实验一 熟悉开发环境

【实验目的】

在 Windows 系统环境中实现 JDK 的安装与配置。

熟悉并运行 Eclipse 软件。

【实验类别】

验证性

【实现指导】

1. 运行 Eclipse 软件，File -> New -> Java Project
   a) 输入工程名称，如 AppStructure01，点击 Finish
2. 选中项目下 Src 文件夹，右键选择 New -> Package
   a) 在 Name 一栏输入包名，如 com.oraclesdp.csg.hello
3. 选中 Src 文件夹下的包 com.oraclesdp.csg.hello，右键选择 New -> Class
   a) 在 Name 一栏输入类名
   b) 勾选 Public static void main(String[] args)一栏，点击 Finish

【实验内容】

1. 输出程序结果

```
class A
{
        void f()
        {
            System.out.println("I am A");
        }
}
class B
{
}
public class Hello
{
    public static void main (String args[ ])
    {
        System.out.println("你好，很高兴学习 Java");
        A a=new A();
```

```
            a.f();
        }
}
```

2. 输出程序结果

```
class Tom
  {
        int leg;
        String head;
        void cry(String s)
        {
            System.out.println(s);
        }
  }
  public    class Example
  {
      public  static void main(String args[])
      {
          Tom cat;
          cat=new Tom();
          cat.leg=4;
          cat.head="猫头";
          System.out.println("腿:"+cat.leg+"条");
          System.out.println("头:"+cat.head);
          cat.cry("我今天要和 Jerry 拼了");
      }
  }
```

实验二 Java 基本语法

【实验目的】

掌握 java 的基本语法。

进一步熟悉的 Eclipse 软件环境

【实验类别】

验证性

【实验内容】

1.熟悉输入和输出数据

```
import java.util.*;
public class Example
{
    public static void main (String args[ ])
    {
        Scanner reader=new Scanner(System.in);
        double a=0,b=0,c=0;
        System.out.println("输入边 a:");
        a=reader.nextDouble();
        System.out.println("输入边 b:");
        b=reader.nextDouble();
        System.out.println("输入边 c:");
        c=reader.nextDouble();
        if(a+b>c&&a+c>b&&b+c>a)
        {
          if(a*a==b*b+c*c||b*b==a*a+c*c||c*c==a*a+b*b)
          {
            System.out.printf("\n%-8.3f%-8.3f%-8.3f 构成是直角三角形
",a,b,c);
          }
        else if(a*a<b*b+c*c&&b*b<a*a+c*c&&c*c<a*a+b*b)
          {
              System.out.printf("\n%-8.3f%-8.3f%-8.3f 构成锐角三角形
```

```
",a,b,c);
                    }
                   else
                    {
                        System.out.printf("\n%-8.3f%-8.3f%-8.3f 构成钝角三角形
",a,b,c);
                    }
                }
               else
                {
                    System.out.printf("%f,%f,%f 不能构成三角形",a,b,c);
                }
            }
    }
```

2.：素数的判定

题目/任务

1. 判断 101-200 之间有多少个素数，并输出所有素数。

知识点说明/训练要点

1. 算术运算
2. 循环语句的使用
3. 关系运算

实现步骤

1. 构建 java 类
2. 构建判断是否为素数的方法
3. 循环判断 101~200 所有的数
4. 输出是素数的值

实现指导

1. 新建一个 java 文件并构建类和入口方法 main
2. 构建一个方法，判断数是否为素数
   a) 通过数值 n 对 2~n-1 所有数的余数进行判断，如果有一个余数为 0，则不是素数，返回 false

