北京邮电大学软件学院

实验报告

课程名称： Java SE 程 序 设 计

项目名称： Java程序编制（基础练习）

项目完成人：

姓名： 何彩

学号： 2017211921

指导教师： 崔毅东

日 期： 2018 年 12 月 26 日

# 实验概述

## 实验目的

掌握类的继承方法；实现类接口；能够使用多线程编程，实现线程间同步、通讯；能够实现socket网络编程，使用UDP、TCP两种方式进行数据的传递

## 实验内容

|  |  |
| --- | --- |
|  | 任务说明 |
|  | 实现类的继承，完成project6 |
|  | 实现抽象类的继承和接口，完成project7 |
|  | 实现多线程编程 |
|  | 实现线程间同步，完成project8 |
|  | 通过TCP实现网络编程 |
|  | 通过UDP实现网络编程，完成project9 |
|  | 对代码进行整理、规范代码 |

## 实验环境

### 硬件环境

普通计算机

### 网络环境

无需网络

### 软件环境

windows10、visual studio2017

## 实验结果

project5:用户输入三角形的三边，能够显示出面积、周长、三边的信息

project6:显示五个GeometricObject对象的howToColor theSquare的信息;显示一个边长为5的八边形的面积和周长信息，用clone复制八边形，并和之前的比较

