữ liệu thu thập được, tuy chưa xác định được ai theo tôn giáo nào, nhưng bạn ước đoán được số lượng tôn giáo cực đại có tín đồ trong trường. Mỗi sinh viên chỉ theo nhiều nhất một tôn giáo.

INPUT & OUTPUT

Có nhiều test. Dòng đầu tiên của mỗi test là hai số nguyên n và m. m dòng tiếp theo có dạng x y thể hiện sinh viên x và sinh viên y có cùng tôn giáo. Các sinh viên đánh số từ 1 đến n. Kết thúc khi n=m=0.

Với mỗi test, in ra số lượng tôn giáo cực đại

Sample Input	Sample Output
10 9 1 2 1 3 1 4 1 5 1 6 1 7 1 8 1 9 1 10 10 4 2 3 4 5 4 8 5 8 0 0	Case 1: 1 Case 2: 7

10608. ĐẾM BẠN

Trong trường chuyên có N học sinh. Thủa ban đầu có một số cặp bạn trong trường. Tuy nhiên sau một hồi chơi chung, "bạn của bạn cũng là bạn của ta" - nghĩa là nếu A và B là bạn, B và C là bạn thì A và C cũng là bạn. Khuê có nhiệm vụ đếm nhóm bạn có số lượng lớn nhất.

INPUT

Dòng đầu là số nguyên thể hiện số test. Sau đó là các bộ test. Ở mỗi bộ test, dòng đầu ghi 2 số nguyên N($1 \le N \le 30000$) và M ($0 \le M \le 500000$), tương ứng là số học sinh của trường và số cặp bạn bè ban đầu. M dòng tiếp theo ghi hai số A và B ($1 \le A \le N$, $1 \le B \le N$, $A \ne B$) mô tả A và B là bạn. (các dòng mô tả này có thể trùng lặp).

OUTPUT

Với mỗi test, in ra số lượng bạn bè trong nhóm bạn cực đại.

Sample Input	Sample Output
2 3 2 1 2 2 3 10 12 1 2 3 1 3 4 5 4 3 5 4 6	3 6

5 2 2 1 7 10 1 2 9 10 8 9	

10685. SINH TỒN THIÊN NHIÊN

Trong tự nhiên luôn luôn tồn tại các chuỗi thức ăn. Thực vật ở mức thấp nhất của chuỗi, các động vật nhỏ ăn thực vật và động vật to ăn động vật nhỏ. Có thể có chu trình trong chuỗi thức ăn, chẳng hạn như khi chết đi, xác động vật bị phân hủy thành mùn và trở thành chất bón (thức ăn) cho thực vật. Trong bài toán này, bạn có nhiệm vụ xác định chuỗi thức ăn lớn nhất giữa một nhóm sinh vật. Nếu A ăn B thì A và B thuộc về cùng một chuỗi.

INPUT

Có nhiều test, mỗi test có khuôn dạng sau. Dòng đầu của test chứa hai số nguyên **C** (**1 ≤ C ≤ 5000**), và **R** (**0 ≤ R ≤ 5000**), tương ứng là số lượng sinh vật và số lượng các quan hệ trong nhóm. Sau đó là **C** dòng mô tả tên các sinh vật, tên sinh vật không quá 30 ký tự, gồm chữ cái Latin (chữ thường và chữ hoa) cùng dấu gạch dưới (_). Sau đó là R dòng mô tả các mối quan hệ gồm tên sinh vật 1 và tên sinh vật 2, ở đây sinh vật 2 ăn thịt sinh vật 1. Không có sinh vật nào ăn thịt chính mình.

OUTPUT

Với mỗi bộ test, in ra chuỗi thức ăn kích thước cực đại.

Sample Input	Sample Output
5 2 caterpillar bird horse elefant herb herb caterpillar caterpillar bird 0 0	3

11503. BẠN ẢO

Những ngày nay, bạn có thể làm mọi thứ online. Ví dụ bạn dùng Facebook để kết bạn ảo. Với một số người, phát triển mạng xã hội (bạn của mình, bạn của bạn, bạn của bạn của bạn,….) là một sở thích gây nghiện. Ai đó sưu tập tem – còn những người nghiện này sưu tập bạn ảo. Nhiệm vụ của bạn là quan sát dữ liệu lấy từ Facebook và xác định kích thước mạng xã hội của một ai đó. Quan hệ bạn bè là 2 chiều, tức là nếu Vô Phúc là bạn của Bất Hiếu thì Bất Hiếu cũng là bạn của Vô Phúc.

INPUT

Dòng đầu tiên là số test. Mỗi test bắt đầu bằng một số nguyên F<100 000 – là số lượng quan hệ bạn bè. F dòng tiếp theo, mỗi dòng chứa tên hai người (cách nhau bằng dấu cách) có quan hệ bạn bè. Tên người là xâu không quá 20 ký tự.

OUTPUT

Mỗi khi có một quan hệ tình bạn mới được thiết lập, in ra kích thước của mạng xã hội của hai người vừa mới kết bạn với nhau.

Sample Input	Sample Output

1 3 Fred Barney Barney Betty Betty Wilma	2 3 4

11690. TIỀN BẠC

Trường chuyên tổ chức cho học sinh khối 10 đi chơi ở vùng núi Ba Vì tuyệt đẹp. Khi đi chơi, mọi người có vay nợ nhau một số tiền để mua sắm, đánh tiến lên, tá lả, hát karaoke. Trong hai ngày ở đó, có quá nhiều sự kiện kinh khiếp đã diễn ra (không thể viết ra ở đây). Chỉ biết hậu quả là rất nhiều câu nói "Tớ không bao giờ chơi với cậu nữa !" đã được cất lên (tất nhiên câu này lịch sự).

Quay về Mễ Trì, nhóm học sinh lớp 10 phát hiện mình chưa thanh toán sòng phẳng các khoản nợ với nhau. Nhưng giờ đây, việc giải quyết nợ nần khó khăn hơn rất nhiều bởi vì nhiều học sinh còn không muốn chạm mặt nhau – nói chi đến việc trả nhau tiền.

Bạn được yêu cầu giúp đỡ, bạn hỏi được mọi người tổng số tiền người đó nợ (hay bị nợ) và người này còn làm bạn với ai. Từ tất cả những thông tin này, bạn có thể kiểm tra xem liệu nhóm học sinh khối 10 có thể thanh toán sòng phẳng cho nhau được không (tiền chỉ có thể trao đổi giữa những cặp học sinh vẫn còn là bạn).

Input

Dòng đầu tiên là số nguyên N – là số test, mỗi test có khuôn dạng sau: Dòng đầu mỗi test là hai số nguyên n ($2 \le n \le 10000$), và m ($0 \le m \le 50000$), tương ứng là số lượng học sinh khối 10 và số lượng mối quan hệ bạn bè còn lại sau chuyến đi chơi. Sau đó là n dòng, mỗi dòng ghi một số nguyên o ($-10000 \le o \le 10000$) thể hiện người đó nợ bao nhiêu (hoặc bị nợ nếu $o < 0$). Tổng n số nguyên này bằng 0. Kế đó là m dòng có dạng x y ($0 \le x < y \le n-1$) thể hiện x và y còn là bạn bè.

Output

Với mỗi test, in ra một dòng `POSSIBLE` hoặc `IMPOSSIBLE`.

