# 12569   Planning mobile robot on Tree (EASY Version)

We are given a connected, undirected graph G on n vertices. There is a mobile robot on one of the vertices; this vertex is labeled s. Each of several other vertices contains a single movable obstacle. The robot and the obstacles may only reside at vertices, although they may be moved across edges. No vertex may ever contain more than one movable entity (robot or obstacles).

In one step, we may move either the robot or one of the obstacles from its current position v to a vacant vertex adjacent to v. Our goal is to move the robot to a designated vertex t using the smallest number of steps possible.

Let us call this graph motion planning with one robot, or GMP1R for short. In this problem, we restrict the graph G to be a tree, namely TMP1R.
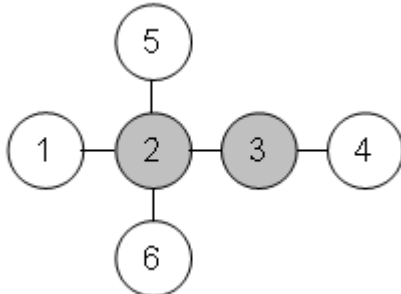
Here are some examples (gray circles represent obstacles).
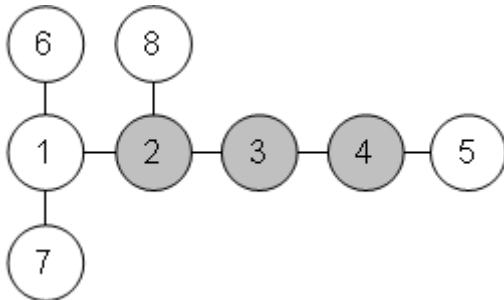
Example 1 (s=1, t=3):
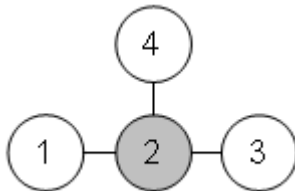
Move the obstacle 2-4, and then move the robot 1-2-3. Total: 3 moves.

Example 2 (s=1, t=4):

Move obstacle 2-5, then 3-2-6, and then move the robot 1-2-3-4. Total: 6 moves.

Example 3 (s=1, t=5):

Move the robot 1-6, then obstacle 2-1-7, then robot 6-1-2-8, then obstacle 3-2-1-6, then 4-3-2-1, and finally robot 8-2-3-4-5. Total: 16 moves.

Input

The first line contains the number of test cases T ($T \leq 340$). Each test case begins with four integers n, m, s, t ($4 \leq n \leq 15$, $0 \leq m \leq n-2$, $1 \leq s, t \leq n$, $s \neq t$), the number of vertices, the number of obstacles and the label of the source and target. Vertices are numbered 1 to n. The next line contains m different integers not equal to s, the vertices containing obstacles. Each of the next $n - 1$ lines contains two integers u and v ($1 \leq u < v \leq n$), that means there is an edge $u - v$ in the tree.

Output

For each test case, print the minimum number of moves k in the first line. Each of the next k lines contains two integers a and b, that means to move the robot/obstacle from a to b. If there is no solution, print '-1'. If there are multiple solutions, any will do.

Print a blank line after each test case.

Sample Input

```
3
4 1 1 3
2
1 2
2 3
2 4
6 2 1 4
2 3
1 2
2 3
3 4
2 5
2 6
8 3 1 5
2 3 4
1 2
2 3
3 4
4 5
1 6
1 7
2 8
```

Sample Output

```
Case 1: 3
2 4
1 2
2 3

Case 2: 6
2 5
3 2
2 6
1 2
```

```
2 3
3 4
Case 3: 16
1 6
2 1
1 7
6 1
1 2
2 8
3 2
2 1
1 6
4 3
3 2
2 1
8 2
2 3
3 4
4 5
```