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： 胡亦璇

学号： 2018211884

指导教师： 崔毅东

日 期： 2019 年 10 月 16 日

# 实验概述

## 实验目的

本实验旨在通过一系列Java语言简单程序的编写，使学生掌握Java通用IDE，练习类的封装使用、Java基本类库的使用、Java GUI编程和事件驱动编程、利用UML进行简单建模，提高编程实践能力。

## 实验内容

|  |  |
| --- | --- |
|  | 任务说明 |
|  | 分析项目需求，细化所需实现的功能 |
|  | 分析哪些数据为输入、输出、存储等 |
|  | 设计UI，设计类 |
|  | 画UML类图 |
|  | 用伪代码写程序大致结构 |
|  | 编程实现 |
|  | 调试，测试 |

## 实验环境

Windows系统下Eclipse IDE（配置javafx zulu11）

### 硬件环境

Lenovo ThinkPad X250 64bit 8G

### 网络环境

无

### 软件环境

操作系统：Windows10 64bit; IDE：Eclipse 64bit

## 实验结果

五个程序均能完成实验指导书的基本要求，及时响应键盘、鼠标输入，通过控制台或Javafx窗口在屏幕上输出，并且计算正确、运行良好。详见“二、实验内容”部分。

# 实验内容

# Project #1分析设计

## 1.1问题分析

### 1.1.1 目标

设计一个GUI程序，模拟四辆小车赛跑。用户可通过文本框输入分别设置四辆小车的速度，1代表最快。

### 1.1.2功能

#### 1.1.2.1功能一：接收键盘输入

#### 1.1.2.2功能二：显示小车运动动画

### 1.1.3性能

#### 1.1.3.1程序响应的性能

在用户进行文本框输入后，程序在0.5秒之内响应并将重置动画显示到屏幕上

#### 1.1.3.2健壮性

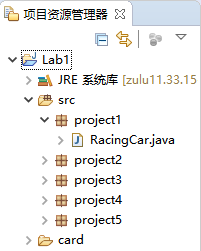
当输入为数字，程序正常执行。若输入不为数字，程序捕捉异常，并在控制台输出“Please input number!”，等待下一次输入。

#### 1.1.3.3异常输入处理

见1.1.3.2健壮性

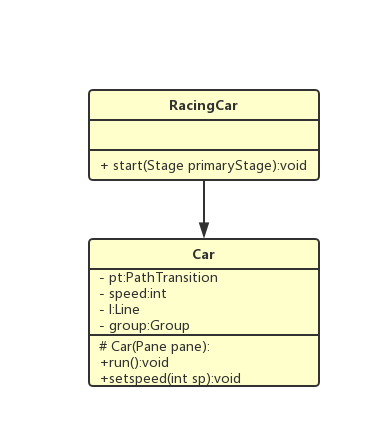
## 1.2设计方案

### 1.2.1模块划分



本项目只包含一个模块 RacingCar

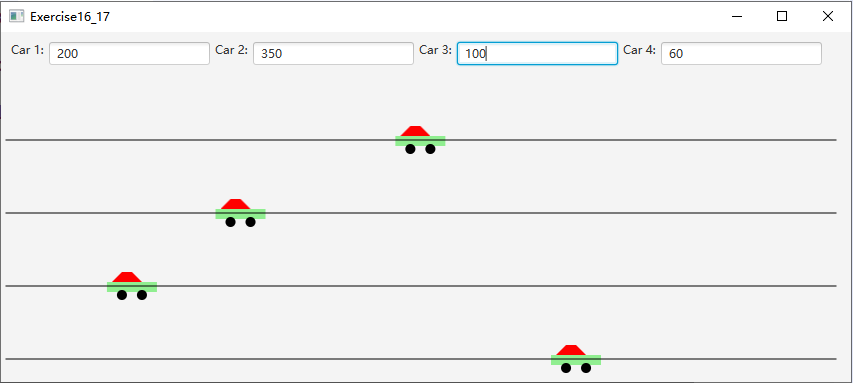
### 1.2.2类图



### 1.2.3算法

无

## 1.3实验结果



运行程序后，屏幕上显示四辆小车、赛道以及四辆小车分别的速度因子输入框（如上图），当输入数字并回车后，小车立刻以相应速度沿赛道运动，到最右端自动停下。若输入不为数字，程序捕捉异常，并在控制台输出“Please input number!”，小车将继续以上一次输入的速度因子运动，直到输入为数字。

## 1.4 调试心得

在一个文本域中回车属于ActionEvent类型，而不是KeyEvent类型。

注意类中构造函数内部变量与外部成员变量的区别，注意变量放置位置，这很有可能导致程序出现意想不到的错误。

自定义类型可通过extends Pane转换成节点类型。

注意各类面板的嵌套关系。

## 1.5源程序

//RacingCar.java

package project1;

import javafx.animation.PathTransition;

import javafx.application.Application;

import javafx.collections.ObservableList;

import javafx.geometry.Insets;

import javafx.scene.Group;

import javafx.scene.Scene;

import javafx.scene.control.Label;

import javafx.scene.control.TextField;

import javafx.scene.shape.\*;

import javafx.scene.layout.\*;

import javafx.scene.paint.Color;

import javafx.stage.Stage;

import javafx.util.Duration;

public class RacingCar extends Application{

@Override

public void start(Stage primaryStage) {

//Create 4 pane

Pane pane1=new Pane();

Pane pane2=new Pane();

Pane pane3=new Pane();

Pane pane4=new Pane();

//Create 4 car and place them in panes

Car car1=new Car(pane1);

Car car2=new Car(pane2);

Car car3=new Car(pane3);

Car car4=new Car(pane4);

//Create a HBox

HBox hbox=new HBox(5);

hbox.setPadding(new Insets(5,5,5,5));

//Create some labels and TextField and put them in HBox

TextField tx1=new TextField();

TextField tx2=new TextField();

TextField tx3=new TextField();

TextField tx4=new TextField();

hbox.getChildren().addAll(new Label("Car 1:"),tx1,new Label("Car 2:"),tx2,

new Label("Car 3:"),tx3,new Label("Car 4:"),tx4);

//Place nodes in a VBox

VBox vbox=new VBox(5);

vbox.setPadding(new Insets(5,5,5,5));

vbox.getChildren().addAll(hbox,pane1,pane2,pane3,pane4);