project7:输入局数，能显示每一轮的A、B产生的随机字符、sleep时间、得分情况，和最终的总得分和输赢

project8:输入局数，能显示ABkid每一轮的得分、每次出的什么和总得分及输赢情况

# 实验内容

# Project #1分析设计

## 1.1问题分析

### 1.1.1 目标

GeometricObject类包含一个图形类的基本信息：颜色、是否填充、创建时间

Triangle 类要继承GeometricObject抽象类，同时包含三角形的周长、面积信息

输入三角形三条边的信息能够得到周长、面积等信息

### 1.1.2功能

#### 1.1.2.1功能一：获得面积、周长、边长信息

如果创造的是无参对象、则输出默认值的信息；

否则由用户输入参数，能单独调用相对应的函数获得信息

#### 1.1.2.2功能二：显示颜色、填充与否、创建时间

在toString()函数里面能显示这些信息

### 1.1.3性能

#### 1.1.3.1功能一、二的性能

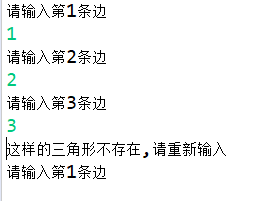
在用户输入文字的同时，程序在每次输入后的0.3秒之内响应并将结果显示到屏幕上

#### 1.1.3.2健壮性

如果输入的三角形不符合三角形的要求，则不会成功

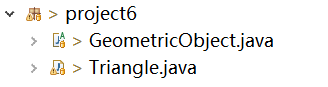
#### 1.1.3.3异常输入处理

当输入的三角形不符合要求的时候，会提示用户重新输入



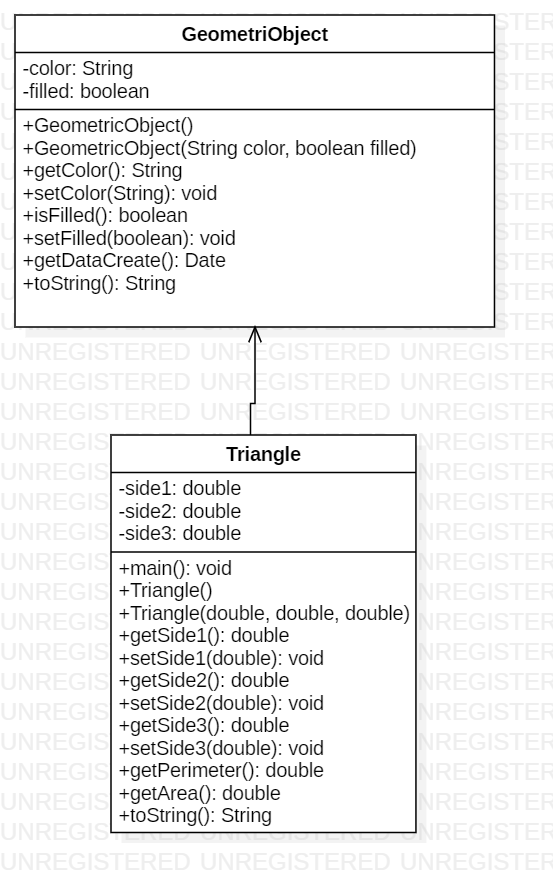
## 1.2设计方案

### 1.2.1模块划分



本项目只包含一个模块 ，两个类

### 1.2.2类图



### 1.2.3算法

#### 1.2.1.1算法一：getArea()

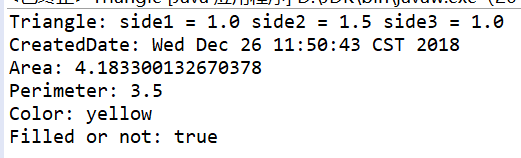
area= 

#### 1.2.1.2算法二：getPerimeter()

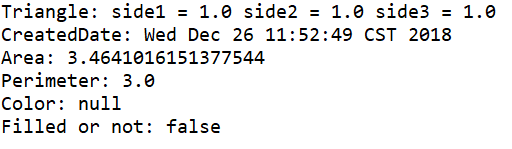
Perimeter=side1+side2+side3

## 1.3实验结果

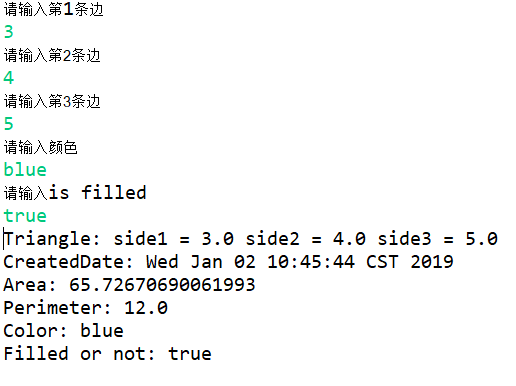
1. 边长为（1.0，1.5，1.0）的情况



1. 默认三角形情况下的输出



1. 用户输入信息后的输出



## 1.4 调试心得

无

## 1.5源程序

GeometricObject.java

package project6;

import java.util.Date;

public abstract class GeometricObject {

private String color;

private boolean filled;

private Date dateCreated=new Date();

public GeometricObject() {

}

public GeometricObject(String color, boolean filled) {

this.setColor(color);

this.setFilled(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 Date getDateCreate() {

return dateCreated;

}

public String toString() {

return( "Color: "+color+"isFilled: "+filled+"CreatedDate: "+dateCreated);

}

}

Triangle.java:

package project6;

import java.util.Scanner;

public class Triangle extends GeometricObject {

private double side1 = 1.0, side2 = 1.0, side3 = 1.0;

public static void main(String[] args) {

//(1,1.5,1)的实例

var t = new Triangle(1, 1.5, 1);

t.setColor("yellow");

t.setFilled(true);

System.out.println(t.toString());

//无参构造的实例

var t2=new Triangle();

System.out.println(t2.toString());

//用户输入

Scanner in=new Scanner(System.in);

double[] side=new double[3];

while(true) {

for(int i=0;i<3;i++) {

System.out.println("请输入第"+(i+1)+"条边");

side[i]=in.nextDouble();

}

if((side[0]+side[1]<=side[2])||(side[1]+side[2]<=side[0])

||(side[2]+side[0]<=side[1])) {

System.out.println("这样的三角形不存在,请重新输入");

continue;

}

else {

var t3=new Triangle(side[0],side[1],side[2]);

System.out.println("请输入颜色");

String color=in.next();

System.out.println("请输入is filled");

boolean filled=in.nextBoolean();

t3.setColor(color);

t3.setFilled(filled);

System.out.println(t3.toString());

break;

}

}

in.close();

}

public Triangle() {

}

public Triangle(double side1, double side2, double side3) {

this.setSide1(side1);

this.setSide2(side2);

this.setSide3(side3);

}

public double getSide1() {

return side1;

}

public void setSide1(double side1) {

this.side1 = side1;

}

public double getSide2() {

return side2;

}

public void setSide2(double side2) {

this.side2 = side2;

}

public double getSide3() {

return side3;

}

public void setSide3(double side3) {

this.side3 = side3;

}

//获得周长

public double getPerimeter() {

return side1 + side2 + side3;

}

//获得面积

public double getArea() {

double p = getPerimeter();

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

}

//打印三角形三边、周长、面积、颜色、创建时间等信息

@Override

public String toString() {

return "Triangle: side1 = " + side1 + " side2 = " + side2 + " side3 = " + side3

+"\nCreatedDate: "+getDateCreate()+"\nArea: "+getArea()

+"\nPerimeter: "+getPerimeter()+"\nColor: "+getColor()

+"\nFilled or not: "+isFilled();

}

}

# Project #2分析设计

## 2.1问题分析

### 2.1.1 目标

一个继承GeometricObject抽象类和Colorable的接口的Squre类

一个继承GeometicObject抽象类和Comparable、Cloneable接口的Octagon类

### 2.1.2功能

#### 2.1.2.1功能一：显示color的方法

如果filled值为true,则调用howToColor()函数会显示how to color的具体方法

#### 2.1.2.2功能二：显示八边形的周长和面积

toString()会显示八边形的周长和面积信息

#### 2.1.2.3功能三：克隆八边形，并将它与被克隆的进行比较

显示克隆的对象和被克隆的对象的toString()信息，并且将这两个对象进行比较

### 2.1.3性能

#### 2.1.3.1功能一、二、三的性能

在用户输入文字的同时，程序在每次输入后的0.2秒之内响应并将结果显示到屏幕上

#### 2.1.3.2健壮性

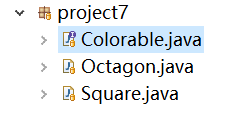
当使用clone()函数的时候可能会克隆失败发出异常，这个时候会将异常信息显示出来