b) 如果余数全部不为 0，则数为素数，返回 true
3. 循环对 101~200 所有书进行遍历，判断是否为素数
4. 如果是素数，则通过控制台输出该数值

3：2000 年到 3000 年中的闰年

题目/任务

1. 求 2000~3000 年间的闰年
   a) 每四年一闰
   b) 百年不闰
   c) 四百年再闰

知识点说明/训练要点

1. 算术运算
2. 循环语句的使用
3. 逻辑运算

实现步骤

1. 构建 java 类
2. 构建判断是否为闰年的方法
3. 循环判断 2000~3000 所有的年份
4. 输出是闰年的值

实现指导

1. 新建一个 java 文件并构建类和入口方法 main
2. 构建一个方法，判断数是否为闰年
   a) 年份余 4 等于 0 并且年份余 100 不等于 0 的年份是闰年
   b) 年份余 400 等于 0 的年份是闰年
3. 循环对 2000~3000 所有书进行遍历，判断是否为闰年
4. 如果是闰年，则通过控制台输出该年份

4：水仙花数

题目/任务

1. 打印出所有的 "水仙花数 "，即一个三位数，其各位数字立方和等于该数本身

知识点说明/训练要点

1. 算术运算
2. If 条件块
3. 循环语句

实现步骤

1. 构建 java 类
2. 判断是否为水仙花数
3. 循环遍历
4. 输出结果

实现指导

1. 新建一个 java 文件并构建类和入口方法 main
2. 通过条件判断一个数是否为水仙花数
   a) 个位数：n%10
   b) 百位数：n/100
   c) 十位数：n/10%10
3. 循环遍历 100~999 所有三位数，判断是否为水仙花数
4. 使用 system.out.println 将结果输出到控制台

5：数列求和

题目/任务

1. 求 2/1，3/2，5/3，8/5，13/8，21/13...的和
2. 累加项数由用户输入决定

知识点说明/训练要点

1. 变量的定义与使用
2. 获取输入数据
3. 循环的嵌套使用
4. 语句输出

实现步骤

1. 构建 java 类
2. 定义所需变量
3. 获取用户输入整数
4. 实现获取分数分子分母的方法
5. 通过 for 循环遍历求和
6. 输出最终结果

实现指导

1. 新建一个 java 文件并构建类和入口方法 main
2. 定义所需要的变量
   a) 用于获取用户输入的 Scanner 变量
   b) 用于累加求和的实数 sum 变量
3. 通过 Scanner.nextInt 获取用户输入的整数
4. 通过递归获取第 n 项的分子和分母的方法
   a) 分子等于前一项的分子+分母
   b) 分母等于前一项的分子
5. 通过 for 循环获取用户输入长度的分式的值
6. 使用 system.out.println 将结果输出到控制台

6：函数

题目/任务

1. 当输入 n 为偶数时，调用函数求 1/2+1/4+...+1/n
2. 当输入 n 为奇数时， 调用函数求 1/1+1/3+...+1/n

知识点说明/训练要点

1. 变量的定义与使用
2. 获取输入数据
3. 循环的嵌套使用
4. 语句输出

实现步骤

1. 构建 java 类
2. 定义所需变量
3. 获取用户输入整数
4. 通过 for 实现偶数求和
5. 通过 for 实现奇数求和

实现指导

1. 新建一个 java 文件并构建类和入口方法 main
2. 定义所需要的变量
   a) 用于获取用户输入的 Scanner 变量
   b) 用于统计求和的实数 sum
3. 通过 Scanner.nextInt 获取用户输入的整数
4. 通过 for 循环实现偶数求和
   a) 分式求值的时候请先转换成实数，否则结果会是整数
5. 通过 for 循环实现奇数求和
   a) 分式求值的时候请先转换成实数，否则结果会是整数

实验三 类与对象

【实验目的】

掌握类和方法的定义，对象的创建和使用。

掌握引用的概念和引用赋值。

掌握方法重载，构造方法的作用及使用。

掌握包的概念和使用。

【实验类别】

设计性

【实验内容】

1：Rectangle 类的定义及使用。

定义一个名为 Rectangle 的类表示矩形，其中含有 length、width 两个 double 型的成员变量表示矩形的长和宽。编写一个 RectDemo 应用程序，在 main() 方法中创建一个矩形对象 rt，通过访问成员变量的方式为两个成员变量赋值，计算并输出它的面积。