Ví dụ

Sample Input	Sample Output

Sample Input	Sample Output
2 5 3 100 -75 -25 -42 42 0 1 1 2 3 4	POSSIBLE IMPOSSIBLE

4 2 15 20 -10 -25 0 2 1 3	

11966. ĐẶT TÊN TINH VÂN

Cao nằm trên sân thượng và ngắm sao. Bởi vì Cao không biết về bất kỳ tinh vân nào cả, nên Cao tự quyết định: nếu 2 ngôi sao đủ gần nhau – chúng sẽ thuộc về cùng một tinh vân. Và bây giờ, Cao sẽ đếm số lượng tinh vân trên bầu trời. Hai ngôi sao thuộc cùng một tinh vân nếu khoảng cách giữa chúng trên mặt phẳng hai chiều không vượt quá D đơn vị.

INPUT

Dòng đầu là số lượng test T ($T \leq 50$). Dòng đầu của mỗi test là 2 số: số lượng ngôi sao N ($0 \leq N \leq 1000$) và khoảng cách D là số thực ($0.00 \leq D \leq 1000.00$). N dòng kế tiếp là tọa độ của các ngôi sao, có dạng X Y ($-1000.00 \leq X, Y \leq 1000.00$). Các số thực chỉ có nhiều nhất 2 chữ số sau dấu thập phân.

OUTPUT

Với mỗi test, in ra dòng `"Case T: N"`. Ở đây T là số thứ tự test (tính từ 1) và *N là số lượng tinh vân*.

Ví dụ

Sample Input	Sample Output

SAMPLE INPUT	SAMPLE OUTPUT
2 5 1.5 1.0 0.1 2.0 0.0 5.0 0.2 6.0 0.4 3.0 -0.1 3 4.0 121.12 254.06 645.04 301.85 912.49 568.96	Case 1: 2 Case 2: 3

MAP-SET

417.

Chúng tôi phát triển một chương trình mã hóa đơn giản, mã hóa các từ có ít hơn năm chữ cái thành một số nguyên.

Xét trên bảng chữ cái tiếng Anh {a, b, c, ..., z}. Trên bảng chữ cái này, xác định tập các từ hợp lệ. Trong từ hợp lệ, các chữ cái được sắp xếp theo chiều từ trái qua phải theo thứ tự từ điển. Ví dụ:

abc aep gwz là ba từ hợp lệ, trong khi aab cat thì không.

Người ta gắn số thứ tự cho mỗi từ hợp lệ, theo quy tắc sau:

a -> 1
b -> 2
.
.
z -> 26
ab -> 27
ac -> 28
.
.
az -> 51
bc -> 52
.
.
vwxyz -> 83.681

Chương trình của bạn đọc vào một loạt các từ có từ 1-5 chữ cái (mỗi từ trên một dòng). Đối với mỗi từ đọc vào, nếu từ đó là không hợp lệ, in ra số 0. Nếu từ hợp lệ, in ra số thứ tự của từ này.

INPUT

Đầu vào bao gồm một loạt các từ đơn, một trong mỗi dòng.

OUTPUT

Với mỗi từ, in ra đáp số cần tìm

Sample Input	Sample Output
z	26
a	1
cat	0
vwxyz	83681

484.

Cho một dãy các số nguyên. Xóa bỏ các ký tự trùng lặp, và in ra danh sách các số nguyên khác nhau trong dãy, và số lần xuất hiện của các số nguyên đó.

INPUT

Một danh sách các số nguyên (số dương, âm, và / hoặc 0).

OUTPUT

Nhiều dòng, mỗi dòng i là 1 cặp số nguyên (ai, bi), trong đó bi là số lần xuất hiện của ai trong dãy, các giá trị ai khác nhau và xuất hiện theo thứ tự xuất hiện trong dãy ban đầu.

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

SHORTEST PATH

1. BFS

UVA321. BIỆT THỰ MỚI

Mr.Black mới mua một ngôi nhà ở ngoại ô. Chỉ có một điều duy nhất làm ông không thoải mái: tuy mọi phòng đều có công tắc đèn nhưng bóng đèn mà công tắc điều khiển không nằm ở trong cùng phòng với công tắc đó. Dù người giúp việc của ông nghĩ đây là một ý kiến hay, ông cảm thấy những người thợ điện có vẻ ngốc khi họ nối dây điện giữa công tắc với bóng đèn. Một đêm, Mr.Black về nhà muộn. Khi đứng ở hành lang, ông nhận ra đèn ở mọi phòng đều tắt. Thật không may, Mr.Black sợ bóng tối, vì vậy ông không bao giờ vào một căn phòng không bật đèn. Sau một lúc suy nghĩ, Mr.Black quyết định tận dụng hệ thống bóng đèn. Ông cần phải đi đến phòng ngủ của mình và tắt đèn ở mọi căn phòng khác. Nhiệm vụ của bạn là phải viết chương trình, cho trước thông tin về căn nhà, xác định cách đi từ hành lang vào phòng ngủ khi ban đầu chỉ có đèn ở hành lang bật. Bạn không bao giờ được vào một căn phòng tối, và khi cuối cùng bạn vào được phòng ngủ thì đèn ở mọi căn phòng khác phải tắt. Nếu có nhiều đường tới phòng ngủ, bạn phải tìm ra đường đi với số bước bé nhất. Di chuyển từ một phòng này sang phòng khác, bật công tắc, tắt công tắc được coi là một bước.

INPUT

Input gồm nhiều test. Mỗi test bắt đầu với 1 dòng chứa 3 số nguyên r, d, s. r là số phòng của căn nhà, tối đa là 10. d là số cánh cửa nối giữa các phòng và s là số công tắc của căn nhà. Các căn phòng được đánh số từ 1 tới r. 1 là hành lang, và r là phòng ngủ. Tiếp đó là d dòng, mỗi dòng chứa 2 số i,j thể hiện có cửa từ phòng i tới phòng j. Sau đó là s dòng chứa 2 số k và l, thể hiện công tắc trong phòng k điều khiển bóng đèn trong phòng l. Giữa các test sẽ có một dòng trống. Input kết thúc khi r=d=s=0. Trường hợp này không cần xử lý.

OUTPUT

Với mỗi căn nhà, bắt đầu với số thự tự của test: "Villa #1", "Villa #2",... Nếu như có phương án cho căn nhà này, in ra số bước ít nhất cần để đi tới phòng ngủ và cách đi. Nếu không có phương án, in ra một dòng duy nhất "The problem cannot be solved." Output một dòng trống sau mỗi test.

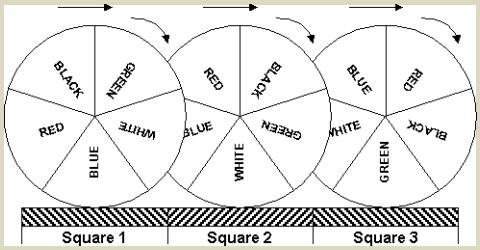
Sample Input	Sample Output
3 3 4 1 2 1 3 3 2 1 2 1 3 2 1 3 2	Villa #1 The problem can be solved in 6 steps: - Switch on light in room 2. - Switch on light in room 3. - Move to room 2. - Switch off light in room 1. - Move to room 3. - Switch off light in room 2.

