Java 语言程序设计 B 实验指导书

刘永磊

电子与信息工程系网络工程教研室

2010-11-18

实验一　面向对象

一、实验目的

1．掌握继承、多态的概念与实现方法；

2．掌握包和接口的定义和使用方法；

3．掌握封装性，多态性的概念；

4．理解 Java 语言是如何体现面向对象编程基本思想。

二、实验要求

1．能实现类的继承关系；

2．用多种方法创建各个类的对象；

3．程序应包括各个被调用方法的执行结果的显示；

4．写出实验报告。要求记录编译和执行 Java 程序当中的系统错误信息提示，并给出解决办法（附运行界面、源代码）。

三、实验环境

1．计算机一台；

2．JDK、Eclipse 工具软件。

四、实验内容

1．分别编写两个类 Point2D，Point3D 来表示二维空间和三维空间的点，使之满足下列要求：

1) Point2D 有两个整型成员变量 x, y（分别为二维空间的 X,Y 方向坐标），Point2D 的构造方法要实现对其成员变量 x, y 的初始化。

2)Point2D 有一个 void 型成员方法 offset(int a, int b)，它可以实现 Point2D 的平移。

3)Point3D 是 Point2D 的直接子类，它有有三个整型成员变量 x,y,z（分别为三维空间的 X,Y,Z 方向坐标），Point3D 有两个构造方法：Point3D(int x,int y,int z)和 Point3D(Point2D p,int z)，两者均可实现对 Point3D 的成员变量 x, y,z 的初始化。

4)Point3D 有一个 void 型成员方法 offset(int a, int b,int c)，该方法可以实现 Point3D 的平移。

5)在 Point3D 中的主函数 main()中实例化两个 Point2D 的对象 p2d1，p2d2，打印出它们之间的距离，再实例化两个 Point2D 的对象 p3d1，p3d2，打印出他们之间的距离。

2．定义抽象类 Shape，抽象方法为 showArea()，求出面积并显示，定义矩形类 Rectangle,正方形类 Square,圆类 Circle，根据各自的属性，用 showArea 方法求出各自的面积，在 main 方法中构造 3 个对象，调用 showArea 方法。

定义接口 DiagArea，其中包含方法 double getDiagonal()求对角线长, double getArea()求面积，定义一个矩形类，实现此接口，并自行扩充成员变量和方法，定义一个正方形类继承矩形类（如矩形有长 w 和宽 h，正方形有边 x，并有相应的构造函数，有一个方法中一次直接显示边长、面积和对角线长）,在另一类中的主方法里使用测试该类。

五、实验步骤

1.（第 1 题)定义 Point2D，及定义它的属性和方法；

定义子类Point3D,及定义它的属性和方法;在Point3D中的主函数main()中实例化两个Point2D的对象，并通过这两个对象调用它们的属性和方法，输出方法执行结果。

2.（第 2 题)定义抽象类 Shape，抽象方法为 showArea()，再定义矩形类 Rectangle,正方形类 Square,圆类 Circle,和各自的属性。定义主类、主方法,在 main 方法中构造 3 个对象,调用 showArea 方法；定义接口 DiagArea，其中包含方法 double getDiagonal()，在主 main 方法中输出方法执行结果。