//TextField's EventHandler

tx1.setOnAction(e-> {

try {

int input1 = Integer.valueOf(tx1.getText()).intValue();

car1.setspeed(input1);

}catch(Exception ex) {

System.out.println("Please input number!");

}

car1.run();

});

tx2.setOnAction(e-> {

try {

int input2 = Integer.valueOf(tx2.getText()).intValue();

car2.setspeed(input2);

}catch(Exception ex) {

System.out.println("Please input number!");

}

car2.run();

});

tx3.setOnAction(e-> {

try {

int input3 = Integer.valueOf(tx3.getText()).intValue();

car3.setspeed(input3);

}catch(Exception ex) {

System.out.println("Please input number!");

}

car3.run();

});

tx4.setOnAction(e->{

try {

int input4 = Integer.valueOf(tx4.getText()).intValue();

car4.setspeed(input4);

}catch(Exception ex) {

System.out.println("Please input number!");

}

car4.run();

});

//Create a scene and place it in stage

Scene scene=new Scene(vbox,850,350);

primaryStage.setTitle("Exercise16\_17");

primaryStage.setScene(scene);

primaryStage.show();

}

}

class Car extends Pane{

//创建一个移动动画类PathTransition的对象

private PathTransition pt = new PathTransition();

private int speed=100;

private Line l=new Line(0,65,830,65);//小车跑道

private Group group=new Group();

Car(Pane pane){//画一辆小车

Rectangle r=new Rectangle(0,50,50,10);

Circle c1=new Circle(15,63,5);

Circle c2=new Circle(35,63,5);

Polygon p=new Polygon();

ObservableList<Double> list=p.getPoints();

list.addAll(15.0,40.0,5.0,50.0,35.0,50.0,25.0,40.0);

r.setFill(Color.LIGHTGREEN);

c1.setFill(Color.BLACK);

c2.setFill(Color.BLACK);

p.setFill(Color.RED);

group.getChildren().addAll(p,r,c1,c2);//将小车各部分形状绑成组

pane.getChildren().addAll(group,l);

}

void run() {//使小车运动的函数

pt.setDuration(Duration.millis(speed\*100));//设置动画周期持续时间

pt.setPath(l);//小车运动轨迹

pt.setNode(group);//设置要移动的图形为小车组

pt.setAutoReverse(false);//设置不自动转向

pt.setCycleCount(1);//设置循环次数为1

pt.play();

return;

}

void setspeed(int sp) {//设置小车速度因子

speed=sp;

}

}

# Project #2分析设计

## 2.1问题分析

### 2.1.1 目标

该程序需要根据四种不同税收类型的政策，打印出相应金额的税收价目表。

### 2.1.2功能

#### 2.1.2.1功能一：从键盘获取金额上界、下界以及打表金额间隔

#### 2.1.2.2功能二：在控制台打印出相应金额段的税收表

### 2.1.3性能

#### 2.1.3.1响应输入性能

在用户输入文字的同时，程序在每次输入后的0.4秒之内响应并将结果显示到屏幕上。

#### 2.1.3.2健壮性

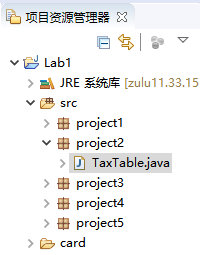
当金额下界不小于上界时，程序抛出异常并捕获，在控制台打印出“downbound must be smaller than upbound!”，等待重新输入；当退出选项用户输入与选项不匹配时，程序抛出异常并捕获，在控制台打印出“Please input only 1 or 2!”，等待重新输入。

#### 2.1.3.3异常输入处理

当输入不为数字时，程序抛出异常并捕获，在控制台打印出“Input should be number!”，等待重新输入；当金额间隔为0时，程序抛出异常并捕获，在控制台打印出“Interval shouldn’t be zero!”，等待重新输入。

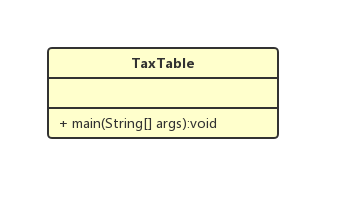
## 2.2设计方案

### 2.2.1模块划分



本项目只包含一个模块 TaxTable

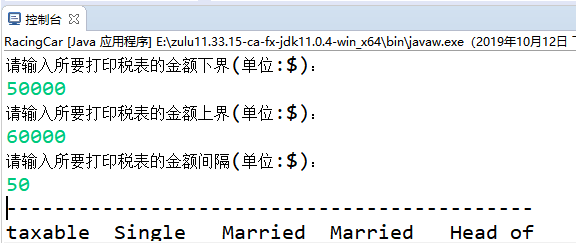
### 2.2.2类图

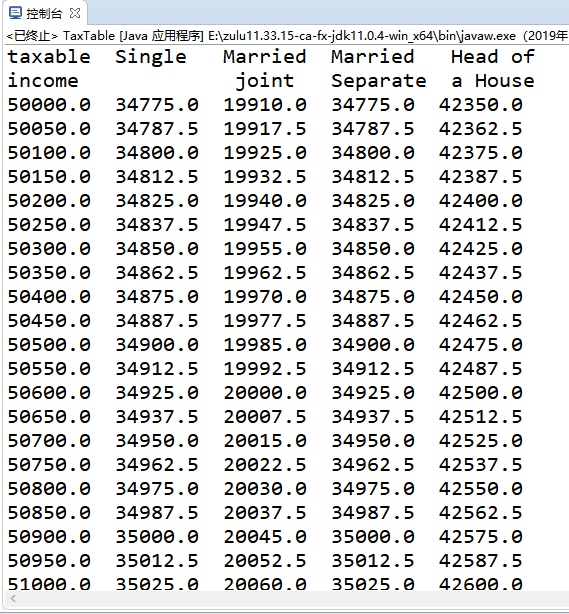


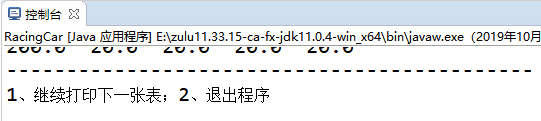
### 2.2.3算法

无

## 2.3实验结果







运行程序后，控制台屏幕上打印出如上图1，用户依次输入三个金额数据后回车，程序计算出四种类型下所应缴纳税额，并且打印如上图2的表格。打印完一张表后，出现上图3退出选项，选1继续；选2退出。

