

# 《Java 程序设计》实验指导书

郑州轻工业学院 计算机与通信工程学院

## 实验一 熟悉 Java 程序的开发

### 一、实验目的

- (1) 学习使用 JDK 开发工具开发 Java 应用程序；
- (2) 掌握 Java Application 程序的开发过程；
- (3) 掌握 Java Applet 程序的开发过程。

### 二、实验内容

#### 上机前的重要提示：

- Java 源代码可在任何文本编辑器中输入，但这里建议使用记事本。
- 所有的 Java 源代码都应具有扩展名 “.java”
- 在包含主类的文件中，文件名应与主类的名称相同，并注意有大小写之分。

#### 1. 基本指导

##### 指导内容 1：

编写并运行第一个 Java Application 程序

##### 实验步骤：

- (1) 开机后，在 java 实验目录下创建 test1 子目录。本阶段的 Java 源程序、编译后的字节码文件都放在这个目录中。
- (2) 打开一个纯文本编辑器（如记事本），键入如下程序（注意大小写）：

```
import java.io.*;

public class MyFirstJavaProgram {

    public static void main(String args[]){

        System.out.println("This is my first Java program! ");

    }

}
```

- (3) 将文件保存起来，命名为 MyFirstJavaProgram.java，保存在自己工作的目录下。
- (4) 进入命令方式（MS—DOS），并转.java 文件所在目录。敲入下述命令，编译上述 Java 文件。

命令格式：javac MyFirstJavaProgram.java

- (5) 利用 Java 解释器运行这个 Java Application 程序并查看运行结果。

命令格式：java MyFirstJavaProgram

以上程序运行结果如图 1-1 所示。



图 1-1

## 指导内容 2:

编写并编译第一个 Java Applet 程序。

- (1) 打开一个纯文本编辑器（如记事本），键入如下程序（注意大小写）:

```
import java.applet.Applet;
import java.awt.Graphics;
public class MyFirstJavaApplet extends Applet{
    public void paint(Graphics g){
        g.drawString("This is my first Java Applet ",15,20);
    }
}
```

- (2) 把文件保存起来，命名为 MyFirstJavaApplet.java，保存在自己测试的目录下。
- (3) 进入命令方式（MS—DOS）并转.java 文件所在目录，敲入下述命令，编译上述 Java 文件。

命令格式：javac MyFirstJavaApplet.java

- (4) 编写配合 Applet 的 HTML 文件，代码如下：

```
<html>
<body>
    <applet code=MyFirstJavaApplet width=300 height=200>
    </applet>
</body>
</html>
```

- (5) 将上述内容存盘为 MyFirstJavaApplet.html,与文件 MyFirstJavaApplet.java 保存在本实验的工作目录下。
- (6) 用模拟的 Applet 运行环境解释运行这个 Java Applet 程序并观察运行结果。  
命令格式：AppletViewer MyFirstJavaApplet.html  
以上程序的运行结果如图 1-2 所示。



图 1-2

## 2. 练习思考

### 练习内容：

运行下面的程序代码，并回答问题。

### 程序代码：

```
import java.awt.*;
import java.applet.*;
public class WhatAmI extends Applet {
    public void paint(Graphics g){
        g.drawString("What am I,Application or Applet?",10,20);
    }
}
```

### 思考问题：

- (1) 上面的程序是 Application 还是 Applet?
- (2) 该程序的运行过程有几步，它们分别是什么？
- (3) DrawString 方法中的第二个参数“10”和第三个参数“20”是什么意思？
- (4) 将上面的程序改成另一种类型的 Java 程序，同样输出字符串“What am I,Application or Applet?”。

## 3. 上机作业

分别编写 Application 和 Applet 程序，使运行后在屏幕上生成如下的图案：

```
*
***
*****
*****
*****
```

## 实验二 Java 语言编程基础

### 一、实验目的

- (1) 掌握如何在 Java 程序中定义变量；
- (2) 掌握各种运算符及其相关表达式运算；
- (3) 学习数组的定义及使用。

### 二、实验内容

#### 1. 基本指导

##### 指导内容 1：

编写 Java Application 程序，分析程序运行结果。



```

        "1000","1001","1010","1011",
        "1100","1101","1110","1111"};

static final int FLAG1=1;
static final int FLAG2=2;
static final int FLAG4=8;
public static void main( String args[] ){
    int flags=0;
    System.out.println("Clear all flags... flags="+binary[flags]);
    flags=flags | FLAG4;
    System.out.println("Set flag4... flags="+binary[flags]);
    flags=flags ^ FLAG1;
    System.out.println("Revert flag1... flags="+binary[flags]);
    flags=flags ^ FLAG2;
    System.out.println("Revert flag2... flags="+binary[flags]);
    int cf1=~FLAG1;
    flags=flags & cf1;
    System.out.println("Clear flag1... flags="+binary[flags]);
    int f4=flags & FLAG4;
    f4=f4>>>3;
    System.out.println("Get flag4... flag4="+f4);
    int f1=flags & FLAG1;
    System.out.println("Get flag1... flag1="+f1);
}
}

