**算法训练 排列问题**

时间限制：1.0s   内存限制：512.0MB

问题描述

　　求一个0～N-1的排列（即每个数只能出现一次），给出限制条件（一张N\*N的表，第i行第j列的1或0，表示为j-1这个数不能出现在i-1这个数后面，并保证第i行第i列为0），将这个排列看成一个自然数，求从小到大排序第K个排列。

数据规模和约定

　　N<=10，K<=500000

输入格式

　　第一行为N和K,接下来的N行，每行N个数，0表示不能，1表示能

输出格式

　　所求的排列

样例输入

3 2

0 1 1

1 0 0

0 1 0

样例输出

1 0 2

解释：

对于N=3的没有任何限制的情况

第一：0 1 2

第二：0 2 1

第三：1 0 2

第四：1 2 0

第五：2 0 1

第六：2 1 0

根据题目所给的限制条件由于2不能出现在1后面，0不能出现在2后面

第一：0 2 1

第二：1 0 2

第三：2 1 0

本题的C++参考代码如下：

#include "iostream"

#include "string"

#include "stdio.h"

#include "ctype.h"

#include "algorithm"

#include "stack"

using namespace std;

const int N =101;

int a[11];

bool map[11][11];

bool isOk(int n)

{

for(int i=0;i<n-1;i++)

if(map[ a[i] ][ a[i+1] ] == 0)

{

return false;

}

return true;

}

void init()

{

for(int i=0;i<10;i++)

a[i]=i;

}

int main()

{

int n;

int k;

cin>>n>>k;

for(int i=0;i<n;i++)

for(int j = 0; j < n; j++)

cin>>map[i][j];

init();

int num=0;

do

{

if(isOk(n))

num++;

if(num==k)

{

cout<<a[0];

for(int i=1;i<n;i++)

cout<<" "<<a[i];

cout<<endl;

break;

}

//cout<<endl;

} while (std::next\_permutation(a,a+n));

}

本题的C参考代码如下：

本题的Java参考代码如下：

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.StreamTokenizer;

import java.util.Collections;

import java.util.LinkedList;

public class Main {

private static int vis[][];

private static int count = 0, m;

private static LinkedList<String> queue = new LinkedList<String>();

public static void main(String[] args) throws IOException {

StreamTokenizer st = new StreamTokenizer(new BufferedReader(new InputStreamReader(System.in)));

st.nextToken();

int n = (int) st.nval;

st.nextToken();

m = (int) st.nval;

int arr[] = new int[n];

vis = new int[n][n];

for (int i = 0; i < n; i++) {

arr[i] = i;

for (int j = 0; j < n; j++) {

st.nextToken();

vis[i][j] = (int) st.nval;

}

}

allsort(arr, 0);

Collections.sort(queue);

System.out.println(queue.get(m - 1));

}

private static void allsort(int arr[], int k) {

if (arr.length == k) {

int ok = 0;

String str = "";

for (int i = 0; i < arr.length - 1; i++) {

if (vis[arr[i]][arr[i + 1]] == 0) {

ok = 1;

return;

}

str += arr[i] + " ";

}

if (ok == 0) {

queue.add(str + arr[k - 1]);

}

return;

}

int tmp;

for (int i = k; i < arr.length; i++) {

tmp = arr[i];

arr[i] = arr[k];

arr[k] = tmp;

allsort(arr, k + 1);

tmp = arr[i];

arr[i] = arr[k];

arr[k] = tmp;

}

}

}