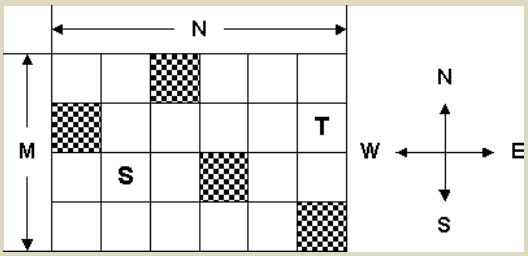
2 1 2 2 1 1 1 1 2 0 0 0	Villa #2 The problem cannot be solved.

UVA10047. XE MỘT BÁNH

Monocycle là một chiếc xe đạp chạy bằng một bánh và chiếc xe của chúng ta sẽ có một chút đặc biệt. Bánh xe sẽ được tô màu theo 5 màu như hình bên. các phần được tô màu bằng nhau, chính xác 72o. Người lái xe đạp phải lái xe một bánh trên các ô vuông. Di chuyển từ tâm của 1 ô đến tâm của một ô tiếp theo khiến bánh xe quay chính xác 72o quanh trục của chính nó. Các hiệu ứng được biểu diễn như hình. Khi xe ở vị trí ô thứ 1, điểm giữa của phần màu xanh chạm đất. Nhưng khi xe đi đến ô kế tiếp (square 2) thì điểm giữa của phần màu trắng chạm đất.

Một số ô có chứa vật chắn và chặn xe không cho đi vào chỗ đó. Xe bắt đầu ở một vị trí nào đó và cố gắng đến vị trí mục tiêu trong thời gian ngắn nhất. Từ bất kì một ô nào, có thể đi thẳng đến ô tiếp theo hoặc đứng yên ở ô đó nhưng phải xoay 90o sang trái hoặc phải. Mỗi một hành động như vậy được tính là 1 giây. Xe luôn bắt đầu quay mặt về phía Bắc và điểm chính giữa màu xanh lá hướng xuống mặt đất. Và tại ô mục tiêu, mặt màu xanh phải hướng xuống đất nhưng không cần quan tâm xe quay theo hướng nào. Trước khi xe chạy, bạn hãy tính xem thời gian tối thiểu để xe đi đến mục tiêu.





INPUT

Input có thể có nhiều test. Dòng đầu tiên của mỗi test là 2 số nguyên M và N, cho biết độ lớn của bản đồ. M dòng kế tiếp chứa N kí tự mỗi dòng. Kí tự `#' cho biết vị trí của một vật cản, phần còn lại xe được tự do di chuyển vào. Vị trí bắt đầu được đánh dấu bằng `S' và mục tiêu được đánh dấu bằng `T'. Input kết thúc khi M và N nhận giá trị 0.

OUTPUT

Mỗi test in ra một dòng chỉ thứ tự giống như trong Sample output. Nếu có thể đến dược vị trí mục tiêu thì in ra thời gian ngắn nhất (theo giây) giống như trong Sample output, Nếu không thể, in ra ``destination not reachable''. In một dòng trống giữa 2 test.

Sample Input	Sample Output
1 3 S#T 10 10 #S.....# #..#.##.##	Case #1 destination not reachable Case #2 minimum time = 49 sec

#.##.##.## .#....##.# ##.##..#.# #..#.##... #.....##. ..##.##... #.###...#. #.....###T 0 0	

UVA11101. ĐẠI SIÊU THỊ

Waterloo có hai siêu trung tâm mua sắm, mỗi cái trải rộng trên nhiều khu dân cư. Kim và Pat muốn mua sắm và đi bộ trong các trung tâm nhưng không muốn đi bộ giữa các trung tâm vì như thế không làm đúng nhiệm vụ: mua sắm. Bởi vậy, họ muốn biết khoảng cách nhỏ nhất giữa các trung tâm mua sắm.

Mỗi khu dân cư là một hình vuông đơn vị có ranh giới là các con phố và đại lộ. Các con phố chạy từ đông sang tây và đại lộ chạy từ bắc đến nam. Cả hai được đặt tên bằng các số nguyên liên tiếp nằm trong khoảng từ 0 → 2000 (đại lộ số nhỏ nằm phía tây đại lộ số cao và con phố số nhỏ nằm phía nam của con phố số cao). Các con phố và đại lộ có độ rộng bằng 0.

Mỗi trung tâm mua sắm là một tập hợp khu dân cư thông nhau. Ta định nghĩa hai khu dân cư thông với nhau qua một dãy các khu dân cư nếu hai khu dân cư liền kề có chung một cạnh. Các trung tâm mua sắm không giao nhau và không bao quanh các khu trống. Các khu trống không nằm trong các trung tâm thì thông nhau.

Input chứa nhiều test. Mỗi test mô tả hai trung tâm mua sắm. Mỗi trung tâm được thể hiện bằng số nguyên p ≥ 4, chu vi của trung tâm, theo sau bằng một hoặc nhiều dòng chứa p cặp (a, s) là tọa độ các giao lộ nằm trên chu vi, theo chiều kim đồng hồ. Kết thúc bằng test chứa số 0.

Với mỗi test, in ra một số nguyên d – giá trị quãng đường đi bộ nhỏ nhất giữa các trung tâm mua sắm, giả sử rằng Kim và Pat luôn luôn đi bộ dọc theo các con phố và các đại lộ.

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

2. DIJKKSTRA

UVA10801. THANG MÁY LÊN XUỐNG

Toà nhà chọc trời có không quá 100 tầng được đánh số từ 0 đến 99. Nó có n ($1 \leq n \leq 5$) thang máy để di chuyển lên hoặc xuống bằng các tốc độ khác nhau. Mỗi thang máy i trong khoảng từ 1 đến n cần Ti ($1 \leq Ti \leq 100$) giây để đi giữa 2 tầng cạnh nhau (lên hoặc xuống). Thang máy không nhất thiết phải dừng lại ở bất kỳ tầng nào, Tệ hơn nữa là một thang máy không nhất thiết phải phải dừng lại ở mọi tầng. Bạn đang ở tầng 0 và bạn muốn đi lên tầng k bằng cách nhanh nhất có thể. Giả sử bạn không cần đợi thang máy đầu tiên để đi vào và để cho đơn giản, việc chuyển thang máy ở một tầng nào đó mất đúng một phút. Dĩ nhiên, cả hai chiếc thang máy phải dừng lại ở tầng đó. Bạn không được sử dụng cầu thang bộ. Không ai đi cùng thang máy với bạn, nên bạn có thể dừng lại nếu bạn muốn. Tính thời gian tối thiểu để đi từ tầng 0 đến tầng k. Đi qua tầng k khi vẫn ở trong thang máy không tính là đến được.

OUTPUT

Input sẽ chứa nhiều test. Trong mỗi test, dòng đầu tiên sẽ bắt đầu bằng 2 số nguyên n và k. Dòng tiếp theo chứa các số T1 , T2,... ,Tn. n dòng tiếp theo chứa danh sách các số nguyên được sắp xếp – dòng đầu tiên sẽ là danh sách các tầng thang máy thứ nhất dừng, tiếp theo sẽ là danh sách các tầng thang máy thứ 2 dừng...

INPUT

Với mỗi test case , in ra 1 số - thời gian để đi từ tầng 0 đến k . Nếu không thể , in ra "IMPOSSIBLE".

