1361：产生数(Produce)

题目描述

给出一个整数n（n≤2000）和k个变换规则（k≤15）。规则：

① 1个数字可以变换成另1个数字；

② 规则中，右边的数字不能为零。

例如：n=234，k=2规则为

2 → 5

3 → 6

上面的整数234经过变换后可能产生出的整数为（包括原数）234，534，264，564共4种不同的产生数。

求经过任意次的变换（0次或多次），能产生出多少个不同的整数。仅要求输出不同整数个数。

输入格式

n
k
x_1 y_1
x_2 y_2

输出格式

格式为一个整数（满足条件的整数个数）。

输入样例

```
234
2
2 5
3 6
```

输出样例

```
4
```

解析

这是一道排列组合的计算问题：我们只要知道原数num的每一位数有多少种变换，然后每一位的可能的变换数相乘即可。

所以问题就拆分为一个一位数在k个变换规则的约束下有多少种变换，题例 2->4 ,3->5。也就是234这个数的第一位2可以变成4，这一位存在两种可能结果，同理第二位3也存在两种可能结果。所以结果就是4=2*2。

2->3,3->4,3->5这就要用到队列了,所以可以考虑BFS搜索来进行操作

代码

```
#include<bits/stdc++.h>

using namespace std;

const int N = 20; // 题目种的变换规则<=15
char num[5];  // 题目中的数<2000最多也就4位数，
int k;
int f[N], t[N]; // 存放变换规则 f原始数字 t目标数字
bool vis[N];// 用来当作标记，访问过的数不能再访问了
long long ans = 1;

long long bfs() {
    int lena = strlen(num);
    queue<int> q;
    for (int i = 0; i < lena; i++)// 对num的每一位进行bfs
    {
        int cur = num[i] - '0';
        q.push(cur);
        //每处理一位数字的时候都要进行重置处理
       memset(vis, true, sizeof(vis));
        int tmp = 1;
        while (!q.empty()) {
            //依次取出队列中的数字
          int top = q.front();
            //分别检查它的变化规律
          for (int j = 0; j < k; j++) {
                //当前数字存在变化规则
              //原始数字相同，目标数字没有变形过
              if (top == f[j] && vis[t[j]]) {
                    tmp++;
                    q.push(t[j]);
                    //标记原始数字已经变化过了，防止循环变化，如下所示：
                vis[f[j]] = false; // 2->5  5->2
                    //标记目标数字已经变化过，表面由多个数字变成1个相同数字引
                vis[t[j]] = false; // 2->5 3->5
               }
```

```
            }
            //检测完的从队列中移除
          q.pop();
        }
        //乘以每个数字的变化可能性就得到了最终的变化数
       ans *= tmp;
    }
    return ans;
}

int main() {
    cin >> num;
    cin >> k;
    for (int i = 0; i < k; i++) {
        cin >> f[i] >> t[i];
    }
    cout << bfs();
    return 0;
}
```


起的重复计数