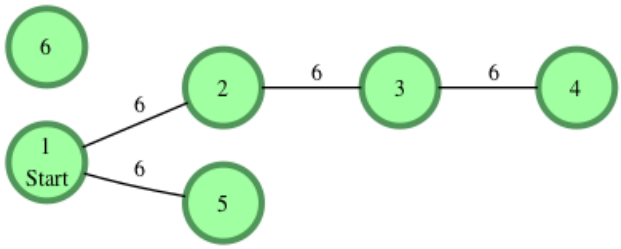
BFS: Shortest Reach in a Graph

Consider an undirected graph consisting of n nodes where each node is labeled from 1 to n and the edge between any two nodes is always of length 6. We define node s to be the starting position for a BFS. Given a graph, determine the distances from the start node to each of its descendants and return the list in node number order, ascending. If a node is disconnected, it's distance should be -1.

For example, there are $n = 6$ nodes in the graph with a starting node $s = 1$. The list of $edges = [[1, 2], [2, 3], [3, 4], [1, 5]]$, and each has a weight of 6.



Starting from node 1 and creating a list of distances, for nodes 2 through 6 we have $distances = [6, 12, 18, 6, -1]$.

Function Description

Define a Graph class with the required methods to return a list of distances.

Input Format

The first line contains an integer, q, the number of queries.

Each of the following q sets of lines is as follows:

- The first line contains two space-separated integers, n and m, the number of nodes and the number of edges.
- Each of the next m lines contains two space-separated integers, u and v, describing an edge connecting node u to node v.
- The last line contains a single integer, s, the index of the starting node.

Constraints

- $1 \leq q \leq 10$
- $2 \leq n \leq 1000$
- $1 \leq m \leq \frac{n \cdot (n-1)}{2}$
- $1 \leq u, v, s \leq n$

Output Format

For each of the q queries, print a single line of $n-1$ space-separated integers denoting the shortest distances to each of the $n-1$ other nodes from starting position s. These distances should be listed sequentially by node number (i.e., $1, 2, \ldots, n$), but *should not* include node s. If some node is unreachable from s, print -1 as the distance to that node.

Sample Input

```
2
4 2
1 2
1 3
1
3 1
2 3
2
```

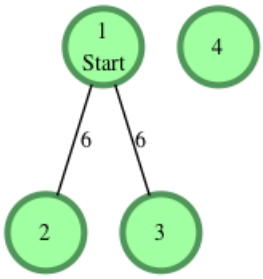
Sample Output

```
6 6 -1
-1 6
```

Explanation

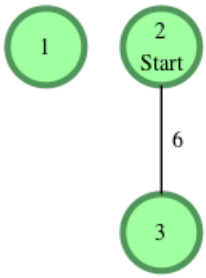
We perform the following two queries:

1. The given graph can be represented as:



   where our *start* node, s, is node 1. The shortest distances from s to the other nodes are one edge to node 2, one edge to node 3, and there is no connection to node 4.

2. The given graph can be represented as:



where our *start* node, s, is node 2. There is only one edge here, so node 1 is unreachable from node 2 and node 3 has one edge connecting it to node 2. We then print node 2's distance to nodes 1 and 3 (respectively) as a single line of space-separated integers: `-1 6`.

Note: Recall that the actual length of each edge is 6, and we print -1 as the distance to any node that's unreachable from s.