Sample Input	Sample Output
2 30 10 5 0 1 3 5 7 9 11 13 15 20 99 4 13 15 19 20 25 30 2 30 10 1 0 5 10 12 14 20 25 30 2 4 6 8 10 12 14 22 25 28 29 3 50 10 50 100 0 10 30 40 0 20 30 0 20 50 1 1 2 0 2 4 6 8 10	275 285 3920 IMPOSSIBLE

Giải thích ví dụ đầu tiên, dùng thang máy đầu tiên để đến tầng 13 (130 giây) , đợi 60 giây để chọn thang máy 2 để đi tới tầng 30 (85 giây), tổng cộng 275 giây. Trong ví dụ thứ 2, đi thang máy 1 lên tầng 10, chuyển sang thang máy 2 và lên tầng 25, quay trở lại thang máy 1 và lên tầng 30. Tổng thời gian là 10*10 + 60 + 15*1 + 60 + 5*10 = 285 s. Trong ví dụ 3, dùng thang máy 1 lên tầng 30, dùng thang mấy 2 xuống tầng 20 và dùng thang máy 3 lên tầng 50. Trong ví dụ 1, thang máy 1 không lên tầng 1.

UVA11367. ĐỔ ĐẦY BÌNH

Sau khi trải qua một chuyến nghỉ hè quanh Châu Âu, bạn nhận ra rằng giá xăng giữa các thành phố là khác nhau. Liệu có thể tiết kiệm tiền xăng nếu bạn khôn ngoan hơn không? Để giúp đỡ các du khách khác hoặc có thể là chính bạn trong lần sau, hãy viết một chương trình xác định đường đi rẻ nhất giữa các thành phố bằng cách đổ xăng trên đường. Coi như mỗi chiếc xe đi đơn vị khoảng cách tốn một đơn vị xăng, bình xăng ban đầu rỗng.

INPUT

Dòng đầu tiên là hai số nguyên $1 \le n \le 1000$ và $0 \le m \le 10000$ tương ứng là số thành phố và số đường đi. Sau đó là một dòng có n số nguyên dương $1 \le p_i \le 100$, trong đó p_i là giá xăng tại thành phố thứ i. Theo sau là m dòng với ba số nguyên $0 \le u$, $v < n$ và $1 \le d \le 100$, cho biết có đường đi giữa u và v với độ dài d. Sau đó là một dòng $1 \le q \le 100$, số lượng các câu hỏi, q dòng tiếp theo với ba số nguyên $1 \le c \le 100$, s và e, trong đó c là dung tích bình xăng, s là thành phố xuất phát, e là thành phố kết thúc.

OUTPUT

Với mỗi câu hỏi, in ra giá tiền của đường đi rẻ nhất giữa thành phố s và e, sử dụng một chiếc xe với dung tích xăng đã cho, hoặc "impossible" nếu không có dường đi giữa s và e với chiếc xe đã cho.

Sample Input	Sample Output
5 5 10 10 20 12 13 0 1 9 0 2 8 1 2 1 1 3 11 2 3 7 2 10 0 3 20 1 4	170 impossible

UVA11492. BABEL

John và Mary là 2 anh em, và họ vô cùng hăng hái với khóa học ngoại ngữ sắp tới. Mỗi người đều tham dự một vài khóa học. Sau đó, họ bàn luận với nhau về từ vựng, ngữ pháp và các nền văn hóa của các nước. Họ nhận ra rằng có một số từ được dùng trong nhiều loại ngôn ngữ khác nhau, kể cả một số từ được dùng với các ngôn ngữ khác nhau. Vì dụ, từ "Amigo" trong tiếng Bồ Đào Nha và Tây Ban Nha đều có cùng nghĩa trong khi đó từ "date" trong tiếng Anh có nghĩa là ngày tháng trên lịch, còn trong tiếng Pháp lại có nghĩa là hoa quả. Còn "red" trong tiếng Tây Ban Nha là mạng và tiếng Anh lại là màu đỏ.

Vô cùng vui sướng vì phát hiện này, hai ạnh em quyết định viết ra tất cả các từ được dùng trong nhiều loại ngôn ngữ mà họ có thể nghĩ ra được, phụ thuộc vào các cặp ngôn ngữ cho trước. John thách đố Mary, cho trước 1 cặp ngôn ngữ viết ra 1 dãy các từ sao cho từ đầu tiên của dãy thuộc ngôn ngữ thứ nhất và từ cuối cùng thuộc ngôn ngữ thứ 2. 2 từ liền kề nhau đều là từ vựng của cùng một ngôn ngữ. Ví dụ như 2 ngôn ngữ Bồ Đào Nha và Pháp, Mary có thể viết ra các từ sau "amigo actual date" (Portuguese/Spanish, Spanish/English, English/French). John vô cùng bất ngờ khi thấy Mary dễ dàng giải quyết vấn đề . Vô cùng khó chịu với em gái mình John quyết định tăng độ phức tạp của vấn đề Mary phải tìm cách in ra độ dài xâu bé nhất thỏa mãn yêu cầu (xâu là tập hợp các từ, không tính các dấu cách phân biệt giữa các từ) và 2 từ liên tiếp phải có chữ cái đầu tiên khác nhau. John đã làm một cuộc tìm kiếm trên Internet và liệt kê ra một loạt các từ và bắt đầu thách đố Mary. Vì có thể có rất nhiều cách giải nên John chỉ yêu cầu Mary đưa ra độ dài xâu nhỏ nhất. Bạn có thể giúp đỡ Mary được không?

INPUT

Gồm nhiều test. Dòng đầu tiên là số m ($1 \le m \le 2000$), biểu thị tổng số lượng từ John đưa ra. Dòng tiếp theo là 2 xâu O và D biểu thị 2 ngôn ngữ. m dòng tiếp theo gồm các xâu i1,i2, và p thể hiện từ xuất hiện trong 2 ngôn ngữ (i1 và i2 khác nhau). Mỗi xâu có độ dài từ 1 đến 50, và chỉ được viết bằng các kí tự thường không viết hoa. Mỗi cặp ngôn ngữ có thể có nhiều từ nhưng

một từ không được phép xuất hiện nhiều lần trong một test. Số không kết thúc Input.

OUTPUT

Với mỗi test, chương trình của bạn phải in ra một số nguyên thể hiện độ dài xâu nhỏ nhất.

Sample Input	Sample Output
4 portugues frances ingles espanhol red espanhol portugues amigo frances ingles date espanhol ingles actual 4 portugues alemao ingles espanhol red espanhol portugues amigo frances ingles date espanhol ingles actual 6 portugues frances ingles espanhol red espanhol portugues amigo frances ingles date frances espanhol la portugues ingles a espanhol ingles actual 0	12 impossivel 5

UVA341. ĐI LIÊN TỤC

Ghét phải đợi trước đèn tín hiệu giao thông khi lái xe, David chuẩn bị rất nhiều bản đồ các khu vực mình thường xuyên đi và đo độ trễ trung bình (theo đơn vị giây) ở các giao lộ trên những khu vực đó. David muốn tìm các tuyến đường giữa những điểm cụ thể trên các giao lộ sao cho thời gian đợi tại các đèn tín hiệu là nhỏ nhất (bất kể việc David có thể phải lái xe dài hơn để tránh đèn). Chú ý các tuyến đường đi từ giao lộ i đến giao lộ j là đường một chiều. Giữa hai giao lộ chỉ có tối đa một đường nối. Hãy giúp David làm được điều đó.