```

- (2) 把文件保存起来，命名为 BitDemo.java，保存在 java 实验目录的 test2 子目录下。
- (3) 进入命令方式（MS—DOS）并转.java 文件所在目录，敲入下述命令，编译上述 Java 文件。

命令格式：javac BitDemo.t.java

- (4) 利用 Java 解释器运行这个 Java Application 程序并查看运行结果。

命令格式：java ArithmeticTest。

以上程序的运行结果如图 2-2 所示。

```

命令提示符
E:\Java\test\test2>javac BitDemo.java

E:\Java\test\test2>java BitDemo
Clear all flags... flags=0000
Set flag4... flags=1000
Revert flag1... flags=1001
Revert flag2... flags=1011
Clear flag1... flags=1010
Get flag4... flag4=1
Get flag1... flag1=0

E:\Java\test\test2>_

```

图 2-2

## 2. 练习思考

### 练习内容：

运行下面的程序代码，并回答问题。

### 程序代码：

```
import java.applet.Applet;
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class DataType extends Applet implements ActionListener
{
    Label prompt=new Label("请分别输入整数和浮点数:");
    TextField input_int=new TextField(6);
    TextField input_double=new TextField(6);
    TextField output=new TextField(35);
    int getInt; double getDouble;

    public void init() {
        add(prompt); add(input_int); add(input_double);
        add(output); output.setEditable(false);
        input_double.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e) {
        getInt=Integer.parseInt(input_int.getText());
        getDouble=Double.parseDouble(input_double.getText());
        output.setText("您输入了整数: "+getInt+"和浮点数: "+getDouble);
    }
}
```

### 思考问题：

- (1) 上面的程序是 Application 还是 Applet?
- (2) 上面的程序用什么方式接受数据的输入和数据的输出?
- (3) 假如在输入整数的文本框，输入了浮点数，运行的结果是什么，为什么?
- (4) 假如在输入浮点数的文本框，输入了整数，运行的结果又是什么，为什么?

### 3. 上机作业

编写一个加密 Application 程序，将一个字母赋给一个变量，输出这个字母加密后的结果。加密操作是将字母变换成倒序的字母，例如 A 变成 Z, B 变成 Y, C 变成 X……。

提示：

- 定义一字符变量 c，用来存放指定的字符；
- 计算变量 c 的倒序字母的 ASCII 码；  
c>=' A' && c<=' Z' 时 倒序字母的 ASCII 码为 155-c;  
c>=' a' && c<=' z' 时 倒序字母的 ASCII 码为 219-c;。

- 用 System.out.println 方法将加密后的字母输出。

## 实验三 Java 语言控制结构

### 一、实验目的

- (1) 掌握一维数组的声明、初始化和引用；
- (2) 熟练使用 if/else 语句和 switch 条件分支语句编程；
- (3) 熟练使用 while 语句、do-while 语句、for 语句等循环语句编程。

### 二、实验内容

#### 1. 基本指导

##### 指导内容 1:

比较两个数的大小并按升序输出。

##### 实验步骤:

- (1) 开机后，在 java 实验目录下创建 test3 子目录。本阶段的 Java 源程序、编译后的字节码文件都放在这个目录中。
- (2) 打开一个纯文本编辑器，键入如下程序（注意大小写）:

```
public class Sort {  
    public static void main (String args[]) {  
        double d1=23.4;  
        double d2=35.1;  
        if (d2>=d1)  
            System.out.println(d2+">="+d1);  
        else  
            System.out.println(d1+">="+d2);  
    }  
}
```

- (3) 将文件保存起来，命名为 Sort.java，保存在 java 实验目录的 test3 子目录下。
- (4) 进入命令方式（MS—DOS），并转.java 文件所在目录。敲入下述命令，编译上述 Java 文件。

命令格式：javac Sort.java

- (5) 利用 Java 解释器运行这个 Java Application 程序并查看运行结果。程序的运行结果如图 3.1 所示。

命令格式：java Sort



图 3.1

## 指导内容 2:

编写程序，输出 1 到 1000 之间，所有可以被 3 整除又可以被 7 整除的数。

(1) 打开一个文本编辑器，键入如下程序（注意大小写）：

```
public class NumTest {
    public static void main (String args[]) {
        int n,num,num1;
        System.out.println("在 1~1000 可被 3 与 7 整除的为");
        for (n=1;n<=1000;n++) {
            num =n%3;
            num1=n%7;
            if (num==0) {
                if (num1==0)
                    System.out.print(n+" ");
            }
        }
        System.out.println(" ");
    }
}
```