#### 2.1.3.3异常输入处理

本工程不存在用户输入的情况

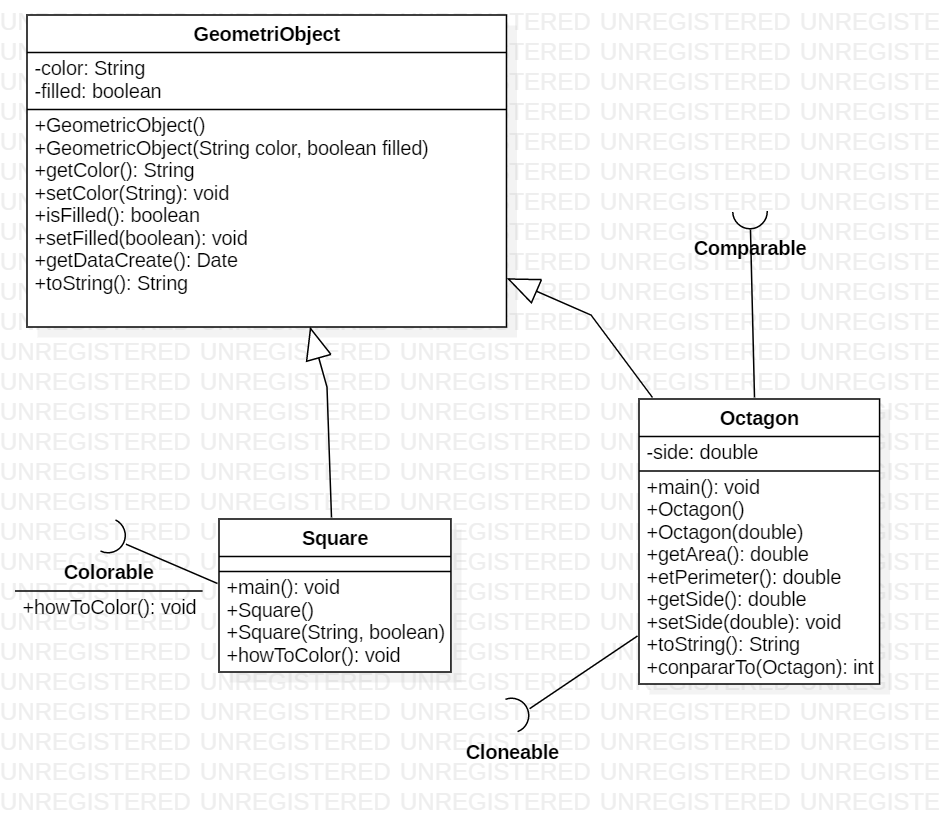
## 2.2设计方案

### 2.2.1模块划分



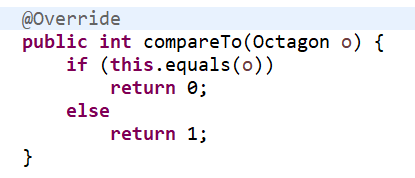
本项目只包含一个模块 ，四个类，其中GeometricObject类在project6中

### 2.2.2类图



### 2.2.3算法

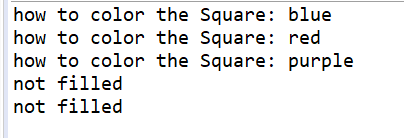
#### 2.2.1.1算法一：比较两个对象



调用equals()函数去比较两个对象

## 2.3实验结果

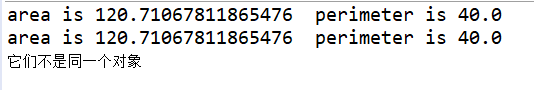
1. 五个GeometricObject对象调用的howToColor()函数的输出信息



1. 第一行为边长为5的八边形的面积和周长信息

第二行为调用clone函数生成的对象的面积和周长信息

第三行为两个对象相比较的结果



## 2.4 调试心得

1.clone()函数返回值是Object类型的，要将它显式地转换成Octagon类型

2.super-class可以指向subclass new出来的对象，但是如果要调用subclass的函数，还是必须将super-class显式转换成subclass类型

## 2.5源程序

Colorable.java:

package project7;

public interface Colorable {

public void howToColor();

}

Square.java

package project7;

import project6.GeometricObject;

public class Square extends GeometricObject implements Colorable {

public static void main(String[] args) {

GeometricObject[] o = new GeometricObject[5];

o[0] = new Square("blue", true);

o[1] = new Square("red", true);

o[2] = new Square("purple", true);

o[3] = new Square("black", false);

o[4] = new Square("while", false);

for (int i = 0; i < 5; i++) {

((Square) o[i]).howToColor();