修改 Rectangle 类，为每个变量定义访问方法和修改方法，定义求矩形周长的方法 perimeter()和求面积的方法 area()。修改 RectDemo 应用程序，在 main()方法中创建矩形对象后通过方法设置其长和宽，然后输出其周长和面积。

为 Rectangle 类编写一个带参数的构造方法，通过用户给出的长、宽创建矩形对象，再编写一个默认的构造方法，在该方法中调用有参数的构造方法，将矩形的长宽分别设置为 2 和 1。编写一个类测试这个矩形类。

2：Box 类的定义及使用。

定义一个名为 Box 的类，该类有三个 double 类型的成员变量，分别为 length, width 和 height，表示盒子的长、宽和高。编写一个应用程序 BoxDemo，创建一个名为 mybox 的 Box 对象，通过访问成员变量的方式为三个成员变量赋值，然后计算该盒子的体积并输出。

修改 Box 类，为其成员变量定义修改方法和访问方法，再定义一个 volume()方法用来求盒子的体积。用 BoxDemo 程序创建一个 Box 对象，测试这些方法的使用。

修改 Box 类，为该类定义一个带有三个 double 型参数的构造方法。假设该构造方法的声明格式为 Box(double length, double width, double height){ }，方法体应如何编写。然后在 BoxDemo 程序中创建一个长、宽、高分别为 10,20,15 的 Box 对象，输出该对象的体积。如果再用 new Box()创建一个对象会怎样，为什么？

3： 定义一个名为 Point 的类来模拟一个点，一个点可用 x 和 y 坐标描述。在定义的类中编写一个 main()方法，完成下面操作。

声明两个 Point 变量，start 和 end，将 start 点的坐标设置为（10,10），将 end 点的坐标设置为（20,30）。

使用输出语句分别打印 start 和 end 对象的 x 和 y 值，代码如下：

```
System.out.println("start.x="+start.x  +", strat.y="  + start.y);
```

System.out.println("end.x="+end.x +", end.y=" + end.y);

声明一个 Point 类型的名为 stray 的变量，将变量 end 的引用值赋予 stray，然后打印 end 和 stray 变量的成员 x 和 y 的值。

为 stray 变量的成员 x 和 y 指定新的值，然后打印 end 和 stray 的成员值。end 的值反映了 stray 内的变化，表明两个变量都引用了同一个 Point 对象；

为 start 变量的成员 x 和 y 指定新的值，然后打印 start 和 end 的成员值。再次编译并运行 Point 类，start 的值仍然独立于 stray 和 end 的值，表明 start 变量仍然在引用一个 Point 对象，而这个对象与 stray 和 end 引用的对象是不同的。

4： 设计一个银行账户类，其中包括：

账户信息：账号、姓名、开户时间、身份证号、账户余额等。

存款方法。

取款方法。

其他方法如“查询余额”和“显示账号”等。

编写应用程序模拟存款和取款过程。

5： 编写一个名为 Input 的类，该类属于 com.tools 包。使用该类实现各种数据类型（字符型除外）数据输入，其中的方法有 readInt()、readDoube()、readString()等。在用户程序中通过调用 Input.readDouble()即可从键盘上输入 double 型数据。例如，下面程序可以读入一个 double 型数据：

```
import com.tools.Input;

public class Test{

        public static void main(String[] args){

double d = Input.readDouble();

        System.out.println("d  = "+d);

}

}
```

提示：使用 java.util 包中的 Scanner 类实现。

实验四　继承

【实验目的】

1. 理解继承的概念，掌握如何定义一个类的子类，掌握成员变量的隐藏和方法的覆盖，掌握 this 和 super 关键字的使用。

2. 掌握访问修饰符的含义和使用。

【实验类别】

设计性

【实验内容】

实验题目 1：假定根据学生的 3 门学位课程的分数决定其是否可以拿到学位，对于本科生，如果 3 门课程的平均分数超过 60 分即表示通过，而对于研究生，则需要平均超过 80 分才能够通过。根据上述要求，请完成以下 Java 类的设计：

