package chat;

import java.io.\*;

import java.io.IOException;

import java.util.Arrays;

import java.util.Random;

import java.net.\*;

import java.nio.IntBuffer;

public class ClientNode{

static Socket clientSocket;

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

clientSocket=new Socket(InetAddress.getLocalHost(),8888);

InputStream ist=clientSocket.getInputStream();

OutputStream ost =clientSocket.getOutputStream();

//建立读数据缓冲区

BufferedReader br=new BufferedReader(new InputStreamReader(ist));

PrintWriter out=new PrintWriter(ost);//PrintWriter(out, autoFlush)

// PrintStream output = new PrintStream(clientSocket.getOutputStream());

// int a=1;

// out.write(a);//写入内存缓冲区

// int b=2;

//out.write(b);

// int c=41;

// output.println(c);

// output.flush();

// out.write(c);

//随机抽样检测误码

// Estimate Estimate=new Estimate(aliceKeys,bobKeys);

// System.out.println("随机测试误码在设定误码率范围内");//A

/\* String md5HashcodeAlice = Utility.getFileMD5(new File("AliceKey.txt")); //A

out.println(md5HashcodeAlice+"\r");

System.out.println(md5HashcodeAlice);

/\* while!md5HashcodeAlice.equals(md5HashcodeBob)){

Cascadei cascadei = new Cascadei(aliceKeys,bobKeys,n,p);//A

}

Enhance enhance = new Enhance(n2,leakedMsg2);//获取数据,把n2和泄露量送到保密增强 //A

\*/

int[][] aliceBlocks1= {{1,0,1},{1,1,1},{0,0,1},{0,0,0}};

for (int i=0; i<aliceBlocks1.length; i++){

int e = parityCheck1(aliceBlocks1[i]);

//建一个缓冲区

// StringBuffer sb= new StringBuffer();

// sb.append(e);

// String d=String.valueOf(e);

out. write(e);//发送奇偶值

// leakedMsg+=1;

}

int[][] BobBlocks1= {{1,1,1},{1,1,1},{0,0,1}};

for (int i=0; i<BobBlocks1.length; i++){

int e = parityCheck1(BobBlocks1[i]);

//建一个缓冲区

// StringBuffer sb= new StringBuffer();

// sb.append(e);

// String d=String.valueOf(e);

out. write(e);//发送奇偶值

// leakedMsg+=1;

}

BufferedReader wt=new BufferedReader(new InputStreamReader(System.in));

while(true){

String content=wt.readLine();

out.flush();

if(content.equals("byebye")){

break;

}

System.out.println(br.readLine()); //read data

}

clientSocket.close();

}

private static int parityCheck1(int[] a) {

// TODO 自动生成的方法存根

int parity1 = 0;

for (int i=0; i<a.length; i++){//一维数组长，blocks的列数长

parity1 ^= a[i];

}

return parity1;

}

}