}

}

public Square() {

super();

}

public Square(String color, boolean filled) {

super(color, filled);

}

@Override

public void howToColor() {

if (isFilled())

System.out.println("how to color the Square: " + getColor());

else

System.out.println("not filled");

}

}

Octagon.java

package project7;

import project6.GeometricObject;

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

private double side;

public static void main(String[] args) {

Octagon o = new Octagon(5);

System.out.println(o.toString());

// clone

Octagon x = null;

try {

x = (Octagon) o.clone();

} catch (CloneNotSupportedException e) {

e.printStackTrace();

}

System.out.println(x.toString());

// 对比结果

if (o.compareTo(x) == 0) {

System.out.println("它们是同一个对象");

} else

System.out.println("它们不是同一个对象");

}

public Octagon() {

}

public Octagon(double side) {

this.side = side;

}

public double getArea() {

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

}

public double getPerimeter() {

return side \* 8;

}

public double getSide() {

return side;

}

public void setSide(double side) {

this.side = side;

}

public String toString() {

return "area is " + getArea() + " perimeter is " + getPerimeter();

}

ok

@Override

public int compareTo(Octagon o) {

if (this.equals(o))

return 0;

else

return 1;

}

}

# Project #3分析设计

## 3.1问题分析

### 3.1.1 目标

两个线程随机生成字母，然后判断生成的两个字母的大小并获得得分。经过指定轮数后，判断两个线程谁的得分高。

### 3.1.2功能

#### 3.1.2.1功能一：随机Sleep,随机生成字母

线程AB会根据随机生成的数组进行sleep,并且会随机生成字母

#### 3.1.2.2功能二：判断A、B生成的字母大小，记录得分

线程C会判断每一轮A B线程生成字母的大小，并且记录得分，将得分、字母、sleep time显示在控制台

### 3.1.3性能

#### 3.1.3.1功能一、二的性能

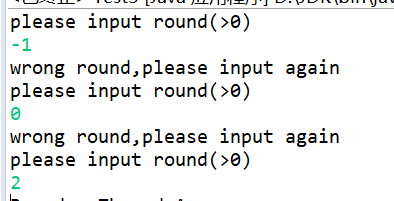
在用户输入轮数后，会等待线程ABsleep的最大时间后，将结果显示在控制台

#### 3.1.3.2健壮性

如果输入的轮数不符合逻辑，程序不会向下运行，会要求再次输入

#### 3.1.3.3异常输入处理

当用户输入的轮数小于等于0的时候，会提示用户输入错误，然后重新输入。



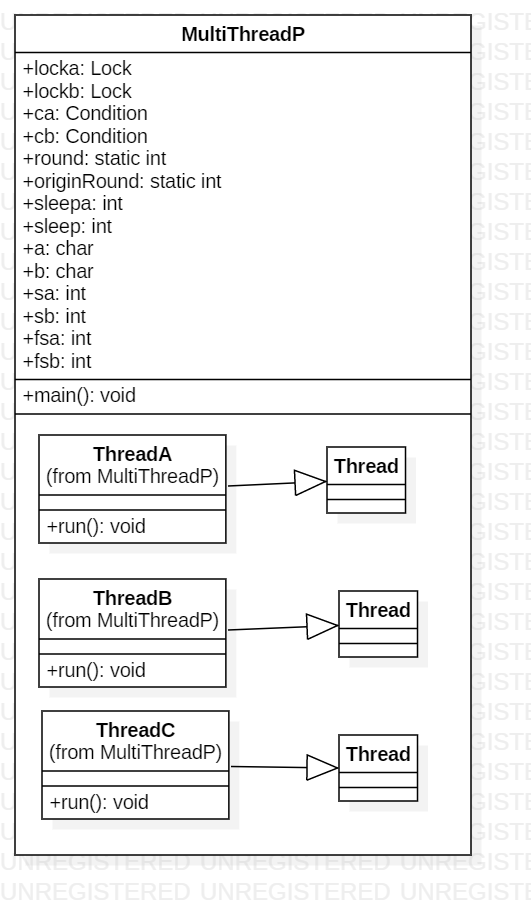