(1)设计一个基类 Student 描述学生的共同特征。

(2)设计一个描述本科生的类 Undergraduate，该类继承并扩展 Student 类。

(3)设计一个描述研究生的类 Graduate，该类继承并扩展 Student 类。

(4)设计一个测试类 StudentDemo，分别创建本科生和研究生这两个类的对象，并输出相关信息。

编写一个 Java Application 程序，创建 Student 类，UnderGraduate 类，Graduate 类，具体参照代码如下：**(填写缺少的代码部分)**

【代码 1】 //定义一个 pdsu 包名

```
class Student{
```

```
    private String name;
    private int classA,classB,classC;
    public Student(String  name,int classA,int classB,int classC){
        【代码 2】//赋初值
    }
    public String getName(){
        return  name;
    }
    public int getAverage(){
        return  (classA+classB+classC)/3;
    }
}
class UnderGraduate 【代码 3】 Student{// 将 UnderGraduate 定义为 Student 的子类
    public UnderGraduate(String  name,int classA,int classB,int classC){
        super(name,classA,classB,classC);
    }
    public void isPass(){
        if(【代码 4】)//找到本科生三门科平均成绩>=60 分的
            System.out.println(" 本 科 生 "+getName()+" 的 三 科 平 均 分 为 : "+getAverage()+",可以拿到学士学位。");
        else
            System.out.println(" 本 科 生 "+getName()+" 的 三 科 平 均 分 为 : "+getAverage()+",不能拿到学士学位。");
    }
}
class Graduate extends Student{
    public Graduate(String  name,int  classA,int classB,int classC){
        super(name,classA,classB,classC);
    }
    public void isPass(){
        if(getAverage()>=80)
        【代码 5】//将可以拿到士学位的研究生姓名，三科平均分输出。
                    else
            System.out.println(" 研 究 生 "+getName()+" 的 三 科 平 均 分 为 : "+getAverage()+",不能拿到硕士学位。");
    }
}
public class StudentDemo{
    public static void main(String[] args){
        UnderGraduate  s1=new UnderGraduate("Tom",55,75,81);
        Graduate s2=new Graduate("Mary",72,81,68);
          【代码 6】//找出能拿到学士学位的学生姓名，三科平均成绩。
          【代码 7】//找出能拿到硕士学位的学生姓名，三科平均成绩。
```

```
    }
}
```

运行结果：

本科生 Tom 的三科平均分为：70，可以拿到学士学位。
研究生 Mary 的三科平均分为：73，不能拿到硕士学位。

实验题目 2：理解类的继承。

（1）定义一个表示长方形的类 Rectangle，其中包含两个 private 的 double 型的成员变量 length 和 width 分别表示长方形的长和宽，定义一个有参数的构造方法 Rectangle(double length, double width)，在其中为长方形对象初始化，定义一个无参数的构造方法，在其中调用有参数的构造方法，使创建的对象的长和宽都为 0。再定义用来求长方形周长的方法 perimeter()和求面积的方法 area()。

（2）定义一个长方体类 Cuboid，使其继承 Rectangle 类，其中包含一个表示高的 double 型成员变量 height；定义一个构造方法 Cuboid(double length, double width, double height)；再定义一个求长方体表面积的方法 area(double height)和求体积的方法 volume()。

（3）编写一个名为 CuboidTest.java 的应用程序，求一个长、宽和高分别为 10、5、2 的长方体的体积。

实验题目 3：学生吃饭

（1）构建人物类并实现 eat 方法

（2）构建学生类，其为 final，不允许被继承，继承人物类并实现 eat 方法

（3）在 main 中创建学生类实例并调用 eat 方法

实验五 抽象

【实验目的】

1. 继续理解继承的概念，掌握如何定义一个抽象类的子类。

2. 理解 final 类和 abstract 类的含义，掌握如何定义和使用 final 类和 abstract 类。

3. 掌握对象的造型与多态的概念，掌握访问修饰符的含义和使用。

【实验类别】

设计性

【实验内容】

1. 定义一个名为 Triangle 的三角形类，使其继承 Shape 抽象类，覆盖 Shape 类中的抽象方法 perimeter()和 area()两个方法。编写一个应用程序测试 Triangle 类的使用。

