**北京邮电大学软件学院**

**2017－2018学年第一学期实验报告**

**课程名称： Java SE程序设计**

**项目名称： Java编程（提高练习）**

**项目完成人：**

**姓名：\_\_\_王子妍\_\_\_\_\_**

**学号：\_\_\_\_2016211980\_\_\_\_**

**指导教师：\_\_\_程保中，孙艺\_\_\_\_\_\_\_\_\_\_\_**

**日 期： 2018 年 1 月 3 日**

1. **实验目的**

希望在这次java实验中提高使用java的熟练度，在训练中更好的使用GUI以及多线程和网络编程。

1. **实验内容**

五个实验分别为：

Project #5 Design Classes with Inheritance

Project #6 Design Classes with Abstract Classes and Interfaces

Project #7 GUI, Graphics, and Event-Driven Programming

Project #8 Multi-Threading programming

Project #9 Socket programming

1. **实验环境**

eclipse+win10

1. **实验结果**

五个实验均按照要求认真完成。所有的实验截图以及重要代码以及解题思路都在附录中。

1. **附录**

**Ps:调试心得或者在构思时候的UML类图写在在每个实验里面，感觉这次实验的前两个比较简单，从第三个实验GUI开始比较难，所以调试心得在后三个实验写的会多些，前两个实验画了一下UML图，展示了一下大概的框架。**

**Project #5 Design Classes with Inheritance**

Draw the UML diagram that involving the classes Triangle and GeometricObject. Implement the class. Write a test program that creates a Triangle object with sides 1, 1.5, 1, setting color yellow and filled true, and displaying the area, perimeter, color, and whether filled or not.

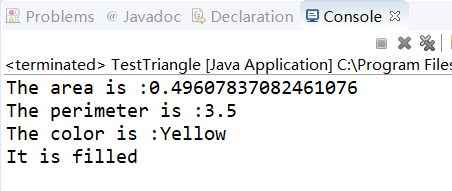
这个实验主要是复习了继承方面的知识，下面是我绘制的UML图：

|  |
| --- |
| **GeometricObject** |
| -color:String  -filled:Boolean  -data:java.utill.Date |
| + GeomtricObject()  + GeomtricObject(String color,boolean filled)  + getColor():String  + setColor(String color):void  + isFilled():boolean  + setFilled(boolean filled): void  + getDateCreat():java.util.Date  + toString():String |

z

|  |
| --- |
| **Triangle** |
| -side1:double  -side2:double  -side3:double |
| +Triangle()  +Triangle(double s1,double s2,double s3)  +Triangle(double s1,double s2,double s3,String color,boolean filled)  +getSide1():double  +getSide2():double  +getSide3():double  +setSide1(double s):void  +setSide2(double s):void  +setSide3(double s):void  +getParimeter():double  +getArea():double |

**运行效果截图为：**



**代码部分：**

**package** project5;

**public** **class** GeomtricObject {

**private** String color="White";

**private** **boolean** filled;

**private** java.util.Date date;

**public** GeomtricObject()

{

date=**new** java.util.Date();

}

**public** GeomtricObject(String color,**boolean** filled)

{

date=**new** java.util.Date();

**this**.color=color;

**this**.filled=filled;

}

**public** String getColor()

{

**return** color;

}

**public** **void** setColor(String color)

{

**this**.color=color;

}

**public** **boolean** isFilled()

{

**return** filled;

}

**public** **void** setFilled(**boolean** filled)

{

**this**.filled=filled;

}

**public** java.util.Date getDateCreat()

{

**return** date;

}

**public** String toString()

{

**return** "created on "+date+"\ncolor:"+color+" and filled: "+filled;

}

}

**package** project5;

**public** **class** Triangle **extends** GeomtricObject{

**private** **double** side1=1.0;

**private** **double** side2=1.0;

**private** **double** side3=1.0;

**public** Triangle()

{

}

**public** Triangle(**double** s1,**double** s2,**double** s3)

{

side1=s1;

side2=s2;

side3=s3;