## 2.4 调试心得

注意Java中获取控制台输入的Scanner是一个类，而不是函数，要实例化。

注意输入时各种异常情况的捕获处理。

## 2.5源程序

//TaxTable.java

package project2;

import java.util.InputMismatchException;

import java.util.Scanner;

public class TaxTable {

public static void main(String[] args) {

while(true) {

double downbound=0,upbound=0,interval=0;

double level1=0,level2=0,level3=0,level4=0,level5=0;

double tax=0;

while(true) {

System.out.println("请输入所要打印税表的金额下界(单位:$)：");

try {

@SuppressWarnings("resource")

Scanner scan1=new Scanner(System.in);

downbound=scan1.nextDouble();

System.out.println("请输入所要打印税表的金额上界(单位:$)：");

upbound=scan1.nextDouble();

System.out.println("请输入所要打印税表的金额间隔(单位:$)：");

interval=scan1.nextDouble();

//若间隔金额为0，抛出异常

if(interval==0)throw new IllegalAccessException("Input is not a number");

//若下界大于上界或输入不为数字，抛出异常

if(downbound>=upbound)throw new InputMismatchException("Downbound must be smaller than upbound");

else break;

}catch(InputMismatchException ex) {

System.out.println("Input should be number!And downbound must be smaller than upbound!");

}

catch(IllegalAccessException e) {

System.out.println("Interval shouldn't be zero!");

}

}

System.out.println("--------------------------------------------");

System.out.println("taxable Single Married Married Head of");

System.out.println("income joint Separate a House");

for(;downbound<=upbound;downbound+=interval){

System.out.print(downbound+" ");

for(int mode=1;mode<=4;mode++){

switch(mode){

case 1://X

level1=7300;

level2=29700;

level3=71950;

level4=15150;

level5=326450;

break;

case 2://Y-1

level1=14600;

level2=59400;

level3=119950;

level4=182800;

level5=326450;

break;

case 3://Y-2

level1=7300;

level2=29700;

level3=59975;

level4=91400;

level5=163225;

break;

case 4://Z

level1=10450;

level2=39800;

level3=102800;

level4=166450;

level5=326450;

break;

}

if(downbound<=level1)tax=downbound\*0.1;

else if(downbound<=level2)tax=level1+(downbound-level1)\*0.15;

else if(downbound<=level3)tax=level2+(downbound-level2)\*0.25;

else if(downbound<=level4)tax=level3+(downbound-level3)\*0.28;

else if(downbound<=level5)tax=level4+(downbound-level4)\*0.33;

else tax=level5+(downbound-level5)\*0.35;

System.out.print(tax+" ");

}

System.out.println(" ");

}

System.out.println("--------------------------------------------");

while(true) {

System.out.println("1、继续打印下一张表；2、退出程序");

try {

@SuppressWarnings("resource")

Scanner scan2=new Scanner(System.in);

int op=scan2.nextInt();

if(op==2) System.exit(0);

else if(op==1)break;

//如果输入不是1或2，抛出异常

else throw new InputMismatchException("Input only can be 1 or 2");

}catch (InputMismatchException ex) {

System.out.println("Please input only 1 or 2!");

}

}

}

}

}

# Project #3分析设计

## 3.1问题分析

### 3.1.1 目标

该程序需要根据控制台用户输入的两个数计算出它们的最小公倍数，并打印到屏幕上。

### 3.1.2功能

#### 3.1.2.1功能一：计算最小公倍数

### 3.1.3性能

#### 3.1.3.1响应输入性能

在用户输入回车后的0.4秒之内程序响应，并将结果显示到屏幕上。

#### 3.1.3.2健壮性

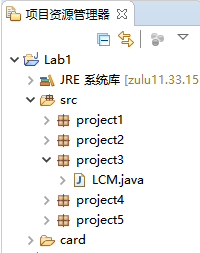
能抛出、捕获并处理输入异常。

#### 3.1.3.3异常输入处理

当输入不为数字时，程序抛出异常并捕获，在控制台打印出“Input should be number!”，等待重新输入。当退出选项用户输入与选项不匹配时，程序抛出异常并捕获，在控制台打印出“Please input only 1 or 2!”，等待重新输入。

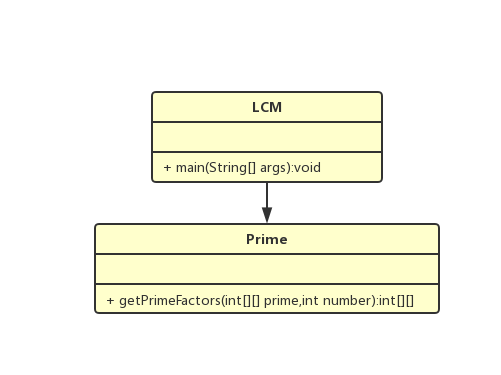
## 3.2设计方案

### 3.2.1模块划分



本项目只包含一个模块 LCM

### 3.2.2类图



### 3.2.3算法

#### 3.2.3.1算法一：计算数的因子

for(int i=2;i<=number;i++) {//从最小因子2开始遍历

while(number%i==0) {//若能整除，则i为因子

prime[j][0]=i;//将i存入数组第一列

prime[j][1]++;//i出现次数加1，存入数组第二列

if(number%i==0&&(number/i)%i!=0) {//若number中不再出现i重复的因子

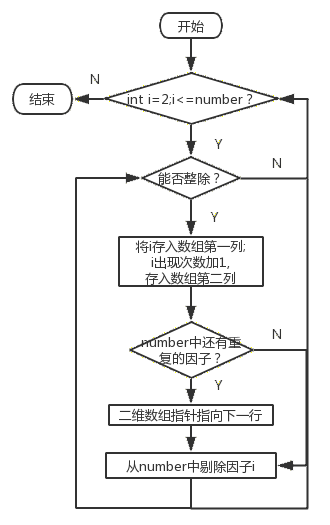
j++;//二维数组指针指向下一行

}

number=number/i;//从number中剔除因子i

}

}



#### 3.2.3.2算法二：由因子计算最小公倍数

for(;prime1[k][0]!=0;) {//遍历第一个数的因子

for(;prime2[j][0]!=0;) {//固定第一个数的某个因子a，遍历第二个数的因子b

if(prime1[k][0]<prime2[j][0]) {//比较a,b;若a<b

result\*=Math.pow(prime1[k][0],prime1[k][1]);//将a乘进result

if((prime1[k+1][0]==0)&&(prime2[j+1][0]!=0))j++;//如果此时a已经是数1的最后一个因子，且b不是数2的最后一个因子，数2指针向后移

k++;//数1指针向后移

break;

