《高级语言程序设计》课程实验指导书

（2012年修订）

课程编号：

实用专业：　　电子商务

学时数：　　72

学分：　　2

编写者：　　谭学清

实验一：Java 编程环境

【实验目的】

1、掌握Java程序编辑、编译和运行的过程。

2、掌握Java程序的构成特点。

3、总结在调试过程中的错误。

【实验内容】

1、下载、安装并设置 Java SDK 软件包。

2、配置运行环境。

3、编写并编译运行简单的Java 应用程序和小程序。

【实验步骤】

1、免费的安装文件可以从Sun公司的主页上下载：Http://java.sun.com/

2、配置运行环境（假设安装目录为e:\j2sdk1.4.2_01\bin）

winXP/NT/2000系统：

在桌面“我的电脑”上右击，选择“属性”菜单，在高级选项卡中，配置“环境变量”：

path－e:\j2sdk1.4.2_01\bin

classpath－.;e:\j2sdk1.4.2_01\lib

进入DOS命令提示符状态，键入： javac回车，若出现帮助信息提示即为安装成功

3、编写一个简单的 Java 程序，运行结果为在屏幕上输出“HELLO WORLD!”。

Application程序的编写：打开记事本

```
public class HelloApplication{
public static void main(String args[]){
System.out.println(" HELLO WORLD!");
}
}
```

程序调试过程:

- 保存为HelloApplication.java
- 在DOS命令提示符状态，键入：javac HelloApplication.java，编译程序自动生成HelloApplication.class
- 在DOS命令提示符状态，键入:java HelloApplication，则可运行程序

4、Applet程序：

```
import java.applet.Applet;
import java.awt.*;
public class HelloApplet extends Applet{
public void paint(Graphics g){
g.drawString("欢迎学习java语言",100,100);
}
}
```

html文件的编写：

```
<html>
<body>
<applet code=HelloApplet.class width=500 height=400>
</applet>
</body>
</html>
```

程序调试过程:

- 保存
- 编译源程序(javac)
- 运行html文件：（1）用IE浏览器（2）用appletviewer 文件名.html

【实验准备知识】

1、什么是Java虚拟机？它的作用是什么？

2、如何在不同的环境下设置环境变量？

3、Java Application和Java Applet的不同之处？

实验二：Java 基本语法练习

【实验目的】

1、掌握Java运算符和表达式应用。

2、掌握各种流程控制语句

3、熟练应用数组的定义和使用

【实验内容】

1、教程例题编辑、调试和运行。

2、编写程序完成计算并输出 n 的阶乘和输出1~100间的所有奇数等程序功能

【实验步骤】

1、 选择语句应用：

实例1：给出3个整型数，找出最大的数。

```
public class U1{
public static void main(String args[]){
int max,a=4,b=3,c=7;
//具体程序内容由学生完成
System.out.println(a+" "+b+" "+c);//如果改为（a+b+c)结果如何？
System.out.println("max="+max);
}
}
```

实例2：根据变量score中存放的考试分数，输出对应的等级。

60分以下为D等；60～69为C等；70～89为B等；90～100为A等。

```
public class U2{
public static void main(String args[]){
int score=55;
```

```
switch(score/10) {
case 0:case 1:case 2:case 3:case 4:
case 5:System.out.println(score+"分是D等");break;
//去掉break结果有何变化？
case 6:System.out.println(score+"分是C等");break;
case 7:
case 8:System.out.println(score+"分是B等");break;
case 9:
case 10:System.out.println(score+"分是A等");break;
default:System.out.println("数据错误");
}
}
}
```

2、循环语句应用

实例3：计算1＋2＋„+100的结果。

程序自己编写

练习：

（1）计算并输出 n 的阶乘（设 n ＝10）。

（2）编写程序，输出1~100间的所有奇数。

提示：满足表达式（i%2!=0）的 i 值为奇数

3、数组的定义和使用

实例4：二维数组的定义和使用

```
public class Shuzu1{
public static void main(String args[]) throws java.io.IOException{
char ch[][]=new char[4][4];
System.out.println("输入16个字母，输入b跳出");
lab1:
for(int i=0;i<4;i++)
for(int j=0;j<4;j++){
ch[i][j]=(char)System.in.read();
System.in.skip(2);
```

```
//输入一个字母回车一次，回车不会被认做字母赋给数组
if(ch[i][j]=='b')
break lab1;// 去掉该处和第5行的lab1，结果又会如何？
}
for(int i=0;i<4;i++)//用于输出数组ch
for(int j=0;j<4;j++)
System.out.print(ch[i][j]+"|"); //"|"用于分隔每个数组元素
System.out.println("跳出");
}
}
```