(2) 把文件命名为 NumTest.java，保存在 java 实验目录的 test3 子目录下。

(3) 进入命令方式（MS—DOS）并转.java 文件所在目录，敲入下述命令，编译上述 Java 文件。

命令格式：javac NumTest.java

(4) 利用 Java 解释器运行这个 Java Application 程序并查看运行结果。程序的运行结果如图 3.2 所示。

命令格式：java NumTest。

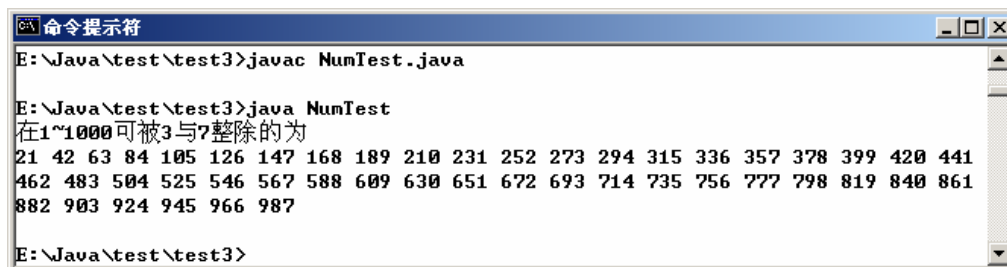


图 3.2

## 2. 练习思考

### 练习内容 1:



使用 while 和 do\_while 循环语句改写指导 2 的程序代码，并上机运行。

### 练习内容 2:

创建一个具有 5 个值的数组，并找出最大值和最小值。

### 程序代码:

```
public class ArrSort {
    public static void main(String[] args) {
        int arr[]=new int[5];
        int i;
        arr[0]=10;
        arr[1]=20;
        arr[2]=-9;
        arr[3]=8;
        arr[4]=98;
        int min=0,max=0;
        for(i=0;i<5;i++){
            if(max<arr[i])
                max=arr[i];
            if(min>arr[i])
                min=arr[i];
        }
        System.out.println("数组的最大值是:" +max);
        System.out.println("数组的最小值是:"+min);
    }
}
```

### 思考问题:

- (1) 将上面的数组进行排序，数组的第一个元素为最小值，最后一个元素为最大值。
- (2) 能根据给定的数组值，找出该数组值在数组中的下标。

### 3. 上机作业

- (1) 编写一个换算 GPA 的 Application 程序，对于学生学习的每门课程，都输入两个整数：考试成绩和学分，考试成绩按如下公式换算：

85~100: 4  
75~84: 3  
60~74: 2  
45~59: 1  
44 以下: 0

GPA 等于换算后每门课的成绩的学分加权平均值 ( $\sum (\text{成绩} \times \text{学分}) / \sum \text{学分}$ )。

- (2) 设 n 为自然数， $n!=1 \times 2 \times 3 \times \cdots \times n$  称为 n 的阶乘，并且规定  $0!=1$ 。试编制程序计算 2!,4!,6!和 10!，并将结果输出到屏幕上。

## 实验四 面向对象的编程技术

### 一、实验目的

- (1) 掌握类与对象的基本概念以及它们之间的关系；
- (2) 掌握定义类与创建对象实例的方法；
- (3) 掌握类方法和属性的定义和使用；
- (4) 掌握构造方法的定义及其使用。

### 二、实验内容

#### 1. 基本指导

##### 指导内容：

定义一个类-圆，并编一个主类测试它。

##### 实验步骤：

- (1) 开机后，在 java 实验目录下创建 test4 子目录。本阶段的 Java 源程序、编译后的字节码文件都放在这个目录中。
- (2) 打开一个纯文本编辑器，定义一个类-圆，代码如下：

```
class CCircle    {  
    double pi;  
    double radius;  
    double getRadius(){  
        return radius;  
    }  
    void setCircle(double r, double p){  
        pi=p;  
        radius=r;  
    }  
}
```

- (3) 在上面的代码后面添加主类代码，创建类-圆的一个实例，并输出该圆的半径：

```
public class TestCCircle{  
    public static void main(String args[])    {  
        CCircle cir1=new CCircle();  
        cir1.setCircle(2.0,3.1416);  
        System.out.println("radius="+cir1.getRadius());  
    }  
}
```

```

    }
}

```

- (4) 把文件命名为 TestCCircle.java, 保存在 java 实验目录的 test4 子目录下。
- (5) 编译并运行该程序, 程序的运行结果如图 4.1 所示



图 4.1

## 2. 练习思考

### 练习内容:

扩展圆的定义, 为其增加可以求圆面积的方法, 并在主类中输出一个实例化的圆的面积。

### 思考问题:

运行扩展后的程序, 思考如下的问题:

- (1) 是否可以将类-圆的定义和主类的源代码放在两个文件中。如果可以的话, 两个文件的命名有何要求, 上机测试后, 给出结论。
- (2) 修改程序, 使圆的属性 pi 定义为最终变量, 其值为 3.14159, 看会出现什么样的结果。如果程序出错, 请调整代码以适合属性 pi 为最终变量的要求。
- (3) 为程序添加构造方法代码, 调用该构造方法, 可以完成圆的半径的初始化。
- (4) 修改主类代码, 测试构造方法的使用。

## 3. 上机作业

- (1) 编写 Book.java, 定义一个类 Book, 具有以下属性和方法:  
属性: 书名(Title); 出版日期(Pdate); 字数(Words)。  
方法: 计算单价 price(): 单价=字数/1000\*35\*日期系数  
上半年的日期系数=1.2; 下半年的日期系数=1.18
- (2) 编写一个类 ComplexNumber 实现复数的运算;

### 复数类 ComplexNumber 的属性:

m\_dRealPart: 实部, 代表复数的实数部分;  
m\_dImaginPart: 虚部, 代表复数的虚数部分;