抽象类 Shape 如下：

```
public abstract class Shape{
  private String name;
  public Shape(){}
  public Shape(String name){
     this.name = name;
  }
  public void setName(String name){
     this.name = name;
  }
  public String getName(){
      return name;
  }
```

```
    public abstract double perimeter();
    public abstract double area();
}
```

2．设计一个汽车抽象类 Auto，其中包含一个表示速度的 double 型的成员变量 speed 和表示启动的 start()方法、表示加速的 speedUp()方法以及表示停止的 stop()方法。再设计一个 Auto 类的子类 Bus 表示公共汽车，在 Bus 类中定义一个 int 型的表示乘客数的成员变量 passenger，另外定义两个方法 gotOn()和 gotOff()表示乘客上车和下车。编写一个应用程序测试 Bus 类的使用。

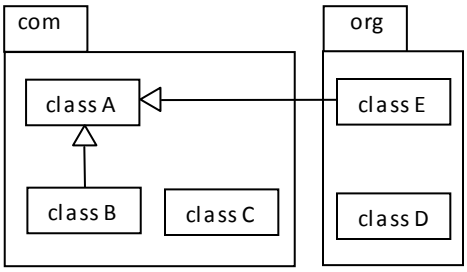
```
public abstract class Auto{
        double speed;
    public Auto(double speed) {
        super();
        this.speed = speed;
        System.out.println("车速"+this.speed);
    }
    public double getSpeed() {
        return speed;
    }
    public void setSpeed(double speed) {
        this.speed = speed;
    }
public abstract void start();
public abstract void speedUp(double speed);
public abstract void stop();
    }
```

运行结果如下，根据运行结果编写程序.

```
车速100.0
车上的人数是：5人
车子开始启动，速度是100.0kilo/h.
设置车速为60.0
车子加速到80.0kilo/h.
```

停车.
上车人数3　车上的人数是:8
车速:0.0人数:8
车子加速到 60.0kilo/h.

3．下图表示两个包 com 和 org，其中 A、B、C 类属于 com 包，D 和 E 类属于 org 包，箭头表示继承关系。



com 包和 org 包结构

假设 A 类的声明如下，请在表格中标出每个类对这些变量是否可见。

```
public class A{
    public int v1;
    private int v2;
    protected int v3;
    int v4;
}
```

完成改题的测试．提示：

```
package com;
public class A {
        public int v1;
        private int v2;
        protected int v3;
        int v4;
}

package org;
import com.A;
```

```
public class D {
	void test(){
		A a=new A();
		System.out.println("v1"+a.v1);
		System.out.println("v2"+a.v2);
		System.out.println("v3"+a.v3);
		System.out.println("v4"+a.v4);
	}
}
```