思考：去掉第5行和第12行的lab1，结果又会如何？为什么会这样？

【实验准备知识】

1、复习Java语言基础知识

2、编写课后练习程序

实验三：面向对象编程练习

【实验目的】

1、 java类的定义和使用

2、 掌握创建和使用类对象的方法。

3、 对象的引用

4、 理解面向对象编程的关键技术：继承和多态

【实验内容】

1、教程例题编辑、调试和运行。

2、类和对象的创建、对象的引用

【实验步骤】

1、定义一个Person类，该类属性（变量）和方法如下：

姓名：name 字符串类型；

性别：sex 字符型；

年龄：age 整型；

2个构造方法：1个是默认的构造方法（由系统完成），另一个可通过参数赋值；

将该3个变量转化成字符串便于显示输出的方法：toString（该名称可自定义）

创建主类，通过Person类创建对象，显示输出该对象的各种属性。

```
class Person{
String name;
char sex;
int age;
public Person(String s,char c,int i){
name=s;
sex=c;
age=i;
}
public String toString(){
String s="姓名："+name+" 性别："+sex+" 年龄："+age;
//返回s的值
}
}
public class T1{
public static void main(String args[]){
Person p1=new Person("张三",'男',20);
//定义对象p2，各个参数分别为：李四，女，28
p1.sex='女'; //将p1的sex属性改为女
System.out.println(p1.toString());//输出p1的各个属性
//将p2的的age改为33
//输出p2的各个属性
}
}
```

2、模拟银行存取款业务。

1) 建立银行帐户类，属性：储户现有款项、存款的方法、取款的方法、显示余额的方法

2) 建立主类，创建对象（模拟储户名），存款1000，取款500，显示余额。

小结：成员变量与成员方法的设计方法。

3、类的继承

题目一：填空，完成并运行程序。写下程序运行结果并回答问题。

理解创建新类B（通过继承现有类A）的方法，使新类B具有类A的功能，并添加新的功能，编写主类考查通过继承创建的类B与父类A

程序一：

```
class A{
int i,j;
void showij(){System.out.println("i and j:"+i+" "+j);}
}
class B { //B类继承A类的属性和方法
int k;
void showk(){System.out.println("k:"+k);}
void sum(){System.out.println("i+j+k:"+(i+j+k));}
}
public class M1{
public static void main(String args[]){
A father =new A();
B son=new B();
father.i=10;father.j=20;
//使用父类A中的方法
son.i=7;son.j=8;son.k=9;
//使用子类B从父类A中继承的方法
//使用子类B新增的方法showk()
son.sum();
}
}
```

思考题：

（1）将父类A中的成员变量i声明为private，编译时观察有哪几处错误？

（2）在程序一中的子类B中添加语句：int i,j;（对父类A中的同名变量i,j 进行了重新定义）观察运行结果有什么不同？为什么？这种现象称为什么？

（3）在程序一中在子类B中添加成员方法：

void showij(){System.out.println (“覆盖了父类的成员方法”);}

（对父类A中的同名方法进行重新定义）观察运行结果有什么不同？为什么？这种现象称为什么？

4、类的多态性

题目：完成程序二中主类中的主方法，内容包括：

（1） 用类Intsort创建对象s

（2） 显示输出两个数的排序10，25

（3） 显示输出三个数的排序10，25 ，17

程序二：

```
class Intsort{
public String sort(int a,int b){ //定义两个数排序的方法

if(a>b)return a+” ”+b;
else return b+” ”+a;
}
public String sort(int a,int b,int c){ //定义三个数排序的方法
int swap;
if(a<b){swap=a;a=b;b=swap;}
if(a<c){swap=a;a=c;c=swap;}
if(b<c){swap=b;b=c;c=swap;}
return a+” ”+b+” ”+c;
}
}
public class M1{ //定义主类考查Intsort类中的方法
public static void main(String args[]){
}
}
```

