Problem A. Cây DFS

Time Limit	2000 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

Được cung cấp một đồ thị vô hướng có trọng số kết nối mà không có vòng lặp tự tham chiếu và cạnh lặp lại. Đồ thị chứa n đỉnh và m cạnh.

Đối với mỗi cạnh (u, v), tìm trọng số nhỏ nhất có thể của cây bao trùm mà chứa cạnh (u, v).

Trọng số của cây bao trùm là tổng trọng số của tất cả các cạnh được bao gồm trong cây bao trùm.

Nhập

Dòng đầu tiên chứa hai số nguyên n và m ($1 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 2 \cdot 10^5$) — số lượng đỉnh và cạnh trong đồ thị.

Mỗi trong số m dòng tiếp theo chứa ba số nguyên u_i, v_i, w_i

($1 \le u_i, v_i \le n, u_i \ne v_i, 1 \le w_i \le 10^9$) — các đầu mút của cạnh thứ i và trọng số của nó.

Xuất

In ra m dòng. Dòng thứ i nên chứa trọng số nhỏ nhất có thể của cây bao trùm mà chứa cạnh thứ i.

Các cạnh được đánh số từ 1 đến m theo thứ tự xuất hiện trong đầu vào.

Ví dụ

Input	Output
5 7 1 2 3 1 3 1 1 4 5 2 3 2 2 5 3 3 4 2 4 5 4	9 8 11 8 8 8 9

Problem B. Cây 1

Time Limit	2000 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

Cho một đồ thị vô hướng, trọng số liên thông mà không có vòng và cạnh lặp.

Hãy nhớ rằng một cây bao trùm của một đồ thị được xác định là một đồ thị con không có chu trình, liên thông của đồ thị đã cho mà bao gồm tất cả các đỉnh của đồ thị. Trọng số của một cây được xác định là tổng trọng số của các cạnh mà cây đã chứa. Cây bao trùm nhỏ nhất (**MST**) của một đồ thị được xác định là cây bao trùm của đồ thị có trọng số nhỏ nhất có thể. Đối với bất kỳ đồ thị nào liên thông thì cây bao trùm nhỏ nhất tồn tại, nhưng trong trường hợp tổng quát, cây bao trùm nhỏ nhất của một đồ thị không phải là duy nhất.

Đối với mỗi cạnh của đồ thị đã cho, bạn cần xác định: liệu nó có được bao gồm trong một MST **bất kỳ** nào không, hoặc được bao gồm **trong ít nhất một MST**, hoặc **không được bao gồm trong bất kỳ** MST nào.

Dữ liệu

Dòng đầu tiên chứa hai số nguyên n và m ($2 \le n \le 10^5, n - 1 \le m \le min(10^5, \frac{n(n-1)}{2})$) — số đỉnh và cạnh của đồ thị, tương ứng. Tiếp theo là m dòng, mỗi dòng chứa ba số nguyên — mô tả các cạnh của đồ thị dưới dạng "$a_i\ b_i\ w_i$" ($1 \le a_i, b_i \le n$, $1 \le w_i \le 10^6$, $a_i \ne b_i$), trong đó a_i và b_i là số của các đỉnh kết nối bởi cạnh thứ i, w_i là trọng số của cạnh. Đảm bảo rằng đồ thị là liên thông và không chứa vòng hoặc cạnh lặp.

Kết quả

In ra m dòng — câu trả lời cho tất cả các cạnh. Nếu cạnh thứ i được bao gồm trong bất kỳ MST nào, in "`any`"; nếu cạnh thứ i được bao gồm ít nhất trong một MST, in "`at least one`"; nếu cạnh thứ i không được bao gồm trong bất kỳ MST nào, in "`none`". In các câu trả lời cho các cạnh theo thứ tự mà các cạnh được chỉ định trong đầu vào.

Ví dụ 1

Input	Output
4 5 1 2 101 1 3 100 2 3 2 2 4 2 3 4 1	none any at least one at least one any

Ví dụ 2

Input	Output
3 3 1 2 1 2 3 1 1 3 2	any any none

Ví dụ 3

Input	Output
3 3 1 2 1 2 3 1 1 3 1	at least one at least one at least one

Ghi chú

Trong ví dụ thứ hai, MST là duy nhất cho đồ thị đã cho: nó chứa hai cạnh đầu tiên.

Trong ví dụ thứ ba, bất kỳ hai cạnh nào cũng tạo thành MST cho đồ thị đã cho. Điều đó có nghĩa là mỗi cạnh được bao gồm ít nhất trong một MST.

Problem C. Cây Fenwick

Time Limit 2000 ms
Mem Limit 262144 kB

Ehab quan tâm đến phép toán bitwise-xor và các đồ thị đặc biệt. Mahmoud đã đưa cho anh ấy một bài toán kết hợp cả hai. Anh ta có một đồ thị đầy đủ gồm n đỉnh được đánh số từ 0 đến n - 1. Với mọi $0 \leq u < v < n$, đỉnh u và đỉnh v được nối với nhau bằng một cạnh vô hướng có trọng số $u \oplus v$ (trong đó $\oplus$ là phép bitwise-xor). Bạn có thể tìm trọng số của cây khung nhỏ nhất của đồ thị đó không?

Bạn có thể đọc về đồ thị đầy đủ tại https://en.wikipedia.org/wiki/Complete_graph

Bạn có thể đọc về cây khung nhỏ nhất tại https://en.wikipedia.org/wiki/Minimum_spanning_tree

Trọng số của cây khung nhỏ nhất là tổng trọng số các cạnh được bao gồm trong nó.

Đầu vào

Dòng duy nhất chứa một số nguyên n ($2 \leq n \leq 10^{12}$), số lượng đỉnh trong đồ thị.

Đầu ra

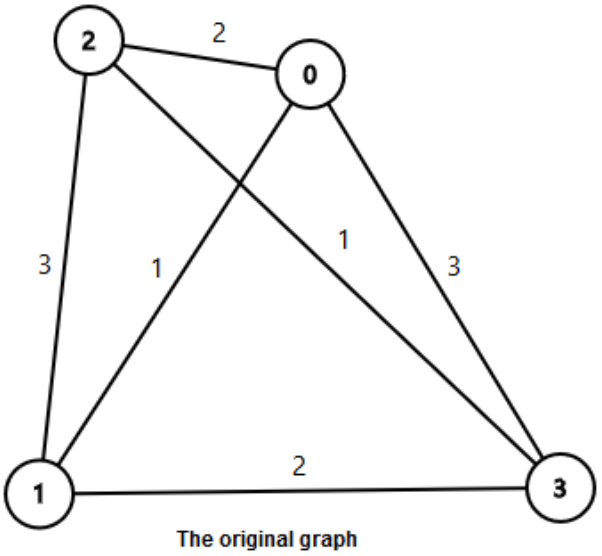
Dòng duy nhất chứa một số nguyên x, trọng số của cây khung nhỏ nhất của đồ thị.

Ví dụ

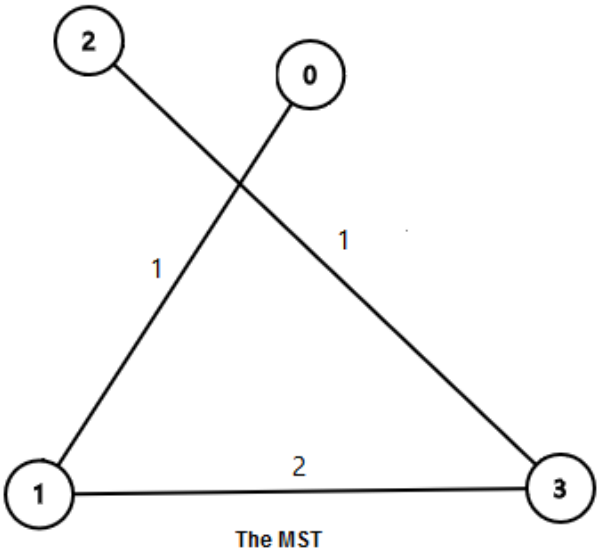
Input	Output
4	4

Ghi chú

Trong ví dụ đầu tiên:



The original graph



The MST

Trọng số của cây khung nhỏ nhất là 1+2+1=4.

Problem D. Cây Dijkstra

Time Limit	1000 ms
Mem Limit	1572864 kB
Code Length Limit	50000 B
OS	Linux

Bạn được cho một đồ thị vô hướng với N đỉnh và M cạnh, trong đó trọng số là duy nhất.

Có một hàm Cost(u, v), được định nghĩa như sau:

Khi có một đường đi giữa đỉnh u và v, xóa cạnh có trọng số nhỏ nhất. Cost(u,v) là tổng trọng số của các cạnh đã bị xóa trong quá trình này.

 graph

Ví dụ, từ đồ thị ở trên (giống như đầu vào mẫu), Cost(2,6) là 2+3+4+5+6 = 20.

Cho một đồ thị vô hướng, nhiệm vụ của bạn là tính tổng của Cost(u,v) cho tất cả các đỉnh u và v, với u < v. Vì kết quả có thể rất lớn, hãy đưa ra kết quả theo modulo 10^9.

Nhập

Dòng đầu tiên của đầu vào bao gồm hai số nguyên, N và M. (1 <= N <= 100,000, 0 <= M <= 100,000)

M dòng tiếp theo bao gồm ba số nguyên, u, v, và w. Điều này có nghĩa là có một cạnh giữa đỉnh u và v với trọng số w. (1 <= u, v <= N, 1 <= w <= 100,000)

Xuất

Xuất tổng được chỉ định trong câu đề bài.

Ví dụ

Input	Output
6 7 1 2 10 2 3 2 4 3 5 6 3 15 3 5 4 4 5 3 2 6 6	256

Problem E. Cây BMW

Time Limit	500 ms
Mem Limit	1572864 kB
Code Length Limit	50000 B
OS	Linux

Mary rất rất hạnh phúc. Bạn cũng sẽ hạnh phúc nếu cha mẹ bạn tặng bạn một chiếc BMW mới nhất vào ngày sinh nhật của bạn. Cô ấy muốn thử nghiệm chiếc xe mới của mình và đó là lý do tại sao cô ấy sẽ đi thăm bà nội.

Có một đồ thị với N đỉnh đại diện cho N thành phố và M cạnh đại diện cho các con đường hai chiều giữa một số cặp thành phố. Chúng ta có thể giả định rằng Mary sống ở thành phố 1 và bà nội của cô ấy sống ở thành phố N. Thật không may, không phải mọi thứ đều tốt đẹp trong cuộc sống và ví dụ là giới hạn tốc độ. Mary quyết định lái xe với tốc độ cố định. Mỗi trong số M con đường này có một tốc độ tối đa cho phép V mà Mary không thể vượt quá. Tốt, sở thích của cô ấy không kết thúc ở đây. Với tình yêu với những trải nghiệm mạo hiểm, cô ấy muốn lái chiếc xe đắt tiền của mình nhanh nhất có thể.

Bạn cần viết một chương trình để tìm ra tốc độ tối đa mà Mary có thể đạt được thành phố của bà nội mà không gặp tranh cãi với cảnh sát (vì tốt cho họ).

Lưu ý: Có thể có nhiều hơn một con đường giữa hai thành phố.

Nhập

Dòng đầu tiên của đầu vào chứa số lượng bài kiểm tra - T (T <= 5). Trên dòng đầu tiên của mỗi bài kiểm tra có hai số nguyên - N (2 <= N <= 50000) và M (1 <= M <= 100000). Trên M dòng tiếp theo có ba số nguyên A, B, và V đại diện cho một con đường hai chiều giữa các thành phố A và B với giới hạn tốc độ V trong đó 1 <= V <= 10^18 (Cuối cùng thì đó là một chiếc BMW).