}

if(prime1[k][0]>prime2[j][0]) {//若a>b

result\*=Math.pow(prime2[j][0],prime2[j][1]);//将b乘进result

j++;//数2指针向后移

if(prime2[j][0]==0) {//如果此时b已经是数2的最后一个因子

result\*=Math.pow(prime1[k][0],prime1[k][1]);//将a也乘进result

k++;//数1指针向后移

}

break;

}

if(prime1[k][0]==prime2[j][0]) {//若a=b

result\*=Math.pow(prime1[k][0],Math.max(prime1[k][1],prime2[k][1]));//取该因子出现最大次数进行乘方运算，乘进result

k++;//数1数2指针均向后移

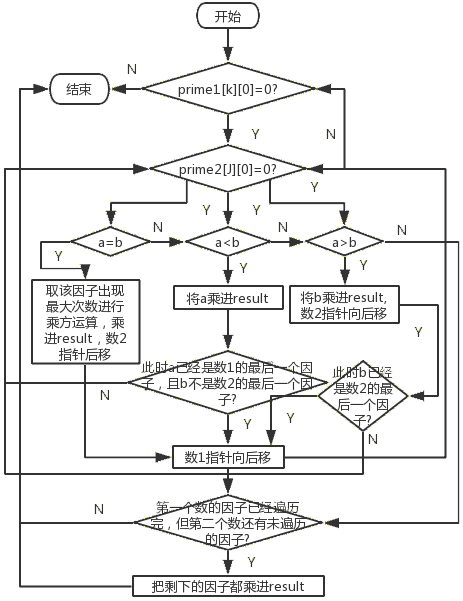
j++;

}

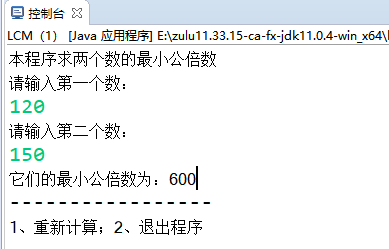
}

}

if(prime2[j][0]!=0)//如果第一个数的因子已经遍历完，但第二个数还有未遍历的因子 for(;prime2[j][0]!=0;j++)result\*=Math.pow(prime2[j][0],prime2[j][1]);//把剩下的因子都乘进result



## 3.3实验结果



运行程序后，控制台屏幕上打印出如上图，用户依次输入两个数据后回车，程序计算出它们的最小公倍数，并且打印在屏幕上。计算完一组数据后，出现退出选项，选1继续；选2退出。

## 3.4 调试心得

计算最小公倍数时，注意两个数组指针变化不一定同步，要考虑全各种可能。

计算数的因子时，注意处理重复出现的因子，注意存入二维数组中的位置。

遇到逻辑错误，要设置断点，单步调试，打开监视窗口观察变量的改变和分支条件跳转是否与设想中一致，从而缩小bug所在范围。

## 3.5源程序

//LCM.java

package project3;

import java.util.InputMismatchException;

import java.util.Scanner;

public class LCM {

public static void main(String[] args) {

while(true) {

int result=1,num1=0,num2=0,flag;

//两个二维数组存放分别两个数的因子和因子出现次数

int[][] prime1=new int[50][2];

int[][] prime2=new int[50][2];

System.out.println("本程序求两个数的最小公倍数");

do {//do-while循环、flag和异常处理一起保证用户输入为数字

flag=0;

System.out.println("请输入第一个数：");

try {

@SuppressWarnings("resource")

Scanner scan1=new Scanner(System.in);

num1=scan1.nextInt();

System.out.println("请输入第二个数：");

num2=scan1.nextInt();

}catch(InputMismatchException ex) {//捕获输入不为数字的异常

System.out.println("Input should be number!");

flag=1;

}

}while(flag==1);

if(num2==1) {

int temp=num1;

num1=num2;

num2=temp;

}

Prime prime\_v=new Prime();//实例化自定义类Prime

prime\_v.getPrimeFactors(prime1,num1);//分别计算两个数的因子，并存入对应数组

prime\_v.getPrimeFactors(prime2,num2);

//跟据因子计算最小公倍数

int k=0,j=0;

for(;prime1[k][0]!=0;) {//遍历第一个数的因子

for(;prime2[j][0]!=0;) {//固定第一个数的某个因子a，遍历第二个数的因子b

if(prime1[k][0]<prime2[j][0]) {//比较a,b;若a<b

result\*=Math.pow(prime1[k][0],prime1[k][1]);//将a乘进result

if((prime1[k+1][0]==0)&&(prime2[j+1][0]!=0))j++;//如果此时a已经是数1的最后一个因子，且b不是数2的最后一个因子，数2指针向后移

k++;//数1指针向后移

break;

}

if(prime1[k][0]>prime2[j][0]) {//若a>b

result\*=Math.pow(prime2[j][0],prime2[j][1]);//将b乘进result

j++;//数2指针向后移

if(prime2[j][0]==0) {//如果此时b已经是数2的最后一个因子

result\*=Math.pow(prime1[k][0],prime1[k][1]);//将a也乘进result

k++;//数1指针向后移

}

break;

}

if(prime1[k][0]==prime2[j][0]) {//若a=b

result\*=Math.pow(prime1[k][0],Math.max(prime1[k][1],prime2[k][1]));//取该因子出现最大次数进行乘方运算，乘进result

k++;//数1数2指针均向后移

j++;

}

}

}

if(prime2[j][0]!=0)//如果第一个数的因子已经遍历完，但第二个数还有未遍历的因子

for(;prime2[j][0]!=0;j++)result\*=Math.pow(prime2[j][0],prime2[j][1]);//把剩下的因子都乘进result

System.out.println("它们的最小公倍数为："+result);

System.out.println("-----------------");

while(true) {

System.out.println("1、重新计算；2、退出程序");

try {

@SuppressWarnings("resource")