INPUT

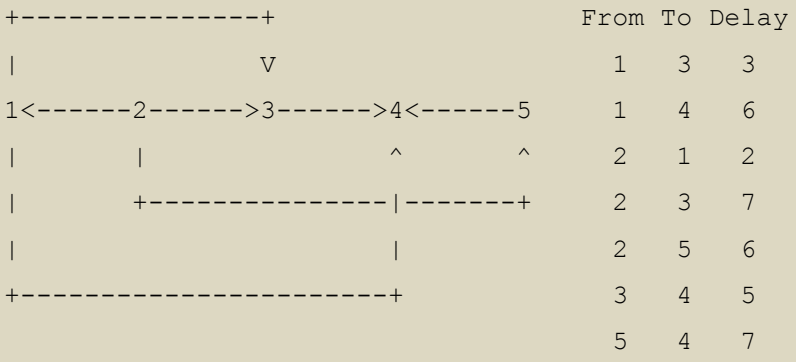
Gồm nhiều test. Dòng đầu tiên là số N1 (1≤m≤10), biểu thị tổng số giao lộ trong khu vực, các giao lộ được đánh số tuần tự, bắt đầu từ 1. Với mỗi giao lộ, input chỉ ra số lượng các tuyến phố có đầu mút là giao lộ đó, ứng với mỗi tuyến phố, có 2 thông số: đầu mút giao lộ bên kia và độ trễ trung bình David phải đợi ở giao lộ đó. Sau phần miêu tả này là 2 con số - tên 2 giao lộ mà David chọn làm điểm khởi đầu và điểm kết thúc. Số 0 duy nhất dùng để đánh dấu kết thúc input.

OUTPUT

Với mỗi test, in ra trên một dòng danh sách các giao lộ mà David sẽ chọn trên tuyến đường có độ trễ trung bình nhỏ nhất và thời gian đợi trung bình khi đi trên tuyến đường đó.

Ví dụ

Giả sử David muốn đi từ giao lộ 2 đến giao lộ 6 trên bản đồ sau (ứng với sample test 1):



From	To	Delay
1	3	3
1	4	6
2	1	2
2	3	7
2	5	6
3	4	5
5	4	7

Sample Input	Sample Output
5 2 3 3 4 6 3 1 2 3 7 5 6 1 4 5 0 1 4 7 2 4 2 1 2 5 1 1 6 1 2 7 4 2 5 3 13 4 8 5 18 2 3 7 6 14 1 6 6 2 3 5 5 9 3 6 2 7 9 4 6 1 7 2 0 1 7 0	Case 1: Path = 2 1 4; 8 second delay Case 2: Path = 1 2; 5 second delay Case 3: Path = 1 2 3 6 7; 20 second delay

UVA929. MÊ CUNG SỐ

Xét một mê cung số là một bảng 2 chiều, trong các ô chứa các số trong khoảng 0 đến 9 như trong hình vẽ bên dưới. Người ta có thể đi trong mê cung theo một trong bốn hướng Đông Tây Nam Bắc. Giả sử số bên trong ô là giá của ô đó. Hãy tìm đường đi có giá tối thiểu từ ô góc trên bên trái đến ô góc dưới bên phải của mê cung có kích thước N×M trong đó $1 \le N, M \le 999$.

0	3	1	2	9
7	3	4	9	9
1	7	5	5	3
2	3	4	2	5

INPUT

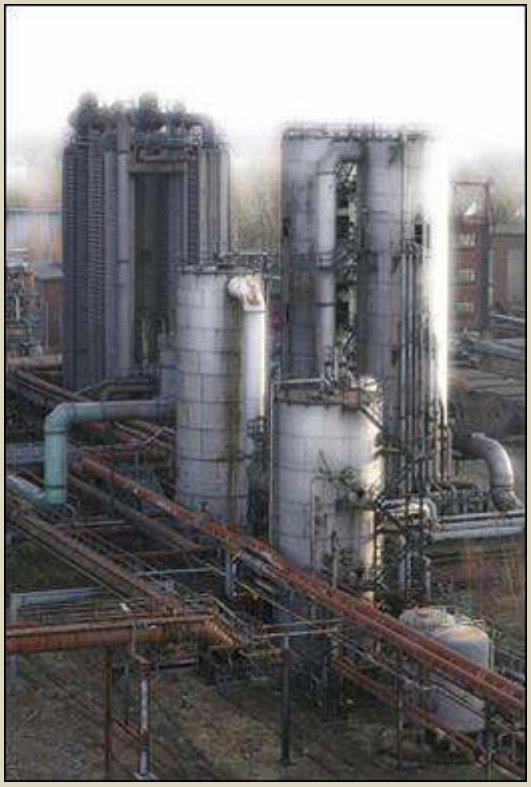
Input có nhiều test. Dòng đầu tiên ghi số test. Ở mỗi test, dòng đầu tiên là số hàng N, dòng kế tiếp là số cột M. N dòng kế tiếp, mỗi dòng có đúng M số cách nhau bằng dấu cách.

OUTPUT

Với mỗi test, in ra giá trị tối thiểu.

Sample Input	Sample Output
2 4 5 0 3 1 2 9 7 3 4 9 9 1 7 5 5 3 2 3 4 2 5 1 6 0 1 2 3 4 5	24 15

UVA12138 - Chemical Plant



Xử lý hóa chất là việc làm mạo hiểm – đặc biệt với hoạt chất (chất dễ có phản ứng hóa học). Hoạt chất trở nên nguy hiểm nếu không được lưu chứa trong môi trường phù hợp. Do đó cần đến những thiết bị đặc biệt để hỗ trợ xử lý tức thì khi di chuyển hoạt chất giữa các khu vực của nhà máy. Tuy nhiên những thiết bị như thế rất đắt nên nhà máy hóa chất xử dụng mạng lưới để chuyển – với một mức tổn hao nào đó.

Ở nhà máy hóa chất của chúng ta, có V pit. Pit là nơi chứa hóa chất. Để giảm chi phí, pit không được trang bị môi trường xử lý hoạt chất. Do đó cứ 1s ở trong pit, hoạt chất sẽ bị phân rã 1 milligram. Các pit được đánh số từ 1 đến V. Có một bể chứa an toàn ở pit S. Hóa chất chuyển từ pit S sang bể chứa an toàn này không tốn thời gian. Có W miligram hoạt chất đến pit 1 vào thời điểm 0. Hoạt chất này cần chuyển đến bể chứa an toàn. Cửa mở vào bể chứa an toàn mở vào thời điểm T. Do đó, nếu hoạt chất đến bể chứa trước T, có thể hoạt chất sẽ phân rã một lượng nào đó trước khi được đưa vào bể chứa – nhưng nếu đến sau T, hoạt chất sẽ không được đưa vào bể chứa. Có E ống truyền trong mạng truyền dẫn và chúng có thể được sử dụng để lưu chuyển hoạt chất (hoạt chất không bị phân rã trong đường ống). Mỗi ống truyền kết nối 2 pit: pit nhập là pit s và pit thoát là pit d. Với mỗi ống t, van mở của đầu pit nhập sẽ mở ra trong đúng 1s nào đó, từ sst đến sct (sst ≤ sct) – nhưng chúng ta không biết chính xác thời điểm cụ thể. Tương tự, van mở của đầu pit thoát sẽ mở trong đúng một giây trong khoảng dst & dct (dst ≤ dct). Vì bạn không rõ thời điểm van mở, nên nếu bạn muốn chuyển hóa chất từ pit s đến pit d bằng ống truyền t thì hóa chất phải đến pit s trước thời điểm sst. Nếu hóa chất đến pit x nào đó qua ống y với khoảng mở van là dsy & dcy (dsy ≤ dcy) và sau đó rời pit x bằng ống z với khoảng mở van là ssz & scz (ssz ≤ scz) thì mức độ phân rã cực đại là scz – dsy. Chú ý rằng, khi chọn một đường ống để thoát khỏi một pit, bạn phải chọn ống có khoảng mở không muộn hơn thời điểm muộn nhất mà hóa chất đi vào pit này. Trong test mẫu 3, hoạt chất luôn rời pit 2 qua ống thứ 3 – kể cả khi nó đến pit 2 ở thời điểm 15. (. Please note that, while choosing the tube to get out from a pit, you must always select a tube with it's opening interval beginning no sooner than the latest possible time for the chemical to enter that pit).

