# 深搜（dfs）

## 【题目描述】

深度优先搜索是一种常见的搜索算法。通过此算法，我们可以从一个无重边、无自环的无向连通图 $G=(V,E)$ ，和某个出发点 $s$，得到一棵树 $T$。

算法的流程描述如下：

1. 将栈 $S$ 设置为空，并令 $T=(V,\emptyset)$，即 $T$ 的边集初始为空。
2. 首先将出发点 $s$ 压入 $S$ 中。
3. 访问栈顶节点 $u$，并将 $u$ 标记为“已访问的”。
4. 如果存在与 $u$ 相邻且未被访问的节点，则**任意地**从这些节点中挑选一个记为 $v$。我们将边 $(u,v)$ 加入 $T$ 的边集中，并将 $v$ 压入栈 $S$ 中，然后**回到步骤 3**。若不存在这样的节点，则从栈中弹出节点 $u$。

可以证明，当图 $G$ 为连通图时，该算法会得到图的某一棵生成树 $T$。但**算法得到的树 $T$ 可能不是唯一的，它取决于搜索的顺序，也就是算法的第 4 步所选取的顶点**。指定出发点 $s$ 后，如果能够选取一种特定的搜索顺序，使得算法得到的树恰好是 $T$，则我们称 $T$ **是 $G$ 的一棵 $s$-dfs 树**。

现在给定一棵 $n$ 个顶点的树 $T$，顶点编号为 $1\sim n$，并额外给出 $m$ 条边。我们保证这 $m$ 条边两两不同，连接不同的顶点，且与 $T$ 中的 $n-1$ 条树边两两不同。我们称额外给出的 $m$ 条边为**非树边**。在这 $n$ 个顶点中，我们指定了恰好 $k$ 个顶点作为**关键点**。

现在你想知道，有多少种选取这 $m$ 条非树边的方法（可以全部不选），使得：将 $T$ 的边与被选中的非树边构成图 $G$ 之后，存在某个**关键点** $s$ ，使得 $T$ 是 $G$ 的一棵 $s$-dfs 树。

由于答案可能十分巨大，你只需要输出方案数在模 $(10^9+7)$ 意义下的值。

## 【输入格式】

从文件 ***dfs.in*** 中读入数据。

输入的第一行包含一个整数 $c$，表示测试点编号。$c=0$ 表示该测试点为样例。

输入的第二行包含三个正整数 $n,m,k$，分别表示顶点个数，非树边的数量，关键点的数量。

接下来 $n-1$ 行，每行包含两个正整数 $u,v$ 表示树 $T$ 的一条边。保证这 $n-1$ 条边构成了一棵树。

接下来 $m$ 行，每行包含两个正整数 $a,b$ 表示一条非树边。保证 $(a,b)$ 不与树上的边重合，且没有重边。

输入的最后一行包含 $k$ 个正整数 $s_1,s_2,\cdots,s_k$，表示 $k$ 个关键点的编号。保证 $s_1,s_2\cdots,s_k$ 互不相同。

### 【输出格式】

输出到文件 ***dfs.out*** 中。

输出一行包含一个非负整数，表示方案数在模 $(10^9+7)$ 意义下的值。

### 【样例 1 输入】

```
0
4 2 2
1 2
2 3
3 4
1 3
2 4
2 3
```

### 【样例 1 输出】

```
3
```

### 【样例 1 解释】

在这个样例中，有三种选取非树边的方法：只选取边 $(1,3)$，只选取边 $(2,4)$，或不选取任何一条非树边。

如果只选取边 $(1,3)$，或者不选取任何一条非树边，则我们发现 $T$ 都是图 $G$ 的 3-dfs 树。指定的搜索顺序如下：

1. 将 3 放入栈 $S$ 中。此时 $S=[3]$。
2. 将 3 标记为“已访问的”。
3. 由于 3 与 2 相连，且 2 是“未访问的”，将 2 放入栈 $S$ 中，并将 $(3,2)$ 加入树 $T$ 中，此时 $S=[3,2]$。
4. 将 2 标记为“已访问的”。
5. 由于 2 与 1 相连，且 1 是“未访问的”，将 1 放入栈 $S$ 中，并将 $(2,1)$ 加入树 $T$ 中，此时 $S=[3,2,1]$。
6. 由于与 1 相邻的点都是“已访问的”，将 1 弹出栈，此时 $S=[3,2]$。
7. 由于与 2 相邻的点都是“已访问的”，将 2 弹出栈，此时 $S=[3]$。
8. 由于 3 与 4 相连，且 4 是“未访问的”，将 4 放入栈 $S$ 中，并将 $(3,4)$ 加入树 $T$ 中，此时 $S=[3,4]$。
9. 由于与 4 相连的点都是“已访问的”，将 4 弹出栈，此时 $S=[3]$。