实验六 接口

【实验目的】

1. 了解接口的概念和实现机制。
2. 掌握接口的定义和实现。

【实验类别】

设计性

【实验内容】

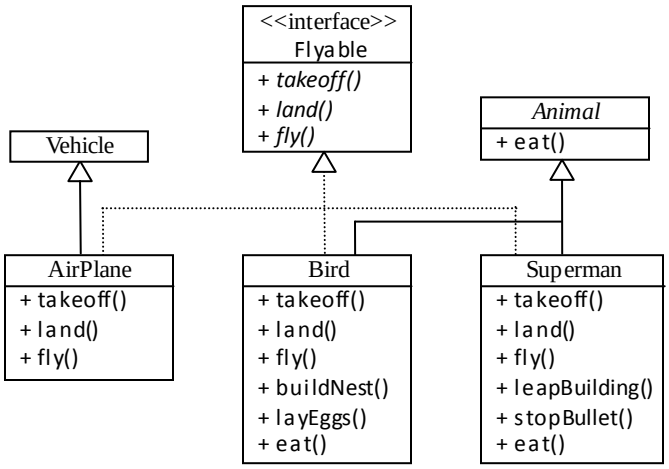
实验题目 1：输入下列程序，分析运行结果。

```
interface Ediable {
   public abstract String howToEat();
}

abstract class Food {
}

abstract class Fruit extends Food implements Ediable {
}

class Beef extends Food implements Ediable {
   public String howToEat() {
       return "Fried Beef Steak";
   }
}

class Mutton extends Food implements Ediable {
   public String howToEat() {
       return "Roast Mutton";
   }
}

class Apple extends Fruit {
   public String howToEat() {
       return "Make Apple Pie";
   }
}

class Orange extends Fruit {
   public String howToEat() {
       return "Make Orange Juice";
   }
}

public class EdibaleTest {
   public static void main(String args[]) {
```

```
        Object obj1 = new Beef();
        Object obj2 = new Mutton();
        Object obj3 = new Apple();
        Object obj4 = new Orange();
        System.out.println(((Ediable) (obj1)).howToEat());
        System.out.println(((Ediable) (obj2)).howToEat());
        System.out.println(((Ediable) (obj3)).howToEat());
        System.out.println(((Ediable) (obj4)).howToEat());
    }
}
```

实验题目２：

参照题目 1 完成下面ＵＭＬ图的设计和实现。



类和接口层次图

实验题目 3：

按以下要求编程

(1)编写 Animal 抽象类，类中声明 eat() 方法

(2)编写 SwimAble 接口，接口中声明 swim()方法。

(3)编写 FlyAble 接口，接口中声明 fly()方法。

(4)定义 Fish 类和 Bird 类继承 Animal 类,分别实现 Swimable 接口和 FlyAble 接口

(5) 实例化这个 Fish 类和 Bird 类，调用成员方法完成信息的输出。

实验七 异常处理

【实验目的】

1. 了解 Java 异常的概念和异常处理机制。
2. 掌握异常的类型和异常的处理方法。
3. 掌握自定义异常的编写方法。

【实验类别】

设计性

【实验内容】

实验题目 1：编写程序，在 main()方法中使用 try 块抛出一个 Exception 类的对象，为 Exception 的构造方法提供一个字符串参数，在 catch 块内捕获该异常并打印出字符串参数。添加一个 finally 块并打印一条消息。

实验题目 2：编写程序，定义一个 static 方法 methodA()，令其声明抛出一个 IOException 异常，再定义另一个 static 方法 methodB()，在该方法中调用 methodA()方法，在 main()方法中调用 methodB()方法。试编译该类，看编译器会报告什么？对于这种情况应如何处理？由此可得到什么结论？

实验题目 3：创建一个自定义的 MyException 异常类，该类继承 Exception 类，为该类写一个构造方法，该构造方法带一个 String 类型的参数。写一个方法，令其打印出保存下来的 String 对象。再编写一个类，在 main()方法中使用 try-catch 结构创建一个 MyException 类的对象并抛出，在 catch 块中捕获该异常并打印出传递的 String 消息。

实验题目 4：编写程序，要求从键盘输入一个 double 型的圆的半径，计算并输出其面积。测试当输入的数据不是 double 型数据（如字符串“abc”）会产生什么结果，怎样处理。提示：使用 Scanner 对象输入数据。如果类型不正确将抛出 java.util.InputMismatchException 异常。

实验八　输入和输出

【实验目的】

1. 了解和掌握 Path 类和 Files 类的使用。

2. 掌握字节输入/输出流类的使用，其中包括 InputStream、OutputStream 类，DataInputStream、DataOutputStream、BufferedInputStream、BufferedOutputStream 和 PrintStream 类。