Xuất

In ra một dòng duy nhất tốc độ tối đa mà Mary có thể đạt được. Nếu cô ấy không thể đến nhà bà nội, in -1.

Ví dụ

Input	Output
1 4 5 1 2 80 3 1 20 2 3 60 4 3 300 2 4 90	80

Problem F. Cây 2

Time Limit	3000 ms
OS	Linux
Statement	PDF

Một trong những hoạt động thiết yếu của hiệp sĩ là tham gia các giải đấu. Thường xuyên, các nhóm hiệp sĩ tụ tập khắp đất nước để so sánh kỹ năng của mình. Trong một ngày thi đấu điển hình như vậy, mọi người có năm giờ để thực hiện mười bộ môn, như đấu kiếm, bắn cung và nhiều hình thức cưỡi ngựa khác nhau. Không cần phải nói, bạn phải tự mang theo ngựa của mình.

Điều này không dễ dàng như vẻ ngoài của nó. Thường thì một hiệp sĩ mất vài ngày để đi từ lâu đài nơi anh ta sống đến nơi tổ chức giải đấu. Nhưng đôi khi ngựa rất, rất bướng bỉnh. Sau khi đã đi được một quãng đường nhất định trong một ngày, chúng đôi khi đơn giản dừng lại và từ chối đi tiếp. May mắn thay, chúng bắt đầu lại vào ngày hôm sau. Để đảm bảo rằng ngựa không từ chối phục vụ trước khi hành trình trong ngày được hoàn thành, một hiệp sĩ muốn chọn một lịch trình mà trong đó chuyến đi dài nhất trong một ngày càng ngắn càng tốt. Vì vậy, một chuyến đi mất nhiều ngày với quãng đường ngắn được ưa thích hơn là một tuyến đường ngắn hơn có nguy cơ ngựa từ chối đi.

Viết một chương trình trả lời các truy vấn từ các hiệp sĩ khắp đất nước về cách tốt nhất để đi từ lâu đài của họ đến nơi tổ chức giải đấu. Cho một mô tả về các địa điểm liên quan (tức là lâu đài, địa điểm của các giải đấu, và nhà trọ cho việc lưu trú qua đêm), chương trình nên nói cho họ biết khoảng cách lớn nhất họ phải vượt qua trong một ngày sao cho khoảng cách này là nhỏ nhất trong tất cả các lịch trình có thể.

Các địa điểm được chỉ định bởi các số nguyên liên tiếp từ 1 đến N, trong khi một con đường được biểu diễn bởi ba số nguyên, cụ thể là điểm xuất phát, điểm đến, và chiều dài của nó. Mỗi con đường có thể được sử dụng theo cả hai hướng, và có ít nhất một tuyến đường (tức là một chuỗi các con đường) giữa bất kỳ hai địa điểm nào. Các hiệp sĩ tuân thủ các con đường đã cho khi di chuyển và chỉ dừng lại qua đêm tại một trong số các địa điểm N.

Đầu vào

Dòng đầu tiên chứa tổng số trường hợp kiểm tra.
Mỗi trường hợp kiểm tra bắt đầu với một dòng đầu tiên chứa số N của các địa điểm ($1 < N < 3000$) theo sau là số R của các con đường ($1 < R < 100000$). Tiếp theo là R dòng với ba số nguyên mỗi dòng (a, b, và l), mỗi số định nghĩa một con đường nối các địa điểm a và b ($1 < a,b < N$) với chiều dài l ($1 < l < 1000000$).
Sau đó, mỗi trường hợp kiểm tra tiếp tục với số Q của các truy vấn trên một dòng riêng biệt

(1 < Q < 1000). Mỗi truy vấn tiếp theo chứa hai số nguyên k và t, chỉ ra một truy vấn của một hiệp sĩ sống ở địa điểm k và cần đi đến giải đấu ở địa điểm t (1 < k,t < N,k ≠ t).

Đầu ra

Đối với mỗi trường hợp kiểm tra, xuất ra một dòng chứa từ 'Case', một khoảng trắng, và số thứ tự của nó (bắt đầu từ 1 cho trường hợp kiểm tra đầu tiên). Sau đó, in ra một dòng cho mỗi truy vấn trong trường hợp kiểm tra này, chứa khoảng cách chuyến đi hàng ngày tối đa nhỏ nhất như đã mô tả ở trên. In ra một dòng trắng sau mỗi trường hợp kiểm tra.

Ví dụ

Input	Output
2 4 4 1 2 100 2 3 100 3 4 100 4 1 200 1 1 4 6 9 2 4 5 5 1 7 3 6 6 3 1 4 2 3 2 1 2 1 6 5 42 4 5 3 4 6 5 4 1 3 3 4 5 4 6 1	Case 1 100 Case 2 2 5 3 5

Problem G. Cây 3

Time Limit	3000 ms
OS	Linux
Statement	PDF

Để chuẩn bị cho "Cuộc thi học sinh toàn quốc lần đầu tiên ACM" (vào năm 20??), thị trưởng thành phố đã quyết định cung cấp nguồn điện ổn định cho tất cả các trường học. (Thị trưởng thực sự lo lắng về tình trạng mất điện). Vì vậy, để làm được điều đó, trạm điện "Tương lai" và một trường học (bất kỳ trường nào) phải được kết nối; ngoài ra, một số trường học khác cũng cần được kết nối.
Bạn có thể giả định rằng một trường học có nguồn điện ổn định nếu nó được kết nối trực tiếp với "Tương lai", hoặc với bất kỳ trường học nào khác có nguồn điện ổn định. Bạn được cung cấp chi phí kết nối giữa một số trường học. Thị trưởng đã quyết định chọn ra hai kế hoạch kết nối rẻ nhất - chi phí của kết nối bằng tổng chi phí kết nối giữa các trường học. Nhiệm vụ của bạn là giúp thị trưởng tìm ra chi phí của hai kế hoạch kết nối rẻ nhất.

Dữ liệu

- Dữ liệu bắt đầu bằng số lượng bộ test, T ($1 < T < 15$) trên một dòng. Sau đó là T bộ test.
- Dòng đầu tiên của mỗi bộ test chứa hai số, được phân tách bởi một khoảng trắng, N ($3 < N < 100$) là số lượng trường học trong thành phố, và M là số lượng kết nối có thể có giữa chúng.
- M dòng tiếp theo chứa ba số A_i, B_i, C_i, trong đó C_i là chi phí kết nối ($1 < C_i < 300$) giữa trường học A_i và B_i. Các trường học được đánh số bằng các số nguyên trong khoảng từ 1 đến N.

Kết quả

- Đối với mỗi bộ test, in ra một dòng chứa hai số được phân tách bởi một khoảng trắng - chi phí của hai kế hoạch kết nối rẻ nhất. Gọi S_1 là chi phí rẻ nhất và S_2 là chi phí rẻ nhì. Điều quan trọng, $S_1 = S_2$ chỉ khi và chỉ khi có hai kế hoạch rẻ nhất, nếu không thì $S_1 < S_2$. Bạn có thể giả định rằng luôn có thể tìm ra chi phí S_1 và S_2.

Ví dụ

Input	Output
2 5 8 1 3 75 3 4 51 2 4 19 3 2 95 2 5 42 5 4 31 1 2 9 3 5 66 9 14 1 2 4 1 8 8 2 8 11 3 2 8 8 9 7 8 7 1 7 9 6 9 3 2 3 4 7 3 6 4 7 6 2 4 6 14 4 5 9 5 6 10	110 121 37 37

Problem H. Cây 4

Time Limit	3000 ms
OS	Linux
Statement	PDF

Chính phủ Bangladesh đang nghiêm túc xem xét giảm bớt tình trạng kẹt xe không thể chịu đựng được ở Dhaka. Họ đã thử nhiều cách để giải quyết vấn đề này. Nhưng tất cả bạn biết khi di chuyển ở Dhaka, gần như là không thể tránh khỏi điều đó :-). Cơ quan cao cấp của Chính phủ hiện đã nhận ra tình hình rằng họ cần một cái nhìn và hành động khác biệt cho nhiệm vụ khổng lồ này. Họ đã đến với bạn và với tư cách là một chuyên gia về khoa học máy tính, bạn đã đưa ra cho họ một giải pháp. Ý tưởng là đơn giản, đó là xây dựng các con đường mới. Nhưng vấn đề là đảm bảo rằng con đường mới được xây dựng được sử dụng một cách hợp lý. Và điều đó chỉ xảy ra nếu chi phí di chuyển được giảm cho một số tuyến đường. Trong trường hợp đó, giao thông trên những tuyến đường đó sẽ tự động sử dụng con đường hiện tại và do đó sẽ được phân bổ tốt hơn trong mạng lưới.

Trong bài toán này, bạn sẽ được cung cấp một bản đồ của Dhaka dưới dạng một đồ thị không hướng. Mỗi nút sẽ biểu thị một trạm và cạnh sẽ biểu thị một con đường. Các nút sẽ là các điểm hình học 2-D đơn giản và chi phí để di chuyển một con đường tỷ lệ với khoảng cách Euclidian giữa hai nút. Bây giờ, bạn sẽ đề xuất xây dựng một con đường tạm thời. Bạn đã cố định tiêu chí để chọn hai nút giữa đó con đường mới sẽ được xây dựng.

Nếu có một con đường giữa (u, v) thì cặp này sẽ không được xem xét.
Ngược lại, $C_{uv} = Sum(PreCost_{ij}—CurCost_{ij})$ với $CurCost_{ij}$ là chi phí đường đi ngắn nhất từ i đến j nếu con đường giữa (u, v) tồn tại. Và $PreCost_{ij}$ là chi phí đường đi ngắn nhất trước khi xây dựng con đường giữa u và v.
Cặp (u,v) với C_{uv} lớn nhất sẽ được chọn.

Đầu vào

Mỗi trường hợp kiểm tra bắt đầu với hai số nguyên dương N (< 50) và M (< 1225). Trong vài dòng tiếp theo, tọa độ (x, y) của các nút sẽ được cung cấp. Giá trị của mỗi tọa độ sẽ nằm trong phạm vi từ -1000 đến 1000. Trong vài dòng tiếp theo, mô tả con đường sẽ được cung cấp. Mỗi con đường bao gồm hai số nguyên dương (u, v) nơi 1 < u,v < N. Điều này có nghĩa là có một con đường hai chiều giữa u và v. Đồ thị được đảm bảo là liên thông.
Đầu vào kết thúc bởi N = M = 0. Trường hợp này không được xử lý.

Đầu ra

Nếu việc xây dựng đường mới giúp cải thiện tình hình giao thông thì đầu ra sẽ là cặp nút (u, v) giữa đó con đường mới sẽ được xây dựng. Nếu có nhiều ứng cử viên thì các nút có khoảng cách ngắn nhất sẽ được chọn. Nếu lại có sự đồng hòa thì cặp nút có chỉ số nhỏ sẽ là câu trả lời.
Mặt khác, nếu mạng lưới đường bộ đã cân bằng (Đường mới sẽ không cải thiện tình hình giao thông), in ra `No road required`. Lưu ý rằng, nếu Cuv nhỏ hơn hoặc bằng 1.0 thì mạng lưới sẽ được coi là cân bằng.