实验二　集合类

一、实验目的

1．掌握 Collection 接口，List 集合，List 接口，List 接口的实现类，Set 集合

2．掌握 Map 集合，Map 接口，Map 接口的实现类

二、实验要求

1．认真分析以上程序，给出运行结果；

2．正确使用集合类。

三、实验环境

1．计算机一台；

2．JDK、Eclipse 工具软件。

四、实验内容

1．分析、运行以下程序

该例示范了 ArrayList 的使用　先声明了一 String 类型的数组，里面存储了“颜色”，是用字符串写出的颜色，将这个字符串数组存入 ArrayList 实例，同时还存入了 awt 包内的颜色实例，全部存入后利用迭代，删除不符要求的假数据，也就是我们用字符串写的颜色，也用到了 instanceof 它是一个二元操作符，类似于 equals 用于判断 instanceof 左边的对象是否是右边对象的实例，若是返回真，这里 就可以判断 ArrayList 里面的真假颜色，假颜色是字符串的实例，所以我们通过迭代一个个对比。只要是 String 的实例就将其从数组中删 除，所以最后 ArrayList 里面仅仅剩下二个元素，运行效果如下：

```
import java.awt.*;
import java.util.*;
public class CollectionTest
{//List 是一个能包含重复元素的已排序的 Collection,有时 list 也称为序列，List 第一个元素的下标为 0
    public String colors[]={"red","white","blue"};//定义一个字符数组

    //构造函数
    public CollectionTest()
    {
        ArrayList list=new ArrayList();//实例化一个 ArrayList
        list.add(Color.magenta);//向里面添加一个元素，这里是颜色

        for(int count=0;count<colors.length;count++)
            list.add(colors[count]);//加入开始声明的数组中的元素
```

```
            list.add(Color.cyan);      //颜色导入 awt 包
            System.out.println("\nArrayList");
            for(int count=0;count<list.size();count++)
               System.out.println(list.get(count)+" ");//从 arrayList 中读取  元素

            removeString(list);
            System.out.println("\n\nArrayList after calling"+"removeString:");
            for(int count=0;count<list.size();count++)
                   System.out.println(list.get(count)+" ");
        }


           public void removeString(Collection collection)
           {
                Iterator itrator=collection.iterator();    //声明一个迭代
                //调用 itrator 的 hasNext 方法判断 Collection 是否还包含元素
                while(itrator.hasNext())
                {
                     //调用 itrator 的 next 方法获得下一个元素的引用
                     if( itrator.next() instanceof String ) // instanceof 判断是否是
String 的实例
                           itrator.remove();    //如果是的则删除
                }
           }

        public  static void main(String[] args)
        {
             new CollectionTest();
        }

    }
```

2. 分析运行以下程序

//Set 是包含独一无二元素的 Collection，HashSet 把它的元素存储在哈希表中，而 TreeSet 把它的元素存储在树中

```
import java.util.*;

public class SetTest
{
     private String colors[]={"orange","tan","orange","white",  "gray"};
     public SetTest()
     {
               ArrayList list;
               list=new ArrayList(Arrays.asList(colors));
               System.out.println("ArrayList:"+list);
               printNonDuplicates(list);
     }


     public void printNonDuplicates(Collection collection)
     {
          //构造 HashSet 删除 Collection 中多余的元素
          HashSet set=new HashSet(collection);
          //将 coolection 放入 HashSet 后即会消除重复元素
          System.out.println("set: "+set);

          Iterator itrator=set.iterator();
          System.out.println("\nNonDuplicates are:");
          while(itrator.hasNext())
          System.out.println(itrator.next()+" ");
          System.out.println();
     }

     public static void main(String[] args)
     {
            new SetTest();
     }

}
```

3. 分析运行以下程序

```
import java.util.HashSet;
import java.util.Iterator;
import java.util.Set;

class TestSet
{
     public static void main(String args[])
     {
         Set set = new HashSet();
         set.add("aaa");
         set.add("bbb");
         set.add("aaa");//后面加入的重复性元素均无效
         set.add("bbb");
         set.add("aaa");
         set.add("bbb");
         set.add("aaa");
         set.add("bbb");
         set.add("aaa");
         set.add("bbb");
         Iterator ite=set.iterator();
         System.out.println(set.size());//the result is 2
         while(ite.hasNext())
         {
             System.out.println("----"+ite.next());
          }
     }
}
```

4. 分析运行以下程序

```
package tt;

import java.util.Arrays;
import java.util.Collections;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;
```

```
public class ListTest {

    public static void baseUse(){
            //链表实现
            List list = new LinkedList();
            //数组实现
            //List list = new ArrayList();
            list.add("a");//向列表的尾部追加"a"
            System.out.println("使用 list 接口的 add()一个参数的方法："+list);
            list.add(0,"b");//在指定位置插入"b"
            System.out.println("使用 list 接口的 add 二个参数的方法："+list);
            list.remove("a");//移除列表中"a"
            System.out.println("使用 list 接口的 remove()方法删除 a："+list);
     }
      public static void useSort(){
            String[] strArray = new String[] {"z", "a", "c","C"};
            List list = Arrays.asList(strArray);
            System.out.println(list);
            Collections.sort(list);//根据元素自然顺序排序
            System.out.println("自然顺序："+list);
            Collections.sort(list, String.CASE_INSENSITIVE_ORDER);//根据指定的字母
方式排序
            System.out.println("指定字母方式："+list);
            Collections.sort(list, Collections.reverseOrder());//根据反转自然顺序
方式排序
            System.out.println("反转自然顺序："+list);
            Collections.sort(list, String.CASE_INSENSITIVE_ORDER);
            System.out.println(list);
            Collections.reverse(list);//反转列表排序
            System.out.println(list);
      }

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