### 复数类 ComplexNumber 的方法:

ComplexNumber(double r,double i): 构造函数, 创建复数对象的同时完成复部的实部、虚部的初始化, r 为实部的初值, i 为虚部的初值;  
getRealPart(): 获得复数对象的实部;  
getImaginaryPart(): 获得复数对象的虚部;  
setRealPart( double d): 把当前复数对象的实部设置为给定的形式的数字;  
setImaginaryPart(double d): 把当前复数对象的虚部设置为给定的形式参数的数

字；

`complexAdd(ComplexNumber c)`: 当前复数对象与形式参数复数对象相加，所得的结果也是复数值，返回给此方法的调用者；

`complexMinus(ComplexNumber c)`: 当前复数对象与形式参数复数对象相减，所得的结果也是复数值，返回给此方法的调用者；

`complexMulti(ComplexNumber c)`: 当前复数对象与形式参数复数对象相乘，所得的结果也是复数值，返回给此方法的调用者；

`toString()`: 把当前复数对象的实部、虚部组合成 `a+bi` 的字符串形式，其中假设 `a` 和 `b` 分别为实部和虚部的数值。

- (3) 编写主类 `TestClass`，在主类中实例化类 `Book` 和 `ComplexNumber`，并输出实例化对象的属性。运行该程序，分析运行的结果，你觉得你学到了什么？

## 实验五 包、接口、类库与向量类

### 一、实验目的

- (1) 掌握创建包与引用包的方法；
- (2) 掌握用接口实现多重继承的机制；
- (3) 熟悉向量类的引入和使用。

### 二、实验内容

#### 1. 基本指导

##### 指导内容 1:

包的创建和引用。

##### 实验步骤:

- (1) 开机后，在 `java` 实验目录下创建 `test5` 子目录。本阶段的 `Java` 源程序都放在这个子目录中。字节码文件则根据建包的情况放在 `test5` 相应的子目录中。
- (2) 打开一个纯文本编辑器，输入如下的代码：

```
package p1;
public class DefiPackage {
    public void display(){
        System.out.println("in method display()");
    }
}
```

- (3) 将文件命名为 `DefiPackage.java`，保存在 `java` 实验目录的 `test5` 子目录下。
- (4) 打开 `MS-DOS` 窗口，转到 `DefiPackage.java` 所在的目录，键入命令：

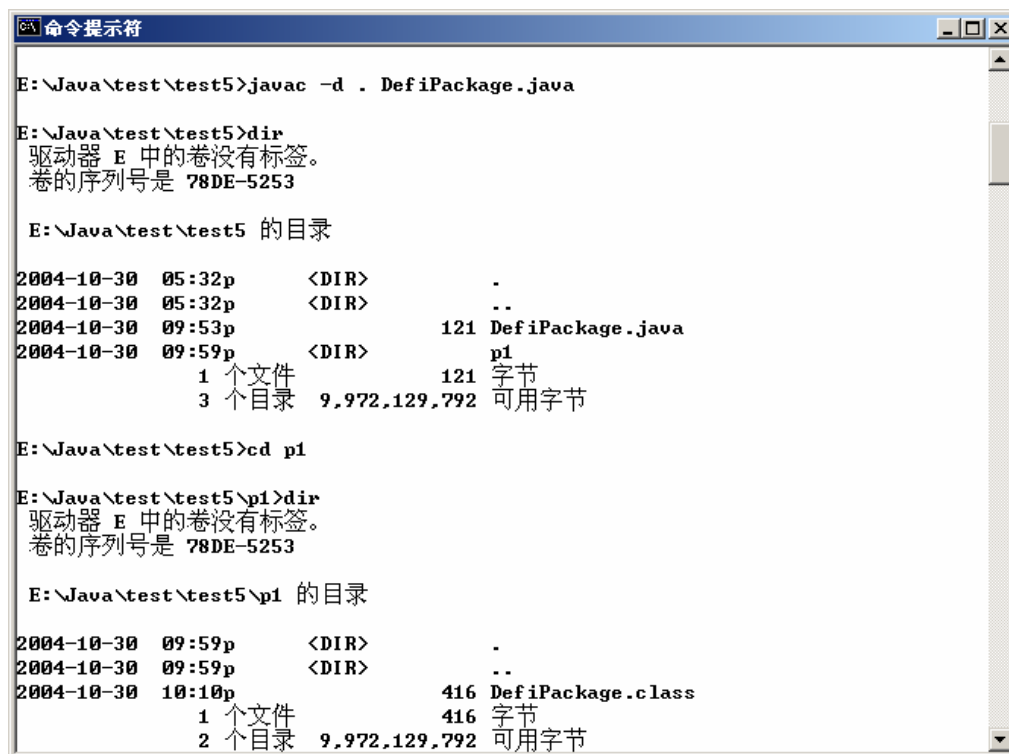
```
javac -d . DefiPackage.java
```

- (5) 键入 Dir 命令, 可以看到在 test5 子目录下创建了 p1 的子文件夹。接着键入下面的命令以查看 p1 下的文件, 可以看到 DefiPackage.class 存储在此文件夹下。

```
cd p1
```

```
dir
```