Ví dụ

Input	Output
4 6 0 0 0 2 2 0 2 2 1 2 1 3 1 4 2 3 2 4 3 4 4 4 0 0 0 2 2 0 2 2 1 2 2 3 3 4 4 1 4 3 0 0 0 2 2 0 2 2 1 2 2 3 3 4 0 0	No road required 1 3 1 3

Problem I. Sửa đường tàu

Time Limit	4000 ms
OS	Linux
Statement	PDF

Peter sống ở Expensive City, một trong những thành phố đắt đỏ nhất thế giới. Peter không có đủ tiền để mua xe hơi, và xe buýt ở Expensive City khá tệ, vì vậy anh ấy sử dụng tàu điện ngầm để đi làm. Cho đến nay, tàu điện ngầm rất rẻ: bạn có thể đi bất cứ đâu chỉ với vé $2. Tháng trước, các quản lý quyết định rằng giá vé quá rẻ nên họ đã phát minh ra EFS (Hệ thống Giá vé Đắt đỏ). Với hệ thống này, người dùng chỉ có thể mua vé hàng tháng giữa các ga liền kề, cho phép họ di chuyển giữa các ga này bất kỳ số lần nào. Giá của vé hàng tháng thay đổi giữa các ga, vì vậy quyết định mua vé nào phải được cân nhắc kỹ lưỡng.

Với kế hoạch tàu điện ngầm trước đây, cách rẻ nhất để di chuyển từ Picadilly đến Victoria và Queensway là mua vé hàng tháng Picadilly-Victoria và Queensway-Victoria, với tổng chi phí là $12.
Peter là một người bán hàng, vì vậy anh ấy cần có thể di chuyển đến bất kỳ phần nào của thành phố. Anh ấy muốn chi tiêu càng ít tiền càng tốt, và đây là nơi bạn xuất hiện. Anh ấy đã thuê bạn để viết một chương trình, cho biết danh sách các ga, giá vé hàng tháng giữa các cặp ga và ga gần nhà Peter nhất, trả về số tiền tối thiểu Peter phải chi tiêu để có thể di chuyển đến bất kỳ ga nào khác. Chương trình này cũng phải trả về giá trị nếu không thể di chuyển từ ga nhà Peter đến tất cả các ga khác, vì trong trường hợp này Peter sẽ bắt đầu cân nhắc sử dụng xe buýt...

Đầu vào

Đầu vào bao gồm một số trường hợp kiểm tra. Một trường hợp kiểm tra bắt đầu với hai số nguyên: $1 < s < 400$ (số lượng ga) và $0 < c < 79800$ (số lượng kết nối) được tách bởi một khoảng trắng. Tiếp theo là s dòng, mỗi dòng chứa tên của một ga tàu điện ngầm. Sau tên của các ga sẽ có c dòng cho thấy các kết nối giữa các ga. Kết thúc trường hợp kiểm tra là một dòng chứa tên của ga mà Peter cần di chuyển đến tất cả các ga khác.
Đầu vào kết thúc với trường hợp kiểm tra ảo '0 0', không được xử lý.

Đầu ra

Đối với mỗi trường hợp kiểm tra, đầu ra sẽ là một dòng chứa một số nguyên, giá vé hàng tháng tối thiểu mà Peter phải trả để di chuyển từ ga đã cho đến tất cả các ga khác, hoặc "Impossible" nếu không thể di chuyển đến tất cả các ga.

Ví dụ

Input	Output
3 3 Picadilly Victoria Queensway Picadilly Victoria 2 Queensway Victoria 10 Queensway Picadilly 20 Picadilly 4 2 Picadilly Victoria Queensway Temple Picadilly Victoria 2 Temple Queensway 100 Temple 0 0	12 Impossible

Problem J. Cây 6

Time Limit	2000 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

Đã biết rằng bọ chét ở Berland chỉ có thể nhảy theo chiều dọc và chiều ngang, và độ dài của cú nhảy luôn bằng s centimet. Một con bọ chét đã tìm thấy mình ở trung tâm của một ô của bảng ô vuông có kích thước $n \times m$ centimet (mỗi ô có kích thước là 1×1 centimet). Cô ấy có thể nhảy theo ý muốn cho một số lần tùy ý, thậm chí cô ấy có thể ghé thăm một ô nhiều lần. Hạn chế duy nhất là cô ấy không thể nhảy ra khỏi bảng.

Bọ chét có thể đếm số lượng ô mà cô ấy có thể đạt được từ vị trí xuất phát (x, y). Hãy ký hiệu số lượng này bằng $d_{x,y}$. Nhiệm vụ của bạn là tìm số vị trí xuất phát (x, y), có giá trị tối đa của $d_{x,y}$.

Nhập

Dòng đầu tiên chứa ba số nguyên n, m, s ($1 \le n, m, s \le 10^6$) — độ dài của bảng, chiều rộng của bảng và độ dài của cú nhảy của bọ chét.

Đầu ra

Xuất ra một số nguyên duy nhất — số lượng vị trí xuất phát cần thiết của bọ chét.

Ví dụ 1

Input	Output
2 3 1000000	6

Ví dụ 2

Input	Output
3 3 2	4

Problem K. Cây DSU Rollback

Time Limit 1000 ms
Mem Limit 262144 kB

Igor đang ở trong bảo tàng và muốn xem càng nhiều bức tranh càng tốt.

Bảo tàng có thể được biểu diễn như một lưới hình chữ nhật gồm $n \times m$ ô. Mỗi ô có thể là trống hoặc không thể đi qua. Các ô trống được đánh dấu bằng '.', các ô không thể đi qua được đánh dấu bằng '*'. Mỗi cặp ô liền kề với hai loại khác nhau (một ô trống và một ô không thể đi qua) được phân chia bởi một bức tường chứa một bức tranh.

Ở đầu, Igor đang ở trong một ô trống nào đó. Mỗi lúc, anh ấy có thể di chuyển đến bất kỳ ô trống nào có chung một cạnh với ô hiện tại.

Đối với một số vị trí bắt đầu, bạn cần tính toán số lượng tối đa các bức tranh mà Igor có thể nhìn thấy. Igor chỉ có thể nhìn thấy bức tranh nếu anh ấy đang ở ô kề cạnh với bức tường chứa bức tranh đó. Igor có rất nhiều thời gian, vì vậy anh ấy sẽ xem xét mọi bức tranh mà anh ấy có thể nhìn thấy.

Nhập

Dòng đầu tiên của đầu vào chứa ba số nguyên n, m và k ($3 \le n, m \le 1000$, $1 \le k \le min(n \cdot m, 100\,000)$) — kích thước của bảo tàng và số vị trí bắt đầu cần xử lý.

Mỗi trong số n dòng tiếp theo chứa m ký tự '.', '*' — mô tả của bảo tàng. Đảm bảo rằng tất cả các ô biên đều không thể đi qua, vì vậy Igor không thể ra khỏi bảo tàng.

Mỗi trong số k dòng cuối cùng chứa hai số nguyên x và y ($1 \le x \le n$, $1 \le y \le m$) — hàng và cột của một trong số vị trí bắt đầu của Igor. Các hàng được đánh số từ trên xuống dưới, cột được đánh số từ trái sang phải. Đảm bảo rằng tất cả các vị trí bắt đầu đều là ô trống.

Đầu ra

In k số nguyên — số lượng tối đa các bức tranh mà Igor có thể nhìn thấy nếu anh ấy bắt đầu từ vị trí tương ứng.

Ví dụ 1

Input	Output
5 6 3 ****** *..*.* ****** *....* ****** 2 2 2 5 4 3	6 4 10

Ví dụ 2

Input	Output
4 4 1 **** *..* *.** **** 3 2	8

Problem L. Cây 7

Time Limit 2000 ms
Mem Limit 262144 kB

Hongcow là người cai trị thế giới. Là người cai trị thế giới, anh ấy muốn làm cho việc di chuyển bằng đường bộ trong các quốc gia trở nên dễ dàng hơn.

Thế giới có thể được mô hình hóa như một đồ thị vô hướng với n nút và m cạnh. k trong số các nút là nơi đóng trụ của chính phủ của k quốc gia tạo nên thế giới.

Có tối đa một cạnh kết nối bất kỳ hai nút nào và không có cạnh nào kết nối một nút với chính nó. Hơn nữa, đối với bất kỳ hai nút tương ứng với chính phủ, **không có đường đi nào giữa hai nút đó**. Bất kỳ đồ thị nào thỏa mãn tất cả các điều kiện này đều là *ổn định*.

Hongcow muốn thêm càng nhiều cạnh càng tốt vào đồ thị trong khi vẫn giữ cho nó ổn định. Xác định số cạnh tối đa mà Hongcow có thể thêm vào.

Nhập

Dòng đầu tiên của đầu vào sẽ chứa ba số nguyên n, m và k ($1 \le n \le 1\,000$, $0 \le m \le 100\,000$, $1 \le k \le n$) — số đỉnh và số cạnh trong đồ thị, và số đỉnh là nơi đóng trụ của chính phủ.

Dòng tiếp theo của đầu vào sẽ chứa k số nguyên $c_1, c_2, \ldots, c_k$ ($1 \le c_i \le n$). Các số nguyên này sẽ đôi một khác nhau và biểu thị các nút là nơi đóng trụ của chính phủ trong thế giới này.

Các dòng tiếp theo m dòng đầu vào sẽ chứa hai số nguyên u_i và v_i ($1 \le u_i, v_i \le n$). Điều này biểu thị một cạnh vô hướng giữa các nút u_i và v_i.

Đảm bảo rằng đồ thị mô tả bởi đầu vào là ổn định.

Đầu ra

Đầu ra một số nguyên, số cạnh tối đa mà Hongcow có thể thêm vào đồ thị trong khi vẫn giữ cho nó ổn định.

Ví dụ 1

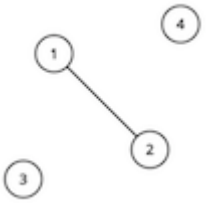
Input	Output
4 1 2 1 3 1 2	2

Ví dụ 2

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

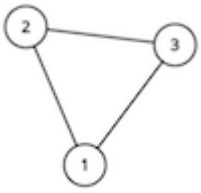
Ghi chú

Đối với bài kiểm tra mẫu đầu tiên, đồ thị trông như thế này:



Đỉnh 1 và 3 là đặc biệt. Giải pháp tối ưu là kết nối đỉnh 4 với các đỉnh 1 và 2. Điều này thêm tổng cộng 2 cạnh. Chúng ta không thể thêm thêm bất kỳ cạnh nào nữa, vì các đỉnh 1 và 3 không thể có bất kỳ đường đi nào giữa chúng.

Đối với bài kiểm tra mẫu thứ hai, đồ thị trông như thế này:



Chúng ta không thể thêm thêm cạnh vào đồ thị này. Lưu ý rằng chúng ta không được phép thêm vòng lặp tự, và đồ thị phải là đơn giản.

Problem M. Cây TST

Time Limit 3000 ms
Mem Limit 262144 kB

Đây là một bài toán tương tác. Trong phần tương tác bên dưới, bạn sẽ thấy thông tin về việc làm mới đầu ra.

Trong bài toán này, bạn sẽ chơi một trò chơi với Hongcow. Thật may mắn cho bạn!

Hongcow có một ma trận ẩn kích thước n nhân n là M. Gọi $M_{i,j}$ là phần tử ở hàng i và cột j của ma trận. Các hàng và cột được đánh số từ 1 đến n.