}

**public** Triangle(**double** s1,**double** s2,**double** s3,String color,**boolean** filled)

{

side1=s1;

side2=s2;

side3=s3;

setColor(color);

setFilled(filled);

}

**public** **double** getSide1()

{

**return** side1;

}

**public** **double** getSide2()

{

**return** side2;

}

**public** **double** getSide3()

{

**return** side3;

}

**public** **void** setSide1(**double** s)

{

side1=s;

}

**public** **void** setSide2(**double** s)

{

side2=s;

}

**public** **void** setSide3(**double** s)

{

side3=s;

}

**public** **double** getParimeter()

{

**return** side1+side2+side3;

}

**public** **double** getArea()

{

**double** p=(side1+side2+side3)/2.0;

**double** ans=Math.*sqrt*(p\*(p-side1)\*(p-side2)\*(p-side3));

**return** ans;

}

}

**package** project5;

**public** **class** TestTriangle {

**public** **static** **void** main(String[] args)

{

Triangle t1=**new** Triangle(1.0,1.5,1.0,"Yellow",**true**);

System.***out***.println("The area is :"+t1.getArea());

System.***out***.println("The perimeter is :"+t1.getParimeter());

System.***out***.println("The color is :"+t1.getColor());

**if**(t1.isFilled()==**true**)

{

System.***out***.println("It is filled");

}

**else**

System.***out***.println("It is not filled");

}

}

**Project #6 Design Classes with Abstract Classes and Interfaces**

This project consists of two separate problems. For each, print the source code and the screen shot of a sample run.

1. Design an interface named Colorable with a void method named howToColor(). Every class of a colorable object must implement the Colorable interface. Design a class named Square that extends GeometricObject and implements Colorable. Implement howToColor to display a message on how to color the square.

Draw a UML diagram that involves Colorable, Square, and GeometricObject.

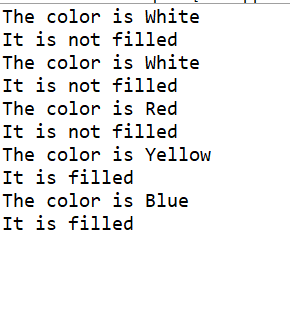
Write a test program that creates an array of five GeometricObject. For each object in the array, invoke its howToColor method if it is colorable.

这个实验主要是复习了接口，下面是我绘制的UML类图

|  |
| --- |
| **GeometricObject** |
| -color:String  -filled:Boolean  -data:java.utill.Date |
| + GeomtricObject()  + GeomtricObject(String color,boolean filled)  + getColor():String  + setColor(String color):void  + isFilled():boolean  + setFilled(boolean filled): void  + getDateCreat():java.util.Date  + toString():String |

|  |
| --- |
| **Square** |
| -side:double |
| +Square()  +Square(double)  +Square(double,String ,boolean)  +setSide(double):void  +getSide():double  +getArea ():double  +getPerimeter():double  +howToColor():void |

测试的时候是建立了一个Square数组然后一一初始化的，用了不同的构造函数初始化，效果截图为：



代码部分为：

**package** project6;

**public** **class** GeometricObject {

**private** String color="White";

**private** **boolean** filled;

**private** java.util.Date date;

**public** GeometricObject()

{

date=**new** java.util.Date();

}

**public** GeometricObject(String color,**boolean** filled)

{

date=**new** java.util.Date();

**this**.color=color;

**this**.filled=filled;

}

**public** String getColor()

{

**return** color;

}

**public** **void** setColor(String color)

{

**this**.color=color;

}

**public** **boolean** isFilled()

{

**return** filled;

}

**public** **void** setFilled(**boolean** filled)

{

**this**.filled=filled;

}

**public** java.util.Date getDateCreat()

{

**return** date;

}

**public** String toString()

{

**return** "created on "+date+"\ncolor:"+color+" and filled: "+filled;

}

}

**package** project6;

**interface** Colorable

{

**public** **void** howToColor();

}

**public** **class** Square **extends** GeometricObject **implements** Colorable

{

**private** **double** side;

**public** Square()

{

side=0;