## 3.2设计方案

### 3.2.1模块划分



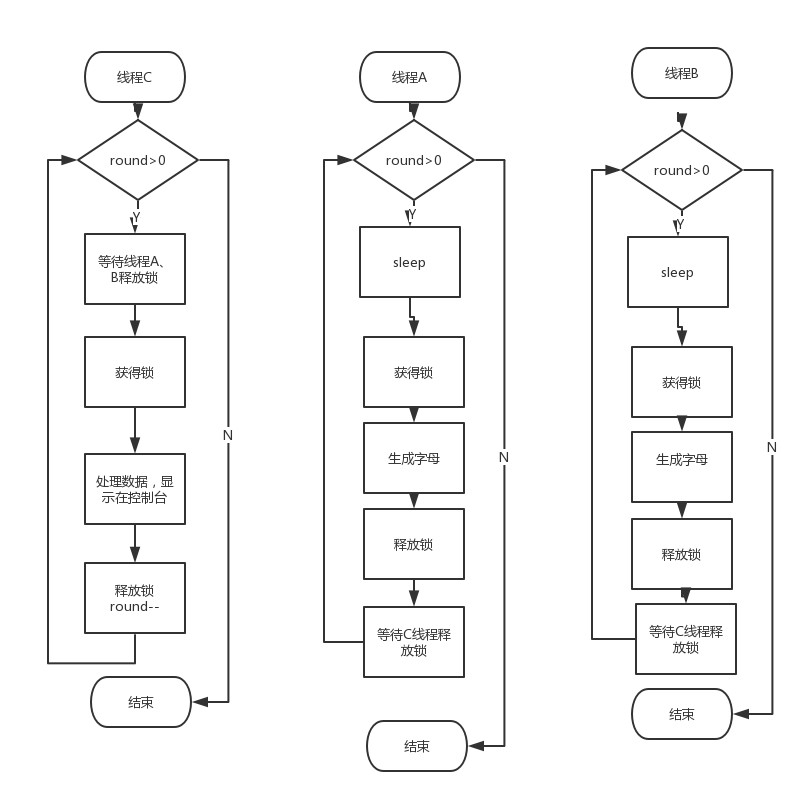
本项目只包含一个模块 ，一个类

### 3.2.2类图

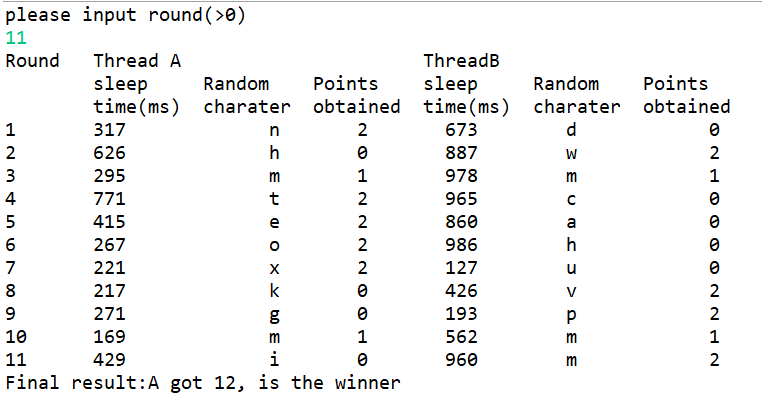


### 3.2.3算法

#### 3.2.1.1算法一：A、B、C之间的通讯



## 3.3实验结果

1. 

## 3.4 调试心得

C线程一开始应该进入等待状态，而AB是操作完了才为等待状态，C需要比AB先运行，不然会发生死锁

## 3.5源程序

MultiThreadP.java

package project8;

import java.util.Scanner;

import java.util.concurrent.locks.Condition;

import java.util.concurrent.locks.Lock;

import java.util.concurrent.locks.ReentrantLock;

public class MultiThreadP {

// 线程AB的锁

Lock locka = new ReentrantLock();

Lock lockb = new ReentrantLock();

Condition ca = locka.newCondition();

Condition cb = lockb.newCondition();

static int round = 0, originRound = 0;// 轮数

int sleepa, sleepb;// 用于输出A、B sleep的时间

char a, b;// A,B输出的字符

int sa = 0, sb = 0, fsa = 0, fsb = 0;// A,B每局得分记录和总得分

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

while (true) {

// 输入轮数

System.out.println("please input round(>0)");

originRound = in.nextInt();

round = originRound;

if (originRound <= 0) {

System.out.println("wrong round,please input again");

} else {

in.close();

break;

}

}

// 结果输出

System.out.println("Round" + " Thread A " + " ThreadB");

System.out.println("\t" + "sleep " + "Random " + "Points " + "sleep " + "Random " + "Points");

System.out.println(

"\t" + "time(ms) " + "charater " + "obtained " + "time(ms) " + "charater " + "obtained ");

// 生成、启动线程

MultiThreadP t = new MultiThreadP();

ThreadC tc = t.new ThreadC();

new Thread(tc).start();

ThreadA ta = t.new ThreadA();

ta.start();

ThreadB tb = t.new ThreadB();

tb.start();

}

// 线程A

class ThreadA extends Thread {

@Override

public void run() {

while (true) {

if (round == 0) {// 线程结束

break;

}

// 产生sleep的随机数并进行sleep

sleepa = (int) (Math.random() \* 1000);

try {

Thread.sleep(sleepa);

} catch (InterruptedException e) {

e.printStackTrace();