Các phần tử trong ma trận nằm trong khoảng từ 0 đến 10^9. Thêm vào đó, $M_{i,i} = 0$ với mọi i hợp lệ. Nhiệm vụ của bạn là tìm giá trị nhỏ nhất trên mỗi hàng, loại trừ phần tử trên đường chéo chính. Cụ thể, với mỗi i, bạn phải tìm $\min_{j \neq i} M_{i,j}$.

Để làm điều này, bạn có thể đặt câu hỏi cho Hongcow.

Một câu hỏi bao gồm việc đưa cho Hongcow một tập hợp các chỉ số phân biệt $\{w_1, w_2, ..., w_k\}$, với $1 \leq k \leq n$. Hongcow sẽ trả lời bằng n số nguyên. Số nguyên thứ i sẽ là giá trị nhỏ nhất của $min_{1 \leq j \leq k} M_{i,w_j}$.

Bạn chỉ được phép hỏi Hongcow tối đa 20 câu hỏi — anh ấy nghĩ bạn chỉ cần hỏi bấy nhiêu câu.

Khi bạn đã sẵn sàng trả lời, hãy in ra một số nguyên duy nhất -1 trên một dòng riêng, sau đó in ra n số nguyên trên dòng tiếp theo. Số nguyên thứ i sẽ là giá trị nhỏ nhất trên hàng i của ma trận, loại trừ phần tử thứ i. Đừng quên làm mới đầu ra cho câu trả lời cuối cùng. Việc in câu trả lời không tính là một câu hỏi.

Bạn sẽ nhận được kết quả `Sai Đáp Án` nếu

- Câu hỏi hoặc câu trả lời của bạn không đúng định dạng như mô tả trong đề bài.
- Bạn hỏi nhiều hơn 20 câu hỏi.
- Câu hỏi của bạn có chỉ số trùng lặp.
- Giá trị của k trong câu hỏi không nằm trong khoảng từ 1 đến n, bao gồm cả hai đầu.
- Câu trả lời cuối cùng của bạn không chính xác.

Bạn sẽ nhận được kết quả `Quá Thời Gian Không Hoạt Động` nếu bạn không in gì hoặc quên làm mới đầu ra, bao gồm cả câu trả lời cuối cùng (thông tin thêm về làm mới

đầu ra bên dưới).

Input

Dòng đầu tiên của đầu vào chứa một số nguyên duy nhất n ($2 \le n \le 1,000$).

Output

Để in câu trả lời cuối cùng, hãy in ra chuỗi `-1` trên một dòng riêng. Sau đó, dòng tiếp theo chứa n số nguyên. Số nguyên thứ i là giá trị nhỏ nhất của hàng i trong ma trận, loại trừ các phần tử trên đường chéo chính. **Đừng quên làm mới câu trả lời của bạn!**

Interaction

Để đặt câu hỏi, hãy in ra một số nguyên duy nhất k trên một dòng riêng, biểu thị kích thước tập con của bạn. Sau đó, dòng tiếp theo chứa k số nguyên $w_1, w_2, \ldots w_k$. Lưu ý, bạn phải làm mới đầu ra để nhận được phản hồi.

Hongcow sẽ trả lời bằng cách in ra một dòng gồm n số nguyên. Số nguyên thứ i trong dòng này biểu thị giá trị nhỏ nhất của M_{i, w_j} với j nằm trong khoảng từ 1 đến k.

Bạn chỉ được phép hỏi tối đa 20 câu hỏi, nếu không bạn sẽ nhận được kết quả `Sai Đáp Án`.

Để làm mới đầu ra bạn có thể dùng (ngay sau khi in số nguyên và xuống dòng):

- `fflush(stdout)` trong C++;
- `System.out.flush()` trong Java;
- `stdout.flush()` trong Python;
- `flush(output)` trong Pascal;
- Xem tài liệu cho các ngôn ngữ khác.

Hacking Để hack ai đó, sử dụng định dạng sau

```
n
M_{1,1} M_{1,2} ... M_{1,n}
M_{2,1} M_{2,2} ... M_{2,n}
```

```
...
M_{n,1} M_{n,2} ... M_{n,n}
```

Tất nhiên, chương trình thí sinh sẽ không thể nhìn thấy đầu vào này.

Examples

Input	Output
3 0 0 0 2 7 0 0 0 4 3 0 8 0 5 4	3 1 2 3 1 3 2 1 2 1 2 1 1 -1 2 5 4

Input	Output
2 0 0 0 0	1 2 1 1 -1 0 0

Note

Trong ví dụ đầu tiên, Hongcow có ma trận ẩn

```
[
 [0, 3, 2],
 [5, 0, 7],
 [4, 8 ,0],
]
```

Dưới đây là phiên bản dễ đọc hơn minh họa cho phần tương tác. Cột bên trái là Hongcow, còn cột bên phải là thí sinh.

```
3
                3
```

```
                1 2 3
0 0 0
                1
                3
2 7 0
                2
                1 2
0 0 4
                1
                2
3 0 8
                1
                1
0 5 4
                -1
                2 5 4
```

Trong ví dụ thứ hai, các phần tử ngoài đường chéo của ma trận có thể bằng 0.

Problem N. Cây 8

Time Limit 2000 ms
Mem Limit 524288 kB

Phát có một cô em gái rất chăm chỉ. Tuy nhiên, lịch sử dường như là môn học gây cản trở cho cô ấy, vì vậy Phát phải giúp cô ấy với điều này. Hôm nay, hai người bạn của chúng ta sẽ ôn lại lịch sử công nghệ và cách nó ảnh hưởng đến những đổi mới trên toàn thế giới.

Như chúng ta đã biết, có khoảng n quốc gia trên thế giới, phát triển độc lập với nhau. Phát sẽ giúp em gái mình ôn lại một vài sự kiện cốt lõi:

- Sự kiện quan trọng đầu tiên là sự phát triển của hàng không. Ngày nay, việc di chuyển bằng đường hàng không vẫn là một trong những hình thức vận chuyển nhanh nhất và an toàn nhất. Việc xây dựng sân bay và các địa điểm hàng không là cần thiết cho một quốc gia phát triển về công nghệ. Vào một thời điểm nào đó, một đường bay thẳng sẽ được mở giữa hai quốc gia u và v. Sau sự kiện này, mọi người có thể bay từ quốc gia u đến quốc gia v, và ngược lại.
- Sự hợp tác toàn cầu cũng là một khía cạnh quan trọng của sự phát triển công nghệ. Vào một thời điểm nào đó, hai quốc gia x và y sẽ ký một thỏa thuận hợp tác trong lĩnh vực công nghệ.
- Có lẽ sự kiện đáng chú ý nhất là một đổi mới. Khi một quốc gia k phát triển một công nghệ mới, quốc gia đó sẽ mời các nhà ngoại giao, cũng như các kỹ sư từ các quốc gia khác mà nó đã ký thỏa thuận hợp tác, đến quốc gia k để tham dự một hội thảo về công nghệ mới nhất này. Tuy nhiên, các sự kiện hội thảo thường được thông báo vào phút chót, vì vậy khách mời sẽ phải đi máy bay để đến sự kiện. Di chuyển bằng đường hàng không rất nhanh, và do đó khách mời có thể thực hiện nhiều chuyến bay, và du lịch qua các quốc gia trung gian nếu cần, để đến quốc gia đích của họ. Tuy nhiên, nếu một quốc gia h nhận được lời mời từ quốc gia k, nhưng không có cách nào để di chuyển bằng đường hàng không đến quốc gia k từ quốc gia h, thì quốc gia h sẽ không gửi đoàn đại biểu đến sự kiện.

Phát đã kể cho em gái mình q sự kiện theo thứ tự thời gian. Tuy nhiên, cô em gái vẫn gặp khó khăn trong việc nhớ tất cả những thông tin này. Anh ấy đặt ra một câu đố để giúp em gái mình nhớ những điều này dễ dàng hơn: cho mỗi hội thảo, có bao nhiêu quốc gia sẽ gửi người của họ đến tham dự hội thảo đó.

Giả sử có n quốc gia độc lập. Ban đầu, không có chuyến bay trực tiếp nào từ quốc gia này sang quốc gia khác, và chưa có thỏa thuận hợp tác nào được ký kết. Dựa vào danh sách q sự

kiện theo thứ tự thời gian, bạn có thể giúp em gái của Phát trả lời câu đố cho mỗi sự kiện hội thảo không?

Đầu vào

Dòng đầu tiên chứa hai số nguyên n và q ($1 \le n \le 2 \cdot 10^5$, $1 \le q \le 6 \cdot 10^5$), biểu thị cho số lượng quốc gia, và số lượng sự kiện, tương ứng.

Các dòng tiếp theo q dòng, mỗi dòng mô tả một sự kiện, có một trong các định dạng sau:

- `1 u v` – các quốc gia u và v ($1 \le u, v \le n$, $u \neq v$) mở một đường bay trực tiếp đến nhau. Được đảm bảo rằng chưa có đường bay trực tiếp nào được mở giữa u và v trước đó.
- `2 x y` – các quốc gia x và y ($1 \le x, y \le n$, $x \neq y$) ký một thỏa thuận hợp tác. Được đảm bảo rằng các quốc gia x và y chưa ký thỏa thuận hợp tác trước đó.
- `3 k` – quốc gia k ($1 \le k \le n$) thực hiện một đổi mới công nghệ, và mời tất cả các quốc gia mà quốc gia k đã ký thỏa thuận hợp tác trước đó đến k tham dự một hội thảo. Được đảm bảo rằng ít nhất một sự kiện như vậy xảy ra vào một thời điểm nào đó.

Đầu ra

Đối với mỗi sự kiện hội thảo (các sự kiện loại `3`), xuất ra số lượng quốc gia được mời tham dự sự kiện này.

Chấm điểm

- Subtask 1 (80 điểm): $n \le 10^3$ và $q \le 3 \cdot 10^3$.
- Subtask 3 (125 điểm): Không có ràng buộc bổ sung

Tổng điểm cho bài toán này là 205.

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

Ghi chú

Sự kiện hội thảo đầu tiên được tổ chức bởi quốc gia 1. Trước sự kiện này, các quốc gia 1 và 4 đã ký một thỏa thuận hợp tác, tuy nhiên quốc gia 1 chỉ có chuyến bay đến quốc gia 3, do đó quốc gia 4 không thể gửi đại diện đến sự kiện.

Sự kiện hội thảo thứ hai được tổ chức bởi quốc gia 4. Tại thời điểm này, quốc gia 1 bây giờ có một chuyến bay trực tiếp đến quốc gia 4. Lưu ý rằng quốc gia 2 **không có** thỏa thuận hợp tác với quốc gia 4, họ chỉ ký thỏa thuận như vậy với quốc gia 1.

Sự kiện hội thảo cuối cùng lại được tổ chức bởi quốc gia 1. Cả hai quốc gia 2 và 4 đều có thể gửi đại diện đến sự kiện. Từ quốc gia 4, khách mời có thể bay trực tiếp đến quốc gia 1. Từ quốc gia 2, khách mời có thể bay đến quốc gia 4, sau đó từ quốc gia 4 đến quốc gia 1 để tham dự sự kiện.

Problem O. Cây 9

Time Limit	3000 ms
OS	Linux
Statement	PDF

Vào thế kỷ 25, nền văn minh bị tàn phá bởi một loạt thảm họa cuối cùng đã dẫn đến việc loài người xây dựng các thành phố có tường bao quanh kết nối với nhau bằng cầu hầm để thuận tiện cho việc vận chuyển. Mỗi thành phố bao quanh có một loại quặng độc đáo cần thiết để xây dựng và sửa chữa tất cả cơ sở hạ tầng bao gồm cả cầu hầm. Vật liệu này, khi kết hợp với các loại quặng từ tất cả các thành phố khác, tạo thành một vật liệu gần như không thể phá hủy được gọi là "oreon".