}

**public** Square(**double** side)

{

**this**.side=side;

}

**public** Square(**double** side,String color,**boolean** filled)

{

**this**.side=side;

setColor(color);

setFilled(filled);

}

**public** **void** setSide(**double** side)

{

**this**.side=side;

}

**public** **double** getSide()

{

**return** side;

}

**public** **double** getArea()

{

**return** side\*side;

}

**public** **double** getPerimeter()

{

**return** side\*4;

}

**public** **void** howToColor()

{

System.***out***.println("The color is "+getColor());

**if**(isFilled()==**true**)

System.***out***.println("It is filled");

**else**

System.***out***.println("It is not filled");

}

}

**package** project6;

**public** **class** TestSquare {

**public** **static** **void** main(String[] args)

{

Square g[]=**new** Square[5];

g[0]=**new** Square();

g[1]=**new** Square(3.0);

g[2]=**new** Square(2.0,"Red",**false**);

g[3]=**new** Square(1.5,"Yellow",**true**);

g[4]=**new** Square(3.5,"Blue",**true**);

g[0].howToColor();

g[1].howToColor();

g[2].howToColor();

g[3].howToColor();

g[4].howToColor();

}

}

2. Develop a class named Octagon (八边形)that extends GeometricObject and implements the Comparable and Cloneable interfaces. Assume that all eight sides of the octagon are of equal size. The area can be computed using the following formula:



Draw the UML diagram that involves Octagon, GeometricObject, Comparable, and Cloneable.

Write a test program that creates an Octagon object with side value 5 and displays its area and perimeter. Create a new object using the clone method and compare the two objects using the compareTo method.

UML类图如下：(Word我写的排版有点问题，画箭头位置总是很奇怪就没有用箭头，大致想表达，Octagon实现两个接口且继承GeometricObject类)

|  |
| --- |
| **GeometricObject** |
| -color:String  -filled:Boolean  -data:java.utill.Date |
| + GeomtricObject()  + GeomtricObject(String color,boolean filled)  + getColor():String  + setColor(String color):void  + isFilled():boolean  + setFilled(boolean filled): void  + getDateCreat():java.util.Date  + toString():String |

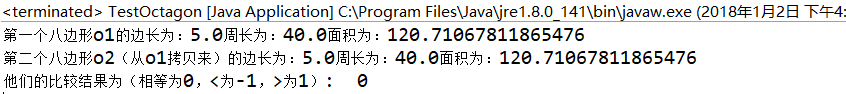
|  |
| --- |
| Comparable<E> |
| +compareTo(E) |

|  |
| --- |
| Cloneable |
|  |

|  |
| --- |
| **Octagon** |
| -side:double |
| +Octagon ()  +Octagon(double)  +setSide(double):void  +getSide():double  +getArea ():double  +getPerimeter():double  +compareTo(Octagon):int  +clone():Octagon |

O1为我直接创建的一个八边形，o2是通过clone o1的一个八边形，里面的内容与o1完全相同的一个新的多边形，在clone方法里new出来并且进行的复制。

结果截图为：



代码为（GeometricObject与之前相同就不列在下面啦）：

**package** project6;

**interface** Comparable<E>

{

**public** **int** compareTo(E o);

}

**interface** Cloneable

{

};

**public** **class** Octagon **extends** GeometricObject **implements** Comparable<Octagon>,Cloneable{

**private** **double** side;

**private** **double** area;

**public** Octagon()

{

side=1;

area=(2+(4/Math.*sqrt*(2)))\*side\*side;

}

**public** Octagon(**double** s)

{

side=s;

area=(2+(4/Math.*sqrt*(2)))\*side\*side;

}

**public** **double** getSide()

{

**return** side;

}

**public** **void** setSide(**int** s)

{

side=s;

}

**public** **double** getPerimeter()

{

**return** side\*8;

}

**public** **double** getArea()

{

**return** area;

}

**public** **int** compareTo(Octagon o)

{

**if**(side<o.side) **return** -1;

**else** **if**(side==o.side) **return** 0;

**else**

**return** 1;

}

**public** Octagon clone()

{

**try** {

Octagon oo=**new** Octagon();

oo.area=area;

oo.side=side;