Hãy viết chương trình tìm cách chuyển hóa chất từ pit 1 đến bể chứa sao cho mức độ tiêu hao bé nhất (lượng phân rã nhỏ nhất). Chú ý hóa chất không thể có trọng lượng âm.

INPUT

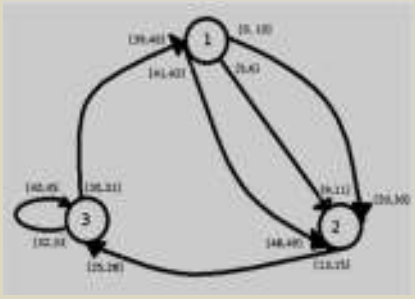
Có nhiều test. Ở mỗi test, dòng đầu là ba số nguyên V (1≤V≤50,000), E (1≤E≤100,000) & W (1≤W≤2,000,000,000). Dòng kế tiếp có hai số nguyên S (1≤S≤V) & T (0≤T≤2,000,000,000). Sau đó là E dòng mô tả các đường ống. Mỗi ống t được mô tả bằng 6 số nguyên, s (1≤s≤V), d (1≤s≤V), sst, sct, dst, dct (0≤sst≤sct<dst≤dct≤2,000,000,000). Input kết thúc với V = E = W = 0. Không cần xử lý test này.

OUTPUT

Với mỗi test, in ra dòng Plant C: L trong đó C là số thứ tự của test và L là khối lượng hóa chất ở bể chứa ở thời điểm T.

Sample Input	Sample Output
3 6 50 2 50 1 2 0 10 20 30 1 2 5 6 9 11 2 3 13 15 25 28 3 3 32 33 40 45 3 1 30 31 39 40 1 2 41 42 48 49 5 13 20 3 1000 3 3 41 41 999 1000 3 3 39 40 1000 1000 5 4 25 25 30 30 1 2 2 2 6 6 1 2 1 1 8 8 2 2 7 7 13 13 2 2 8 8 15 15 2 3 14 14 20 20 2 3 16 16 20 20 4 3 30 30 40 40 4 3 32 32 41 41 3 5 21 21 25 25 3 5 22 22 25 25 3 3 50 3 30 1 2 5 10 15 25 2 3 20 20 30 30 2 3 25 25 30 30 0 0 0	Plant 1: 27 Plant 2: 15 Plant 3: 30

Hình vẽ dưới minh họa test mẫu 1. Hoạt chất khởi đầu ở pit 1 vào thời điểm 0. Sau đó nó đến pit 2 theo cung ([5,6] [9,11]). Độ hao hụt tối thiểu ở pit 1 là 6. Sau đó chuyển từ pit 2 qua pit 3 qua cạnh ([13,15] [25,28]). Độ hao hụt tối thiểu là 6. Sau đó từ 3 qua 1 qua ống ([30,31] [39,40]). Độ hao hụt tối thiểu là 6. Cuối cùng từ pit 1 qua pit 2 theo ống ([41,42] [48,49]). Độ hao hụt tối thiểu là 3



UVA1202. Finding Nemo

Nemo là cậu bé nghịch ngợm. Một ngày, cậu lang thang đi chơi và bị lạc không thể tìm đường về nhà. Do đó, cậu gửi tín hiệu cho ông bố Marlin để xin trợ giúp. Sau khi kiểm tra bản đồ, Marlin thấy bản đồ giống như một mê cung với các bức tường và các cửa. Tất cả các bức tường đều song song với trục tung và trục hoành, độ dầy của bức tường tính là 0. Tất cả các cửa gắn trên tường đều mở và có độ dài 1. Marlin không thể đi xuyên tường trừ phi có cửa. Vì bước qua cánh cửa có thể gặp nguy hiểm (có ác quỷ) nên Marlin muốn đi qua ít cửa nhất có thể. Hình dưới minh họa một mê cung ví dụ và tuyến đường Marlin đi tìm Nemo. Chúng ta giả định ban đầu Marlin ở vị trí (0,0) trên mặt phẳng tọa độ. Với vị trí của Nemo và vị trí bức tường và cửa, hãy viết chương trình tính số lượng cửa tối thiểu mà Marlin phải đi qua để cứu Nemo.

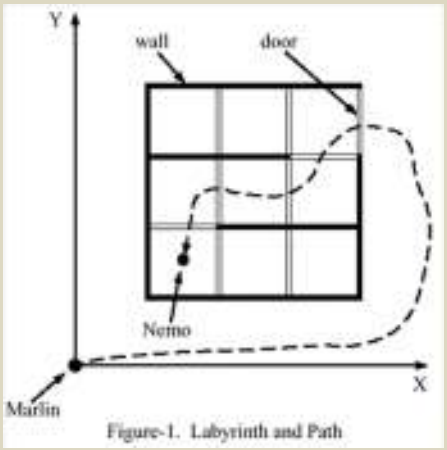


Figure-1. Labyrinth and Path

INPUT

Có nhiều test. Ở mỗi test, dòng đầu là hai số nguyên M và N, tương ứng là số lượng tường và số lượng cửa trong mê cung. Sau đó là M dòng, mỗi dòng có 4 số nguyên dạng x y d t, trong đó: (x,y) là tọa độ góc dưới bên trái của tường. t là chiều dài của bức tường, d là hướng của bức tường (nhận giá trị 0 – bức tường song song với trục X và nhận giá trị 1 – bức tường song song với trục Y). Tọa độ nằm trong khoảng [1,199]. Kế đó là N dòng, mỗi dòng có dạng x y d. x, y và d có ý nghĩa tương tự như trên, vì cửa có chiều dài cố định là 1 nên không cần t. Dòng cuối cùng của test là hai số thực f1 f2 là tọa độ của Nemo (không nằm trên tường và cửa). Kết thúc input là M = -1 và N = -1 – không phải xử lý.

OUTPUT

Với mỗi test, in ra số cửa tối thiểu mà Marlin phải đi để gặp được Nemo. Nếu không gặp được Nemo, in ra giá trị -1.