```
        baseUse();
    //    useSort();
    }

}
```

5. 分析运行以下程序

```
package tt;

import java.util.Iterator;
import java.util.TreeSet;

public class TreeSetTest {

     public static void main(String args[]){
          TreeSet a = new TreeSet();
          a.add("1167014513046,hondanna_mousepress");
          a.add("1167014512046,hondanna_mousepress_usefull");
          a.add("1167014511046,hondanna_mousepress_num");
          a.add("1167014515437,hondanna_mousepress");
          a.add("1167014514438,hondanna_mousepress_usefull");
         Iterator iterator = a.iterator();
         while(iterator.hasNext())
             System.out.println(iterator.next());
         }
}
```

6. 分析运行以下程序

```
package com.mwq;
import java.util.HashMap;
import java.util.Map;
public class TestCollection {
public static void main(String[] args) {
System.out.println("开始：");
Map<Integer, String> map = new HashMap<Integer, String>();
map.put(220180, null);
map.put(220181, "马先生");
```

```
System.out.println("get()方法的返回结果：");
System.out.print("------ " + map.get(220180));
System.out.print("    " + map.get(220181));
System.out.println("    " + map.get(220182));
System.out.println("containsKey()方法的返回结果：");
System.out.print("------ " + map.containsKey(220180));
System.out.print("    " + map.containsKey(220181));
System.out.println("    " + map.containsKey(220182));
System.out.println("结束！");
}
}
```

7. 分析运行以下程序

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import java.util.TreeMap;

public class HashMain {
 /**
 * @param args
 */
public static void main(String[] args) {
Map map=new HashMap();
map.put("a", "aaa");
map.put("b", "bbb");
map.put("c", "ccc");
map.put("d", "ddd");

Iterator iter=map.keySet().iterator();
while(iter.hasNext())
{
Object key=iter.next();
System.out.println("map key "+key.toString()+" value=---"+map.get(key));
}
TreeMap tab=new TreeMap();
```

```
tab.put("a", "aaa");
tab.put("b", "bbb");
tab.put("c", "ccc");
tab.put("d", "ddd");

Iterator iter2=tab.keySet().iterator();
while(iter2.hasNext())
{
Object key=iter2.next();
System.out.println("tab key "+key.toString()+" value=---"+tab.get(key));
}
}
}
```

五、实验步骤

在 Eclipse 环境下运行以上程序，给出运行结果。

实验三　输入/输出

一、实验目的

1．理解流的概念，流的划分；

2．掌握文件描述，顺序处理，随机访问处理的方法；

3．能够熟练的使用过滤流；掌握字符流处理的方法；

4．理解对象串行化的概念和方法。

二、实验要求

1．验证以上程序；

2．给出运行结果。

三、实验环境

1．计算机一台；

2．JDK、Eclipse 工具软件。

四、实验内容

1．使用标准数据流的应用程序。标准数据流指在字符方式下（如 DOS 提示符）程序与系统进行输入输出的方式，键盘和显示器屏幕是标准输入输出设备，数据输入的起点为键盘，数据输出的终点是屏幕，输出的数据可以在屏幕上显示出来。程序功能：将键盘上输入的字符在屏幕上显示出来。

```
class LX5_3{
public static void main(String[] args) throws java.io.IOException {
byte buffer[]=new byte[10];
System.out.println("从键盘输入不超过 10 个字符，按回车键结束输入：");
int count =System.in.read(buffer);//读取输入的字符并存放在缓冲区 buffer 中
System.out.println("保存在缓冲区 buffer 中元素的个数为："+count);
System.out.println("buffer 中各元素的值为：");
for (int i=0;i<count;i++){
System.out.print(" "+ buffer[i]);//在屏幕上显示 buffer 元素的值
}
System.out.println();
System.out.println("输出 buffer 字符元素：");
System.out.write(buffer, 0, buffer.length);
}
}
```

2．使用文件输入输出流的应用程序。将保存在本地机当前文件夹中的 LX5_1.HTML 文本文件的内容在屏幕上显示出来，然后将其另存为 LX5_1.txt 文件。

```
import java.io.*;
public class LX5_4 {
public static void main(String[] args) throws IOException {
FileReader in=new FileReader("LX5_1.HTML");//建立文件输入流
BufferedReader bin=new BufferedReader(in);//建立缓冲输入流
FileWriter out=new FileWriter(" LX5_1.txt",true);//建立文件输出流
String str;
while ((str=bin.readLine())!=null) {
//将缓冲区内容通过循环方式逐行赋值给字符串 str
System.out.println(str);//在屏幕上显示字符串 str
out.write(str+"\n");//将字符串 str 通过输出流写入 LX5_1.txt 中
}
in.close();
out.close();
}
}
```

