**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（基础练习）**

**项目完成人：**

**姓名：\_\_王子妍\_\_\_\_\_\_**

**学号：\_\_2016211980\_\_\_\_\_\_**

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

**日 期： 2017 年 11 月 15 日**

1. **实验目的**

通过使用Java语言进行基本程序的开发，掌握Java通用IDE，练习类的封装使用、Java基本类库的使用、利用UML进行简单建模。

1. **实验内容**

全部内容在附录中展开描述。

1. **实验环境**
   1. Eclipse
   2. Windows 10
2. **实验结果**

四个实验都已完成，代码和思考结构见附录。

1. **附录**

**Lab1**

Submit the following items:

1. Analysis: Restate the problem in your own words (e.g., what is input, output if any, what needs to be computed, what data needs to be stored and their type).

随机生成两个int类型的整数，让用户从键盘上输入这两个数的和为string类型，转换为int类型，然后和正确的和进行比对，然后对他的答案进行判断。

1. Design: Clearly describe the steps to solve this problem using English or pseudo code.

Main

{

Int n1=random;

Int n2=random;

//get two random number by System.currentTimeMillis()%10

Print n1,n2 on the screen

Get a string from the user

Judge if it could parse into int , have a try block

If it could be a inteager;

Compare this with the right answer

Use the ? : to compare

And tell the user he is right or not by a dialog on screen

If catch couldn’t change into int

Remaind the user that he input a wrong answer.

}

1. Coding: Implement the solution in Java.

**package** firstlab;

**import** javax.swing.JOptionPane;

**public** **class** AMathLearningTool {

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

{

**int** number1=(**int**) (System.*currentTimeMillis*()%10);

**int** number2=(**int**) (System.*currentTimeMillis*()/3%10);

String input=JOptionPane.*showInputDialog*("What is "+number1+" + "+number2);

**try**

{

**int** Ans=Integer.*parseInt*(input);

**int** ans = number1+number2;

**boolean** c=(ans == Ans)?**true**:**false**;

**while**(c)

{

JOptionPane.*showMessageDialog*(**null**, number1+" + "+number2+" = "+Ans+" is true", "true", JOptionPane.***INFORMATION\_MESSAGE***);

**break**;

}

**while**(!c)

{

JOptionPane.*showMessageDialog*(**null**, number1+" + "+number2+" = "+Ans+" is wrong", "wrong", JOptionPane.***INFORMATION\_MESSAGE***);

**break**;

}

}

**catch**(Exception ex)

{

JOptionPane.*showMessageDialog*(**null**, "你太调皮啦，输入啦非法字符哦", "出错啦", JOptionPane.***ERROR\_MESSAGE***);

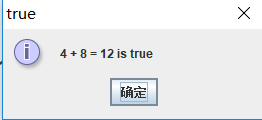
}

}

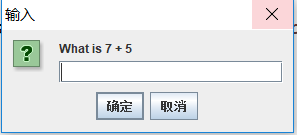
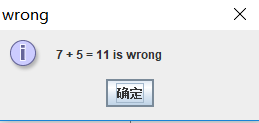
1. } Testing: Make sure that the numbers appear random. Submit the screen shots of two sample runs.

Example1:

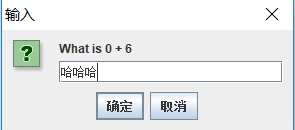


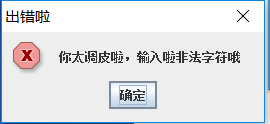


Example 2:

Example 3:





**Lab 2**

Submit the following items:

1. Analysis: Restate the problem in your own words (e.g., what is input, output if any, what needs to be computed, what data needs to be stored and their type).