**return** oo;

}

**catch**(Exception ex)

{

**return** **null**;

}

}

}

**package** project6;

**public** **class** TestOctagon {

**public** **static** **void** main(String[] args) {

Octagon o1=**new** Octagon(5);

Octagon o2=o1.clone();

System.***out***.println("第一个八边形o1的边长为："+o1.getSide()+"周长为："+o1.getPerimeter()+"面积为："+o1.getArea());

System.***out***.println("第二个八边形o2（从o1拷贝来）的边长为："+o2.getSide()+"周长为："+o2.getPerimeter()+"面积为："+o2.getArea());

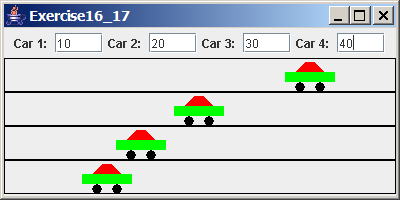
System.***out***.println("他们的比较结果为（相等为0，<为-1，>为1）: "+o1.compareTo(o2));

}

}

**Project #7 GUI, Graphics, and Event-Driven Programming**

Develop a GUI application that simulates four cars racing, as shown in the following figure. You can set the speed for each car with 1 the highest.



Design:

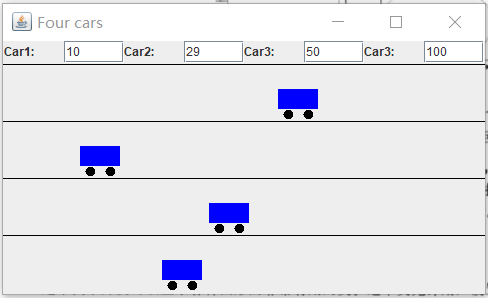
1. Draw a sketch of the user interface that shows the components, containers, and the layout managers.
2. Design a class named Car for illustrating one racing car with appropriate data fields, constructors, and methods. Car should extend JPanel.
3. Draw a UML diagram that involves the main frame class, its superclass, and the Car class.

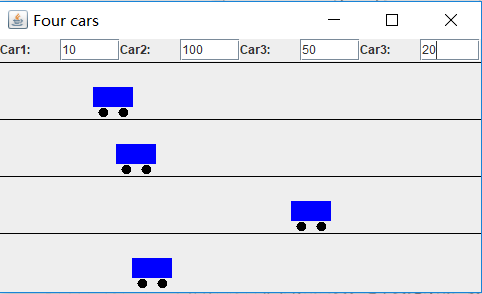
Implementation:

1. Implement the Car class.
2. Implement the main application class.

**这个实验主要是GUI，是期中考试前学的，有些忘记了，借着做实验的机会又重新看了一下GUI然后再写的，这个实验我感觉比较重要的就是要会用Timer类，会用ActionEvent进行事件的监听，以及比较基础的panel还有frame来用Layout布局。虽然做的时候花了一些时间，但是看到小车跑起来的效果还是很开心的**。

下面是两张效果截图：





下面是代码部分：

**package** project7;

**import** java.awt.\*;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** javax.swing.\*;

**public** **class** FourCars **extends** JFrame{

Timer t1=**new** Timer(1000,**new** TimerListener(1));

Timer t2=**new** Timer(1000,**new** TimerListener(2));

Timer t3=**new** Timer(1000,**new** TimerListener(3));

Timer t4=**new** Timer(1000,**new** TimerListener(4));

Car car1=**new** Car();

Car car2=**new** Car();

Car car3=**new** Car();

Car car4=**new** Car();

**public** FourCars()

{

//第一部分，先画出上面的文字输入部分，用p1完成

JPanel p1=**new** JPanel(**new** GridLayout(1,0));

p1.add(**new** JLabel("Car1:"));

JTextField jtf1=**new** JTextField();

p1.add(jtf1);

p1.add(**new** JLabel("Car2:"));