3. 使用随机文件类的应用程序。使用文件输入类 FileReader 只能将文件内容全部读入。如果要选择读入文件的内容，可使用随机文件类 RandomAccessFile。建立数据流，通过指针有选择的读入文件内容。

```
import java.io.*;
class LX5_5 {
public static void main(String args[]) {
String str[]={"First line\n","Second line\n","Last line\n"};
try {
RandomAccessFile rf=new RandomAccessFile("LX5_5.txt", "rw");
System.out.println("\n 文件指针位置为："+rf.getFilePointer());
System.out.println("文件的长度为："+rf.length());
rf.seek(rf.length());
System.out.println("文件指针现在的位置为："+rf.getFilePointer());
for (int i=0; i<3; i++)
rf.writeChars(str[i]); // 字符串转为字节串添加到文件末尾
rf.seek(10);
System.out.println("\n 选择显示的文件内容：");
String s;
while ((s=rf.readLine())!=null)
```

```
System.out.println(s);
rf.close();
}
catch (FileNotFoundException fnoe) {}
catch (IOException ioe) {}
}
}
```

4.使用数据输入输出流与文件输入输出流类的应用程序。使用数据输出流 DataOutputStream 和数据输入流 DataInputStream 可以读取或写入任何 Java 类型的数据，不用关心它们的实际长度是多少字节。一般与文件输入流 FileInputStream 和输出流类 FileOutputStream 一起使用。将整型数据和字符串对象通过数据输出流写到文件中。将文件中的整型数据和字符串对象通过数据输出流读出，并在屏幕上显示文件中的内容。

```
import java.io.*;
public class LX5_6{
public static void main(String arg[]){
try
{ //添加方式创建文件输出流
FileOutputStream fout = new FileOutputStream("LX5_6.txt",true);
DataOutputStream dout = new DataOutputStream(fout);
dout.writeInt(1);
dout.writeChars("罗马"+"\n");
dout.writeInt(2);
dout.writeChars("北京"+"\n");
dout.close();
}catch (IOException ioe){}
try{
FileInputStream fin = new FileInputStream("LX5_6.txt");
DataInputStream din = new DataInputStream(fin);
int i = din.readInt();
while (i!=-1) {
 //输入流未结束时，输入流结束时 i 为-1
System.out.print(i+" ");
char ch ;
while ((ch=din.readChar())!='\n') //字符串未结束时
System.out.print(ch);
```

```
System.out.println();
i = din.readInt();
}
din.close();
}catch (IOException ioe){}
}
}
```

5. 使用对象输入输出流的应用程序。使用对象流可以直接写入或读取一个对象。由于一个类的对象包含多种信息，为了保证从对象流中能够读取到正确的对象，因此要求所有写入对象流的对象都必须是序列化的对象。一个类如果实现了 Serializable 接口，那么这个类的对象就是序列化的对象。Serializable 接口没有方法，实现该接口的类不需要实现额外的方法。程序功能：保存对象信息到文件，并将文件中的对象信息显示出来。

```
import java.io.*;
public class LX5_7 implements Serializable //序列化接口
{
int bh=1;int nl=21;
String xm;
LX5_7(int bh,String xm,int nl)//构造方法
{
this.bh = bh;
this.xm = xm;
this.nl = nl;
}
LX5_7()//构造方法
{
this(0,"",21);
}
void save(String fname)//保存到文件中的方法
{
try
{
FileOutputStream fout = new FileOutputStream(fname);//输出文件流
ObjectOutputStream out = new ObjectOutputStream(fout);//输出对象流
out.writeObject(this); //写入对象
out.close();
```

```
}catch (FileNotFoundException fe){}
catch (IOException ioe){}
}
void display(String fname)//显示文件中对象信息的方法
{
try{
FileInputStream fin = new FileInputStream(fname);//输入文件流
ObjectInputStream in = new ObjectInputStream(fin);//输入对象流
LX5_7 OO = (LX5_7)in.readObject(); //读取对象
System.out.println(" 类名： "+OO.getClass().getName()+"
"+OO.getClass().getInterfaces()[0]);
System.out.println(" "+OO.bh+" "+OO.xm+" "+OO.nl);
in.close();
}
catch (FileNotFoundException fe){}
catch (IOException ioe){}
catch (ClassNotFoundException ioe) {}
}
public static void main(String arg[])
{
String fname = "LX5_7.obj";
LX5_7 O1= new LX5_7(1,"张驰",14);
O1.save(fname);
O1.display(fname);
}
}
```