Sample Input	Sample Output
8 9 1 1 1 3 2 1 1 3 3 1 1 3 4 1 1 3 1 1 0 3 1 2 0 3 1 3 0 3 1 4 0 3 2 1 1 2 2 1 2 3 1 3 1 1 3 2 1 3 3 1 1 2 0 3 3 0 4 3 1 1.5 1.5 4 0 1 1 0 1 1 1 1 1 2 1 1 1 1 2 0 1 1.5 1.7 -1 -1	5 -1

UVA10166. Travel

Xiao Ming, một cậu bé thông minh và lười biếng, rất ưa ngủ và lập trình. Ming rất ghét đi xe điện. Ấy vậy mà sáng sớm ngày mai, Ming phải đi từ Liuzhou đến Xi'an để tham gia kỳ thi NOI2001. Lo đến muộn không được thi, Ming tìm kiếm chuyến tàu điện để đến Xian sớm nhất có thể. Nhưng cậu ghét đến trạm xe điện sớm, do đó nếu có nhiều cách đi đến đích cùng lúc, cậu chọn cách đi mà thời điểm xuất phát muộn nhất. Ming yêu cầu bạn giúp để cậu ta có thể ngủ dậy muộn nhất có thể. Bạn có danh sách lịch trình của các tuyến tầu hỏa, bạn phải tìm đường đi nhanh nhất nhưng có thời thời gian xuất phát muộn nhất đi từ điểm này đến điểm kia. Chú ý: có thể coi thời gian chuyển từ tàu này sang tàu khác bằng 0.

INPUT

Có nhiều test, mỗi test có 3 phần. Phần 1 là tên tất cả các thành phố trong mạng lưới tàu hỏa. Phần này bắt đầu bằng một số nguyên C ($1 \le C \le 100$), sau đó là C dòng, mỗi dòng là tên một thành phố (tên của thành phố không vượt quá 20 ký tự). Phần 2 mô tả lịch trình các tuyến tàu trong ngày. Dòng đầu là số $T \le 100$, sau đó là T phần mô tả các chuyến tàu. Mỗi chuyến tàu bắt đầu bằng một số nguyên $ti < 100$, sau đó là ti dòng mô tả thời gian, mỗi dòng gồm 2 phần: phần thời gian và tên thành phố - có ý nghĩa là hành khách có thể lên hay xuống tàu tại thành phố đó, vào thời điểm đó. Thời gian có khuôn dạng hhmm (giờ - phút). Phần 3 có 3 dòng: Dòng 1 là thời điểm xuất phát sớm nhất có thể, dòng 2 và dòng 3 tương ứng là tên thành phố khởi hành và đích đến. Kết thúc test khi C=0.

OUTPUT

Với mỗi test, in ra trên một dòng 2 thời điểm dưới dạng hhmm: thời điểm xuất phát và thời điểm đến. Nếu không đến được, in ra dòng "No connection".

Sample Input	Sample Output
3 Liuzhou Guilin Xian 3 2 0900 Liuzhou 1200 Guilin 2 1200 Guilin 2200 Xian 3 0900 Liuzhou 1200 Guilin 2300 Xian 0800 Liuzhou Xian 3 Liuzhou Guilin Xian 1 3 0900 Liuzhou 1200 Guilin 2300 Xian 1000 Liuzhou Xian 0	0900 2200 No connection

UVA10269. Adventure of Super Mario

Sau khi cứu được công chúa, Super Mario cần tìm đường về nhà – dĩ nhiên cùng với công chúa :-) Quen thuộc với 'Super Mario World', Mario không cần đến bản đồ, nhưng muốn tìm tuyến đường tốt nhất để tiết kiệm thời gian. Có A Làng, đánh số từ 1 đến A và B Lâu Đài, đánh số từ A+1 đến A+B. Có những con đường hai chiều kết nối giữa chúng. Mario sống ở Làng 1. Lâu đài Mario xuất phát là A+B. Hai địa điểm được nối với nhau bởi nhiều nhất 1 con đường và một địa điểm không có con đường nào dẫn đến chính nó. Mario đã đo chiều dài của tất cả các con đường – nhưng không muốn đi bộ vì Mario đi bộ rất chậm – 1 đơn vị thời gian đi được 1 đơn vị khoảng cách. Thật may mắn, ở lâu đài cứu được công chúa, Mario tìm thấy một đôi ủng thần kỳ. Nếu đi ủng, Mario có thể chạy siêu tốc từ địa điểm này sang địa điểm khác trong thời gian bằng 0 (tất nhiên là khi chạy nhanh, Mario sẽ cõng theo công chúa). Bởi vì trong các Lâu đài có bẫy, nên Mario sẽ không bao giờ chạy siêu tốc qua bất kỳ Lâu đài nào, Mario sẽ dừng ngay khi gặp một Lâu đài trên đường đi. Hơn thế nữa, Mario bắt đầu, kết thúc quá trình chạy Siêu tốc ở các Làng và Lâu đài. Tuy nhiên, đôi ủng quá cũ, nên Mario không thể dùng nó để chạy quá L km trong một lần chạy và Mario không thể dùng quá K lần.

INPUT

Dòng đầu tiên là số nguyên T, là số lượng test. ($1 \le T \le 20$). Mỗi test bắt đầu bằng 5 số nguyên A, B, M, L và K tương ứng là số Làng, số Lâu đài ($1 \le A,B \le 50$), số lượng con đường, khoảng cách tối đa trong một lần chạy siêu tốc. M dòng tiếp theo, mỗi dòng chứa ba số nguyên Xi, Yi và Li – có con đường có chiều dài Li nối giữa hai địa điểm Xi và Yi. ($1 \le Li \le 100$).

OUTPUT

Với mỗi test, in ra thời gian tối thiểu để Mario về nhà (cùng công chúa).

Sample Input	Sample Output
1 4 2 6 9 1 4 6 1 5 6 10 4 5 5 3 5 4 2 3 4 1 2 3	9

UVA10278. TRẠM XE CỨU HỎA

Thành phố có một số trạm cứu hỏa, một số thị dân phàn nàn rằng khoảng cách từ nhà họ đến trạm cứu hỏa gần nhất là quá xa, do đó cần xây dựng thêm trạm cứu hỏa mới. Bạn phải chọn vị trí trạm cứu hỏa để giảm thiểu khoảng cách từ nhà tới trạm cứu hỏa của các cư dân hay phàn nàn. Thành phố có 500 giao lộ, nối với nhau bằng các cung đường có độ dài khác nhau. Có thể bỏ qua khoảng cách từ nhà hay trạm cứu hỏa đến các giao lộ - nên có thể xem nhà hay trạm cứu hỏa ở ngay các giao lộ. Mỗi giao lộ nối với không quá 20 cung đường.

INPUT

Dòng đầu tiên là một số nguyên – thể hiện số test. Sau đó là một dòng trống. Giữa các test có một dòng trống. Ở mỗi test, dòng đầu tiên là hai số nguyên dương f ($f \le 100$ – là số trạm cứu hỏa) và i ($i \le 500$ – là số giao lộ). Các giao lộ được đánh số lần lượt từ 1 đến i. f dòng tiếp theo, mỗi dòng là số thứ tự của giao lộ chứa trạm cứu hỏa. Sau đó là một số dòng, mỗi dòng chứa 3 số nguyên: a b c – thể hiện có cung đường độ dài c nối 2 giao lộ $a \ne b$. Các cung đường là hai chiều.

OUTPUT

Với mỗi test, in ra một số nguyên duy nhất – là số thứ tự bé nhất của giao lộ có thể đặt trạm cứu hỏa sao cho có thể giảm thiểu khoảng cách xa nhất từ bất kỳ giao lộ nào đến trạm cứu hỏa gần nhất. Mỗi test cách nhau một dòng trống.