Bên ngoài các thành phố có tường bao là những kẻ man rợ vũ trang với vũ khí lỗi thời nhưng phá hủy được mọi phương tiện vận chuyển hàng không, nhưng chỉ làm hỏng và không thể xuyên qua cầu hầm. Do đó, mỗi thành phố được kết nối với nhiều thành phố khác nhằm đảm bảo có đường dự phòng trong trường hợp một trong những cầu hầm kết nối bị hỏng.

Nếu một cầu hầm bị hỏng, nó sẽ không thể đi qua và sẽ cần một lượng lớn "oreon" để sửa chữa. Khi một thành phố bị cô lập, tức là tất cả các kết nối của nó đều bị hỏng, "oreon" không thể được sản xuất, có thể dẫn đến sự phá hủy cuối cùng của bức tường bảo vệ thành phố. Bạn, với tư cách là trưởng đơn vị phòng thủ quốc gia, có nhiệm vụ đảm bảo rằng tất cả các thành phố vẫn có thể tiếp cận được thông qua ít nhất một cầu hầm kết nối vào mọi lúc. Với lực lượng lao động hạn chế trong đơn vị phòng thủ, bạn phải xác định cầu hầm nào cần được bảo vệ bằng số lượng người ít nhất và đảm bảo không có thành phố nào bị cô lập.

Hình 2 cho thấy bản đồ các thành phố có tường bao, cầu hầm kết nối của chúng và số lượng nhân viên an ninh.

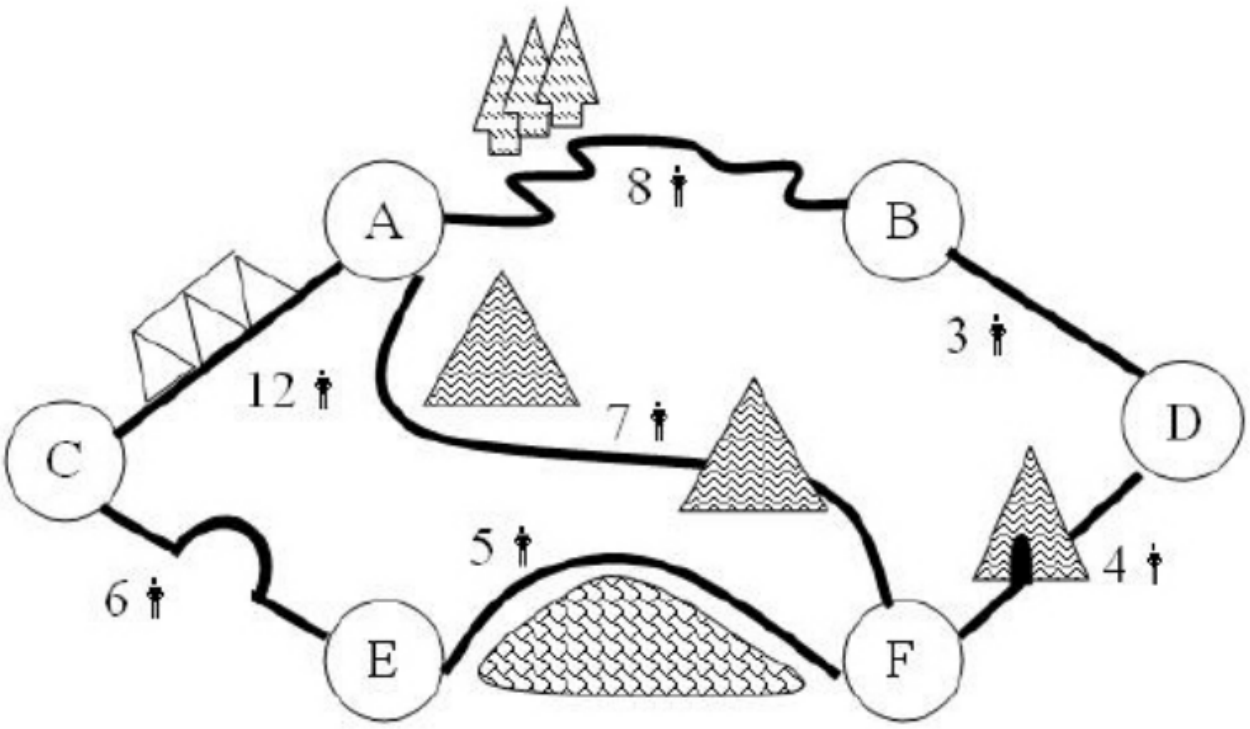


Figure 2: Map of six cities and its interconnecting tunnels

Đầu vào

Đầu vào sẽ chứa nhiều trường hợp kiểm tra. Dòng đầu tiên chỉ ra số lượng trường hợp kiểm tra. Mỗi trường hợp kiểm tra bắt đầu với một số đại diện cho số lượng thành phố có tường bao. Các thành phố được đánh dấu bằng chữ cái theo bảng chữ cái tiếng Anh. Các dòng tiếp theo chứa số lượng nhân viên an ninh cần thiết để bảo vệ cầu hầm kết nối mỗi thành phố với tất cả các thành phố khác. Giá trị bằng không ngụ ý không cần nhân viên an ninh vì không có cầu hầm. Bạn cần xuất ra cầu hầm nào cần được bảo vệ và cần bao nhiêu nhân viên cho mỗi cầu hầm.

Đầu ra

Đầu ra cho thấy kết nối cầu hầm được đặt tên theo các thành phố mà nó kết nối và số lượng nhân viên cần thiết để bảo vệ cầu hầm.

Ví dụ

Input	Output
1 6 0, 8, 12, 0, 0, 7 8, 0, 0, 3, 0, 0 12, 0, 0, 0, 6, 0 0, 3, 0, 0, 0, 4 0, 0, 6, 0, 0, 5 7, 0, 0, 4, 5, 0	Case 1: B-D 3 D-F 4 E-F 5 C-E 6 A-F 7

Problem P. Cây 10

Time Limit	3000 ms
OS	Linux
Statement	PDF

Để giải quyết vấn đề với các két sắt ở Panda Land gần đây bị trộm, Hãng Bảo Mật Panda (PSA) đã phát triển một loại khóa mới an toàn hơn với nhiều chìa khóa. Thay vì chỉ sử dụng một chìa khóa để mở khóa, giờ đây khóa có thể có đến N chìa khóa, tất cả phải được mở khóa trước khi có thể mở két. Cách hoạt động của khóa mới như sau:

- Ban đầu, các chữ số được đặt ở 0000.
- Chìa khóa có thể được mở khóa theo bất kỳ thứ tự nào, bằng cách đặt các chữ số trên khóa để khớp với chìa khóa mong muốn và sau đó nhấn nút UNLOCK.
- Một nút JUMP kỳ diệu có thể chuyển các chữ số thành bất kỳ chìa khóa nào đã được mở khóa mà không cần phải quay.
- Két sẽ được mở khóa nếu và chỉ nếu tất cả các chìa khóa được mở khóa với tổng số lượt quay tối thiểu, không tính JUMP.
- Nếu số lượt quay vượt quá, thì các chữ số sẽ được đặt lại về 0000 và tất cả các chìa khóa sẽ bị khóa lại. Nói cách khác, trạng thái của khóa sẽ được đặt lại nếu việc phá khóa thất bại.

PSA rất tự tin rằng hệ thống mới này sẽ làm chậm lại việc phá khóa, đủ thời gian để xác định và bắt giữ kẻ trộm. Để xác định số lượt quay tối thiểu cần thiết, PSA muốn bạn viết một chương trình. Cho tất cả các chìa khóa, hãy tính số lượt quay tối thiểu cần thiết để mở khóa két.

Đầu vào

Dòng đầu tiên của đầu vào chứa một số nguyên T, số lượng trường hợp kiểm tra. Mỗi trường hợp bắt đầu với một số nguyên N (1 < N < 500), số lượng chìa khóa. N dòng tiếp theo, mỗi dòng chứa đúng bốn chữ số (cho phép số 0 đứng đầu) đại diện cho các chìa khóa cần được mở khóa.

Đầu ra

Đối với mỗi trường hợp, in ra trên một dòng số lượng lượt quay tối thiểu cần thiết để mở khóa tất cả các chìa khóa.

Giải thích cho trường hợp thứ hai:

- Chuyển 0000 thành 1111, số lượt quay: 4
- Chuyển 1111 thành 1155, số lượt quay: 8
- Nhảy từ 1155 về 1111, chúng ta có thể làm điều này vì 1111 đã được mở khóa trước đó.
- Chuyển 1111 thành 5511, số lượt quay: 8

Tổng số lượt quay = 4 + 8 + 8 = 20

Ví dụ

Input	Output
4 2 1155 2211 3 1111 1155 5511 3 1234 5678 9090 4 2145 0213 9113 8113	16 20 26 17

Problem Q. Cây 11

Time Limit	3000 ms
OS	Linux
Statement	PDF

Trong một tập của chương trình Dick Van Dyke, cậu bé Richie đã nối các vết tàn nhang trên lưng của bố mình để tạo thành hình ảnh của Quả Chuông Tự Do. Thật không may, một trong những vết tàn nhang hóa ra là một vết sẹo, vì vậy việc tham gia vào chương trình Ripley's của cậu đã không thành công.

Hãy coi lưng của Dick như một mặt phẳng với các vết tàn nhang ở các vị trí (x, y) khác nhau. Nhiệm vụ của bạn là bảo Richie cách nối các điểm để giảm thiểu lượng mực sử dụng. Richie nối các điểm bằng cách vẽ các đường thẳng giữa các cặp điểm, có thể nâng bút giữa các đường. Khi Richie hoàn thành, phải có một chuỗi các đường liên kết từ bất kỳ vết tàn nhang nào đến vết tàn nhang khác.

Đầu vào

Đầu vào bắt đầu với một số nguyên dương duy nhất trên một dòng cho biết số lượng các trường hợp tiếp theo, mỗi trường hợp được mô tả như dưới đây. Dòng này được theo sau bởi một dòng trắng, và cũng có một dòng trắng giữa hai đầu vào liên tiếp.

Dòng đầu tiên chứa $0 < n < 100$, số lượng vết tàn nhang trên lưng của Dick. Đối với mỗi vết tàn nhang, một dòng theo sau; mỗi dòng tiếp theo chứa hai số thực chỉ tọa độ (x, y) của vết tàn nhang.

Đầu ra

Đối với mỗi trường hợp kiểm tra, đầu ra phải tuân theo mô tả dưới đây. Các đầu ra của hai trường hợp liên tiếp sẽ được tách biệt bằng một dòng trắng.

Chương trình của bạn in ra một số thực duy nhất với hai chữ số thập phân: tổng chiều dài tối thiểu của các đường mực có thể nối tất cả các vết tàn nhang.