计算在50000到60000之间的值所对应的tax table，分别计算每个类型的税，分为四个类型。然后以表格的形式，展示到屏幕上。每个类型用它的要求来实现。

1. Design: Clearly describe the steps to solve this problem using English or pseudo code.

1.give 4 methods to calculate the different condition of tax

2.in the main

3.first, print the name of tax in a line

4.then, call the 4 methods and get 4 double value and display them on the screen

5.and use the %15f or %15d to control the output

1. Coding: Implement the solution in Java.

**package** firstlab;

**public** **class** TaxTable {

**static** **double** single(**int** num)

{

**if**(num<=7300)

**return** num\*0.1;

**else** **if**(num<=29700)

**return** 730+0.15\*(num-7300);

**else** **if**(num<=59975)

**return** 4090+0.25\*(num-29700);

**else** **if**(num<=91400)

**return** 11658.75+0.28\*(num-59975);

**else** **if**(num<=163225)

**return** 20457.75+0.33\*(num-91400);

**else**

**return** 44160+0.35\*(num-163225);

}

**static** **double** marrage(**int** num)

{

**if**(num<=14600)

**return** num\*0.1;

**else** **if**(num<=59400)

**return** 1460+0.15\*(num-14600);

**else** **if**(num<=119950)

**return** 8180+0.25\*(num-59400);

**else** **if**(num<=182800)

**return** 23317.5+0.28\*(num-119950);

**else** **if**(num<=326450)

**return** 40915.5+0.33\*(num-182800);

**else**

**return** 88320+0.35\*(num-326450);

}

**static** **double** saparately(**int** num)

{

**if**(num<=7300)

**return** num\*0.1;

**else** **if**(num<=29700)

**return** 730+0.15\*(num-7300);

**else** **if**(num<=59975)

**return** 4090+0.25\*(num-29700);

**else** **if**(num<=91400)

**return** 11658.75+0.28\*(num-59975);

**else** **if**(num<=163225)

**return** 20457.75+0.33\*(num-91400);

**else**

**return** 44160+0.35\*(num-163225);

}

**static** **double** household(**int** num)

{

**if**(num<=10450)

**return** num\*0.1;

**else** **if**(num<=39800)

**return** 1045+0.15\*(num-10450);

**else** **if**(num<=102800)

**return** 5447.50+0.25\*(num-39800);

**else** **if**(num<=166450)

**return** 21197.50+0.28\*(num-102800);

**else** **if**(num<=326450)

**return** 39019.50+0.33\*(num-166450);

**else**

**return** 91819.50+0.35\*(num-326450);

}

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

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

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

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

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

System.***out***.println("Head of a House");

//一列15个字符

**for**(**int** i=50000;i<=60000;i=i+50)

{

System.***out***.printf("%15d",i);

**double** sin=*single*(i);

System.***out***.printf("%15f",sin);

**double** mar=*marrage*(i);

System.***out***.printf("%15f",mar);

**double** sep=*saparately*(i);

System.***out***.printf("%15f",sep);

**double** hos=*household*(i);

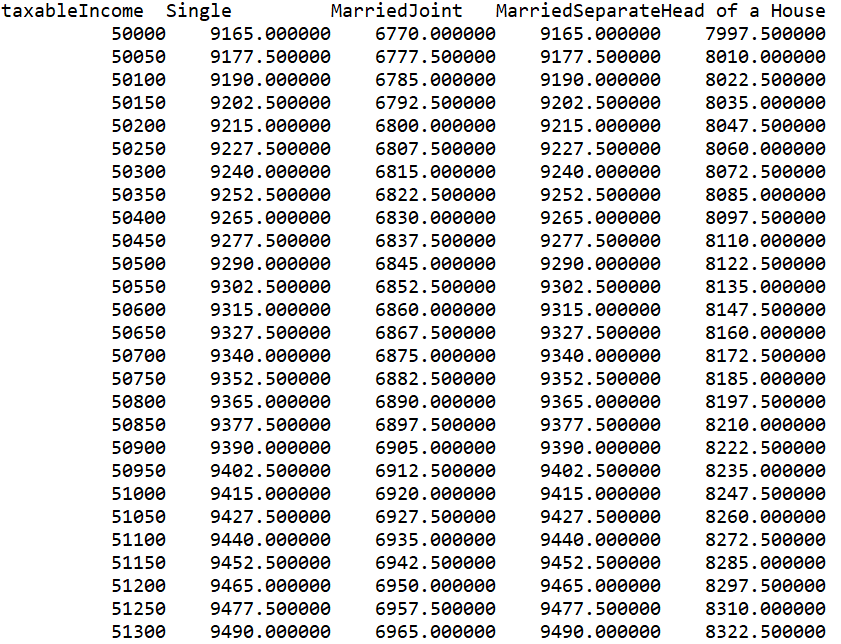
System.***out***.printf("%15f\n",hos);

