Need for Speed : Algorithm

One of your friends is a fan of the famous racing game Need For Speed `A.K.A` NFS. He plays vigorously and already finished the latest release "Need For Speed : Shift". Being a good gamer, he has a belief that he can do those stunts while driving in Dhaka city also. Several times he was speeding but managed to bypass police. Recently DMP (Dhaka Metropolitan Police) has got some cool cars just as your friend has and he understands that blind racing can't save the day now. Thats why he is pathetically in "Need for Algorithm".

The Street Racing area has n intersections and in total m roads between them. Intersections are numbered 1...n. All roads are equal in length. Your friend starts racing from node F and police starts from node P. Some of the intersections are marked as "safepoints". True friends like you live in those safepoints and they will help him to hide along with the car. So, now the objective of your friend is to reach any safepoint, ofcourse strictly before police. Otherwise they can catch him when he reaches there. Now your friend cries for your help. Given the city map, you have to tell him which of the safepoints can be his hiding place, how far is it from his starting point, and how far is it from the place where the police starts.

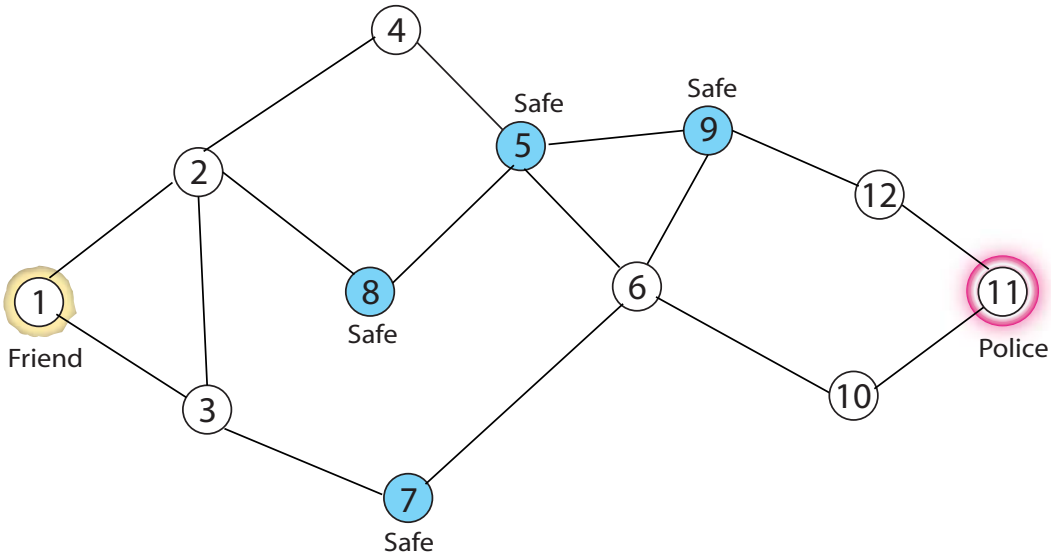

Figure 1: An example graph

In the example graph shown in figure 1, $n = 12, m = 16$. Suppose $F = 1, P = 11$. Safepoints are `5, 7, 8, 9`. Of them, intersection `7` and `8` are useful to your friend. Your friend should not target safepoint `5` or `9`, because he can't reach there before police.

Input Specification

First line of input consists of two integers n and m. $1 \leq n \leq 100$ and $1 \leq m \leq 4950$. First integer denotes number of vertices in the graph, and the second integer denotes number

of edges. The next m lines each contain two integers u, v which means there is an edge u, v in the graph. After all these, a line contains three integers. First one is F, starting point for the friend, next one is the starting point P for the police and last one is S, the number of safe points. Next line contains S integers each in the range $[1..n]$, denoting the safepoints in the map. The input file may contain multiple scenarios. Each of the scenario descriptions will be separated by a single blank line. You should stop taking inputs when 0 is input for both n and m.

Output Specification

For each of the scenario, at first print one line `x valid safepoints.` Then, for each of the useful safepoints in the input file, print one line which contains its vertex number, its shortest distance from your friend and its shortest distance from police. The lines should be sorted in the sequence of increasing distance from your friend. If two useful safepoints has same distance from your friend, then the one that is far from the police should be reported before. If there is again a tie, the one that has smaller vertex number should come before the one that has larger vertex number. If there is no useful safepoint in the scene, print "Caught.". Output for different scenarios should be separated by a blank line. Check the sample output section for clarification.

Sample Input

```
12 16
1 3
1 2
2 3
2 4
2 8
7 3
8 5
4 5
6 7
6 5
9 5
6 9
10 6
10 11
11 12
12 9
1 11 4
5 7 8 9

6 6
1 2
2 3
3 4
4 5
5 6
1 6
```

```
1 6 2
4 5

0 0
```

Sample Output

```
2 valid safepoints.
8 2 4
7 2 3

0 valid safepoints.
Caught.
```

Remarks

Graph Representation You must use an adjacency list representation for the graph. Your graph should have an associated data structure, with which you will be able to find all the neighbors of a node u in $O(deg[u])$ time. Try to follow the adjacency represenstation style shown in the class. You can use `STL` data structures also.

Algorithm You may use the idea of `BFS` (Breadth First Search) algorithm to complete this assignment.

Input Output Explanation In the sample input given, there are two scenes. The first scene starts with the line `12 16` and ends with the line `5 7 8 9`. Output for that scenario are the first 3 lines in the sample output. The second scene starts with `6 6` in the sample input file, and ends with `4,5`. Output for that scene is 5th and 6th line in the sample output. However, the last line of input is `0 0`. Obviously, you should not process this as a graph, rather just exit from your program. So your code should take input in a loop and process them.

```
while(scanf("%d%d",&n,&m)==2)
{
    if(n==0 && m==0) break;

    // some initialization here...

    for(i=0;i<m;i++)
    {
        scanf("%d%d",&u,&v);

        // insert the edge (u,v) in the adjacency list
        // ........
    }

    // .... process and print output
}
```

Md. Tanvir Al Amin
tanviralamin@gmail.com