Sample Input	Sample Output
1 1 6 2 1 2 10 2 3 10 3 4 10 4 5 10 5 6 10 6 1 10	5

UVA 10389 – Subway

You have just moved from a quiet Waterloo neighbourhood to a big, noisy city. Instead of getting to ride your bike to school every day, you now get to walk and take the subway. Because you don't want to be late for class, you want to know how long it will take you to get to school.

You walk at a speed of 10 km/h. The subway travels at 40 km/h. Assume that you are lucky, and whenever you arrive at a subway station, a train is there that you can board immediately. You may get on and off the subway any number of times, and you may switch between different subway lines if you wish. All subway lines go in both directions.

INPUT

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.

Input consists of the x,y coordinates of your home and your school, followed by specifications of several subway lines. Each subway line consists of the non-negative integer x,y coordinates of each stop on the line, in order. You may assume the subway runs in a straight line between adjacent stops, and the coordinates represent an integral number of metres. Each line has at least two stops. The end of each subway line is followed by the dummy coordinate pair -1,-1. In total there are at most 200 subway stops in the city.

OUTPUT

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.

Output is the number of minutes it will take you to get to school, rounded to the nearest minute, taking the fastest route.

Sample Input	Sample Output
1 0 0 10000 1000 0 200 5000 200 7000 200 -1 -1 2000 600 5000 600 10000 600 -1 -1	21

UVA 10603 – Fill

There are three jugs with a volume of a, b and c liters. (a, b, and c are positive integers not greater than 200). The first and the second jug are initially empty, while the third

is completely filled with water. It is allowed to pour water from one jug into another until either the first one is empty or the second one is full. This operation can be performed zero, one or more times.

You are to write a program that computes the least total amount of water that needs to be poured; so that at least one of the jugs contains exactly d liters of water (d is a positive integer not greater than 200). If it is not possible to measure d liters this way your program should find a smaller amount of water d' < d which is closest to d and for which d' liters could be produced. When d' is found, your program should compute the least total amount of poured water needed to produce d' liters in at least one of the jugs.

INPUT

The first line of input contains the number of test cases. In the next T lines, T test cases follow. Each test case is given in one line of input containing four space separated integers - a, b, c and d.

OUTPUT

The output consists of two integers separated by a single space. The first integer equals the least total amount (the sum of all waters you pour from one jug to another) of poured water. The second integer equals d, if d liters of water could be produced by such transformations, or equals the closest smaller value d' that your program has found.

Sample Input	Sample Output
2 2 3 4 2 96 97 199 62	2 2 9859 62

UVA10986. Sending email

There are n SMTP servers connected by network cables. Each of the m cables connects two computers and has a certain latency measured in milliseconds required to send an email message. What is the shortest time required to send a message from server S to server T along a sequence of cables? Assume that there is no delay incurred at any of the servers.

INPUT

The first line of input gives the number of cases, N. N test cases follow. Each one starts with a line containing n (2<=n<20000), m (0<=m<50000), S (0<=S<n) and T (0<=T<n). S!=T. The next m lines will each contain 3 integers: 2 different servers (in the range [0, n-1]) that are connected by a bidirectional cable and the latency, w, along this cable (0<=w<=10000).

OUTPUT

For each test case, output the line "Case #x:" followed by the number of milliseconds required to send a message from S to T. Print "unreachable" if there is no route from S to T.

Sample Input	Sample Output
3 2 1 0 1 0 1 100 3 3 2 0 0 1 100 0 2 200 1 2 50 2 0 0 1	Case #1: 100 Case #2: 150 Case #3: unreachable

UVA11377. Airport Setup

There are N cities in the kingdom 'Nameless' and among them K cities have airport. There the airline companies are willing to setup their flight directly between M pairs of the cities. They think that it will not be profitable for them to run flights between other pairs of cities. Note that if there are airports in both of the cities of any of their M pairs only then they will run flight between that pair otherwise not. Because of this policy of the airline companies and also because all the cities don't have airport many people can't travel between their desired cities. So some people of the country have demanded that the king should do something so that they can travel to their destinations. There are Q demands which came to the king. Every demand is a pair (x, y) which means some people want a route between city x and city y. The king really wants to satisfy the demands but he does not want to setup too many airports. So for every demand (x, y) the king wants to know the minimum number of airports he needs to setup to establish a route between city x and city y.

INPUT

The first line of the input contains X the number of test cases which is at most 10. Each case starts with three numbers N ($1 \leq N \leq 2{,}000$), M ($0 \leq M \leq 10{,}000$) and K ($1 \leq K \leq N$). Here N is the number of cities, M is the number of direct air routes the airlines companies are willing to setup and K is the number of cities which has airport. Then there will be a line containing K integers which denote the name of the cities which have airport (city name ranges from 1 to N). Each city is separated by a single space. Then there will be M lines each of which contains one pair "a b". It means that the airline companies are willing to run their flights between city a and b. The next line contains Q ($1 \leq Q \leq 50$) which is the number of demands came to the king. Then there will be Q lines each containing a pair "x y" which means that the king should build minimum airport to establish a route between city x and city y. There is a blank line between every consecutive test case.

OUTPUT

For every test case you have to output the case number first (See the sample). Then you have to output Q lines each containing the minimum number of airports the king has to setup to satisfy that demand. If it's impossible to satisfy that demand then output -1. You have to output a blank line after each test case.

Sample Input	Sample Output
1 6 4 4 1 2 5 6 1 2 3 5 2 4 4 5 3 1 2 1 3 1 6	Case 1: 0 2 -1

UVA 11833. Route Change

The road system of a country connects all N cities so that it is possible to travel between any pair of cities using existing roads. Each road connects two different cities, is two-way and one has exactly one toll booth (a toll is paid for both directions of traffic). Roads intersect only in a city and no pair of cities is interconnected by two or more roads.

Dias Tranport offers a one-day parcel delivery service between cities. Each parcel must be transported from a city A to another city B. The management of Dias Transport defines, for each parcel, a service route, consisting of C cities and C - 1 roads: the first city on the service route is the origin of the parcel, the final city is the destination of the parcel. The service route never passes twice through the same city, and the vehicle chosen to deliver a parcel can only travel by the service route defined.

One day, however, a vehicle broke down and was taken for repairs in a city that was not among the cities in its service route. The management of Dias Transport wants to know which is the lowest total cost, in terms of tolls, for delivering the parcel (that is, to take the vehicle from the city it was repaired to the destination city), but with

an additional constraint: if at some point the vehicle reaches one of the cities that make up its service route, it should go back to following its service route.

INPUT

The input contains several test cases. The first line of a test case contais four integers N, M, C and K ($4 \leq N \leq 250$, $3 \leq M \leq N \times (N - 1)/2$, $2 \leq C \leq N - 1$ e $C \leq K \leq N - 1$), representing, respectively, the number of cities, the number of roads, the number of cities in the service route and the city where the vehicle was taken for repair. The cities are identified by integers from 0 to N - 1. The service route is 0, 1,..., C - 1, that is, the origin is 0, from 0 goes to 1, from 1 to 2 and so on, until the destination C - 1. The next M lines describe the road system. Each of those lines describes one road and contains three integers U, V and P ($0 \leq U, V \leq N - 1$, $U \neq V$, $0 \leq P \leq 250$), indicating that there exists a road connecting cities U and V with a toll of cost P.

The last test case is followed by a line containing four zeros separated by blank spaces.

OUTPUT

For each test case, your program should print a single line, containing a single integer, the minimum total toll cost for the vehicle to reach the destination city.

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