Scanner scan2=new Scanner(System.in);

int op=scan2.nextInt();

if(op==2) System.exit(0);

else if(op==1)break;

//如果输入不是1或2，抛出异常

else throw new InputMismatchException("Input only can be 1 or 2");

}catch (InputMismatchException ex) {

System.out.println("Please input only 1 or 2!");

}

}

}

}

}

class Prime{

//计算数的因子及其出现次数，存入一个二维数组

public int[][] getPrimeFactors(int[][]prime,int number){

int j=0;

for(int i=2;i<=number;i++) {//从最小因子2开始遍历

while(number%i==0) {//若能整除，则i为因子

prime[j][0]=i;//将i存入数组第一列

prime[j][1]++;//i出现次数加1，存入数组第二列

if(number%i==0&&(number/i)%i!=0) {//若number中不再出现i重复的因子

j++;//二维数组指针指向下一行

}

number=number/i;//从number中剔除因子i

}

}

return prime;

}

}

# Project #4分析设计

## 4.1问题分析

### 4.1.1 目标

写一个程序能够创建矩形对象并设置其宽、高、颜色等属性，并能计算其周长、面积，将其打印在控制台。

写一个程序能够创建时间对象并设置其毫秒数，并能以此换算从1970年1月1日开始计时，经过该毫秒数后的时间，将当前时间时分秒以及换算后的时分秒打印在控制台。

### 4.1.2功能

#### 4.1.2.1功能一：创建矩形对象并显示其各种属性

#### 4.1.2.2功能二：创建时间对象并显示其时分秒

### 4.1.3性能

#### 4.1.3.1程序性能

在用户运行程序后的0.4秒之内响应并将结果显示到屏幕上。

#### 4.1.3.2健壮性

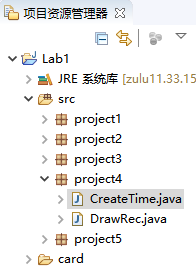
无

#### 4.1.3.3异常输入处理

无控制台输入

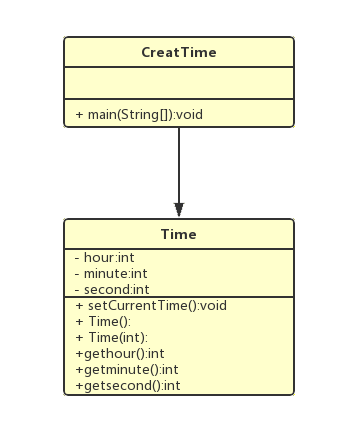
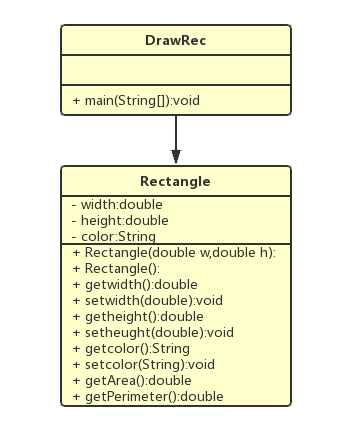
## 4.2设计方案

### 4.2.1模块划分



本项目包含两个模块，分别为：DrawRec，CreateTime

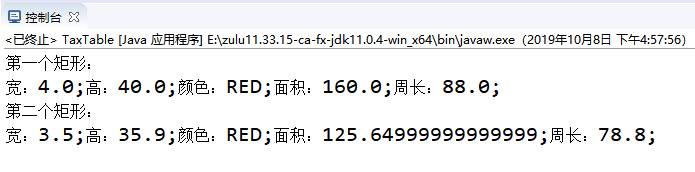
### 4.2.2类图

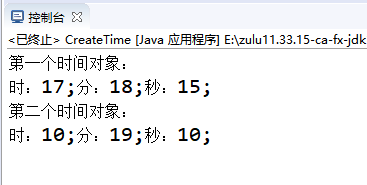


### 4.2.3算法

无

## 4.3实验结果





运行创建矩形程序后，控制台屏幕上打印出如上图1，分别计算显示两个矩形变量的宽、高、颜色、面积、周长。

运行创建时间变量程序后，控制台屏幕上打印出如上图2，分别计算显示当前时间时分秒和从1970年1月1日开始计算的时分秒。

## 4.4 调试心得

无

## 4.5源程序

//DrawRec.java

package project4;

public class DrawRec {

public static void main(String[] args) {

//创建两个矩形对象

Rectangle r1= new Rectangle(4,40);

Rectangle r2= new Rectangle(3.5,35.9);

r1.setcolor("RED");

r2.setcolor("RED");

//显示两个矩形的属性并计算面积和周长

System.out.println("第一个矩形：");

System.out.println("宽："+r1.getwidth()+";"+"高："+r1.getheight()+";"

+"颜色："+r1.getcolor()+";"+"面积："+r1.getArea()+";"+"周长："+r1.getPerimeter()+";");

System.out.println("第二个矩形：");

System.out.println("宽："+r2.getwidth()+";"+"高："+r2.getheight()+";"

+"颜色："+r2.getcolor()+";"+"面积："+r2.getArea()+";"+"周长："+r2.getPerimeter()+";");

}

}

class Rectangle{

private double width,height;

private String color;

public Rectangle(){//默认矩形构造函数

width=1;

height=1;

color="WHITE";

}

public Rectangle(double w,double h){//指定宽高的构造函数

width=w;

height=h;

}

public double getwidth() {//获取宽

return width;

}

public void setwidth(double w) {//设置宽

this.width=w;

}

public double getheight() {//获取高

return height;

}

public void setheight(double h) {//设置高

this.height=h;

}

public String getcolor() {//获取颜色

return color;

}

public void setcolor(String c) {//设置颜色

this.color=c;

}

public double getArea() {//计算面积

return width\*height;

}

public double getPerimeter() {//计算周长

return 2\*(width+height);

}

}

//CreateTime.java

package project4;

import java.util.Calendar;

import java.util.GregorianCalendar;

public class CreateTime {

public static void main(String[] args) {

//创建两个时间对象

Time t1=new Time();

Time t2=new Time(555550000);

//显示两个时间对象的时分秒

System.out.println("第一个时间对象：");

System.out.println("时："+t1.gethour()+";"+"分："+t1.getminute()+";"

+"秒："+t1.getsecond()+";");

System.out.println("第二个时间对象：");