五、实验步骤

在 Eclipse 环境下编写 java 程序，验证以上程序，给出运行结果。

实验四　Java 图形界面

一、实验目的

1．掌握 Java 图形组件和布局管理器的使用；

2．掌握使用 Java 事件处理机制的使用；

3．掌握图形界面的各种控件的使用，如：标签、文本框、按钮、复选框、列表框、窗框等。

二、实验要求

1．采用布局管理器进行界面的布局；

2．学会对不同的事件用相应的事件处理器；

3．写出实验报告。要求记录编译和执行 Java 程序当中的系统错误信息提材示，并给出解决办法。

三、实验环境

1．计算机一台；

2．JDK、Eclipse 工具软件。

四、实验内容

1．在窗口中添加组件

```
import java.awt.*;
import java.awt.event.*;
public class LX6_6 extends Frame implements ActionListener {
Button btn1, btn2;
TextField f,tf1,tf2;
TextArea Area;
LX6_6() {
super("添加组件的窗口");
addWindowListener(new WindowAdapter() {
public void windowClosing(WindowEvent e) {
System.exit(0);
}
});
setSize(350,250); //设置窗口大小
setLocation(200,200);//设置窗口显示位置
setFont(new Font("Arial",Font.PLAIN,12)); //设置字体
setLayout(new FlowLayout());
Area=new TextArea (6,40);
tf1=new TextField(10); tf2=new TextField(10);
```

```
btn1=new Button("显示"); btn2=new Button("退出");
f=new TextField(20);
add(Area); add(new Label("用户名"));
add(tf1); add(new Label("电话"));
add(tf2); add(f); add(btn1); add(btn2);
tf1.addActionListener(this); tf2.addActionListener(this);
btn1.addActionListener(this); btn2.addActionListener(this);
show();
}
public static void main(String args[]) {
new LX6_6();
}
public void actionPerformed(ActionEvent e) {
if (e.getSource()==btn1)
f.setText("你按下了“" + e.getActionCommand() + "”按钮");
if (e.getSource()==tf1)
Area.append("用户名："+tf1.getText()+"\n");
if (e.getSource()==tf2)
Area.append("电 话："+tf2.getText()+"\n");
if (e.getSource()==btn2) {
for (int i=0; i<100000000; i++);
dispose();//只关闭当前窗口,注销该对象
}
}
}
```

2. 为窗口添加菜单

```
import java.awt.*;
import java.awt.event.*;
public class LX6_7 extends Frame implements ActionListener {
Panel p=new Panel();
Button b=new Button("退出");
MenuBar mb=new MenuBar(); // 以下生成菜单组件对象
Menu m1=new Menu("文件");
MenuItem open=new MenuItem("打开");
MenuItem close=new MenuItem("关闭");
```

```
MenuItem exit=new MenuItem("退出");
Menu m12=new Menu("编辑");
MenuItem copy=new MenuItem("复制");
MenuItem cut=new MenuItem("剪切");
MenuItem paste=new MenuItem("粘贴");
Menu m2=new Menu("帮助");
MenuItem content=new MenuItem("目录");
MenuItem index=new MenuItem("索引");
MenuItem about=new MenuItem("关于");
LX6_7() {
super("添加菜单的窗口");
setSize(350,200);
add("South",p);
p.add(b);
b.addActionListener(this);
m1.add(open); // 将菜单项加入到菜单 m1 中
m1.add(close);
m1.addSeparator(); //在菜单中添加分隔条
m1.add(exit);
open.addActionListener(this); //给菜单项 open 注册事件监听器
exit.addActionListener(this);
mb.add(m1); // 将菜单 m1 加入到菜单栏 mb 中
m12.add(copy); m12.add(cut); m12.add(paste);
m1.add(m12);//将 m12 作为 2 级菜单添加到 m1 菜单项中
m2.add(content); m2.add(index); m2.addSeparator(); m2.add(about);
mb.add(m2);
setMenuBar(mb); // 设置菜单栏为 mb
show();// 显示组件}
public static void main(String args[]) {
new LX6_7();}
public void actionPerformed(ActionEvent e) {
if (e.getActionCommand()=="退出")
System.exit(0);
if (e.getActionCommand()=="打开")
new LX6_6();}
```

}

在窗口中添加菜单栏，在菜单栏添加菜单项，并添加下拉菜单和 2 级菜单，通过选择菜单项可以执行不同操作，如“打开”可打开 LX6_6 类生成的窗口。

3．设计一个简单计算器，如下图所示。在“操作数”标签右侧的两个文本框输入操作数，当单击操作符＋，－，×，÷按钮时，对两个操作数进行运算并将结果填入到“结果”标签右侧的文本框中。

五、实验步骤

1．运用一种或多种布局管理器，绘制出一个简单的计算器；

2．为按钮注册事件监听器，使其点击按钮，并在显示区域同步显示当前输入或运算结果；

3．编译运行程序，检查计算器的正确性。