**实验一、友谊悖论验证**

输入：任意图的邻接矩阵（考察随机图和社会网络两种图，模拟生成）

输出：符合友谊悖论的节点占比

相关定义：友谊悖论指大多数人认为, 自己的朋友比自己拥有更多的朋友

算法分析：

把友谊悖论的概念抽象到社会网络无向图G（V，E）中，其中V为图中的节点集，表示社交网络中的人；E为图中的边集，表示社交网络中的人际关系。两个人i与j是朋友当且仅当i，j∈V且(i，j)∈E, 即在图G中节点i和节点j之间存在边。定义d(v)为节点的度，即与它相连的节点个数，对应着表示一个人的朋友个数。

若图G中存在一个节点v满足友谊悖论，那么他的朋友的朋友比他自身的朋友要多。即d(v)比与它相邻节点的平均度数要小。

用数学公式表示为：

其中，1E(u,v)指映射

根据以上思路可以找到网络中满足友谊悖论的节点，再与网络中的总节点数相除，即可得到图中满足友谊悖论的节点比例R。

随机网络理论类似于社会网络结构，我们可以通过构建和刻画真正随机的网络来模拟社会的随机性。在实验过程中，生成多个随机网络并计算网络中满足友谊悖论的节点比例R，通过实验结果证明友谊悖论。

代码实现如下：

import java.util.\*;  
public class Experiment1\_friendship\_paradox {  
 public static void main(String[] args) {  
  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.println("请输入图的节点个数：");  
 int n=sc.nextInt();  
 String t=sc.nextLine();  
 String[] strs=new String[n];  
 int[][] mtr=new int[n][n];  
 System.*out*.println("请输入图的邻接矩阵，数字之间以空格分隔");  
  
 for(int i=0;i<n;i++){  
 strs[i]=sc.nextLine();  
 String[] ss=strs[i].split(" ");  
 int j=0;  
 for (String s:ss){  
 mtr[i][j]=Integer.*parseInt*(s);  
 j++;  
 }  
 }  
  
 int[] fri=new int[n];  
  
 for (int i = 0; i < n; i++) {  
 ArrayList<Integer> friends=new ArrayList();  
 for (int j = 0; j < n; j++) {  
 if (mtr[i][j]==1){  
 fri[i]+=1;  
 friends.add(j);  
 }  
 }  
 }  
 int count=0;  
 for (int i = 0; i < n; i++) {  
 ArrayList<Integer> friends=new ArrayList();  
 for (int j = 0; j < n; j++) {  
 if (mtr[i][j]==1){  
 friends.add(j);  
 }  
 }  
 if(*is*(i,friends,fri,mtr)){  
 count++;  
 }  
 }  
  
 double x = (double)count/(double)n ;  
 System.*out*.println(String.*format*("%.2f", x\*100)+"%");  
 }  
  
 public static boolean is(int i, ArrayList<Integer> friends, int[] fri, int[][] mtr){  
  
 int number=0;  
  
 for (int k = 0; k < friends.size(); k++) {  
 number+=fri[friends.get(k)];  
 }  
 double avg=(double)number/(double)fri[i];  
  
 if (avg>(double)fri[i]){  
 return true;  
 }  
 return false;  
  
 }  
  
}

测试结果如下：