System.out.println("时："+t2.gethour()+";"+"分："+t2.getminute()+";"

+"秒："+t2.getsecond()+";");

}

}

class Time{

private int hour,minute,second;

public void setCurrentTime() {

Calendar calendar=new GregorianCalendar();

this.hour=calendar.get(Calendar.HOUR\_OF\_DAY);

this.minute=calendar.get(Calendar.MINUTE);

this.second=calendar.get(Calendar.SECOND);

}

public Time(){//默认构造函数

setCurrentTime();

}

public Time(int ms){//指定时间的构造函数

hour=((ms/1000)/3600)%24;

minute=(((ms/1000)%3600)/60)%60;

second=((ms/1000)%3600)%60;

}

public int gethour() {//获取时

return hour;

}

public int getminute() {//获取分

return minute;

}

public int getsecond() {//获取秒

return second;

}

}

# Project #5分析设计

## 5.1问题分析

### 5.1.1 目标

该程序旨在设计一个24点的小游戏，能判断玩家输入表达式正误，产生答案，并更新题目。

### 5.1.2功能

能够提供界面和函数，帮助玩家进行一个24点的小游戏。

### 5.1.3性能

#### 5.1.3.1响应输入性能

在玩家按下按钮后的0.4秒之内程序响应。

#### 5.1.3.2健壮性

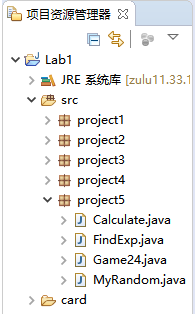
程序能够捕获输入异常。

#### 5.1.3.3异常输入处理

当玩家输入不构成表达式时，程序抛出异常并捕捉，在显示文本框中显示“Wrong!”。

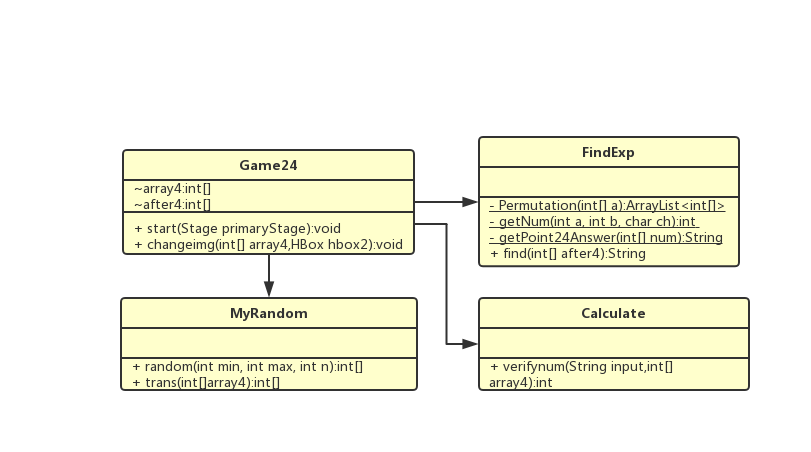
## 5.2设计方案

### 5.2.1模块划分



本项目包含四个模块，分别为：Calculate，FindExp，Game24以及MyRandom

### 5.2.2类图



### 5.2.3算法

#### 5.2.3.1 算法一：生成不重复的指定范围的指定个数随机数（部分代码参考网络）

public int[] random(int min, int max, int n){//参数分别为范围下界，上界，生成个数

if (n > (max - min + 1) || max < min) { //若参数不满足条件

return null;

}

int[] result = new int[n];

int count = 0;

while(count < n) {

int num = (int) (Math.random() \* (max - min)) + min;

boolean flag = true;

for (int j = 0; j < n; j++) {

if(num == result[j]){

flag = false;

break;

}

}

if(flag){

result[count] = num;

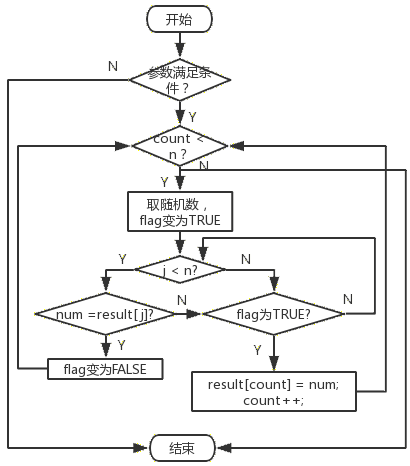
count++;

}

}

return result;

}



#### 5.2.3.2 算法二：检验输入的表达式是否用上了扑克的四个面值

public int verifynum(String input,int[] array4) {

int[] flag4= {0,0,0,0};

String result="";

int a=0,b=0,c=0,d=0;

//将字符串中的非数字字符转换为空格

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

if(Character.isDigit(input.charAt(i))){

result+=input.charAt(i);

}

else{

result+=' ';

}

}

String[] arrStr=result.trim().split("\\s+");//去掉前后空格后，再以空格分割字符串

a=Integer.parseInt(arrStr[0]);//分别将四个数(char)转换成int，赋值给a,b,c,d

b=Integer.parseInt(arrStr[1]);

c=Integer.parseInt(arrStr[2]);

d=Integer.parseInt(arrStr[3]);

for(int k=0;k<4;k++) {

if(a==array4[k]||b==array4[k]||c==array4[k]||d==array4[k])

flag4[k]=1;//若与任一面值匹配，flag置1

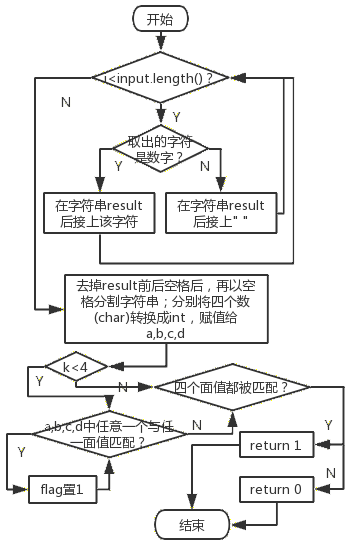
}

if(flag4[0]==1&&flag4[1]==1&&flag4[2]==1&&flag4[3]==1)//如果四个数都匹配

return 1;

else return 0;