【实验准备知识】

1、复习面向对象的三个特征

2、静态变量与普通变量的区别？

3、编写课后练习程序

实验四：包、接口与异常处理练习

【实验目的】

1．理解包是如何组织存放类的

2．熟悉和理解Java中异常的概念和处理机制。

3．熟悉和掌握异常的抛出和捕捉。

【实验内容】

1、教程例题编辑、调试和运行。

2、包是如何组织存放类的

3、系统接口的调用。

4、异常处理机制实现

【实验步骤】

1、包是如何组织存放类的

Java允许把多个类收集在一起成为一组，称作包（package）。引入包的目的是组织类的存放位置。

程序一：

```
package P1;
class C1{}
public class C2{
public static void main(String args[]){
System.out.println("察看编译结果");
}}
interface I1{}
package P2;
class C3{}
```

将上述程序分别保存并调试，写下运行结果和编译结果。

调试过程：

1) 保存文件C2.java(注意：在一个java源程序中只能有一条包语句)

2) 编译：javac –d . C2.java

3) 运行：java P1.C2

4) 保存文件C3.java

5) 编译：javac –d . C3.java

6) 运行：java P2.C3

2、系统接口的调用。

程序二：运行程序，回答问题：

```
import java.awt.*;
import java.awt.event.*;
import java.applet.Applet;
public class M extends Applet implements
MouseListener,MouseMotionListener{
int x1,y1,x2,y2;
public void init(){addMouseListener(this); addMouseMotionListener(this);}
public void paint(Graphics g){g.drawLine(x1,y1,x2,y2); //语句1
g.drawString("x1:"+x1+" y1:"+y1+" x2:"+x2+" y2:"+y2,50,50);}
public void mousePressed(MouseEvent e){ x1=e.getX();y1=e.getY();}
public void mouseClicked(MouseEvent e){}public void
mouseExited(MouseEvent e){}
public void mouseEntered(MouseEvent e){}public void
mouseReleased(MouseEvent e){}
public void mouseDragged(MouseEvent e){x2=e.getX();y2=e.getY();repaint();}
//语句2
public void mouseMoved(MouseEvent e){} }
```

（1）程序中蓝色字的语句打括号是空的，能否删除？

（2）带下划线的语句有什么含义，将语句1改为画矩形；将语句2删除会有什么影响？

（3）接口MouseListener和MouseMotionListener中有哪些方法？

程序三：自定义接口的定义和使用

（1）定义接口N1

```
interface N1{
int year=2006; //year默认为static final常量
int age(); //无方法体，在使用时需加入方法体
void output(); }
```

（2）引用自定义接口

```
public class M implements N1{
String xm;int sr; //成员变量：xm为姓名，sr为出生所在年（如：1988）
M(String a,int b){xm=a;sr=b;} //构造方法
public int age(){ } //实现接口的方法，返回年龄值
public void output(){System.out.println(xm+"的年龄为："+this.age());} //输出年龄
public static void main(String aa[]){
//使用本类M创建对象a，姓名为张弛，出生所在年1988
//调用output（）方法输出
} }
```

3、异常处理机制

程序四：

在扣分程序中如果扣分值为负值，显示“扣分结果为负值”的信息。

扣分程序：

```
public class C1{
public static void main(String args[]){
System.out.println("平时分满分20");
System.out.println("扣分"+kf(1));} //如果将1改为-1，观察程序结果的变化并说明原因
static int kf(int n) {
} //返回总分20-n
}
```

平时分满分20

扣分21

平时分满分20

扣分19

程序五：如果输入扣分值为负值，程序二结果并非我们所期望的运行结果，改造程序二使其会自动处理该种错误，并显示输入的扣分是负值。（程序中蓝色字体部分是添加了异常处理机制的语句）

```
public class C2{
public static void main(String args[]){
System.out.println("平时分满分20");
1．  {System.out.println("扣分"+kf(－1));}
2．  (Exception e){System.out.println("扣分是负值");}
}
static int kf(int n) throws C3{
C3 d=new C3();
if(n<0)throw d; //抛出C3异常
return 20-n; }
}
class C3 extends Exception{ //自定义的异常类
C3(){super("扣分输入错误");}
}
```

完成程序并写出程序结果。

程序六：上机输入以下程序，在程序try中的语句如果可能产生多种异常，注意与catch参数进行匹配。观察程序的运行结果，体会Java中的异常处理机制。

```
class Test{
static int a=3,b=0; //运行后，将变量b改成非零值，观察结果
static String c[]={“数组元素c[0]”};
public static void main(String[] args) {
try{ System.out.println(a);
System.out.println(a/b);
System.out.println(c[b]); //若上条语句产生异常，本语句将不被运行，要想本语句始终运行，可将其放在finally中，写下运行结果
}
catch(ArithmeticException e){
```

```
System.out.println ("捕捉到一个算术异常") ;
}
catch(ArrayIndexOutOfBoundsException e){
System.out.println ("捕捉到一个数组下标越界异常") ;
}
catch(Exception e){
System.out .println ("捕捉到一个系统一般的异常");
}
finally{
System.out.println("程序结束");
}
}}
```

思考题：

(1) 系统中哪条语句会抛出异常?哪条语句捕捉了异常?