}

}

}

1. Testing: Submit the screen shots for first 20 lines in the tax table.



**Lab 3:**

Submit the following items:

1. Analysis: Restate the problem in your own words (e.g., what is input, output if any, what needs to be computed, what data needs to be stored and their type).

输入两个数，输出它们的最小公倍数。主要是要判断他们的质因子，然后算最小公倍数。

1. Design: Clearly describe the steps to solve this problem using English or pseudo code. (Hint: The prime factor table can be represented using a two-dimensional array. Write a method named getPrimeFactors(int number) that returns a two-dimensional array for the prime factor table.)
2. decline a method to calculate some prime number and store in an array.
3. Use the method in 1 and write two methods to store the prime factors in a two-dimensional array.
4. In the main function, first call the method to calculate some prime number, then call the method to store the prime factors in a two-dimensional array. then, calculate the LCM.
5. Coding: Implement the solution in Java.

**package** firstlab;

**import** java.util.\*;

**public** **class** LeastCommonMultiple {

**static** **int**[] *pri* =**new** **int**[60];

**static** **int**[][] *factor1*=**new** **int**[51][3];

**static** **int**[][] *factor2*=**new** **int**[51][3];

**static** **void** prime()

{

**int** number=2;

**int** count=0;

**while**(count<50)

{

**boolean** isPrime=**true**;

**for**(**int** divisor=2;divisor<=number/2;divisor++)

{

**if**(number%divisor==0)

{

isPrime=**false**;

**break**;

}

}

**if**(isPrime)

{

*pri*[count]=number;

count++;

}

number++;

}

}

**static** **int** table1(**int** num)

{

**int** temp=0;

**int** a=0;

**int** count=0;

**while**(**true**)

{

**if**(num==0||num==1) **break**;

**if**(num%(*pri*[temp])==0)

{

a++;

**if**(a==1)

{

*factor1*[count][0]=*pri*[temp];

*factor1*[count][1]++;

count++;

}

**else** **if**(*factor1*[count-1][0]!=*pri*[temp])

{

*factor1*[count][0]=*pri*[temp];

*factor1*[count][1]++;

count++;

}

**else**

{

*factor1*[count-1][1]++;

}

num=num/(*pri*[temp]);

}

**else**

temp++;

}

**return** count;

}

**static** **int** table2(**int** num)

{

**int** temp=0;

**int** count=0;

**int** a=0;

**while**(**true**)

{

**if**(num==0||num==1) **break**;

**if**(num%(*pri*[temp])==0)

{

a++;

**if**(a==1)

{

*factor2*[count][0]=*pri*[temp];

*factor2*[count][1]++;

count++;

}

**else** **if**(*factor2*[count-1][0]!=*pri*[temp])

{

*factor2*[count][0]=*pri*[temp];

*factor2*[count][1]++;

count++;

}

**else**

*factor2*[count-1][1]++;

num=num/(*pri*[temp]);

}

**else**

temp++;

}

**return** count;

}

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

System.***out***.println("请输入需要进行取最大公约数的第一个数字 ");

Scanner input = **new** Scanner(System.***in***);

**int** number1=input.nextInt();

System.***out***.println("请输入需要进行取最大公约数的第二个数字 ");

**int** number2=input.nextInt();

*prime*();