3. 掌握字符输入/输出流类的使用，其中包括 InputStreamReader、OutputStreamWriter、PrintWriter 等类的使用。

4. 了解 ObjectInputStream、ObjectOutputStrema 类和 SeekableByteChannel 类的使用。

【实验类别】

设计性

【实验内容】

实验题目 1：DataInputStream 和 DataOutputStream 类的使用。

（1） 编写程序，随机生成 100 个 1000 到 2000 之间的整数，将它们写到一个文件 out.dat 中。写出数据要求使用 DataOutputStream 类的 writeInt(int i)方法。

（2）编写程序，要求从上题得到的 out.dat 中读出 100 个整数，程序中按照从小到大的顺序对这 100 个数排序，在屏幕上输出，同时写到同一个名为 sort.dat 的文件中。读出数据使用 DataInputStream 的 readInt()方法。

实验题目 2：PrintStream 类的使用。

随机产生 100 个 100 到 200 之间的整数，然后使用 PrintStream 对象输出到文件 output.txt 中。

实验题目 3：定义一个 Student 类，然后编写程序使用对象输出流将一个 Student 对象和一个字符串对象写入 student.dat 文件中，使用对象输入流读出对象。

实验九 集合与泛型（一）

【实验目的】

1. 掌握 Java 集合框架的接口和类的使用。
2. 重点掌握 ArrayList、HashSet、TreeSet、HashMap 和 TreeMap 类的使用。

【实验类别】

设计性

【实验内容】

实验题目 1：编写程序，将 5 个 Integer 对象存放到 ArrayList 对象中，然后按正序和倒序访问其中的每个元素。

实验题目 2：掌握集合元素的访问方法。

（1）编写程序，随机生成 20 个一位数，将它们分别添加到 HashSet 对象和 TreeSet 对象中。

（2）使用增强的 for 循环访问集合中的每个元素。

（3）使用 Iterator 迭代器访问集合中的每个元素。为什么集合中的元素不是 20 个？

实验题目 3：掌握对象顺序的实现。

（1）定义表示员工的 Employee 类，其中包括 empid,ename,eage,esalary 成员分别表示员工号,员工姓名,员工年龄和员工薪水，定义该类，类中有相应的方法。

（2）编写程序，创建 5 个员工对象，将它们存放到一个 HashMap 对象中，遍历每个员工键 empid，遍历每个员工 value，遍历每个员工映射 。

实验十 集合与泛型（二）

【实验目的】

1. 掌握集合中元素的访问方法，掌握对象顺序的概念及如何实现 Comparable 接口。
2. 了解 Arrays 类和 Connections 类的常用方法的使用。了解 Java 泛型的概念。

【实验类别】

设计性

【实验内容】

实验题目 1：编写一个类实现 Comparator 接口，使用该类对象实现 Student 对象按姓名顺序排序。

实验题目 2：

（1）定义表示员工的 Employee 类，其中包括 empid 和 ename 成员分别表示员工号和员工姓名，为该类定义一个带 2 个参数的构造方法。定义该类实现 Comparable 接口，实现其中的 compareTo()方法实现员工按员工号大小比较。

（2）编写程序，创建 5 个员工对象，将它们存放到一个 TreeSet 对象中，输出每个员工信息，看是否是按顺序存放的。

实验十一 多线程

【实验目的】

1. 掌握多线程的概念及线程实现的两种方法。
2. 掌握 Java 线程的状态及状态的改变。

【实验类别】

验证性

【实验内容】

实验题目 1：下面是一个多线程的程序：

```
public class SimpleThread extends Thread {
    public SimpleThread(String str) {
        super(str);
    }
    public void run() {
        for (int i = 0; i < 10; i++) {
            System.out.println(i + " " + getName());
            try {
                sleep((long)(Math.random() * 1000));
            } catch (InterruptedException e) {}
        }
        System.out.println("DONE! " + getName());
    }
}
public class TwoThreadsTest {
    public static void main (String[] args) {
        new SimpleThread("Jamaica").start();
        new SimpleThread("Fiji").start();
    }
}
```