JTextField jtf2=**new** JTextField();

p1.add(jtf2);

p1.add(**new** JLabel("Car3:"));

JTextField jtf3=**new** JTextField();

p1.add(jtf3);

p1.add(**new** JLabel("Car3:"));

JTextField jtf4=**new** JTextField();

p1.add(jtf4);

jtf1.addActionListener(**new** CarTimeListener(1));

jtf2.addActionListener(**new** CarTimeListener(2));

jtf3.addActionListener(**new** CarTimeListener(3));

jtf4.addActionListener(**new** CarTimeListener(4));

add(p1,BorderLayout.***NORTH***);

//第二部分，准备画小车

JPanel p2=**new** JPanel(**new** GridLayout(4,1));

p2.add(car1);

p2.add(car2);

p2.add(car3);

p2.add(car4);

add(p2,BorderLayout.***CENTER***);

}

**class** CarTimeListener **implements** ActionListener

{

**int** whichCar=0;

CarTimeListener(**int** a)

{

whichCar=a;

}

**public** **void** actionPerformed(ActionEvent e)

{

**int** v=Integer.*parseInt*(e.getActionCommand());

**if**(whichCar==1)

{

t1.setDelay(1000/v);

t1.start();

}

**else** **if**(whichCar==2)

{

t2.setDelay(1000/v);

t2.start();

}

**else** **if**(whichCar==3)

{

t3.setDelay(1000/v);

t3.start();

}

**else** **if**(whichCar==4)

{

t4.setDelay(1000/v);

t4.start();

}

}

}

**class** TimerListener **implements** ActionListener

{

**int** whichCar=0;

TimerListener(**int** a)

{

whichCar=a;

}

**public** **void** actionPerformed(ActionEvent e)

{

**if**(whichCar==1)

{

car1.repaint();

}

**else** **if**(whichCar==2)

{

car2.repaint();

}

**else** **if**(whichCar==3)

{

car3.repaint();

}

**else** **if**(whichCar==4)

{

car4.repaint();

}

}

}

**public** **static** **void** main(String[] args)

{

FourCars frame=**new** FourCars();

frame.setSize(500,300);

frame.setTitle("Four cars");

frame.setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);

frame.setLocationRelativeTo(**null**);

frame.setVisible(**true**);

}

**class** Car **extends** JPanel{

**int** dis=0;

**protected** **void** paintComponent(Graphics g)

{

**super**.paintComponent(g);

g.setColor(Color.***BLACK***);

g.drawLine(0, 0,getWidth(), 0);

g.drawLine(0, getHeight(),getWidth(), getHeight());

**if**(dis>500)

dis=0;

**else**

dis=dis+1;

g.setColor(Color.***BLUE***);

g.fillRect(dis,25, 40, 20);

g.setColor(Color.***BLACK***);

g.fillOval(dis+5, 45, 10, 10);

g.fillOval(dis+25, 45, 10, 10);

}

}

}

**Project #8 Multi-Threading programming**

evelop a multi-threaded program.

1. There are three threads in the program.
2. Thread A and Thread B both sleep a random time (no more than 2 seconds) and then each produces a random integer between 0-10;
3. Thread C then compares the two numbers produced by Thread A, and B. The thread producing bigger integer will get 2 points and the thread producing smaller integer will get 0 point. If two integers are equal, then each thread get 1 point.
4. Thread C prints the result in the following format. You can use either console or GUI as the program’s UI.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Round | **Thread A** | | | **Thread B** | | |
| Sleep time | Random integer | Points obtained | Sleep time | Random integer | Points obtained |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| … |  |  |  |  |  |  |
| N |  |  |  |  |  |  |

Final result: A (or B or None) is the winner

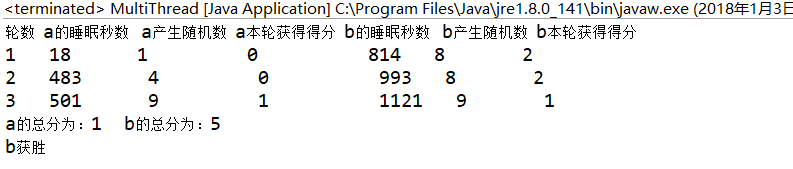
(Hint: You do NOT need to print the table frame)