(2) 为什么程序不会打印出“捕捉到一个系统一般的异常”? 将两个catch语句块交换位置，程序能够编译通过吗?系统将给出什么错误提示?为什么？

原因：try后的catch语句只有一个参数（需要捕捉的异常类的对象），如果在try语句块的程序能够抛出多个异常。那么就需要使用多个异常处理语句(catch语句)来进行捕捉。一般地，处理较具体和较常见的异常的catch块应放在前面，而可以与多种异常相匹配的catch块应放在较后的位置，特别是在多个异常类是相互继承的时候更是如此。例中，如果将捕捉Exception的catch语句放在了捕捉ArithmeticException的catch语句之前，那么就会出错，因为ArithmeticExceptioa是继承自Exception的，如果程序抛出ArithmeticException，那么就会与捕捉Exception的catch语句相匹配(向上转型的思想)，系统会提示捕捉ArithmeticException的catch语句不能执行到，当然这是一个程序的故障(因此应该避免)。

(3)finally语句块中的语句一定会被执行吗?将程序中的变量b改成非零值。写下运行结果并说明为什么？程序也会打印出“程序结束”吗?

(4)将try-catch-finally语句去掉，直接进行编译，能够编译通过吗?观察程序的运行结果。

【实验准备知识】

1、 理解包是如何存放类的

2、 带有包语句的源程序如何调试及运行

3、 理解java异常处理机制的原理及其程序实现

实验五：输入输出流类练习

【实验目的】

1. Java标准输入输出流

2. 数据流应用

【实验内容】

1、教程例题编辑、调试和运行。

2、Java标准输入输出流

3、数据流应用

【实验步骤】

1、**Java**标准输入输出流

程序1：

```
import java.io.*;
public class T1{
public static void main(String args[]) throws IOException{
int ch;
System.out.println("请输入一个字符：");
ch=System.in.read();
//输出ch值（将ch强制转换成字符型）
}
}
```

程序2：用if语句编写程序，实现以下功能：从键盘输入数字1显示good，输入其他键时显示bad。

程序3：编程输出26个英文字母。

程序4：程序功能：从键盘中读取字符串数据，读懂并运行该程序。

```
import java.io.*;
public class C1{
```

```
public static void main(String args[]){
String s;
InputStreamReader ir;
BufferedReader in;
ir=new InputStreamReader(System.in);
//建立与系统标准输入之间的输入流联系
in=new BufferedReader(ir);
try{
do{s=in.readLine(); //从键盘读入一行字符串
if(s!=null){
System.out.println("Read: "+s); //将读取的数据输出
}
}while(s!=null); //判断是否读完数据
}catch(Exception e){};
}}
```

2、数据流应用

程序5：

程序功能：将t1.txt中内容添加到t2.txt中。（注这2个文本文件要事先建立）

t1.txt的内容：

天天向上！

t2.txt的内容：

好好学习，

```
import java.io.*;
public class T3{
public static void main(String args[]) throws IOException{
FileReader in=new FileReader("t1.txt");
BufferedReader bin=new BufferedReader(in);
FileWriter out=new FileWriter("t2.txt",true);
//建立“输入”→“缓冲”→“输出”对象
```

```
String s1;
while((s1=bin.readLine())!=null){
System.out.println(s1); //将s1输出到显示器
out.write(s1+"\n"); //将s1写入out对象中}
in.close();
out.close();
}}
```

调试并运行程序。

【实验准备知识】

1、 理解字节流和字符流

2、 体会例题中的编程要领

实验六：图形界面设计练习

【实验目的】

1. 掌握文本组件、按钮和单复选按钮组件的使用

2. 掌握列表的使用，鼠标键盘事件的处理

3、掌握布局控制的方法

【实验内容】

1、教程例题编辑、调试和运行。

2、编写一个简单的计算器程序，实现加、减、乘、除、清零等功能。

【实验步骤】

1、程序名：X08_08_CalculatorGUI.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.applet.Applet;
public class X08_08_CalculatorGUI extends JFrame implements ActionListener
{
private final static int NBUTTONS = 16; // Constants
private JPanel keyPadPanel; // Panel to hold keyPad
```