**int** n1=*table1*(number1);

**int** n2=*table2*(number2);

**int** i=0,j=0;

**int** ans=1;

**while**(i<n1&&j<n2)

{

**if**(*factor1*[i][0]==*factor2*[j][0])

{

**int** temple=0;

**if**(*factor1*[i][1]>*factor2*[j][1])

{

temple=*factor1*[i][1];

}

**else**

temple=*factor2*[j][1];

**for**(**int** k=1;k<=temple;k++)

{

ans=ans\**factor1*[i][0];

}

i++;

j++;

}

**else** **if**(*factor1*[i][0]<*factor2*[j][0])

{

**for**(**int** k=1;k<=*factor1*[i][1];k++)

{

ans=ans\**factor1*[i][0];

}

i++;

}

**else** **if**(*factor1*[i][0]>*factor2*[j][0])

{

**for**(**int** k=1;k<=*factor2*[j][1];k++)

{

ans=ans\**factor2*[j][0];

}

j++;

}

}

**if**(i>=n1)

{

**while**(j<n2)

{

**for**(**int** k=1;k<=*factor2*[j][1];k++)

{

ans=ans\**factor2*[j][0];

}

j++;

}

}

**if**(j>=n2)

{

**while**(i<n1)

{

**for**(**int** k=1;k<=*factor1*[i][1];k++)

{

ans=ans\**factor1*[i][0];

}

i++;

}

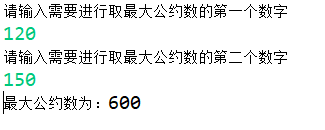
}

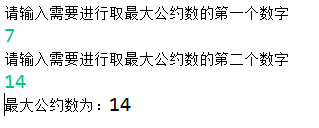
System.***out***.println("最大公约数为："+ans);

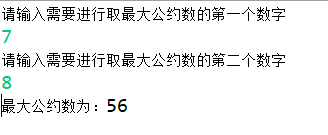
}

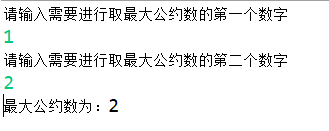
}

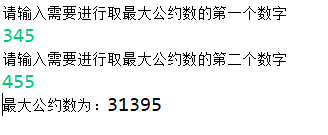
1. Testing: Test your program to find the LCM for (120, 150), (7, 14), (7, 8), (1, 2), and (345, 455)











**Lab4:**

Draw the UML diagram for the class. Implement the class.

Write a test program that creates two Rectangle objects. Assign width 4 and height 40 to the first object and width 3.5 and height 35.9 to the second object. Assign color red to all Rectangle objects. Display the properties of both objects and find their areas and perimeters.

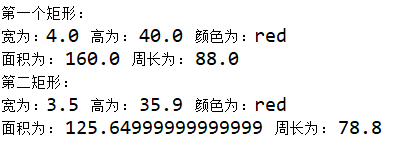
|  |
| --- |
| **Rectangle** |
| **Width: double**  **Height: double**  **Color: String** |
| **Rectangle ()**  **Rectangle (newWidth: double, newHeight: double)**  **getWidth():double**  **getHeight():double**  **getColor():String**  **setWidth( newWidth: double):void**  **setHeight( newHeight: double):void**  **setColor( newColor: String):void**  **getArea():double**  **getPerimeter():double** |

|  |
| --- |
| **Rec1:Rectangle** |
| **Width=4**  **Height=40**  **Color: “red”** |

|  |
| --- |
| **Rec2:Rectangle** |
| **Width=3.5**  **Height=35.9**  **Color: “red”** |

**Rectangle类：**

**屏幕显示：**



代码：

**package** firstlab;

**public** **class** TestRectangle {

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

{

Rectangle rec1=**new** Rectangle(4,40);

Rectangle rec2=**new** Rectangle(3.5,35.9);

rec1.setColor("red");

rec2.setColor("red");

System.***out***.println("第一个矩形：");