10. 由于与 3 相连的点都是“已访问的”，将 3 弹出栈，此时 $S$ 重新变为空。

如果只选取边 $(2,4)$，则我们可以说明 $T$ 是图 $G$ 的 2-dfs 树。指定的搜索顺序如下：

1. 将 2 放入栈 $S$ 中。此时 $S=[2]$。
2. 将 2 标记为“已访问的”。
3. 由于 2 与 3 相连，且 3 是“未访问的”，将 3 放入栈 $S$ 中，并将 (2 3) 加入树 $T$ 中，此时 $S=[2,3]$。
4. 将 3 标记为“已访问的”。
5. 由于 3 与 4 相连，且 4 是“未访问的”，将 4 放入栈 $S$ 中，并将 $(3,4)$ 加入树 $T$ 中，此时 $S=[2,3,4]$。
6. 由于与 4 相邻的点都是“已访问的”，将 4 弹出栈，此时 $S=[2,3]$。
7. 由于与 3 相邻的点都是“已访问的”，将 3 弹出栈，此时 $S=[2]$。
8. 由于 2 与 1 相连，且 1 是“未访问的”，将 1 放入栈 $S$ 中，并将 $(2,1)$ 加入树 $T$ 中，此时 $S=[2,1]$。
9. 由于与 1 相连的点都是“已访问的”，将 1 弹出栈，此时 $S=[2]$。
10. 由于与 2 相连的点都是“已访问的”，将 2 弹出栈，此时 $S$ 重新变为空。

**【样例 2】**

见选手目录下的 ***dfs/dfs2.in*** 与 ***dfs/dfs2.ans***。
这个样例满足测试点 $4\sim6$ 的约束条件。

**【样例 3】**

见选手目录下的 ***dfs/dfs3.in*** 与 ***dfs/dfs3.ans***。
这个样例满足测试点 $10\sim11$ 的约束条件。

**【样例 4】**

见选手目录下的 ***dfs/dfs4.in*** 与 ***dfs/dfs4.ans***。
这个样例满足测试点 $12\sim13$ 的约束条件。

**【样例 5】**

见选手目录下的 ***dfs/dfs5.in*** 与 ***dfs/dfs5.ans***。
这个样例满足测试点 $14\sim16$ 的约束条件。

**【样例 6】**

见选手目录下的 ***dfs/dfs6.in*** 与 ***dfs/dfs6.ans***。
这个样例满足测试点 $23\sim25$ 的约束条件。

## 【数据范围】

对于所有测试数据保证：$1 \le k \le n \le 5 \cdot 10^5$，$1 \le m \le 5 \cdot 10^5$。

| 测试点编号 | $n \le$ | $m \le$ | $k \le$ | 特殊性质 |
|---|---|---|---|---|
| $1 \sim 3$ | 6 | 6 | $n$ | 无 |
| $4 \sim 6$ | 15 | 15 | 6 | |
| $7 \sim 9$ | 300 | 300 | | |
| $10 \sim 11$ | | | $n$ | A |
| $12 \sim 13$ | | | | B |
| $14 \sim 16$ | | | | 无 |
| $17 \sim 18$ | $2 \cdot 10^5$ | $2 \cdot 10^5$ | | A |
| $19 \sim 21$ | | | | B |
| 22 | | | | 无 |
| $23 \sim 25$ | $5 \cdot 10^5$ | $5 \cdot 10^5$ | | |

特殊性质 A：保证在 $T$ 中，$i$ 号点与 $i+1$ 号点相连（$1 \le i < n$）。

特殊性质 B：保证若将 $T$ 的边与所有 $m$ 条非树边构成一个图 $G$，则 $T$ 是 $G$ 的一棵 1-dfs 树。

**请注意，1 号点不一定是 $k$ 个关键点之一。**