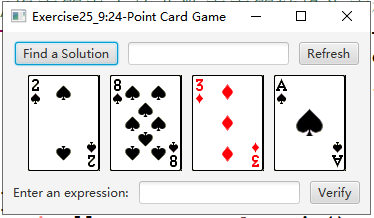
}



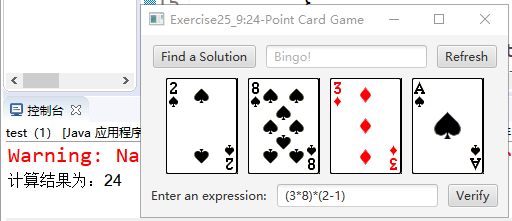
#### 5.2.3.3 算法三：生成24点表达式

(代码源于https://blog.csdn.net/qq\_33172274/article/details/83684595，详见源代码部分FindExp.java)

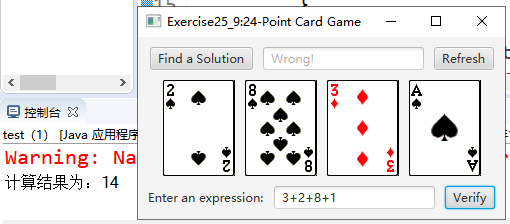
## 5.3实验结果



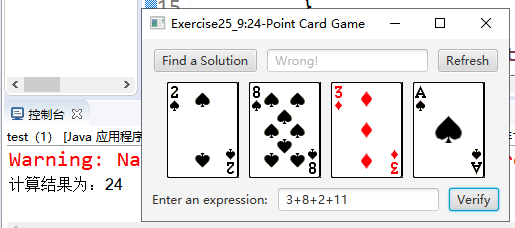
运行程序，javafx出现以上窗口，图中扑克牌图片为随机产生。



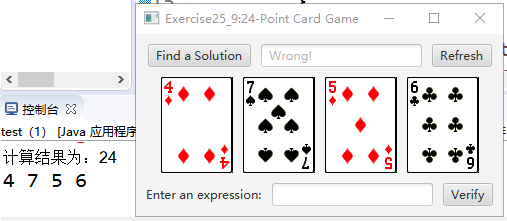
如上图，玩家利用扑克面值计算24点，在最下面一行的输入框中输入表达式。点击“Verify”按钮，最上面一行的显示框里立刻计算并显示该表达式的正误（正确为“Bingo!”；错误为“Wrong!”），并在控制台里打印出表达式的计算结果，方便玩家自查计算正误。



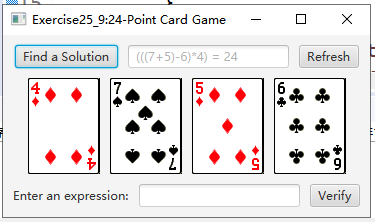
如上图，若玩家输入的表达式结果不等于24，显示框提示“Wrong!”。



如上图，若玩家输入的表达式没有用完四张扑克的面值，显示框提示“Wrong!”。



如上图，点击“Refresh”按钮，更新四张牌，输入框内容清空，控制台打印四张新的扑克面值，方便玩家计算。



如上图，点击“Find a Solution”按钮，最上面一行的显示框里立刻计算并显示一种答案表达式。



如上图，若该牌面组合没有答案，则显示“NO ANSWER!”。

## 5.4 调试心得

当使用javafx出现运行异常时，应该逐段注释掉节点相关代码，逐个运行，以检查出异常出现的大致段，再在该段中设置断点单步调试。

## 5.5源程序

//Game24.java

package project5;

import javax.script.ScriptEngine;

import javax.script.ScriptEngineManager;

import javafx.application.Application;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.control.\*;

import javafx.scene.image.Image;

import javafx.scene.image.ImageView;

import javafx.scene.layout.\*;

import javafx.stage.Stage;

public class Game24 extends Application{

int[] array4= {0,0,0,0};//存储随机数表示图片编号(1~52)

int[] after4= {0,0,0,0};//存储将图片编号转换后的扑克面值(1~13)

@Override

public void start(Stage primaryStage) {

//New 4 random numbers

MyRandom myrandom=new MyRandom();

array4=myrandom.random(1, 52, 4);//产生4个1~52的不重复随机数

after4=myrandom.trans(array4);//将随机数转换成扑克面值

//控制台输出四个面值便于计算

System.out.println(after4[0]+" "+after4[1]+" "+after4[2]+" "+after4[3]);

//UI

//Create 4 images

Image img1=new Image("file:card/"+array4[0]+".png");

Image img2=new Image("file:card/"+array4[1]+".png");

Image img3=new Image("file:card/"+array4[2]+".png");

Image img4=new Image("file:card/"+array4[3]+".png");

ImageView imgview1=new ImageView(img1);

ImageView imgview2=new ImageView(img2);

ImageView imgview3=new ImageView(img3);

ImageView imgview4=new ImageView(img4);

//Create buttons, TextFields, and Labels

Button bt1=new Button("Find a Solution");

Button bt2=new Button("Refresh");

Button bt3=new Button("Verify");

TextField tf1=new TextField();

TextField tf2=new TextField();

Label l=new Label("Enter an expression:");

//Create 3 HBox

HBox hbox1=new HBox();

HBox hbox2=new HBox();

HBox hbox3=new HBox();

//Place nodes into HBox

hbox1.getChildren().addAll(bt1,tf1,bt2);

hbox1.setSpacing(10);

hbox1.setAlignment(Pos.CENTER);

hbox2.getChildren().addAll(imgview1,imgview2,imgview3,imgview4);

hbox2.setSpacing(10);

hbox2.setAlignment(Pos.CENTER);

hbox3.getChildren().addAll(l,tf2,bt3);

hbox3.setSpacing(10);

hbox3.setAlignment(Pos.CENTER);

//Create a VBox and put 3 HBox in it

VBox vbox=new VBox();

vbox.setPadding(new Insets(10));

vbox.setSpacing(10);

vbox.setAlignment(Pos.CENTER);

vbox.getChildren().addAll(hbox1,hbox2,hbox3);

//Buttons' EventHandlers

bt2.setOnAction(e->{//更新按钮

array4=myrandom.random(1, 52, 4);//重新产生四个随机数

after4=myrandom.trans(array4);//重新转换成面值

changeimg(array4,hbox2);//更新图片

tf2.clear();//将上一题输入框2内容清除

});

bt1.setOnAction(e->{//Find a Solution按钮

FindExp findexp=new FindExp();

tf1.setPromptText(findexp.find(after4));//在文本框1中显示出找到的24点表达式

});

bt3.setOnAction(e->{//验证按钮

ScriptEngine jse=new ScriptEngineManager().getEngineByName("JavaScript");