```
    private JTextField accumulator; // Calculator s display
    private JButton keyPad[]; // Internal keyPad array
    private String label[] = // 16 keyPad Button Labels
    { "1", "2", "3", "+", "4", "5", "6", "-", "7", "8", "9", "*", "C", "0","=", "/" };
    private X08_08_Calculator calcMachine = new X08_08_Calculator(); // The
calculator
    public X08_08_CalculatorGUI(String title) { // Set up the keypad grid layout
    super(title);
    keyPadPanel = new JPanel();
    keyPadPanel.setLayout(new GridLayout(4, 4, 1, 1));
    keyPad = new JButton[NBUTTONS];// Create an array of buttons
    for (int k = 0; k < keyPad.length; k++) {
    // For each array slot Create a button
    keyPad[k] = new JButton(label[k]);
    // And a listener for it
    keyPad[k].addActionListener(this);
    // And add it to the panel
    keyPadPanel.add(keyPad[k]);
    } // for
    // Set up the accumulator display
    accumulator = new JTextField("0", 20);
    accumulator.setEditable(false);
    // Add components to the frame with Border layout
    getContentPane().setLayout(new BorderLayout(10, 10));
    getContentPane().add("North", accumulator);
    getContentPane().add("Center", keyPadPanel);
    } // X08_08_CalculatorGUI()
    public void actionPerformed(ActionEvent e) {
    JButton b = (JButton) e.getSource();
    // And its label
    String key = b.getText();
```

```
String result = calcMachine.handleKeyPress(key, accumulator.getText());
accumulator.setText(result);
} // actionPerformed()
public static void main(String args[]) {
X08_08_CalculatorGUI calc = new X08_08_CalculatorGUI("Calculator");
calc.setSize(400, 400);
calc.setVisible(true);
calc.addWindowListener(new WindowAdapter() {
public void windowClosing(WindowEvent e) {
System.exit(0);
}
});
} // main()
} // X08_08_CalculatorGUI
```

2、程序名：X08_08_Calculator.java

```
import java.lang.*;
public class X08_08_Calculator {
public final static int NOOP = -1; // Operations
public final static int EQUAL = 0;
public final static int ADD = 1;
public final static int SUBTRACT = 2;
public final static int MULTIPLY = 3;
public final static int DIVIDE = 4;
public final static int CLEAR = 5;
public final static int APPEND = 0; // Display states
public final static int REPLACE = 1;
private int accumulator; // The actual accumulator
// Internal memory for the visible textField
private int displayRegister;
// The operation that s waiting for Operand2
private int opCode;
```

```
// State of the external display (REPLACE or APPEND)
private int displayState;
// Set true if divide-by-zero error
private boolean error;
public X08_08_Calculator() {
initialize();
} // constructor
private void initialize() {
// Reset the machine s registers
accumulator = 0;
displayRegister = 0;
// Replace the display with the next digit typed
displayState = REPLACE;
opCode = NOOP; // No current operation
error = false;
} // initialize()
private int keyToIntCode(char keyCh) {
switch (keyCh) {
case 'C':
return CLEAR;
case '=':
return EQUAL;
case '+':
return ADD;
case '-':
return SUBTRACT;
case '*':
return MULTIPLY;
case '/':
return DIVIDE;
}
```

```
return -1; // Error Return
}
private int doCurrentOp(int keyCode) { // do current opCode
if (keyCode == CLEAR) {
initialize();
return accumulator;
}
switch (opCode) {
case NOOP:
accumulator = displayRegister;
break;
case ADD:
accumulator = accumulator + displayRegister;
break;
case SUBTRACT:
accumulator = accumulator - displayRegister;
break;
case MULTIPLY:
accumulator = accumulator * displayRegister;
break;
case DIVIDE:
if (displayRegister == 0)
error = true;
else
accumulator = accumulator / displayRegister;
break;
} // switch
if (keyCode == EQUAL)
opCode = NOOP; // Reset opCode
else
opCode = keyCode; // Set up for next operation
```

