**Ví dụ**

**Input**

2

4

2

0 1

1 2

* Người 0 bạn với người số 1
* Người 1 bạn với người số 2

Nếu muốn cả 4 người đều có được thông tin, thì chỉ cần truyền cho người số 0 và số 3

0 → 1 → 2

3

⇒ 2

3

0

Truyền dữ liệu cho từng người riêng lẻ

Truyền cho 0

Truyền cho 1

Truyền cho 2

⇒ 3

**Output**

2

3

Nếu người a biết được thông tin, anh ta sẽ truyền nó cho bạn của mình (b).

Nếu a và b là bạn, thì tạo 1 cạnh nối giữa họ.

⇒ Mỗi 1 thành phần liên thông, chỉ cần đúng 1 người biết thông tin

**Giải thuật**

**Bước 1:** Nhập dữ liệu.

**Bước 2:** Duyệt từng đỉnh, nếu đỉnh đó chưa được thăm, tiến hành thăm từ đỉnh đó, đồng thời tăng biến đếm lên 1.

**Bước 3:** In ra kết quả.

**Mã giả:**

**function DFS(src): {**

**s = stack;**

**s.push(src);**

**visited[src] = true;**

**while s.size > 0: {**

**u = s.top;**

**s.pop;**

**for v in graph[u]: {**

**if visited[v] == false: {**

**visited[v] = true;**

**s.push(v);**

**}**

**}**

**}**

**}**

**read(T);**

**for test = 1 to T: {**

**read(n);**

**read(E);**

**visited = []; // Mảng đánh dấu có n phần tử**

**graph = []; // Mảng động chứa n mảng động**

**for i = 1 to E: {**

**read(a, b);**

**graph[a].push(b);**

**graph[b].push(a);**

**}**

**for i = 0 to n-1:**

**visited[i] = false;**

**result = 0;**

**for i = 0 to n-1:**

**if visited[i] == false:**

**DFS(i);**

**print(result);**

**}**

Độ phức tạp: **O(T \* (V + E))**

**T** là số lượng testcase

**V** là số lượng đỉnh (n)

**E** là số lượng cạnh