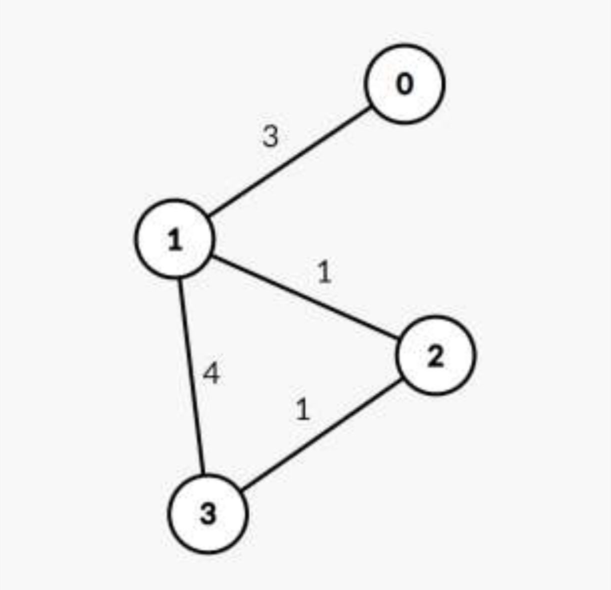
City With the Smallest Number of Neighbors at a Threshold Distance

There are n cities numbered from 0 to n-1. Given the array edges where **edges[i] = [from$_i$, to$_i$,weight$_i$]** represents a bidirectional and weighted edge between cities from$_i$ and to$_i$, and given the integer distance Threshold. You need to find out a city with the smallest number of cities that are reachable through some path and whose distance is **at most** Threshold Distance, If there are multiple such cities, our answer will be the city with the greatest number.

Note: that the distance of a path connecting cities ***i*** and ***j*** is equal to the sum of the edges' weights along that path.

Example 1:

```
Input:
N=4,M=4
edges = [[0,1,3],[1,2,1],[1,3,4],[2,3,1]]
distanceThreshold = 4
Output:3
Explaination:
```



```
The neighboring cities at a distanceThreshold = 4 for
each city are:
```

```
City 0 -> [City 1, City 2]
City 1 -> [City 0, City 2, City 3]
City 2 -> [City 0, City 1, City 3]
City 3 -> [City 1, City 2]
Cities 0 and 3 have 2 neighboring cities at a
distanceThreshold = 4, but we have to return city 3 since
it has the greatest number.
```

Your Task:
You don't need to read input or print anything. Your task is to complete the function **findCity( )** which takes a No of nodes N and vector of edges and ThresHold Distance. and Return the city with the smallest number of cities that are reachable through some path and whose distance is **at most** Threshold Distance, If there are multiple such cities, return the city with the greatest number.

Expected Time Complexity: O(V^2 + EVlogV)
Expected Auxiliary Space: O(N^3)

Constraints:
1 ≤ N≤ 100
1 <= edges.length <= n*(n-1)/2
edges[i].length == 3
0 <= from$_i$ < to$_i$ < n
1 <= weight, distanceThreshold <= 10^4
All pairs (from$_i$, to$_i$) are distinct