System.***out***.println("宽为："+rec1.getWidth()+" 高为： "+rec1.getHeight()+" 颜色为："+rec1.getColor());

System.***out***.println("面积为： "+rec1.getArea()+" 周长为： "+rec1.getPerimeter());

System.***out***.println("第二矩形：");

System.***out***.println("宽为："+rec2.getWidth()+" 高为： "+rec2.getHeight()+" 颜色为："+rec2.getColor());

System.***out***.println("面积为： "+rec2.getArea()+" 周长为： "+rec2.getPerimeter());

}

}

**class** Rectangle

{

**double** width=1;

**double** height=1;

String color="white";

Rectangle()

{}

Rectangle(**double** newWidth,**double** newHeight)

{

width=newWidth;

height=newHeight;

}

**double** getWidth()

{

**return** width;

}

**double** getHeight()

{

**return** height;

}

String getColor()

{

**return** color;

}

**void** setWidth(**double** newWidth)

{

width=newWidth;

}

**void** setHeight(**double** newHeight)

{

height=newHeight;

}

**void** setColor(String newColor)

{

color=newColor;

}

**double** getArea()

{

**return** width\*height;

}

**double** getPerimeter()

{

**return** 2\*(width+height);

}

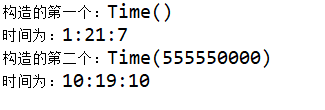
}

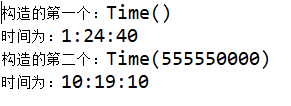
**Time类：**

Draw the UML diagram for the class. Implement the class. Write a test program that creates two Time objects (using new Time() and new Time(555550000)) and display their hour, minute, and second.

Hint: The current time can be obtained using System.currentTime(), as shown in Listing 2.8, ShowCurrentTime.java. The other constructor sets the hour, minute, and second for the specified elapse time. For example, if the elapse time is 555550000 milliseconds, the hour is 10, the minute is 19, and the second is 10.

**先展示两个测试结果图：**





|  |
| --- |
| **Time** |
| **Hour:long**  **Minute:long**  **Second:long** |
| **Time()**  **Time(TotalMilliseconds:long)**  **getHour():long**  **getMinute():long**  **getSecond():long** |

下面部分是代码实现：

**package** firstlab;

**public** **class** TestTime {

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

{

Time t1=**new** Time();

Time t2=**new** Time(555550000);

System.***out***.println("构造的第一个：Time()");

System.***out***.println("时间为："+t1.getHour()+":"+t1.getMinute()+":"+t1.getSecond());

System.***out***.println("构造的第二个：Time(555550000)");

System.***out***.println("时间为："+t2.getHour()+":"+t2.getMinute()+":"+t2.getSecond());

}

}

**class** Time

{

**private** **long** hour;

**private** **long** minute;

**private** **long** second;

Time()

{

**long** TotalMilliseconds=System.*currentTimeMillis*();

**long** TotalSeconds=TotalMilliseconds/1000;

**long** temp=TotalSeconds/(24\*3600);

TotalSeconds=TotalSeconds-temp\*(24\*3600);

hour=TotalSeconds/3600;

TotalSeconds=TotalSeconds-hour\*3600;

minute=TotalSeconds/60;

TotalSeconds=TotalSeconds-minute\*60;

second=TotalSeconds;

}

Time(**long** TotalMilliseconds)

{

**long** TotalSeconds=TotalMilliseconds/1000;

**long** temp=TotalSeconds/(24\*3600);

TotalSeconds=TotalSeconds-temp\*(24\*3600);

hour=TotalSeconds/3600;

TotalSeconds=TotalSeconds-hour\*3600;

minute=TotalSeconds/60;

TotalSeconds=TotalSeconds-minute\*60;

second=TotalSeconds;

}

**long** getHour()

{

**return** hour;

}

**long** getMinute()

{

**return** minute;

}

**long** getSecond()

{

**return** second;

}

}