}

// 产生字母

locka.lock();

a = (char) (Math.random() \* 100 % 26 + 97);

ca.signal();

try {

ca.await();// 等待C处理完后放出信号

locka.unlock();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

// 线程B

class ThreadB extends Thread {

@Override

public void run() {

while (true) {

// 线程结束

if (round == 0) {

break;

}

// 产生sleep参数并且sleep

sleepb = (int) (Math.random() \* 1000);

try {

Thread.sleep(sleepb);

} catch (InterruptedException e) {

e.printStackTrace();

}

// 产生b

lockb.lock();

b = (char) (Math.random() \* 100 % 26 + 97);

cb.signal();

try {

cb.await();

lockb.unlock();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

}

// 线程C

class ThreadC implements Runnable {

@Override

public void run() {

while (round > 0) {

try {

// 等待AB先完成一次

locka.lock();

lockb.lock();

ca.await();

cb.await();

} catch (InterruptedException e) {

e.printStackTrace();

}

// 输出信息

if (a > b) {

sa = 2;

sb = 0;

}

if (a == b) {

sa = 1;

sb = 1;

}

if (a < b) {

sa = 0;

sb = 2;

}

System.out.println(originRound - round + 1 + "\t" + sleepa + "\t" + "\t" + a + "\t" + sa + "\t" + sleepb

+ "\t " + b + "\t" + "\t" + sb);

fsa = fsa + sa;

fsb = fsb + sb;

round--;

// 释放锁

ca.signal();

cb.signal();

lockb.unlock();

locka.unlock();

}

// 最终得分

if (fsa > fsb)

System.out.println("Final result:A got " + fsa + ", is the winner");

else if (fsa < fsb)

System.out.println("Final result:B got " + fsb + ", is the winner");

else

System.out.println("Final result:end in a draw,A and B both got " + fsa);