String strs=tf2.getText();//获取输入表达式

try {

Calculate calculate=new Calculate();

int temp=Integer.parseInt(jse.eval(strs).toString());//将表达式转换成int并计算结果

//若表达式结果等于24，且每个扑克面额都用到了

if((temp==24)&&(calculate.verifynum(tf2.getText(),after4)==1))

tf1.setPromptText("Bingo!");

else tf1.setPromptText("Wrong!");

System.out.println("计算结果为："+jse.eval(strs));//控制台显示计算结果便于检查是否算错

}catch(Exception t) {//若输入不构成表达式

tf1.setPromptText("Wrong!");

}

});

//Create a scene and place it in stage

Scene scene=new Scene(vbox);

primaryStage.setTitle("Exercise25\_9:24-Point Card Game");

primaryStage.setScene(scene);

primaryStage.show();

}

public void changeimg(int[] array4,HBox hbox2) {//更新图片

hbox2.getChildren().clear();//将原有图片节点清空

//添加新图片

Image img1=new Image("file:card/"+array4[0]+".png");

Image img2=new Image("file:card/"+array4[1]+".png");

Image img3=new Image("file:card/"+array4[2]+".png");

Image img4=new Image("file:card/"+array4[3]+".png");

ImageView imgview1=new ImageView(img1);

ImageView imgview2=new ImageView(img2);

ImageView imgview3=new ImageView(img3);

ImageView imgview4=new ImageView(img4);

hbox2.getChildren().addAll(imgview1,imgview2,imgview3,imgview4);

//重新在控制台输出四个面值

System.out.println(after4[0]+" "+after4[1]+" "+after4[2]+" "+after4[3]);

return;

}

}

//MyRandom.java

package project5;

public class MyRandom{

//生成不重复的指定范围的指定个数随机数

public int[] random(int min, int max, int n){//参数分别为范围下界，上界，生成个数

if (n > (max - min + 1) || max < min) { //若参数不满足条件

return null;

}

int[] result = new int[n];

int count = 0;

while(count < n) {

int num = (int) (Math.random() \* (max - min)) + min;

boolean flag = true;

for (int j = 0; j < n; j++) {

if(num == result[j]){

flag = false;

break;

}

}

if(flag){

result[count] = num;

count++;

}

}

return result;

}

//将图片编号转换成扑克面值

public int[] trans(int[]array4) {

int[] after4= {0,0,0,0};

for(int i=0;i<4;i++) {

after4[i]=(array4[i]%13);//每种花色有13张

if(array4[i]%13==0) after4[i]=13;

}

return after4;

}

}

//FindExp.java

//代码源于https://blog.csdn.net/qq\_33172274/article/details/83684595

package project5;

import java.util.ArrayList;

public class FindExp {

//全排列

private static ArrayList<int[]> Permutation(int[] a) {

ArrayList<int[]> arrayList = new ArrayList<>();

for (int i = 0; i < 4; i++) {

for (int j = 0; j < 4; j++) {

if (i == j)

continue;

for (int k = 0; k < 4; k++) {

if (j == k || k == i)

continue;

for (int m = 0; m < 4; m++) {

if (m == k || m == j || m == i)

continue;

int[] b = new int[] { a[i], a[j], a[k], a[m] };

arrayList.add(b);

}

}

}

}

return arrayList;

}

//获取表达式中的数字

private static int getNum(int a, int b, char ch) {

switch (ch) {

case '+':

return a + b;

case '-':

return a - b;

case '\*':

return a \* b;

case '/':

return a / b;

}

return 0;

}

//生成24点表达式

private static String getPoint24Answer(int[] num) {

try {

char[] t = new char[] { '+', '-', '\*', '/' };

ArrayList<int[]> intlist = Permutation(num);

for (int z = 0; z < intlist.size(); z++) {

int a[] = intlist.get(z);

for (int i = 0; i < 4; i++) {

if ((i == 3 && (a[0] % a[1] != 0 || a[1] % a[0] != 0))) {

break;

}

int a12 = getNum(a[0], a[1], t[i]);

for (int j = 0; j < 4; j++) {

if ((j == 3 && (a12 % a[2] != 0 || a[2] % a12 != 0))) { break;

}

int a123 = getNum(a12, a[2], t[j]);

for (int k = 0; k < 4; k++) {

if ((k == 3 && (a123 % a[3] != 0 || a123 % a[3] != 0)))

break;

int a1234 = getNum(a123, a[3], t[k]);

if (a1234 == 24)

return "(((" + a[0] + t[i] + a[1] + ")" + t[j] + a[2] + ")" + t[k] + a[3] + ") = 24";

}

}

}

}

return "NO ANSWER!";//若无答案

}catch(Exception e) {

return "NO ANSWER!";//若无答案

}

}

//该类提供一个公有方法find()用来获得24点表达式

public String find(int[] after4) {

return getPoint24Answer(after4);//返回找到的表达式

}

}

//Calculate.java

package project5;

public class Calculate {

//检验输入的表达式是否用上了扑克的四个面值

public int verifynum(String input,int[] array4) {

int[] flag4= {0,0,0,0};

String result="";

int a=0,b=0,c=0,d=0;

//将字符串中的非数字字符转换为空格

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

if(Character.isDigit(input.charAt(i))){

result+=input.charAt(i);

}

else{

result+=' ';

}

}

String[] arrStr=result.trim().split("\\s+");//去掉前后空格后，再以空格分割字符串

a=Integer.parseInt(arrStr[0]);//分别将四个数(char)转换成int，赋值给a,b,c,d

b=Integer.parseInt(arrStr[1]);

c=Integer.parseInt(arrStr[2]);

d=Integer.parseInt(arrStr[3]);

for(int k=0;k<4;k++) {

if(a==array4[k]||b==array4[k]||c==array4[k]||d==array4[k])

flag4[k]=1;//若与任一面值匹配，flag置1

}

if(flag4[0]==1&&flag4[1]==1&&flag4[2]==1&&flag4[3]==1)//如果四个数都匹配

return 1;

else return 0;

}

}