**Ví dụ**

**2**

**4**

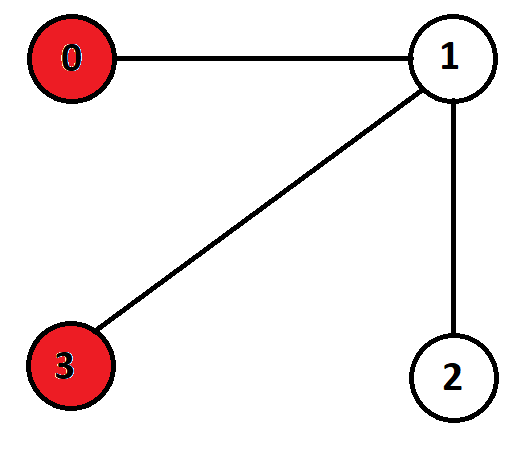
**3**

**0 1**

**2 1**

**1 3**

**0 3**

****

Xuất phát từ 0

Tập kết tại 3

**Cách 1:**

Cho 1 lính đi đặt bom ở 1: 0 → 1 → 3: 2 đơn vị

Cho 1 lính đi đặt bom ở 2: 0 → 1 → 2 → 1 → 3: 4 đơn vị

→ 4 đơn vị thời gian

**Cách 2:**

Cho 1 lính đi: 0 → 1 → 2 → 1 → 3: 4 đơn vị

→ 4 đơn vị thời gian

**⇒ 4**

Cho một lính đặt tại 0: 0 → 1 → 3: 2 đơn vị

Cho một lính đặt tại 1: 0 → 1 → 3: 2 đơn vị

Cho một lính đặt tại 2: 0 → 1 → 2 → 1 → 3: 4 đơn vị

Cho một lính đặt tại 3: 0 → 1 → 3: 2 đơn vị

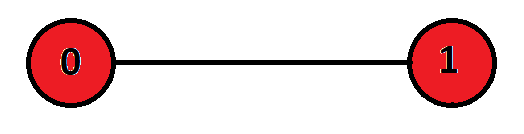
⇒ 4

**2**

**1**

**0 1**

**1 0**

****

Xuất phát từ 1

Tập kết tại 0

**Cách đi:** 1 → 0: 1 đơn vị thời gian

**⇒ 1**

Xuất phát từ s, tập kết tại d

Đặt bom ở tất cả các đỉnh

Số lượng lính và số lượng bom là vô hạn

Cho mỗi lính đặt một quả bom ở một đỉnh duy nhất, rồi về tập kết

**Tìm đường đi ngắn nhất từ s → u → d (với mọi u từ 0 → N-1)**

s → 0

s → 1

…

s → N-1

⇒ Tìm đường đi ngắn nhất từ s tới tất cả các đỉnh còn lại distS

0 → d

1 → d

…

N-1 → d

⇒ Tìm đường đi ngắn nhất từ d tới tất cả các đỉnh còn lại distD

Thời gian đặt bom cho u: distS[u] + distD[u]

**Kết quả: max(distS[u] + distD[u]) với mọi u từ 0 tới N-1**

**Giải thuật:**

**Bước 1:** Nhập dữ liệu.

**Bước 2:** Dijkstra từ đỉnh s và d.

**Bước 3:** Tìm **max(distS[u] + distD[u])** .

**Bước 4:** Đưa ra kết quả theo format.

**Mã giả**

**INF = 10^9;**

**function Dijkstra(s) {**

**dist = []; // N phần tử**

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

**dist[i] = INF;**

**pq = minHeap();**

**ps.push((0, s));**

**dist[s] = 0;**

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

**(w, u) = pq.top();**

**pq.pop();**

**if w > dist[u]:**

**continue;**

**for (weight, v) in graph[u]:**

**if w + weight < dist[v]: {**

**dist[v] = w + weight;**

**pq.push((dist[v], v));**

**}**

**}**

**return dist;**

**}**

**read(T);**

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

**read(N, R);**

**graph = [];**

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

**read(u, v);**

**graph[u].push((v, 1));**

**graph[v].push((u, 1));**

**}**

**read(s, d);**

**distS = Dijkstra(s);**

**distD = Dijkstra(d);**

**res = 0;**

**for u = 0 to N-1:**

**res = max(res, distS[u] + distD[u]);**

**print(‘Case ’, test,’: ‘, res);**

**}**

**Độ phức tạp: O(T \* E \* log(V))**

**T: số lượng testcase**

**E: số lượng cạnh**

**V: số lượng đỉnh**