}

}

}

# Project #4分析设计

## 4.1问题分析

### 4.1.1 目标

kidA,kidB随机出石头、剪刀、布，给出每一轮A，B的得分和最后的总得分及输赢

### 4.1.2功能

#### 4.1.2.1功能一：A，B随机Sleep和生成符号

A，B会根据随机生成的sleeptime进行Sleep,以及生成代表剪刀、石头、布的三种符号中的一个符号

#### 4.1.2.2功能二：判断A，B的输赢

在A，B都生成符号后，会判断A，B每一轮的得分并将得分输出在控制台。当轮数结束后，会输出总成绩和输赢情况

### 4.1.3性能

#### 4.1.3.1功能一、二的性能

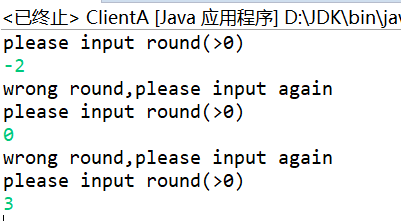
在输入轮数后，经过A，B线程sleeptime最大的值后，会将结果显示在控制台

#### 4.1.3.2健壮性

如果输入的轮数不符合逻辑，会要求重新输入

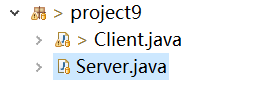
#### 4.1.3.3异常输入处理

当输入的轮数<=0的时候，会显示重新输入轮数



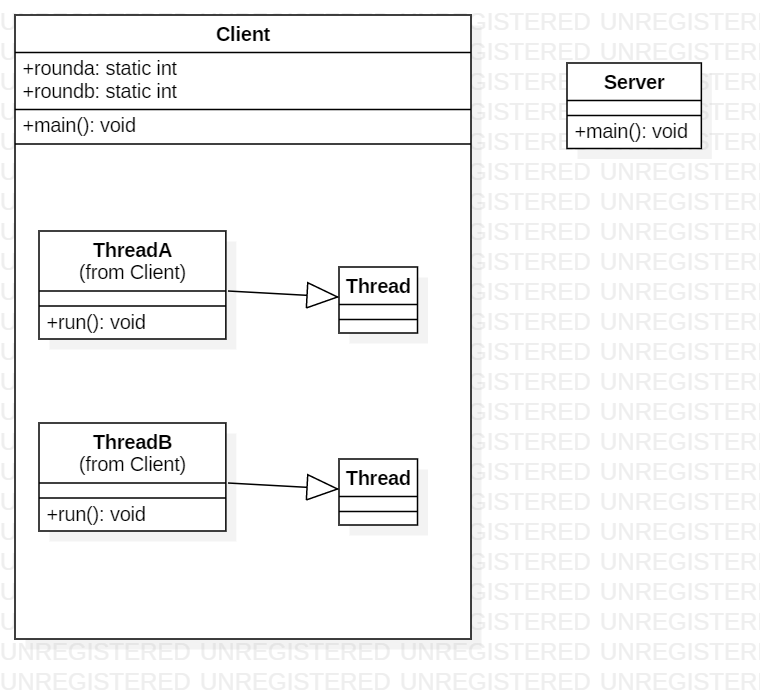
## 4.2设计方案

### 4.2.1模块划分



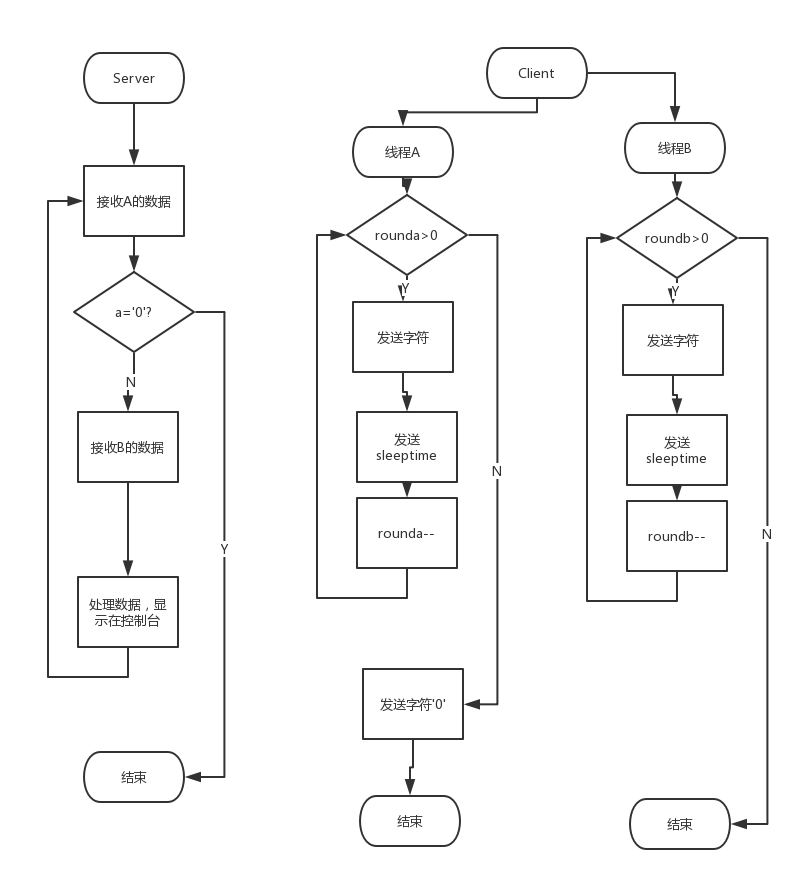
本项目只包含一个模块 ，两个类

### 4.2.2类图



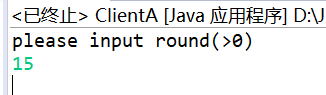
### 4.2.3算法

#### 4.2.1.1算法一：Server和Client之间的通信

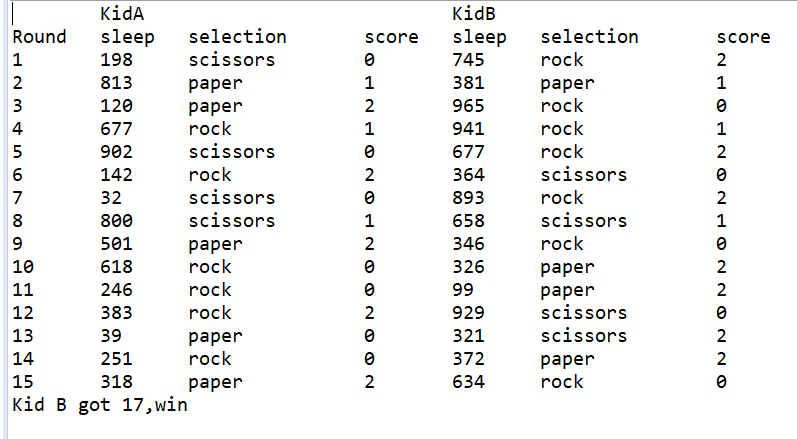


## 4.3实验结果

1. client:



1. server:



## 4.4 调试心得

在实验的过程中发现，即使没有用project8中锁的方式进行同步，Server仍然能依次收到每一轮的正确数据。经查阅资料得到：TCP默认模式下是阻塞模式。也就是说只有A发送了，Server接收了，A才能继续发送，Server也才能继续接收而进入下一步。这样就实现了线程间的同步。

## 4.5源程序

Client.java:

package project9;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.net.Socket;

import java.net.SocketException;

import java.net.UnknownHostException;

import java.util.Scanner;

import java.util.concurrent.locks.Lock;

import java.util.concurrent.locks.ReentrantLock;

public class Client {

static int rounda = 0;

static int roundb = 0;

public static void main(String[] args) {

// 输入轮数

Scanner in = new Scanner(System.in);

while (true) {

System.out.println("please input round(>0)");

rounda = in.nextInt();

if (rounda > 0) {

in.close();

break;