- (3)、(4)、(5) 的操作步骤如图 5-1 所示。



```
命令提示符

E:\Java\test\test5>javac -d . DefiPackage.java

E:\Java\test\test5>dir
驱动器 E 中的卷没有标签。
卷的序列号是 78DE-5253

E:\Java\test\test5 的目录

2004-10-30 05:32p    <DIR>        .
2004-10-30 05:32p    <DIR>        ..
2004-10-30 09:53p                121 DefiPackage.java
2004-10-30 09:59p    <DIR>        p1
                        1 个文件      121 字节
                        3 个目录  9,972,129,792 可用字节

E:\Java\test\test5>cd p1

E:\Java\test\test5\p1>dir
驱动器 E 中的卷没有标签。
卷的序列号是 78DE-5253

E:\Java\test\test5\p1 的目录

2004-10-30 09:59p    <DIR>        .
2004-10-30 09:59p    <DIR>        ..
2004-10-30 10:10p                416 DefiPackage.class
                        1 个文件      416 字节
                        2 个目录  9,972,129,792 可用字节
```

图 5.1

- (6) 在另一个文件中输入如下的代码:

```
import p1.DefiPackage;

public class TestPackage {

    public static void main(String[] args) {
        DefiPackage t=new DefiPackage();
        t.display();
    }
}
```

- (7) 把文件命名为 TestPackage.java, 保存在 java 实验目录的 test5 子目录下。

- (8) 编译并运行该程序, 程序的运行结果如图 5.2 所示



```
命令提示符

E:\Java\test\test5>javac TestPackage.java

E:\Java\test\test5>java TestPackage
in method display()

E:\Java\test\test5>
```

图 5.2

- (9) 在文件 TestPackage.java 中加入包定义语句: [package p2;], 重新正确地编译和运行该程序, 从中理解包的概念。

## 指导内容 2:

创建两个 Vector 类，分别记录凭证的名称和日期。并根据给定的凭证日期，查询满足条件的凭证名称，或根据给定的凭证名称，查询凭证的日期。

### 实验步骤:

- (1) 打开一个纯文本编辑器，输入如下的代码:

```
import java.util.*;
public class CreateVector {
    public static void main(String[] args) {
        Vector voucherName=new Vector();
        Vector voucherDate=new Vector(3);
        voucherName.add("收款凭证 001");
        voucherName.add("收款凭证 002");
        voucherName.add("收款凭证 003");
        voucherName.add("收款凭证 004");
        voucherDate.add("2004/01/06");
        voucherDate.add("2004/01/06");
        voucherDate.add("2004/01/08");
        voucherDate.add("2004/01/08");
        System.out.println(voucherName);
        System.out.println(voucherDate);
    }
}
```

- (2) 将文件命名为 CreateVector.java，保存在 java 实验目录的 test5 子目录下。  
(3) 编译并运行该程序，程序的运行结果如图 5-3 所示。



```
E:\Java\test\test5>javac CreateVector.java

E:\Java\test\test5>java CreateVector
[收款凭证001, 收款凭证002, 收款凭证003, 收款凭证004]
[2004/01/06, 2004/01/06, 2004/01/08, 2004/01/08]

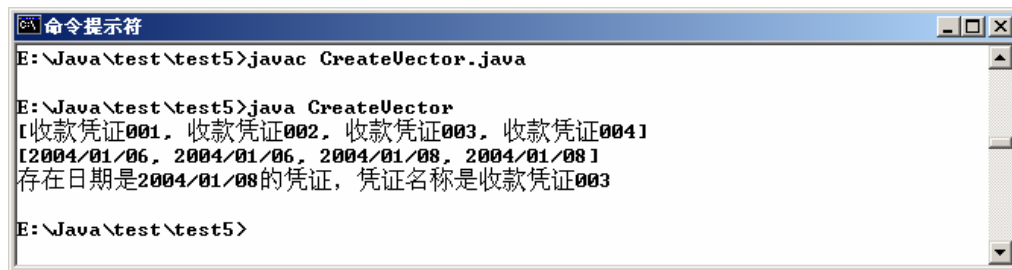
E:\Java\test\test5>
```

图 5-3

- (4) 在上面程序的 main 方法中接着添加如下的程序代码:

```
if(voucherDate.contains("2004/01/08")){
    String res="存在日期是 2004/01/08 的凭证，凭证号是"+
        voucherName.elementAt(voucherDate.indexOf("2004/01/08"));
    System.out.println(res);
}
```

- (5) 重新编译并运行该程序，程序的运行结果如图 5-4 所示。



```
命令提示符
E:\Java\test\test5>javac CreateVector.java

E:\Java\test\test5>java CreateVector
[收款凭证001, 收款凭证002, 收款凭证003, 收款凭证004]
[2004/01/06, 2004/01/06, 2004/01/08, 2004/01/08]
存在日期是2004/01/08的凭证, 凭证名称是收款凭证003

E:\Java\test\test5>
```