Ví dụ

Input	Output
1 3 1.0 1.0 2.0 2.0 2.0 4.0	3.41

Problem R. Cây 12

Time Limit	3000 ms
OS	Linux
Statement	PDF

Ở quốc gia Graphland, có nhiều thành phố nhưng không có đường sá. Chính phủ liên bang muốn thay đổi tình hình này và có kế hoạch xây dựng đường bộ và đường sắt sao cho tất cả các thành phố trong quốc gia được kết nối thông qua hệ thống giao thông mới này. Để làm cho hệ thống mới hiệu quả hơn, Graphland sẽ chỉ xây dựng đường bộ giữa các thành phố trong cùng một bang và sẽ sử dụng đường sắt để kết nối các thành phố ở các bang khác nhau. Để mục đích của bài toán này, xem xét rằng nếu khoảng cách giữa bất kỳ hai thành phố nào là tối đa r thì chúng thuộc cùng một bang. Để giảm thiểu chi phí xây dựng đường bộ và đường sắt, chính phủ cũng muốn chỉ xây dựng số lượng tối thiểu cần thiết của đường bộ và đường sắt sao cho có một lộ trình giữa bất kỳ cặp thành phố nào trong toàn quốc. Bạn đã được thuê để xác định hệ thống mạng lưới giao thông tối ưu mà Graphland phải xây dựng.

Dòng đầu tiên của đầu vào chứa số lượng trường hợp kiểm tra theo sau. Trên dòng đầu tiên của mỗi trường hợp kiểm tra, sẽ có hai số nguyên, n (1 < n < 1000), số lượng thành phố tạo nên Graphland, và r (0 < r < 40000), giá trị ngưỡng để xác định nếu hai thành phố ở trong cùng một bang. Trên các dòng tiếp theo, sẽ có một danh sách các tọa độ nguyên x - y (-10000 < x,y < 10000) trong kế hoạch cho mỗi thành phố ở Graphland. Chương trình của bạn phải xuất ra số lượng bang ở Graphland và số lượng tối thiểu cần mở rộng (làm tròn đến số nguyên gần nhất) của cả đường bộ và đường sắt phải được xây dựng để đáp ứng điều kiện của dự án.

Đầu vào

Dòng đầu tiên của đầu vào cho số lượng trường hợp, T (1 < T < 20). T trường hợp kiểm tra theo sau. Trên dòng đầu tiên của mỗi trường hợp kiểm tra, sẽ có hai số nguyên, n (1 < n < 1000), số lượng thành phố tạo nên Graphland, và r (0 < r < 40000), giá trị ngưỡng để xác định nếu hai thành phố ở trong cùng một bang. Trên các dòng tiếp theo, sẽ có một danh sách các tọa độ nguyên x - y (-10000 < x,y < 10000) trong kế hoạch cho mỗi thành phố ở Graphland.

Đầu ra

Đầu ra bao gồm một dòng cho mỗi bộ dữ liệu đầu vào. Dòng này xác định bộ dữ liệu bằng một số (bắt đầu từ một và tăng lên ở mỗi bộ dữ liệu mới), theo sau là số lượng bang ở

Graphland và số lượng tối thiểu cần mở rộng (làm tròn đến số nguyên gần nhất) của cả đường bộ và đường sắt phải được xây dựng để đáp ứng điều kiện của dự án.

Lưu ý: Lưu ý rằng, theo định nghĩa, nếu A và B ở trong cùng một bang, và B và C ở trong cùng một bang, thì A và C cũng ở trong cùng một bang.

Ví dụ

Input	Output
3 3 100 0 0 1 0 2 0 3 1 0 0 100 0 200 0 4 20 0 0 40 30 30 30 10 10	Case #1: 1 2 0 Case #2: 3 0 200 Case #3: 2 24 28

Problem S. Cây 13

Time Limit	2000 ms
OS	Linux
Statement	PDF

Trong thời kỳ kinh tế khó khăn ngày nay, ngay cả ở Byteland. Để giảm chi phí hoạt động, chính phủ Byteland đã quyết định tối ưu hóa chiếu sáng đường bộ. Cho đến nay, mỗi con đường đều được chiếu sáng suốt đêm, chi phí 1 Đô la Byteland cho mỗi mét và mỗi ngày. Để tiết kiệm tiền, họ đã quyết định không còn chiếu sáng mọi con đường nữa, mà sẽ tắt đèn chiếu sáng của một số con đường. Để đảm bảo rằng cư dân Byteland vẫn cảm thấy an toàn, họ muốn tối ưu hóa việc chiếu sáng theo cách mà sau khi tối một số con đường vào ban đêm, vẫn còn ít nhất một lộ trình được chiếu sáng từ mọi ngã tư ở Byteland đến mọi ngã tư khác.

Số tiền tối đa hàng ngày mà chính phủ Byteland có thể tiết kiệm mà không làm cho cư dân của họ cảm thấy không an toàn là bao nhiêu?

Đầu vào

Tệp đầu vào chứa một số trường hợp kiểm tra. Mỗi trường hợp kiểm tra bắt đầu với hai số m và n, số lượng ngã tư ở Byteland và số lượng đường ở Byteland, tương ứng. Đầu vào kết thúc bằng m = n = 0. Nếu không, 1 < m < 200000 và m - 1 < n < 200000. Sau đó theo sau là n bộ ba số nguyên x, y, z chỉ ra rằng sẽ có một con đường hai chiều giữa x và y với chiều dài z mét (0 < x, y < m và x ≠ y). Đồ thị được chỉ định bởi mỗi trường hợp kiểm tra là liên thông. Tổng chiều dài của tất cả các con đường trong mỗi trường hợp kiểm tra là nhỏ hơn 2^31.

Đầu ra

Đối với mỗi trường hợp kiểm tra, in ra một dòng chứa số tiền tối đa hàng ngày mà chính phủ có thể tiết kiệm.

Ví dụ

Input	Output
7 11 0 1 7 0 3 5 1 2 8 1 3 9 1 4 7 2 4 5 3 4 15 3 5 6 4 5 8 4 6 9 5 6 11 0 0	51

Problem T. Cây 14

Time Limit	1000 ms
OS	Linux
Statement	PDF

Nhằm thu hút đầu tư, Chính phủ của một quốc gia đang phát triển muốn cải thiện giao thông ở một trong những khu vực khó tiếp cận nhất của mình. Khu vực này bao gồm một số địa điểm quan trọng cần phải có quyền truy cập vào sân bay.

Một trong những lựa chọn, là xây dựng một sân bay ở mỗi địa điểm này, nhưng có thể sẽ rẻ hơn nếu xây dựng ít sân bay hơn và có đường liên kết chúng với tất cả các địa điểm khác. Vì đây là các con đường dài kết nối các địa điểm chính trong quốc gia (ví dụ như thành phố, làng lớn, khu công nghiệp), tất cả các con đường đều là hai chiều. Cũng có thể có nhiều hơn một con đường trực tiếp giữa hai khu vực, bởi vì có thể có nhiều cách để liên kết hai khu vực (ví dụ, một con đường đào qua núi trong khi con đường khác vòng qua núi v.v.) với chi phí có thể khác nhau.

Một địa điểm được coi là có quyền truy cập vào sân bay nếu nó chứa một sân bay hoặc nếu có thể di chuyển bằng đường bộ từ đó đến một địa điểm khác có sân bay.

Bạn được cung cấp chi phí xây dựng một sân bay và danh sách các con đường có thể xây dựng giữa các cặp địa điểm và chi phí tương ứng của chúng. Chính phủ giờ đây cần sự giúp đỡ của bạn để quyết định cách rẻ nhất để đảm bảo rằng mọi địa điểm đều có quyền truy cập vào sân bay. Mục tiêu là làm cho việc truy cập sân bay dễ dàng nhất có thể, vì vậy nếu có nhiều cách để đạt được chi phí tối thiểu, hãy chọn cách có nhiều sân bay nhất.

Đầu vào

- Dòng đầu tiên của đầu vào chứa số nguyên $T(T < 25)$, số lượng trường hợp kiểm tra. Phần còn lại của đầu vào bao gồm T trường hợp.
- Mỗi trường hợp bắt đầu với hai số nguyên N, M và $A(0 < N < 10,000, 0 < M < 100,000, 0 < A < 10,000)$ được tách bởi khoảng trắng. N là số lượng địa điểm, M là số lượng con đường có thể được xây dựng, và A là chi phí xây dựng một sân bay.
- M dòng tiếp theo mỗi dòng chứa ba số nguyên X, Y và C $(1 < X, Y < N, 0 < C < 10,000)$, được tách bởi khoảng trắng. X và Y là hai địa điểm, và C là chi phí xây dựng một con đường giữa X và Y.

Đầu ra

- Xuất ra trên T dòng, mỗi dòng cho một trường hợp. Mỗi dòng nên có dạng `Case #X: Y Z` trong đó X là số thứ tự trường hợp, Y là chi phí tối thiểu để xây dựng đường và sân bay sao cho tất cả các địa điểm có quyền truy cập ít nhất một sân bay, và Z là số lượng sân bay cần được xây dựng. Như đã đề cập trước đó, nếu có nhiều câu trả lời với chi phí tối thiểu, hãy chọn cách tối đa hóa số lượng sân bay.

Ví dụ

Input	Output
2 4 4 100 1 2 10 4 3 12 4 1 41 2 3 23 5 3 1000 1 2 20 4 5 40 3 2 30	Case #1: 145 1 Case #2: 2090 2

Problem U. Cây HLD

Time Limit	3000 ms
OS	Linux
Statement	PDF

Trong một đồ thị vô hướng có trọng số cạnh, cây khung nhỏ nhất là một tập hợp các cạnh có tổng trọng số nhỏ nhất sao cho bất kỳ hai nút nào cũng được kết nối bằng một đường đi chỉ chứa các cạnh này. Một thuật toán phổ biến để tìm cây khung nhỏ nhất T trong đồ thị là như sau:

- Ban đầu, để T rỗng.
- Xem xét các cạnh e-1,...,em theo thứ tự tăng dần của trọng số.
  — Thêm ei vào T nếu hai điểm cuối của ei không được kết nối bởi một đường đi trong T.

Một thuật toán khác là như sau:

- Ban đầu, để T là tập hợp tất cả các cạnh.
- Trong khi có một số chu trình C trong T.
  — Loại bỏ cạnh e khỏi T nơi e có trọng số nặng nhất trong C.

Nhiệm vụ của bạn là thực hiện một hàm liên quan đến thuật toán này. Cho một đồ thị vô hướng G với trọng số cạnh, nhiệm vụ của bạn là xuất ra tất cả các cạnh là cạnh nặng nhất trong một số chu trình của G.

Đầu vào

Đầu vào đầu tiên của mỗi trường hợp bắt đầu với số nguyên n và m với 1 < n < 1,000 và 0 < m < 25,000 nơi n là số lượng nút và m là số lượng cạnh trong đồ thị. Theo sau là m dòng chứa ba số nguyên u, v, và w mô tả một cạnh có trọng số w nối các nút u và v nơi 0 < u, v < n và 0 < w < 2^31. Đầu vào kết thúc với một dòng chứa n = m = 0; trường hợp này không được xử lý. Bạn có thể giả định không có hai cạnh nào có cùng trọng số và không có hai nút nào được kết nối trực tiếp bởi nhiều hơn một cạnh.

Đầu ra

Đầu ra cho mỗi trường hợp đầu vào bao gồm một dòng chứa trọng số của tất cả các cạnh là cạnh nặng nhất trong một số chu trình của đồ thị đầu vào. Các trọng số này nên xuất hiện theo thứ tự tăng dần và các trọng số liên tiếp nên được tách bằng một khoảng trắng. Nếu đồ thị không có chu trình thì xuất ra văn bản 'forest' thay vì các số.