```
displayState = REPLACE;
displayRegister = 0;
return accumulator;
} // doCurrentOp
public String handleKeyPress(String keyStr, String external_display) {
String resultStr; // Stores the result
char keyCh = keyStr.charAt(0); // Convert the key label to a char
if (Character.isDigit(keyCh)) { // If this was a digit key
if (displayState == APPEND) // either append it
resultStr = external_display + keyCh;
else { // or use it to replace the display
displayState = APPEND;
resultStr = keyStr;
}
} else { // If not a digit key, it must be an opCode
// Get display value
displayRegister = Integer.parseInt(external_display);
// Perform an operation
int result = doCurrentOp(keyToIntCode(keyCh));
if (!error)
resultStr = Integer.toString(result); // Return result
else {
resultStr = "Error";
initialize();
}
}
return resultStr; // return the result
} // handleKeyPress()
} // X08_08_Calculator Class
```

【实验准备知识】

1、 AWT有哪些组件和容器，他们各自的使用方法。

2、 什么是Swing，它和AWT比有什么优点？使用上有什么区别？

实验七：JavaApplet 的编写

【实验目的】

1. 熟悉Applet的运行过程

2. 掌握Applet程序中常用的方法

【实验内容】

1、教程例题编辑、调试和运行。

2、设计和实现一个RockPaperScissors的Applet子类，主要实现用户和计算机之间玩剪刀、石头或布的游戏，并统计胜负记录。

【实验步骤】

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.*;
public class X07_06_JRockPaperScissors extends JApplet implements
ActionListener {
JLabel greeting = new JLabel("Rock, Paper, Scissors");
JLabel promptLabel = new JLabel("Choose one button");
Font headlineFont = new Font("Helvetica", Font.BOLD, 30);
Font mediumFont = new Font("Helvetica", Font.BOLD, 18);
JButton rock = new JButton("Rock");
JButton paper = new JButton("Paper");
JButton scissors = new JButton("Scissors");
JLabel resultsLabel = new JLabel(" ");
JLabel result1 = new JLabel();
JLabel result2 = new JLabel();
JLabel result3 = new JLabel();
String userPick = "";
String computerPick = "";
```

```
Container con = getContentPane();
int playerCount = 0;
int computerCount = 0;
int tieCount = 0;
final int LOW = 1;
final int HIGH = 3;
public void init() {
greeting.setFont(headlineFont);
con.add(greeting);
promptLabel.setFont(mediumFont);
con.add(promptLabel);
con.add(rock);
con.add(paper);
con.add(scissors);
resultsLabel.setFont(mediumFont);
con.add(resultsLabel);
con.add(result1);
con.add(result2);
con.add(result3);
con.setLayout(new FlowLayout());
rock.addActionListener(this);
paper.addActionListener(this);
scissors.addActionListener(this);
repaint();
}
public void actionPerformed(ActionEvent e) {
int computer;
String msg;
computer = LOW + (int) (Math.random() * HIGH);
if (computer == 1)
computerPick = "rock";
```

```
else if (computer == 2)
computerPick = "paper";
else
computerPick = "scissors";
if (e.getSource() == rock) {
userPick = "rock";
if (computer == 1) {
msg = "tie";
++tieCount;
} else if (computer == 2) {
msg = "computer";
++computerCount;
} else {
msg = "you";
++playerCount;
}
} else if (e.getSource() == paper) {
userPick = "paper";
if (computer == 2) {
msg = "tie";
++tieCount;
} else if (computer == 3) {
msg = "computer";
++computerCount;
} else {
msg = "you";
++playerCount;
}
} else /* 100 */
{
userPick = "scissors";
```

```
        if (computer == 3) {
        msg = "tie";
        ++tieCount;
        } else if (computer == 1) {
        msg = "computer";
        ++computerCount;
        } else {
        msg = "you";
        ++playerCount;
        }
        }
        con.remove(resultsLabel);
        con.remove(result1);
        con.remove(result2);
        con.remove(result3);
        resultsLabel.setText("------Results------");
        result1.setText("You picked " + userPick + " ---- Computer picked " +
computerPick);
        result2.setText("Winner: " + msg);
        result3.setText("You: " + playerCount + " Computer: " + computerCount + "
Ties: " + tieCount);
        con.add(resultsLabel);
        con.add(result1);
        con.add(result2);
        con.add(result3);
        validate();
        resultsLabel.setLocation(40, 70);
        result1.setLocation(30, 100);
        result2.setLocation(30, 120);
        result3.setLocation(30, 140);
        repaint();
```

```
        }
        public void paint(Graphics gr) {
        float startx = 200f;
        float starty = 180f;
        float width = 60f;
        float height = 60f;
        Graphics2D g = (Graphics2D) gr;
        BasicStroke aStroke=new BasicStroke(6.0f,
BasicStroke.CAP_ROUND,BasicStroke.JOIN_ROUND);
        g.setStroke(aStroke);
        g.setColor(Color.WHITE);
        g.draw(new Ellipse2D.Float(startx, starty, width, height));
        g.draw(new Rectangle2D.Float(startx, starty, width, height));
        g.draw(new Line2D.Float(startx, starty, startx + 40, starty + 40));
        g.draw(new Line2D.Float(startx + 40, starty, startx, starty + 40));
        g.draw(new Ellipse2D.Float(startx - 20, starty + 40, 40f, 40f));
        g.draw(new Ellipse2D.Float(startx + 30, starty + 40, 40f, 40f));
        g.setColor(Color.BLACK);
        promptLabel.repaint();
        resultsLabel.repaint();
        result1.repaint();
        result2.repaint();
        result3.repaint();
        rock.repaint();
        paper.repaint();
        scissors.repaint();
        if (userPick.equals("rock")) {
        g.draw(new Ellipse2D.Float(startx, starty, width, height));
        } else if (userPick.equals("paper")) {
        g.draw(new Rectangle2D.Float(startx, starty, width, height));
        } else if (userPick.equals("scissors")) {
```

```
g.draw(new Line2D.Float(startx, starty, startx + 40, starty + 40));
g.draw(new Line2D.Float(startx + 40, starty, startx, starty + 40));
g.draw(new Ellipse2D.Float(startx - 20, starty + 40, 40f, 40f));
g.draw(new Ellipse2D.Float(startx + 30, starty + 40, 40f, 40f));
}
}
}
```

