5. Best Telescope Site

Scientists want to select the best site to build their new space observatory. The best site is defined as the city with the least ambient light from surrounding cities which would reduce the quality of their telescope images.

There are *city_nodes* cities numbered from *1* to *city_nodes*. The cities are connected by bidirectional edges to form a connected graph. The weight of each edge represents the distance between the connected cities. There is a *distanceThreshold* that represents the minimum desired distance from any other city. Determine the city with the smallest number of neighboring cities that are nearer than the *distanceThreshold*. If there are multiple answers, choose the higher city number.

Example

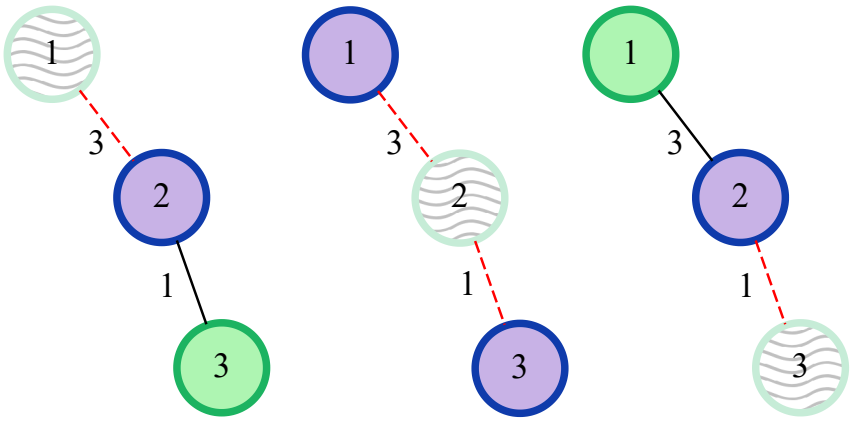
distanceThreshold = 3

city_nodes = 3

city_from = [1, 2]

city_to = [2, 3]

city_weight = [3, 1]



The distance between cities 1 and 3 is 3 + 1 = 4. In the graphs above, the nodes with wavy lines are the cities being considered. The blue nodes represent cities within the threshold distance of the city under consideration, and their edges are highlighted as red dashed lines. City 2 has two neighbors within the threshold distance while the others have only one. Of the cities 1 and 3, the higher city number is 3.

Function Description

Complete the function *findBestCity* in the editor below. The function has to return a single integer denoting the capital city.

findBestCity has the following parameter(s):

int distanceThreshold: Neighboring cities within this distance should be minimized

int city_nodes: the number of cities.

int city_from:[city_edges] integer array of size *city_edges,* such that *city_from[i]* denotes the first endpoint of the i^{th} edge in the city.

int city_to[city_edges]: integer array of size *city_edges,* such that *city_to[i]* denotes the second endpoint of the i^{th} edge in the city.

int city_weight[city_edges] : weight of i^{th} edge.

Constraints

- $2 \leq city\_nodes \leq 10^3$
- $n - 1 \leq city\_edges \leq min(10^3 , (n \times (n - 1)) / 2)$
- $0 \leq distanceThreshold \leq 10^3$
- $1 \leq city\_weight \leq 10^3$
- *Between any two cities no more than one road exists*
- *Each road connects two different cities.*
- *Each city is connected to every city by at least one path.*

Input Format Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

In the first line, there is a single integer *distanceThreshold* .

In the following line, there are two space-separated integers, *city_nodes* and *city_edges.*

city_edges lines follow.

In the i^{th} of them, there are *3* space-separated integers *u, v, w* which denotes that there is a bidirectional edge between *u* and *v* of weight *w*.

Sample Case 0

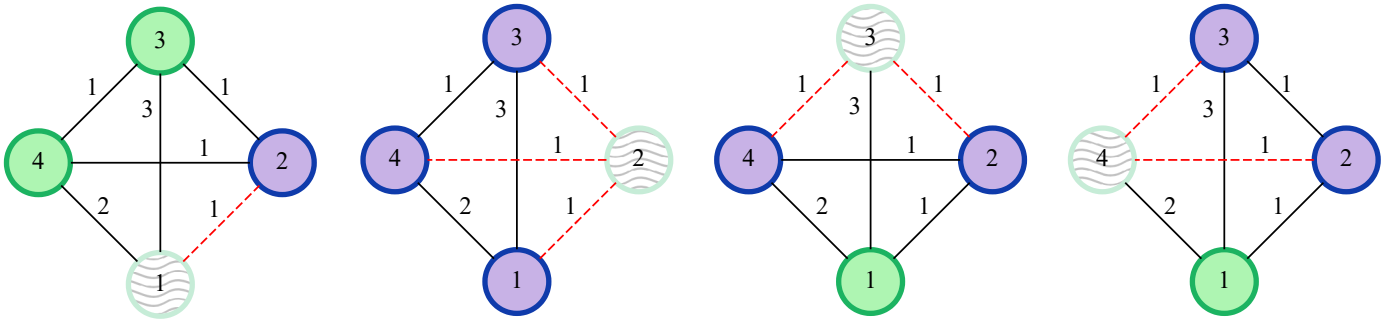
Sample Input

```
STDIN       Function
-----       --------
1       →   distanceThreshold = 1
4 6     →   city_nodes = 4, city_edges = 6
1 2 1   →   city_from[0] = 1, city_to[0] = 2, city_weight[0] = 1
1 3 3   →   city_from[1] = 1, city_to[1] = 3, city_weight[1] = 3
1 4 2   .
2 3 1   .
2 4 1   .
3 4 1   →   city_from[5] = 3, city_to[5] = 4, city_weight[5] = 1
```

Sample Output

```
1
```

Explanation



In the graphs above, the city being considered has wavy lines. The cities in blue are less than or equal to the threshold distance. The best location is in city 1 which has only 1 other city within the threshold distance.

Sample Case 1