1. Repeat step 2 - step 4 at least 3 times.
2. Finally, the server compares the total points obtained by thread A and thread B respectively to declare who is the winner: A (or B or None) is the winner.
3. At least one of the three threads must inherits from Thread class; And at least one of the three threads must implement Runnable interface.
4. Maybe you need to use one or more methods such as: sleep(); wait(); notify(); notifyall(); join(); etc.

**感觉主要就是写三个线程，然后我的第一个是implements Runnable实现的，剩下两个是extends Thread实现的，两个的区别也不是很大，就是注意一下在start和sleep的时候注意就可以，随机数字是(int)(Math.*random*()\*2000)**

**这个之前学过的方法。**

**效果截图为：**



**代码部分为：**

**package** project8;

**public** **class** MultiThread {

**static** **int** []*numbera*=**new** **int**[5];

**static** **int** []*numberb*=**new** **int**[5];

**public** **static** **void** main(String[] args)

{

System.***out***.println("轮数 a的睡眠秒数 a产生随机数 a本轮获得得分 b的睡眠秒数 b产生随机数 b本轮获得得分");

//TaskThread1 ta=new TaskThread1();

//TaskThread2 tb=new TaskThread2();

**for**(**int** i=1;i<=3;i++)

{

TaskThread3 tc=**new** TaskThread3(i);

tc.start();

**try** {

tc.*sleep*(3000);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**int** ansa=0,ansb=0;

**for**(**int** i=1;i<=3;i++)

{

ansa=ansa+*numbera*[i];

ansb=ansb+*numberb*[i];

}

System.***out***.println("a的总分为："+ansa+" b的总分为："+ansb+" ");

**if**(ansa>ansb)

System.***out***.println("a获胜");

**else** **if**(ansa<ansb)

System.***out***.println("b获胜");

**else** **if**(ansa==ansb)

System.***out***.println("a与b平局");

//tb.start();

//ta.start();

}

//public static class TaskThread1 extends Thread{

**public** **static** **class** TaskThread1 **implements** Runnable{

**private** **int** randomnum;

**private** **int** sleeptime;

TaskThread1()

{

sleeptime=(**int**)(Math.*random*()\*2000);

}

**public** **int** getNum()

{

**return** randomnum;

}

**public** **int** getSleepTime()

{

**return** sleeptime;

}

**public** **void** run()

{

**try**

{

Thread.*sleep*(sleeptime);

}

**catch**(Exception ex)

{}

randomnum=(**int**)(Math.*random*()\*11);

//System.out.print(randomnum+" ");

}

}

**public** **static** **class** TaskThread2 **extends** Thread{

**private** **int** randomnum;

**private** **int** sleeptime;

TaskThread2()

{

sleeptime=(**int**)(Math.*random*()\*2000);

}

**public** **int** getNum()

{

**return** randomnum;

}

**public** **int** getSleepTime()

{

**return** sleeptime;

}

**public** **void** run()

{

**try**

{

TaskThread2.*sleep*(sleeptime);

}

**catch**(Exception ex)

{}

randomnum=(**int**)(Math.*random*()\*11);

//System.out.print(randomnum+" ");

}

}

**public** **static** **class** TaskThread3 **extends** Thread{

**private** **int** t;

TaskThread3(**int** times)

{

t=times;

}

**public** **void** run()

{

**int** n1,n2;

**int** s1,s2;

TaskThread1 ta=**new** TaskThread1();

TaskThread2 tb=**new** TaskThread2();

tb.start();

Thread aa=**new** Thread(ta);

aa.start();

**try** {

TaskThread3.*sleep*(2000);

}

**catch** (InterruptedException e) {

e.printStackTrace();

}

n1=ta.getNum();

n2=tb.getNum();

s1=ta.getSleepTime();

s2=tb.getSleepTime();

**if**(n1>n2)

{

*numbera*[t]+=2;

}

**else** **if**(n1<n2)

{

*numberb*[t]+=2;

}

**else**

{

*numbera*[t]+=1;

*numberb*[t]+=1;

}

System.***out***.println(t+" "+s1+" "+n1+" "+*numbera*[t]+" "+s2+" "+n2+" "+*numberb*[t]+" ");

}

}

}

Project #9 Socket programming

Develop two socket programs: one server, one client.