【实验准备知识】

1、 如何创建Applet

2、 Applet的生命周期及主要方法。

实验八：综合实验

【实验目的】

1、熟悉并掌握所学过的知识，并能进行简单的程序开发

2、分析应用程序中使用了哪些系统类，指出使用该类的变量与方法。说明创建了什么类。包含什么变量与方法。

3、能根据实际需要使用不同的系统类编写应用程序

【实验内容】

1、 模拟小车运行，用画图来表示车的位置

2、在Applet界面上点击鼠标，得到点击位置的x和y坐标值，并显示在文本框中。

【实验步骤】

1、模拟小车运行

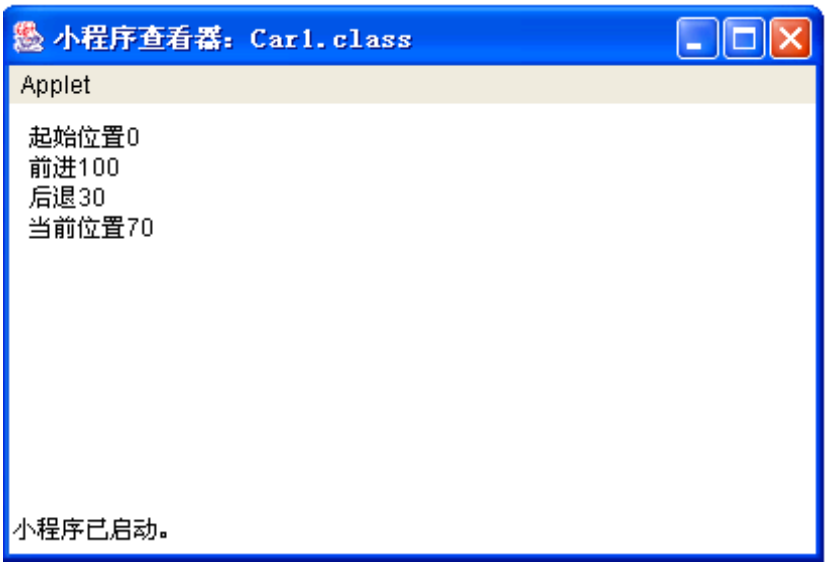
1) 建立小车类Car，拥有属性：位置、前进方法、后退方法、查询位置方法

2) 建立主类，调用前进和后退的方法模拟小车运行，最后查询小车的位置

```
import java.awt.*;
import java.applet.Applet;
public class UseCar extends Applet{
public void paint(Graphics g){
```

```
//定义小车类对象mycar
g.drawString("起始位置"+mycar.getwz(),10,20);
g.drawString("前进100",10,35);
mycar.qj(100);
g.drawString("后退30",10,50);
mycar.ht(30);
g.drawString("当前位置"+mycar.getwz(),10,65);
}
}
class Car{
private int wz;
public Car(){wz=0;}
//定义前进的方法
//定义后退的方法
//定义查询位置的方法
}
```