（1）输入该程序并运行之，体会多线程的程序的编写方法。

（2）将该程序改为通过实现 Runnable 接口的方式实现多线程。

实验题目 2：编写程序，创建并运行二个线程：第一个线程打印 100 次字母 a，第二个线程打印 1~100 的整数。（用课上的三种形式完成）

方法一

```
class Thread_Test   implements Runnable{
    Thread t1,t2;
    Thread_Test(){
        t1=new Thread(this);   //实例化线程 t1
        t2=new Thread(this);    //实例化线程 t2
    }
    public void run(){
        if(Thread.currentThread()==t1)
```

```
            {for(int i=1; i<50;i=i+2)
            System.out.print(" "+i);
               System.out.print(Thread.currentThread().getName()+" 结束！ ");}
            else{for(int i=2; i<50;i=i+2)
                  System.out.print(" "+i);
            System.out.print(Thread.currentThread().getName()+" 结束！ ");}

}
}


public class odd {
      public static void main (String[] args) {
            Thread_Test tt=new Thread_Test();
            tt.t1.start();//启动线程
            tt.t2.start();
      }
}
```

方法二

```
class Thread_Test1    extends Thread{

      public void run(){

            for(int i=1; i<50;i=i+2)
            System.out.print(" "+i);
               System.out.print(Thread.currentThread().getName()+" 结束！ ");}


}


class Thread_Test2    extends Thread{

      public void run(){


            for(int i=2; i<50;i=i+2)
                  System.out.print(" "+i);
            System.out.print(Thread.currentThread().getName()+" 结束！ ");}

}
```

```
public class odd2 {
	public static void main (String[] args) {
		Thread t1=new Thread_Test1();
		Thread t2=new Thread_Test2();

		t1.start();//启动线程
		t2.start();
	}
}
```

方法三

```
class Tt1    extends Thread{

	public void run(){

		for(int i=1; i<50;i=i+2)
		System.out.print(" "+i);
		   System.out.print(Thread.currentThread().getName()+" 结束！ ");}


}

class Tt2    implements Runnable{

	public void run(){


		for(int i=2; i<50;i=i+2)
			System.out.print(" "+i);
		System.out.print(Thread.currentThread().getName()+" 结束！ ");}

}
public class odd3 {
	public static void main (String[] args) {
		Thread t1=new Tt1();

		Runnable r=new Tt2();
		Thread t2=new Thread(r);
		t1.start();//启动线程
		t2.start();
	}
}
```

实验十二 GUI

【实验目的】

1. 掌握容器的布局方法及容器布局管理器。

2. 掌握使用中间容器 JPanel 构建复杂界面的方法。

3. 了解 Java 事件处理模型，掌握事件处理的步骤。

4.掌握 JLabel、JButton、JTextField、JTextArea、JMenu 等常用组件的使用方法。

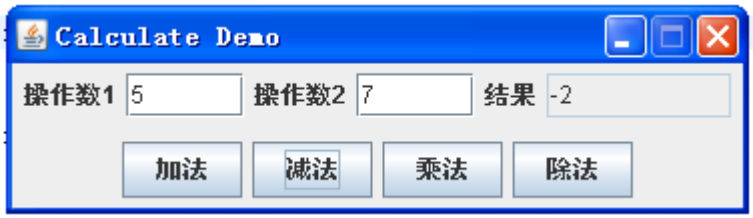
【实验类别】

设计性

【实验内容】

实验题目 1：编写程序，实现如图所示的界面。要求在文本框中输入有关信息，点击“确定”按钮，在下面的文本区域中显示信息，点击“清除”按钮将文本框中的数据清除。

实验题目 2：编写一个如图所示的简单计算器程序，实现数据的加减乘除功能。


Calculate Demo
操作数1
5
操作数2
7
结果
-2
加法
减法
乘法
除法