图 5-4

(6) 继续添加代码，查询当给定凭证名称为“收款凭证 002”时的凭证日期。

## 2. 练习思考

### 练习内容:

创建接口 Speakable 和 Runner，然后创建两个类 Dog 和 Person 实现该接口。

### 程序代码:

```
interface Speakable {
    public void speak();
}
interface Runner {
    public void run();
}
class Dog implements Speakable, Runner {
    public void speak() {
        System.out.println("狗的声音:汪、汪！");
    }
    public void run() {
        System.out.println("狗用四肢跑步");
    }
}
class Person implements Speakable, Runner {
    public void speak() {
        System.out.println("人们见面时经常说:您好！");
    }
    public void run() {
        System.out.println("人用两腿跑步");
    }
}
public class TestInterface {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.speak(); d.run();
        Person p = new Person();
        p.speak(); p.run();
    }
}
```

### 思考问题：

运行上面的程序，思考如下的问题：

- (1) 该程序编译后生成几个字节码文件？
- (2) 创建一个类 Bird（鸟），给出其声音特征，并在主类中创建一个 Bird 类的实例，输出其特征。
- (3) 如何编写抽象类代替程序中的接口，实现程序同样的功能。试比较它们的不同。

### 3. 上机作业

- (1) 创建一个名称为 Vehicle 的接口，在接口中添加两个带有一个参数的方法 start()和 stop()。在两个名称分别为 Bike 和 Bus 的类中实现 Vehicle 接口。创建一个名称为 interfaceDemo 的类，在 interfaceDemo 的 main()方法中创建 Bike 和 Bus 对象，并访问 start()和 stop()方法。
- (2) 创建一个名称为 MainPackage 的包，使它包含 ParentClass 和 SubClass。ParentClass 包含变量声明，其值从构造函数中输出。SubClass 类从父类派生而来，完成对父类变量的赋值。创建一个名称为 DemoPackage 的主类，使它不在 MainPackage 包中，在该类中创建一个 SubClass 类的对象。

## 实验六 图形界面容器及布局管理器

### 一、实验目的

- (1) 掌握 Frame 容器的使用；
- (2) 掌握 Panel 容器的使用；
- (3) 掌握主要布局管理器的用法。

### 二、实验内容

#### 1. 基本指导

##### 题目内容：

编写代码，创建标题为“基本 GUI 编程”的窗口

##### 实验步骤：

- (1) 在 Java 程序编辑器中输入如下的程序代码

```
import java.awt.*;
//创建一个 Frame 类的子类，以便创建一个框架窗体
public class BasicFrame extends Frame{
    public BasicFrame(){
        //设置窗体的标题
        this.setTitle("基本 GUI 编程");
        this.setSize(200,200);
    }
}
```



```

    }
    public static void main(String[] args) {
        BasicFrame frm=new BasicFrame();
        frm.setVisible(true);
    }
}

```

- (2) 将程序保存为 BasicFrame.java，编译运行该程序，运行结果为：

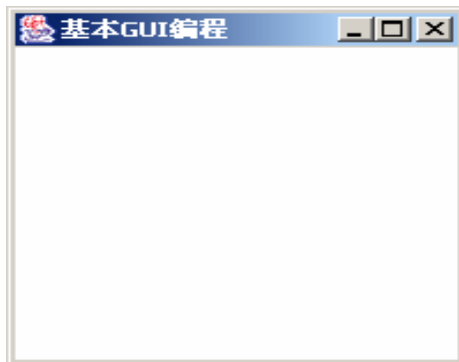


图 6-1 BasicFrame.java 输出结果

- (3) 在上面的 Frame 窗体中加入一面板 Panel，设面板的尺寸为 (80, 80)，背景色为绿色。代码如下：

```

public class BasicFrame extends Frame{
    public BasicFrame(){
        this.setTitle("基本 GUI 编程");
        this.setSize(200,200);
    }
    public static void main(String[] args) {
        BasicFrame frm=new BasicFrame();
        Panel pan=new Panel();
        frm.setLayout(null);//取消默认布局管理器
        pan.setSize(80,80);
        pan.setBackground(Color.green);
        frm.add(pan);
        pan.setLocation(40,40);
        frm.setVisible(true);
    }
}

```

- (4) 将程序保存，编译运行该程序，运行结果为：



图 6-2 BasicFrame.java 加入面板后的输出结果

## 2. 练习思考

### 题目内容:

编写代码，使用按钮排出 BorderLayout 布局的五个方向。

### 程序代码:

```
import java.awt.*;

public class BorderFrame extends Frame {

    public BorderFrame() {
        this.setTitle("BorderLayout 布局");
        this.setSize(200,200);
    }

    public static void main(String[] args) {
        BorderFrame frm=new BorderFrame();
        Button btn1=new Button("北");
        Button btn2=new Button("南");
        Button btn3=new Button("中");
        Button btn4=new Button("西");
        Button btn5=new Button("东");
        frm.add("North",btn1);
        frm.add("South",btn2);
        frm.add("Center",btn3);
        frm.add("West",btn4);
        frm.add("East",btn5);
        frm.setSize(200,200);
        frm.setVisible(true);
    }
}
```

### 思考问题:

运行上面的程序，思考下面的问题：

- (1) 如果并没有在每个位置都安排一个部件，比如将 frm.add(“West”,btn4)注释掉（在前在前面加上“//”号），程序的运行结果会怎样？
- (2) 如果在中间的位置不安排部件，程序的运行结果是怎样的呢？。
- (3) 怎样调整窗口组件间的横向和纵向间距为 10 个像素？
- (4) 根据上面对 BorderLayout 布局管理器的学习，编写一个程序，使界面如图 5I-3 所示。