2、接前一个例子用画图来表示车的位置(运行结果见下页)

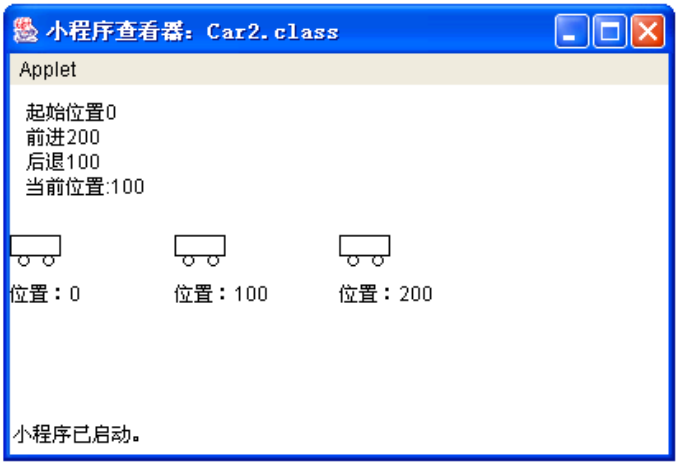


1) 添加一个用于画出小车的类Hcar，属性：位置、设置位置的方法、画出小车的方法

2) 在主类中调用Hcar类画出小车在改变位置时不同的位置。

```
import java.awt.*;
import java.applet.Applet;
public class Car2 extends Applet{
public void paint(Graphics g){
Car mycar=new Car();
//用Hcar类创建对象hcar
g.drawString("起始位置"+mycar.getwz(),10,20); //小车起始位置
hcar.setwz(mycar.x); //设置画小车位置
hcar.draw(g); //在该位置画出小车
g.drawString("前进300",10,35);
//小车前进300，位置发生变化
//设置画小车位置
//在该位置画出小车
g.drawString("后退100",10,50);
//小车后退100，位置发生变化
//设置画小车位置
//在该位置画出小车
g.drawString("当前位置:"+mycar.getwz(),10,65);
}
}
class Car{
public int x;
public Car(){x=0;}
public void qj(int wz1){x+=wz1;}
public void ht(int wz1){x-=wz1;}
public int getwz(){return x;}
}
class Hcar{ //定义画出小车的类
int xpos=20;
public void setwz(int xx){xpos=xx;}
```

```
public void draw(Graphics g){
g.drawRect(xpos,90,30,12);
g.drawOval(xpos+5,90+12,6,6);
g.drawOval(xpos+20,90+12,6,6);
g.drawString("位置："+xpos,xpos,90+40);
}
```



3、在Applet界面上点击鼠标，得到点击位置的x和y坐标值，并显示在文本框中。

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
public class Mouse1 extends Applet implements MouseListener{
int x,y;TextField t1=new TextField(12);
public void init(){ add(t1); addMouseListener(this);}
public void mousePressed(MouseEvent e){
x=e.getX();y=e.getY();
t1.setText("坐标值："+Integer.toString(x)+","+Integer.toString(y));
}
public void mouseClicked(MouseEvent e){}
public void mouseEntered(MouseEvent e){}
public void mouseReleased(MouseEvent e){}
public void mouseExited(MouseEvent e){}
}
```

实验要求：

观察运行结果，并修改源程序：

（1） 再添加一个文本框t2，可以容纳10个字符；

（2） 修改mousePressed(MouseEvent e)方法，x坐标在文本框t1中显示，y坐标在文本框t2中显示；

（3） 使用适配器类（裁剪器）将没有使用的方法去掉。

4、 键盘事件

KeyListener接口中有三个方法：

```
public void keyPressed(KeyEvent e)
public void keyReleased(KeyEvent e)
public void keyTyped(KeyEvent e)
```

程序2：

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
public class Key1 extends Applet implements KeyListener{
String a="";
public void init(){ addKeyListener(this);requestFocus();}
public void keyPressed(KeyEvent ke){showStatus(“按下");}
public void keyReleased(KeyEvent ke){showStatus(“抬起");}
public void keyTyped(KeyEvent ke){
a+=ke.getKeyChar();repaint();}
public void paint(Graphics g){
g.drawString(a,10,20);}
}
```

实验要求：

观察运行结果，注意要先用鼠标点击Applet界面，改变焦点，键盘事件才能响应。

【实验准备知识】

复习 Java 程序设计高级功能