Ví dụ

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

Problem V. Cây Centroid

Time Limit	2000 ms
Mem Limit	65536 kB
Input File	stdin
Output File	stdout

Công ty của Nick đã thuê n người. Bây giờ Nick cần xây dựng một cấu trúc cây về mối quan hệ «cấp trên - cấp dưới» trong công ty (nghĩa là mỗi nhân viên, trừ một người, đều có đúng một cấp trên). Có m đơn xin việc được viết dưới dạng sau: «nhân viên a_i sẵn sàng trở thành cấp trên của nhân viên b_i với chi phí bổ sung là c_i». Trình độ q_j của mỗi nhân viên đã biết, và đối với mỗi đơn xin việc, điều sau đây là đúng: $q_{a_i} > q_{b_i}$.

Bạn có thể giúp Nick tính toán chi phí tối thiểu để xây dựng cấu trúc này, hoặc xác định rằng việc xây dựng không thể thực hiện được.

Nhập vào

Dòng đầu tiên chứa số nguyên n ($1 \le n \le 1000$) — số lượng nhân viên trong công ty. Dòng tiếp theo chứa n số được phân tách bằng dấu cách q_j ($0 \le q_j \le 10^6$)— trình độ của các nhân viên. Dòng tiếp theo chứa số m ($0 \le m \le 10000$) — số lượng đơn xin việc nhận được. m dòng tiếp theo chứa các đơn xin việc, mỗi đơn trong dạng ba số được phân tách bằng dấu cách: a_i, b_i và c_i ($1 \le a_i, b_i \le n$, $0 \le c_i \le 10^6$). Các đơn xin việc khác nhau có thể giống nhau, tức là chúng có thể đến từ cùng một nhân viên đã đề nghị trở thành cấp trên của cùng một người nhưng với chi phí khác nhau. Đối với mỗi đơn xin việc $q_{a_i} > q_{b_i}$.

Đầu ra

Đầu ra chỉ gồm một dòng — chi phí tối thiểu để xây dựng cấu trúc này, hoặc `-1` nếu không thể thực hiện được.

Ví dụ 1

Input	Output
4 7 2 3 1 4 1 2 5 2 4 1 3 4 1 1 3 5	11

Ví dụ 2

Input	Output
3 1 2 3 2 3 1 2 3 1 3	-1

Ghi chú

Trong ví dụ 1, một trong những cách có thể để xây dựng cấu trúc là chấp nhận các đơn xin việc có chỉ số 1, 2 và 4, cho tổng chi phí tối thiểu là 11. Trong ví dụ 2, không thể xây dựng cấu trúc cần thiết, vì vậy câu trả lời là `-1`.

Problem W. Cây nói nhé

Time Limit 1000 ms
Mem Limit 131072 kB

Mô tả đề bài

Có n thành phố và m con đường nối giữa chúng. Tuy nhiên, tất cả các con đường hiện tại đều không thể sử dụng vì tình trạng hư hỏng nặng. Nhiệm vụ của bạn là chọn một số con đường để sửa chữa sao cho có thể đi từ bất kỳ thành phố nào đến bất kỳ thành phố nào khác (tức là toàn bộ hệ thống giao thông được kết nối).

Mỗi con đường có một chi phí sửa chữa nhất định. Bạn cần tìm cách kết nối tất cả các thành phố với chi phí sửa chữa nhỏ nhất.

Dữ liệu

- Dòng đầu tiên chứa hai số nguyên n và m — số thành phố và số con đường ($1 \leq n \leq 10^5$, $1 \leq m \leq 2 \cdot 10^5$).
- m dòng tiếp theo, mỗi dòng gồm ba số nguyên a, b, c — biểu thị có một con đường hai chiều nối thành phố a và b với chi phí sửa chữa là c ($1 \leq a, b \leq n$, $1 \leq c \leq 10^9$).

Mỗi cặp thành phố chỉ có tối đa một con đường nối giữa chúng, và không có đường nào nối một thành phố với chính nó.

Kết quả

- In ra một số nguyên là tổng chi phí sửa chữa nhỏ nhất để đảm bảo mọi thành phố đều được kết nối.
- Nếu không thể kết nối tất cả các thành phố, in ra `"IMPOSSIBLE"`.

Ví dụ

Input	Output
5 6 1 2 3 2 3 5 2 4 2 3 4 8 5 1 7 5 4 4	14

Problem X. Lát gạch

Time Limit 1000 ms
Mem Limit 262144 kB

Bờm được thuê lát gạch mặt sàn một quảng trường bằng các viên gạch hình chữ nhật có kích thước khác nhau. Mặt sàn có thể được xem là một lưới các ô vuông, vị trí các điểm trên mặt sàn được xác định bằng hệ tọa độ Descartes. Bờm đặt các viên gạch không theo thứ tự nào nhưng cẩn thận đặt góc của viên gạch trùng với điểm trên mặt sàn và cạnh của gạch thì song song với trục tọa độ (trục hoành và trục tung). Các viên gạch có thể tiếp xúc nhau nhưng không nằm chồng lên nhau. Vì không đặt gạch theo thứ tự nên sau khi lát được N viên gạch thì Bờm đã tạo ra nhiều khu vực đã lát gạch. Một khu vực đã lát gồm các viên gạch có cạnh tiếp xúc nhau hoặc có chung góc. Ông chủ chỉ trả tiền cho một khu vực đã lát nên Bờm muốn tìm khu vực có diện tích lớn nhất để tính công.

Yêu cầu: Viết một chương trình tính diện tích khu vực đã lát gạch lớn nhất.

Input

Dòng đầu tiên chứa một số nguyên dương N cho biết số viên gạch Bờm đã lát.

N dòng tiếp theo mô tả vị trí và kích thước của các viên gạch đã lát. Mỗi dòng chứa 4 số nguyên X, Y, D, C với (X, Y) là tọa độ của góc dưới bên trái viên gạch trên mặt sàn và D là chiều dài (độ lớn của cạnh song song với trục hoành), C là chiều cao (độ lớn của cạnh song song với trục tung) của viên gạch.

Ràng buộc: Tất cả các test có giá trị X, Y, diện tích khu vực đã lát trong giới hạn số nguyên có dấu 32 bit và $0 < D, C \leq 500$.

Output

In ra một số nguyên cho biết diện tích khu vực đã lát gạch lớn nhất.

Scoring

Subtask	Điểm	Giới hạn
1	20	$0 < N \leq 100$
2	20	$100 < N \leq 1\,000$
3	60	$1\,000 < N \leq 50\,000$

Input	Output
6 6 4 3 1 6 1 1 1 4 7 1 1 3 3 1 2 4 6 2 1 4 1 2 2	7

Problem Y. Hàng rào 1

Input File fencedin.in
Output File fencedin.out

Farmer John has realized that many of his cows are strangely agoraphobic (being fearful of large open spaces). To try and make them less afraid of grazing, he partitions his large field into a number of smaller regions by building vertical (north-south) and horizontal (east-west) fences.

The large field is a rectangle with corner points at $(0, 0)$ and (A, B). FJ builds n vertical fences ($0 \leq n \leq 25,000$) at distinct locations $a_1 \ldots a_n$ ($0 < a_i < A$); each fence runs from $(a_i, 0)$ to (a_i, B). He also builds m horizontal fences ($0 \leq m \leq 25,000$) at locations $b_1 \ldots b_m$ ($0 < b_i < B$); each such fence runs from $(0, b_i)$ to (A, b_i). Each vertical fence crosses through each horizontal fence, subdividing the large field into a total of $(n + 1)(m + 1)$ regions.

Unfortunately, FJ completely forgot to build gates into his fences, making it impossible for cows to leave their enclosing region and travel around the entire field! He wants to remedy this situation by removing pieces of some of his fences to allow cows to travel between adjacent regions. He wants to select certain pairs of adjacent regions and remove the entire length of fence separating them; afterwards, he wants cows to be able to wander through these openings so they can travel anywhere in his larger field.

For example, FJ might take a fence pattern looking like this:

```
+---+--+
|   |  |
+---+--+
|   |  |
|   |  |
+---+--+
```

and open it up like so:

```
+---+--+
|      |
+---+  +
|      |
|      |
+---+--+
```

Please help FJ determine the minimum total length of fencing he must remove to accomplish his goal.

INPUT FORMAT (file fencedin.in):

The first line of input contains A, B, n, and m ($1 \leq A, B \leq 1,000,000,000$). The next n lines contain $a_1 \ldots a_n$, and the next m lines after that contain $b_1 \ldots b_m$.

OUTPUT FORMAT (file fencedin.out):

Please write the minimum length of fencing FJ must remove. Note that this might be too large to fit into a standard 32-bit integer, so you may need to use 64-bit integer types (e.g., "long long" in C/C++).

Input	Output
15 15 5 2 2 5 10 6 4 11 3	44

Problem credits: Brian Dean

Problem Z. Hàng rào 10

Input File fencedin.in
Output File fencedin.out

Farmer John has realized that many of his cows are strangely agoraphobic (being fearful of large open spaces). To try and make them less afraid of grazing, he partitions his large field into a number of smaller regions by building vertical (north-south) and horizontal (east-west) fences.

The large field is a rectangle with corner points at $(0, 0)$ and (A, B). FJ builds n vertical fences ($0 \leq n \leq 2000$) at distinct locations $a_1 \ldots a_n$ ($0 < a_i < A$); each fence runs from $(a_i, 0)$ to (a_i, B). He also builds m horizontal fences ($0 \leq m \leq 2000$) at locations $b_1 \ldots b_m$ ($0 < b_i < B$); each such fence runs from $(0, b_i)$ to (A, b_i). Each vertical fence crosses through each horizontal fence, subdividing the large field into a total of $(n + 1)(m + 1)$ regions.

Unfortunately, FJ completely forgot to build gates into his fences, making it impossible for cows to leave their enclosing region and travel around the entire field! He wants to remedy this situation by removing pieces of some of his fences to allow cows to travel between adjacent regions. He wants to select certain pairs of adjacent regions and remove the entire length of fence separating them; afterwards, he wants cows to be able to wander through these openings so they can travel anywhere in his larger field.

For example, FJ might take a fence pattern looking like this:

```
+---+--+
|   |  |
+---+--+
|   |  |
|   |  |
+---+--+
```

and open it up like so:

```
+---+--+
|      |
+---+  +
|      |
|      |
+---+--+
```

Please help FJ determine the minimum total length of fencing he must remove to accomplish his goal.

INPUT FORMAT (file fencedin.in):

The first line of input contains A, B, n, and m ($1 \leq A, B \leq 1,000,000,000$). The next n lines contain $a_1 \ldots a_n$, and the next m lines after that contain $b_1 \ldots b_m$.

OUTPUT FORMAT (file fencedin.out):

Please write the minimum length of fencing FJ must remove. Note that this might be too large to fit into a standard 32-bit integer, so you may need to use 64-bit integer types (e.g., "long long" in C/C++).

Input	Output
15 15 5 2 2 5 10 6 4 11 3	44

Problem credits: Brian Dean

Problem AA. Cây rau má

Time Limit 1000 ms
Mem Limit 262144 kB

In the country of IOI, there was a railway mogul JOI that owned all of the railroads. However, JOI has passed away and the railways he owns are to be distributed as a split inheritance.

In IOI, there are N cities and there are M railroads that connect the cities. The cities are numbered 1 to N, and the railroads are numbered 1 to M. A railroad i runs between city A_i and B_i in both directions and generates a *distinct* profit of C yen. There may be several lines between the same two cities.