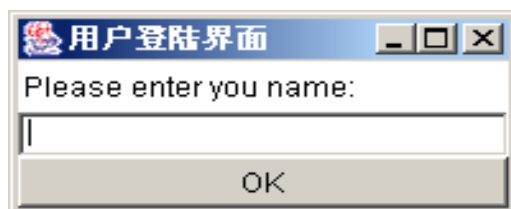


图 6-3

## 3. 上机作业

编写一个程序，模拟如图 6-4 所示的小键盘界面。编写程序。（图中的按钮类型为 JButton 类型，使用该种按钮类型的方法与 Button 类型类似，面板类型为 JPanel，其使用方法与 Panel

类似，只是需将 javax.swing.\*包引进来)

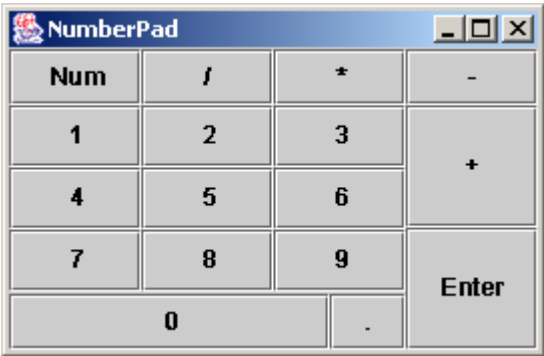


图 6-4

提示：此题要用到布局管理器的组合。

## 实验七 Java 事件处理机制

### 一、实验目的

- (1) 熟悉 JDK1.1 的事件处理机制；
- (2) 掌握处理各种鼠标与键盘事件的编程方法；
- (3) 熟悉事件适配器的使用方法。

### 二、实验内容

#### 1. 基本指导

**题目内容：**

编写程序 keyeventDemo.java。当窗体获得焦点时按下键盘，窗体中将实时的显示所按下的是哪一个键。

**实验步骤：**

- (1) 在 Java 程序编辑器中输入如下的程序代码

```
import java.awt.*;
import java.awt.event.*;
public class keyeventDemo {
public static void main(String[] args) {
    keyeventFrame frm=new keyeventFrame();
    frm.show();
}
}
class keyeventFrame extends Frame implements KeyListener{
    Label label = new Label(" ");
    public keyeventFrame() {
        setTitle("测试键盘事件");
```

```

setSize(300, 200);
Panel panel = new Panel();
add(panel);
label.setAlignment(1);
label.setFont(new java.awt.Font("Dialog", 1, 80));
panel.add(label, BorderLayout.CENTER);
//将窗体与键盘事件监听器相关联
this.addKeyListener(this);
}
//实现监听器接口中的 keyPressed、keyReleased、keyTyped 方法
public void keyPressed(KeyEvent e) {
    label.setText(""+e.getKeyChar());
}
public void keyReleased(KeyEvent e) {}
public void keyTyped(KeyEvent e) {}
}

```

(2) 编译运行上述程序，按下键盘“A”时，运行结果为：

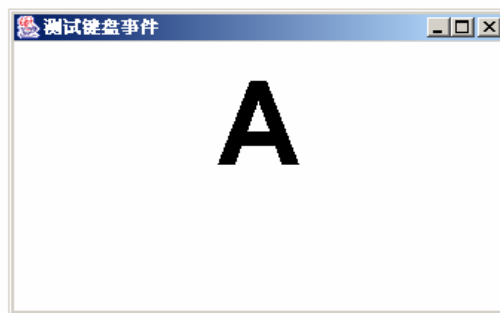


图 7-1 keyeventFrame.java 输出结果

(3) 修改上面的程序代码，以包含窗体关闭事件，并通过事件适配器简化窗体事件处理方法。在上面的源程序中加入一个用于关闭窗口的类 closeWin:

```

class closeWin extends WindowAdapter{
    public void windowClosing(WindowEvent e){
        Frame frm=(Frame)(e.getSource());
        frm.dispose();
        System.exit(0);
    }
}

```

(4) 在 keyeventFrame 类的构造函数中增加一句：

```
this.addWindowListener(new closeWin());
```

将窗体与窗体事件监听器相关联

(5) 按照通过事件适配器简化窗体事件处理的方法，用事件适配器简化键盘事件处理，简化后程序的完整代码如下：

```

import java.awt.*;
import java.awt.event.*;
public class keyeventDemo {
    public static void main(String[] args) {

```

```

        keyeventFrame frm=new keyeventFrame();
        frm.show();
    }
}
class keyeventFrame extends Frame {
    Label label = new Label(" ");
    public keyeventFrame() {
        setTitle("测试键盘事件");
        setSize(300, 200);
        Panel panel = new Panel();
        add(panel);
        label.setAlignment(1);
        label.setFont(new java.awt.Font("Dialog", 1, 80));
        panel.add(label, BorderLayout.CENTER);
        this.addKeyListener(new MykeyPressed());
        this.addWindowListener(new closeWin());
    }
}
class closeWin extends WindowAdapter{
    public void windowClosing(WindowEvent e){
        Frame frm=(Frame)(e.getSource());
        frm.dispose();
        System.exit(0);
    }
}
class MykeyPressed extends KeyAdapter{
    public void keyPressed(KeyEvent e) {
        keyeventFrame frm=(keyeventFrame)(e.getSource());
        frm.label.setText(""+e.getKeyChar());
    }
}
}