1. The client has two threads (You can deem each thread as a kid):
   1. Thread A (i.e. Kid A) and Thread B (i.e. Kid B) both sleep a random time (no more than 2 seconds) and then each produces a random integer between 0-10;
   2. Thread A use TCP to send data to server
   3. Thread B use UDP to send data to server
2. The server receives data from the client (the server acts as both TCP server and UDP server)
3. Once the server receives two integers (one from TCP socket and another from UDP socket), the server compares the two numbers (denoted by "Kid A" and "Kid B"). The Kid with a bigger integer will get 2 points and the Kid with a smaller integer will get 0 point. If two integers are equal, then each Kid get 1 point.
4. The server prints the result in the following format. You can use either console or GUI as the program’s UI.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Round | **Thread A** | | | **Thread B** | | |
| Sleep time | Random integer | Points obtained | Sleep time | Random integer | Points obtained |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| … |  |  |  |  |  |  |
| N |  |  |  |  |  |  |

Final result: A (or B or None) is the winner

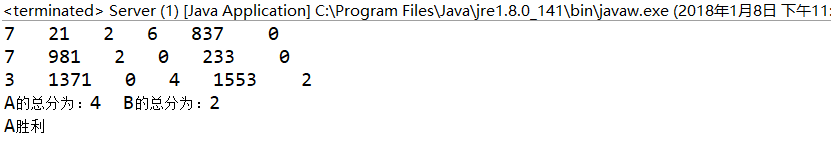
(Hint: You do NOT need to print the table frame)

1. Repeat step 1 - step 4 at least 3 times.
2. Finally, the server compares the total points obtained by thread A and thread B respectively to declare who is the winner: A (or B or None) is the winner.

(Hints: You can use multiple threads in the server program if needed)

**这个是基于上一个实验完成的，感觉还是难度比较大的，因为网络编程掌握的不是很好，所以在UDP编程的时候感觉还是困难比较大，但是好在后面又看了老师前面给的示例代码然后对自己的程序做了一些改进，感觉用多线程还有网络的时候eclipse不太好单步执行，所以纠错会比较的困难，一直用print语句来debug。总的来说就是多线程和网络方面还是掌握的不好，要继续努力。**