} else {

System.out.println("wrong round,please input again");

continue;

}

}

roundb = rounda;

// 启动线程

Client client = new Client();

ThreadA ta = client.new ThreadA();

ta.start();

ThreadB tb = client.new ThreadB();

tb.start();

}

// 对应kidA

class ThreadA extends Thread {

@Override

public void run() {

try {

// 连接server

Socket socket = new Socket("127.0.0.1", 8888);

var out = new DataOutputStream(socket.getOutputStream());

char a;

int sleepa;

while (rounda > 0) {

// 传送字符

a = (char) (Math.random() \* 100 % 3 + 97);

out.writeChar(a);

// 传送sleeptime

sleepa = (int) (Math.random() \* 1000);

out.writeInt(sleepa);

// 进行sleep

Thread.sleep(sleepa);

rounda--;

}

// 将结束信息写给server

out.writeChar('0');

socket.close();

} catch (IOException | InterruptedException e) {

e.printStackTrace();

}

}

}

//对应KidB

class ThreadB extends Thread {

@Override

public void run() {

InetAddress server;

DatagramPacket sndPkt;

try {

// 连接server

DatagramSocket socket = new DatagramSocket();

server = InetAddress.getByName("localhost");

char b;

int sleepb;

while (roundb > 0) {

// 传送字符

b = (char) (Math.random() \* 100 % 3 + 97);

byte[] buf = Character.valueOf(b).toString().getBytes();

sndPkt = new DatagramPacket(buf, buf.length);

sndPkt.setAddress(server);

sndPkt.setPort(8000);

sndPkt.setData(buf);

socket.send(sndPkt);

// 传送sleeptime

sleepb = (int) (Math.random() \* 1000);

buf = Integer.valueOf(sleepb).toString().getBytes();

sndPkt.setData(buf);

socket.send(sndPkt);

// 进行Sleep

Thread.sleep(sleepb);

roundb--;

}

socket.close();

} catch (IOException | InterruptedException e1) {

e1.printStackTrace();

}

}

}

}

Server.java:

package project9;

import java.io.DataInputStream;

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.Arrays;

public class Server {

public static void main(String[] args) {

try {

char a, b;// A,B对应字符

int sleepa, sleepb;// A，B对应sleep时间

int sa, sb;// 单局得分

int fsa = 0, fsb = 0;// 最终得分

int round = 0;

String outa = null, outb = null;// 用于输出剪刀石头布

System.out.println("\t" + "KidA" + "\t\t\t\t" + "KidB");

System.out.println("Round" + "\t" + "sleep" + "\t" + "selection" + "\t" + "score" + "\t" + "sleep" + "\t"

+ "selection" + "\t" + "score");

// TCP：ThreadA的监听窗口， 监听并等待接受连接

ServerSocket servera = new ServerSocket(8888);

Socket tcpsocket = servera.accept();

// 接收A的数据流

var inA = new DataInputStream(tcpsocket.getInputStream());

// UDP，用于接收B

byte[] buf = new byte[1024];

DatagramPacket rcvPkt = new DatagramPacket(buf, buf.length);

DatagramSocket udpsocket = new DatagramSocket(8000);

while (true) {

a = inA.readChar();// 读入A的字符

if (a == '0')// 结束条件

break;

sleepa = inA.readInt();// 读入A的sleeptime

// 接收B的字符

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

rcvPkt.setData(buf);

udpsocket.receive(rcvPkt);

b = new String(buf).trim().charAt(0);

// 接收B的sleeptime

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

rcvPkt.setData(buf);

udpsocket.receive(rcvPkt);

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

// 进行判断

if ((a == 'a' && b == 'b') || (a == 'b' && b == 'c') || (a == 'c' && b == 'a')) {

sa = 2;

sb = 0;

fsa = fsa + sa;

} else if ((b == 'a' && a == 'b') || (b == 'b' && a == 'c') || (b == 'c' && a == 'a')) {

sb = 2;

sa = 0;

fsb = fsb + sb;

} else {

sa = 1;

sb = 1;

fsa = fsa + 1;

fsb = fsb + 1;

}

// 进行输出

if (a == 'a')

outa = "rock";

if (a == 'b')

outa = "scissors";

if (a == 'c')

outa = "paper";

if (b == 'a')

outb = "rock";

if (b == 'b')

outb = "scissors";

if (b == 'c')

outb = "paper";

round++;

System.out.println(round + "\t" + sleepa + "\t" + outa + " \t" + sa + "\t" + sleepb + "\t" + outb

+ " \t" + sb);

}

// 最终输出

if (fsa > fsb) {

System.out.println("Kid A got " + fsa + ",win");

} else if (fsa < fsb) {

System.out.println("Kid B got " + fsb + ",win");

} else {

System.out.println("Kid A and Kid B end in a draw ,both got " + fsa);

}

udpsocket.close();

tcpsocket.close();

servera.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}