**5-24一般解空间搜索问题**

1、问题描述：试设计一个用回溯法搜索一般解空间的函数。该函数的参数包括:生成解空间中下一扩展结点的函数、结点可行性判定函数和上界函数等必要的函数，并将此函数用于解图的 m 着色问题。

图的 m 着色问题描述如下:给定无向连通图 G 和 m 种不同的颜色。用这些颜色为图 G 的各顶点着色，每个顶点着一种颜色。如果有一种着色法使 G 中每条边的 2 个顶点着不同颜 色，则称这个图是 m 可着色的。图的 m 着色问题是对于给定图 G 和 m 种颜色，找出所有不同的着色法。

2、算法设计：对于给定的无向连通图 G 和 m 种不同的颜色，编程计算图的所有不同的着色法。

3、数据输入：第 1 行有 3 个正整数 n，k 和 m，表示给定的图 G 有 n个顶点和 k 条边，m 种颜色。顶点编号为 1，2，…，n。接下来的 k 行中，每行有 2 个正整 数 u,v，表示图 G 的一条边(u,v)。

4、程序代码：

package Chapter5HuiSuFa;

import java.util.Scanner;

public class YiBanJieKongJianSouSuo {

private static int n,k,m;

private static int[] x;

private static int sum;

private static boolean[][] edge;

public static void main(String[] args){

Scanner input = new Scanner(System.in);

while (true){

sum = 0;

n = input.nextInt();

k = input.nextInt();

m = input.nextInt();

edge = new boolean[n+1][n+1];

x = new int[n+1];

for(int i=1; i<=k; i++){

int x = input.nextInt();

int y = input.nextInt();

edge[x][y] = true;

edge[y][x] = true;

}

backtrack(1);

output();

}

}

private static void backtrack(int t){

if(t > n) record();

else

for(int i=f(n,t); i<=g(n,t); i++){

x[t] = h(i);

change(t);

if(constraint(t) && bound(t)) backtrack(t+1);

restore(t);

}

}

private static void record(){

sum++;

}

private static boolean constraint(int t){

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

if(edge[t][j] && x[j]==x[t]) return false;

return true;

}

private static boolean bound(int t){

return true;

}

private static void change(int t){

}

private static void restore(int t){

x[t] = 0;

}

private static int f(int n, int t){

return 1;

}

private static int g(int n, int t){

return m;

}

private static int h(int i){

return i;

}

private static void output(){

System.out.println(sum);

}

}