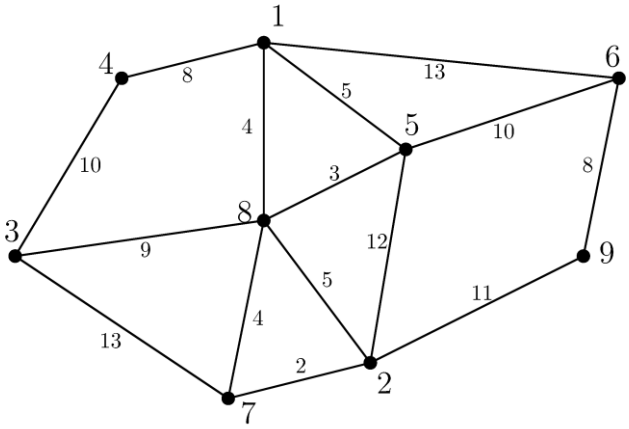
접선 (Contact)

두 명의 비밀요원이 간선(edge) 가중치가 있는 그래프로 표현되는 지역에 상주하고 있다. 이 둘은 특정한 시간에 어떤 정해진 장소에서 서로 만나는 접선 임무를 수행해야 한다. 두 요원은 이 지역의 어떤 지점에서 가장 빠른 시간에 서로 만나려고 한다. 단 두 비밀요원은 한 곳에 움직임 없이 가만히 있으면 매우 위험하기 때문에 어떤 사람이 한 곳에 먼저 도착해서 다른 사람을 기다려서는 안 된다. 즉 두 사람은 미리 정한 지점에 동시에 도착하여 만난 뒤에 비밀자료를 서로 교환하고 돌아가야 한다. 두 비밀요원이 거주하는 초기 장소는 서로 다른 정점(vertex)이며 두 사람이 움직이는 속도는 같다. 두 사람이 가장 빠른 시간에 동시에 만날 수 있는 위치는 정점, 간선(edge), 모두 가능하다. 아래 그래프와 같은 도시의 경우의 예를 들어 보자.

만일 두 요원이 처음 1, 3번 정점 위에 있다면 이 둘이 가장 빨리 만날 수 있는 지점은 간선(3,8) 위의 한 지점, 즉 3번 정점에서 6.5 거리만큼 떨어져 있는 곳이다. 만일 두 사람이 각각 5번, 2번 정점에서 출발한다면 두 요원이 가장 빨리 만날 수 있는 장소는 간선 (2,8)위의 한 점이다. 이 점은 정점 2에서 거리가 4만큼 떨어진 지점이다. 여러분은 두 요원의 초기 상태와 도시의 연결 그래프를 받아서 두 요원이 기다리지 않고 가장 빨리 만날 수 있는 지점을 찾아야 한다.



[입력]
입력 파일의 제일 첫째 줄에는 파일에 포함된 케이스의 수 T가 주어진다. 단, $T \leq 90$이다.

각 케이스의 첫 줄에 정점의 수 N, 간선의 수 M, 질문(query)의 개수 K, 이 3개의 정수가 순서대로 주어진다. 각 변수의 크기는 $1 \leq N \leq 500,\ 1 \leq M \leq 10{,}000$ 이다. K는 두 요원의 초기 정점 위치가 표시된 질문(query)의 개수이며 그 크기는 $1 \leq K \leq 3$이다. 정점은 1번부터 N번까지 연속된 정수로 표현된다. 이어지는 M개의 줄에 각 간선 (i,j)의 두 정점의 번호 i, j와 그 간선의 길이를 나타내는 정수 w, 이렇게 3개의 정수 'i j w'가 공백을 두고 한 줄에 모두 주어진다. 간선의 길이는 1 이상 10,000이하이다. 두 정점 사이에는 최대 하나의 간선만 존재한다. 이렇게 M개의 간선 정보가 M개의 줄에 하나씩 제시된 다음, 두 비밀 요원의 초기 위치를 포함한 K개 질문(query)을 나타내는 두 요원의 서로 다른 초기 정점의 번호 'a b'가 K개 줄에 각각 주어진다. 단 제시된 지역 그래프가 분리(disconnected)되어 두 사람이 만나지 못하는 경우는 없다.

입력은 다음의 세 가지 종류로 주어진다.

- Set 1: $1 \leq N \leq 20,\ 1 \leq M \leq 200, K = 1.$
- Set 2: $1 \leq N \leq 100,\ 1 \leq M \leq 1{,}000, K = 2.$
- Set 3: $1 \leq N \leq 500,\ 1 \leq M \leq 10{,}000, K = 3.$

[출력]

K개의 질문에 대해 두 요원이 만나야 하는 접선 지점이 포함된 간선 (i, j)를 찾아서 규칙(1), 규칙(2), 규칙(3)에 따라서 해당 간선에 관한 정보 'i j'를 출력해야 한다. 지금부터 모든 간선 (i, j)는 $i \leq j$인 형태로만 표현한다.

- **규칙(1)** 정점 i와 정점 j 사이 간선에서 만날 경우 'i j'를 한 줄에 출력한다. 단 $i < j$ 이다.
- **규칙(2)** 만일 두 요원이 정확하게 정점 i 위에서 만나는 경우에는 이 정점은 루프(loop)로 구성된 간선 (i, i)와 동일하므로 'i i'를 출력해야 한다.
- **규칙(3)** 만약 규칙(1), (2)를 만족하는 간선이 하나 이상이라면 간선 (x,y)의 첫번째 정점 번호 x가 가장 작은 간선을 선택한다. 만약 이 첫번째 정점 번호가 같을 경우에는 그 다음 두번째 정점 번호 y가 가장 작은 간선을 선택해서 출력해야 한다.

예를 들어 접선 장소로 가능한 위치를 포함한 간선들이 (8,8), (3,6), (2,7), (5,6), 이렇게 4개인 경우에는 첫번째 정점 번호가 가장 빠른 간선인 '2 7'을 출력해야 한다. 만일 찾아낸 간선이 (5,6), (3,3), (2,9), (2,6) 이렇게 4개라면 '2 6'을 출력해야 한다.

[입출력 예] 위 그림에 표시된 지역이 아래 입력의 첫번째 케이스로 주어져 있다.

입력

```
1            // T=1, 1개의 testing case
9 15 3    //  case 1. 정점은 9개, 간선은 15개, 질문(query)은 3개
1 4 8      // i j w
1 5 5
1 8 4
1 6 13
2 8 5
2 7 2
9 2 11
2 5 12
3 4 10
3 8 9
3 7 13
5 8 3
5 6 10
6 9 8      // 14번째 간선의 정보 (6,9) 그 길이는 8
7 8 4      // 15번째 정보. 그 다음 줄부터는 요원 위치 3건
1 3        // 두 요원의 초기 위치는 각각 정점 1과 3
5 2        // 두 요원의 초기 위치는 각각 정점 5와 2
3 6        // 두 요원의 초기 위치는 각각 정점 3와 6
```

출력

```
3 8              // 초기 위치가 1,3 일 때 접선 지점은 간선 (3,8)위에 있음
2 8              // 접선 지점은 간선 (2,8)위에 있음
5 8              // 초기 위치가 3,6 일 때 접선 지점은 간선 (5,8)위에 있음
```