The railroads will be divided among the K number of children that JOI has. Each child is given a number from 1 to K in order of their age, with the oldest child being number 1. Each child will receive some number of railroads out of the total M railroads (possibly receiving 0). First, child 1 will choose a set of railroads to inherit from the set of M railroads. Next, child 2 will choose anohter set of railroads to inherit from the remaining number of railroads. Following the above pattern, the K children decide which railroads to inherit.

However, there are a couple of rules about how the children will choose their railroads.

1. A child won't choose a railroad that has already been chosen. For example, if the first child chooses railroad i, the second child wouldn't be able to choose railroad i anymore.
2. A child won't choose railroads that form a "cycle". A "cycle" is formed when you can start at a city and can get back to it using each edge in a distinct set of edges once.
3. As all the children are as greedy as their father, they will always choose the set of railroads that yield the optimal profit. It is guaranteed that there is only one inheritance arrangement that meets the above rules.

If a railroad is not chosen by any child, it becoms publicly owned.

Given the railroads and the children, determine which child will choose which railroads.

Input

The first line contains 3 space-separated integers N, M, and K, which are the number of cities, railways, and children respectively.

The following M lines contain the space-separated integers A_i, B_i, and C_i, indicating that there is a railway connecting A_i and B_i that generates C_i yen.

Output

The output should consist of M lines.
Line i should contain the number of the child that receives the i th railway.
If no child owns the railway, output 0 instead.

Constraints

$2 \leq N \leq 1000$
$1 \leq M \leq 300000$
$1 \leq K \leq 10000$
$1 \leq A_i, B_i \leq N (1 \leq i \leq M)$
$A_i \neq B_i (1 \leq i \leq M)$
$1 \leq C_i \leq 1000000000$
$C_i \neq C_j (1 \leq i < j \leq M)$

Влез	Излез
3 5 2 1 2 3 1 2 1 2 3 4 2 3 6 1 3 2	1 0 2 1 2

Влез	Излез
3 6 5 1 2 1 1 2 2 2 3 3 2 3 4 3 1 5 3 1 6	4 3 2 1 2 1

Problem AB. Cây FFT

Time Limit	2000 ms
Mem Limit	262144 kB
Input File	stdin
Output File	stdout

Có một cách dễ dàng để lấy một nhiệm vụ mới từ một nhiệm vụ cũ gọi là "Đảo ngược vấn đề": chúng ta đưa ra một đầu ra của nhiệm vụ gốc và yêu cầu tạo ra một đầu vào, sao cho giải pháp cho vấn đề gốc sẽ tạo ra đầu ra mà chúng ta đã cung cấp. Nhiệm vụ khó khăn của Topcoder Open 2014 Vòng 2C, InverseRMQ, là một ví dụ tốt.

Bây giờ hãy tạo một nhiệm vụ theo cách này. Chúng ta sẽ sử dụng nhiệm vụ: bạn được cho một cây, hãy tính khoảng cách giữa bất kỳ cặp nút nào của nó. Vâng, điều này rất dễ, nhưng phiên bản đảo ngược thì khó hơn một chút: bạn được cho một $n \times n$ ma trận khoảng cách. Xác định xem nó có phải là ma trận khoảng cách của một cây có trọng số (tất cả trọng số phải là số nguyên dương) hay không.

Input

Dòng đầu tiên chứa một số nguyên n ($1 \le n \le 2000$) — số lượng nút trong đồ thị đó.

Tiếp theo n dòng mỗi dòng chứa n số nguyên $d_{i,j}$ ($0 \le d_{i,j} \le 10^9$) — khoảng cách giữa nút i và nút j.

Output

Nếu tồn tại một cây như vậy, xuất "CÓ", nếu không xuất "KHÔNG".

Examples

Input	Output
3 0 2 7 2 0 9 7 9 0	YES

Input	Output
3 1 2 7 2 0 9 7 9 0	NO

Input	Output
3 0 2 2 7 0 9 7 9 0	NO

Input	Output
3 0 1 1 1 0 1 1 1 0	NO

Input	Output
2 0 0 0 0	NO

Note

Trong ví dụ đầu tiên, cây yêu cầu tồn tại. Nó có một cạnh giữa các nút 1 và 2 với trọng số 2, một cạnh khác giữa các nút 1 và 3 với trọng số 7.

Trong ví dụ thứ hai, điều này là không thể vì $d_{1,1}$ nên là 0, nhưng nó là 1.

Trong ví dụ thứ ba, điều này là không thể vì $d_{1,2}$ nên bằng $d_{2,1}$.

Problem AC. Cây GCD

Time Limit 2000 ms
Mem Limit 262144 kB

You are given an array a of n ($n \geq 2$) positive integers and an integer p. Consider an undirected weighted graph of n vertices numbered from 1 to n for which the edges between the vertices i and j ($i < j$) are added in the following manner:

- If $gcd(a_i, a_{i+1}, a_{i+2}, \ldots, a_j) = min(a_i, a_{i+1}, a_{i+2}, \ldots, a_j)$, then there is an edge of weight $min(a_i, a_{i+1}, a_{i+2}, \ldots, a_j)$ between i and j.
- If $i + 1 = j$, then there is an edge of weight p between i and j.

Here $gcd(x, y, \ldots)$ denotes the greatest common divisor (GCD) of integers x, y, ...

Note that there could be multiple edges between i and j if both of the above conditions are true, and if both the conditions fail for i and j, then there is no edge between these vertices.

The goal is to find the weight of the minimum spanning tree of this graph.

Input

The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases.

The first line of each test case contains two integers n ($2 \leq n \leq 2 \cdot 10^5$) and p ($1 \leq p \leq 10^9$) — the number of nodes and the parameter p.

The second line contains n integers $a_1, a_2, a_3, \ldots, a_n$ ($1 \leq a_i \leq 10^9$).

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

Output t lines. For each test case print the weight of the corresponding graph.

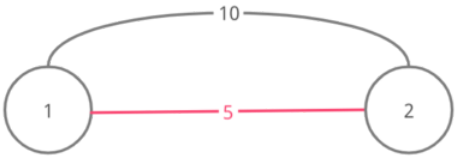
Examples

Input	Output
4 2 5 10 10 2 5 3 3 4 5 5 2 4 9 8 8 5 3 3 6 10 100 9 15	5 3 12 46

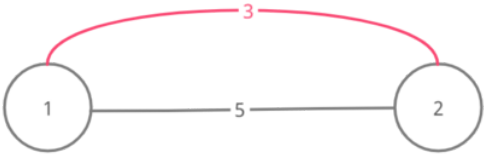
Note

Here are the graphs for the four test cases of the example (the edges of a possible MST of the graphs are marked pink):

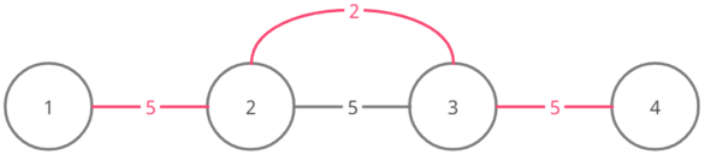
For test case 1



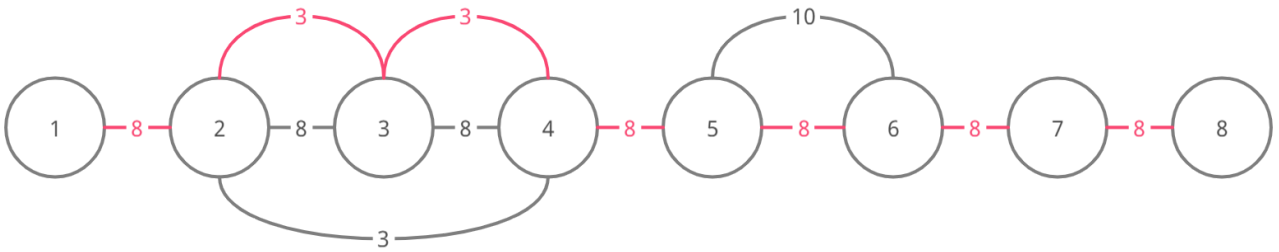
For test case 2



For test case 3



For test case 4


3
3
10
1
8
2
8
3
8
4
8
5
8
6
8
7
8
8
3

Problem AD. Cây vàng 304

Time Limit 1000 ms
Mem Limit 262144 kB

Bear là môt nhà du thám (du lịch và thám hiểm) nổi tiếng với khả năng di chuyển và trải nghiệm trong những điều kiện vô cùng khắc nghiệt. Trong chuyến du thám sắp tới, anh sẽ đến với một vùng biển hẻo lánh của biển Nam Thái Bình Dương. Quần đảo này gồm N đảo (các đảo được đánh số thứ tự từ 1 đến N). Việc di chuyển giữa N đảo này được đáp ứng bởi m chuyến phà (mỗi chuyến phà đáp ứng nhu cầu di chuyển đi lại giữa hai đảo cố định nào đó), đủ đảm bảo để từ mỗi đảo có thể đến được bất kỳ đảo khác bằng cách trực tiếp hoặc thông qua các tuyến phà trung gian.

Sau khi có được những thông tin cần thiết, Bear đặt ra nhiệm vụ như sau cho chuyến du thám: Chọn ra $N-1$ trong số M tuyến phà để thực hiện hành trình đến N đảo, mỗi đảo ít nhất một lần, sao cho tổng thời gian thực hiện (đơn vị tính là phút) là nhỏ nhất. Bear sẽ đổ bộ xuống đảo 1, thực hiện hành trình, quay về đảo 1 rồi thoát khỏi máy bay để hoàn thành nhiệm vụ đặt ra. Hành trình có thể phải lặp lại nhiều hơn 1 lần đối với một số đảo cũng như tuyến phà.

Các thông tin mà Bear có được bao gồm:

- M tuyến phà với thời gian di chuyển tương ứng bởi tuyến đó;
- Thời gian mà Bear cần để thoát ra khỏi mỗi đảo kể từ lúc đặt chân đến.

Yêu cầu: Hãy tính xem trong chuyến du thám của mình, Bear có thể hoàn thành nhiệm vụ đặt ra với tổng thời gian nhỏ nhất là bao nhiêu?

Input

Cho trong file văn bản `BEAR.INP` có cấu trúc như sau:

- Dòng đầu tiên ghi hai số nguyên N và M $(5 \leq N \leq 10000,\ N < M \leq 100000)$;
- Dòng thứ 2 ghi N số nguyên $S_1,\ S_2, \cdots,\ S_N$ cho biết: S_i là số đơn vị thời gian tối thiểu để Bear thoát ra khỏi đảo i $(1 \leq S_i \leq 1000;\ i = 1,\ 2, \cdots, N)$;
- Mỗi dòng trong M dòng tiếp theo ghi thông tin một tuyến phà bao gồm 3 số nguyên $u,\ v$ và T $(1 \leq u,\ v \leq N,\ 1 \leq T \leq 1000)$ với ý nghĩa: T là số đơn vị thời gian di chuyển theo tuyến phà giữa các đảo u và v.

Lưu ý: nói riêng tại đảo 1, mỗi lần thâm nhập vào đảo này, Bear cũng đều cần thời gian tối thiểu S_1 mới thoát ra khỏi nó.

Output

Ghi ra tệp văn bản `BEAR.OUT` duy nhất một số nguyên, là tổng thời gian nhỏ nhất mà Bear có thể cần để hoàn thành nhiệm vụ.

Scoring

50% số Test ứng với 50% số điểm của bài có $N \leq 1000$

Input	Output
6 10 5 2 7 4 5 8 1 3 5 2 3 6 3 1 4 2 4 7 5 6 3 4 5 8 2 6 6 5 3 5 2 5 9 3 4 4	105