**实验效果截图为：**



**代码部分为：**

**package project9;**

**import java.io.\*;**

**import java.net.DatagramPacket;**

**import java.net.DatagramSocket;**

**import java.net.InetAddress;**

**import java.net.Socket;**

**import java.util.Arrays;**

**public class Client {**

**private DataOutputStream outputToServer;**

**public static void main(String[] args)**

**{**

**new Client();**

**}**

**public static class TaskThreadA extends Thread{**

**private int randomnum;**

**private int sleeptime;**

**private Socket s;**

**TaskThreadA(Socket socket)**

**{**

**sleeptime=(int)(Math.random()\*2000);**

**s=socket;**

**randomnum=(int)(Math.random()\*11);**

**}**

**public int getNum()**

**{**

**return randomnum;**

**}**

**public int getSleepTime()**

**{**

**return sleeptime;**

**}**

**public void run()**

**{**

**try**

**{**

**TaskThreadA.sleep(sleeptime);**

**}**

**catch(Exception ex)**

**{}**

**}**

**}**

**public static class TaskThreadB extends Thread{**

**private DatagramSocket socket;**

**private byte[] buf=new byte[256];**

**private DatagramPacket sendPacket;**

**private int randomnum;**

**private int sleeptime;**

**TaskThreadB(DatagramSocket socket)**

**{**

**Arrays.fill(buf, (byte) 0);**

**buf[0]=(byte) sleeptime;**

**sleeptime=(int)(Math.random()\*2000);**

**this.socket=socket;**

**randomnum=(int)(Math.random()\*11);**

**buf[1]=(byte)randomnum;**

**}**

**public int getNum()**

**{**

**return randomnum;**

**}**

**public int getSleepTime()**

**{**

**return sleeptime;**

**}**

**public void run()**

**{**

**try**

**{**

**TaskThreadB.sleep(sleeptime);**

**}**

**catch(Exception ex)**

**{**

**System.out.println("Error in Thread b");**

**}**

**}**

**}**

**public Client()**

**{**

**try {**

**Socket socketA=new Socket("localhost",8000);**

**TaskThreadA threadA=new TaskThreadA(socketA);**

**threadA.start();**

**outputToServer=new DataOutputStream(socketA.getOutputStream());**

**outputToServer.writeInt(threadA.randomnum);**

**outputToServer.writeInt(threadA.sleeptime);**

**DatagramSocket socketB=new DatagramSocket();**

**TaskThreadB threadB=new TaskThreadB(socketB);**

**threadB.start();**

**InetAddress address=InetAddress.getByName("localhost");**

**DatagramPacket sendPacket1 = new DatagramPacket(threadB.buf, threadB.buf.length, address, 8000);**

**sendPacket1.setData(new Integer(threadB.randomnum).toString().getBytes());**

**socketB.send(sendPacket1);**

**DatagramPacket sendPacket2 = new DatagramPacket(threadB.buf, threadB.buf.length, address, 8000);**

**sendPacket2.setData(new Integer(threadB.sleeptime).toString().getBytes());**

**socketB.send(sendPacket2);**

**}**

**catch(Exception ex)**

**{**

**}**

**}**

**}**

**package project9;**

**import java.io.DataInputStream;**

**import java.net.\*;**

**import java.util.Arrays;**

**public class Server {**

**public static void main(String[] args)**

**{**

**new Server();**

**}**

**public Server()**

**{**

**int ansa=0,ansb=0;**

**try {DatagramSocket socketB = new DatagramSocket(8000);**

**ServerSocket serverSocket=new ServerSocket(8000);**

**for(int i=1;i<=3;i++)**

**{**

**byte[] buf = new byte[256];**

**Socket socket=serverSocket.accept();**

**DataInputStream inputFromClient=new DataInputStream(**

**socket.getInputStream());**

**int num,sleept;**

**num=inputFromClient.readInt();**

**sleept=inputFromClient.readInt();**

**System.out.print(num+" "+sleept+" ");**

**DatagramPacket receivePacket = new DatagramPacket(buf, buf.length);**

**Arrays.fill(buf, (byte) 0);**

**socketB.receive(receivePacket);**

**int numb,sleepb;**

**numb=Integer.parseInt(new String(buf).trim());**

**DatagramPacket receivePacket2 = new DatagramPacket(buf, buf.length);**

**Arrays.fill(buf, (byte) 0);**

**socketB.receive(receivePacket2);**

**sleepb=Integer.parseInt(new String(buf).trim());**

**if(num>numb)**

**{**

**System.out.print("2 ");**

**ansa=ansa+2;**

**}**

**else if(num==numb)**

**{**

**System.out.print("1 ");**

**ansa=ansa+1;**

**ansb=ansb+1;**

**}**

**else**

**{**

**System.out.print("0 ");**

**ansb=ansb+2;**

**}**

**System.out.print(numb+" "+sleepb+" ");**

**if(num>numb)**

**System.out.println("0 ");**

**else if(num==numb)**

**System.out.println("1 ");**

**else**

**System.out.println("2 ");**

**}**

**System.out.println("A的总分为："+ansa+" B的总分为："+ansb);**

**if(ansa>ansb) System.out.println("A胜利");**

**else if(ansa==ansb) System.out.println("B胜利");**

**else System.out.println("A与B平局啦");**

**}**

**catch(Exception ex)**

**{**

**System.out.println("Error in Server");**

**}**

**}**

**}**