```

(6) 运行该程序，查看输出结果。

## 2. 练习思考

### 题目内容：

编写一个 Applet 程序，跟踪鼠标的移动，并把鼠标的当前位置用不同的颜色显示在鼠标所在的位置上，同时监测所有的鼠标事件，把监测到的事件名称，显示在 Applet 的状态条中。

### 程序代码：

```

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class mousemove extends Applet implements
MouseListener,MouseMotionListener
{
    int x,y; String s="";

```

```
public void init()
{
    this.addMouseListener(this);
    this.addMouseMotionListener(this);
}
public void paint(Graphics g)
{
    g.drawString(s,50,100);
    float a,b,c;
    a=(float)Math.random();
    b=(float)Math.random();
    c=(float)Math.random();
    g.setColor(new Color(a,b,c));
    g.drawString("鼠标当前的位置是:("+x+", "+y+")",x,y);
}
public void mouseClicked(MouseEvent e)
{
    x=e.getX();
    y=e.getY();
    if(e.getClickCount() ==1)
        s="您单击了鼠标";
    else if(e.getClickCount() ==2)
        s="您双击了鼠标";
    repaint();
}
public void mouseEntered(MouseEvent e)
{
    x=e.getX();
    y=e.getY();
    s="鼠标进入 Applet.";
    repaint();
}
public void mouseExited(MouseEvent e)
{
    x=e.getX();
    y=e.getY();
    s="鼠标离开 Applet.";
    repaint();
}
public void mousePressed(MouseEvent e)
{
    x=e.getX();
    y=e.getY();
    s="您按下的鼠标.";
    repaint();
}
public void mouseReleased(MouseEvent e)
{
    x=e.getX();
    y=e.getY();
    s="您松开了鼠标.";
```

```

        repaint();
    }
    public void mouseDragged(MouseEvent e)
    {
        x=e.getX();
        y=e.getY();
        s="您拖动了鼠标.";
        repaint();
    }
    public void mouseMoved(MouseEvent e)
    {
        x=e.getX();
        y=e.getY();
        s="您移动了鼠标.";
        repaint(); }
}

```

### 思考问题：

运行上面的程序（需将其字节码文件嵌入到 HTML 文件中，运行该 HTML 文件），思考下面的问题：

- (1) 如果在程序中删除事件处理方法 mouseMoved()的定义，会出现什么错误，为什么？
- (2) 观察鼠标事件处理方法 mouseReleased()是什么时候被触发的。
- (3) 如果本题只要求处理鼠标单击事件，获取鼠标单击时鼠标所在的位置，请用事件适配器简化鼠标事件处理代码？
- (4) 如何将鼠标当前的位置显示在状态栏中？

程序经过 (3) 和 (4) 步改进后，运行结果如下：

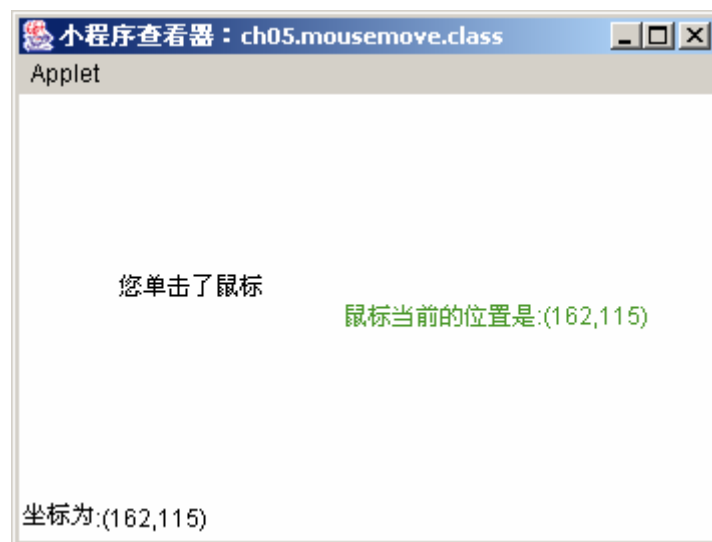


图 7-2 mousemove.java 输出结果

### 3. 上机作业

#### 题目内容：

编写一个 Applet 程序，首先捕捉用户的一次鼠标点击，然后记录点击的位置，从这个位置开始复制用户所敲击的键盘。实验一下，如果不点击鼠标而直接敲击键盘，能否捕捉到键盘事件？为什么？

## 实验八 AWT 基本组件

### 一、实验目的

- (1) 熟悉常用的 AWT 组件及其方法；
- (2) 掌握使用 AWT 组件的一般步骤。

### 二、实验内容

#### 1. 基本指导

##### 指导内容：

为文本区设置字体显示效果及前、背景色。图 8-1 为该程序某一时刻运行的效果图。

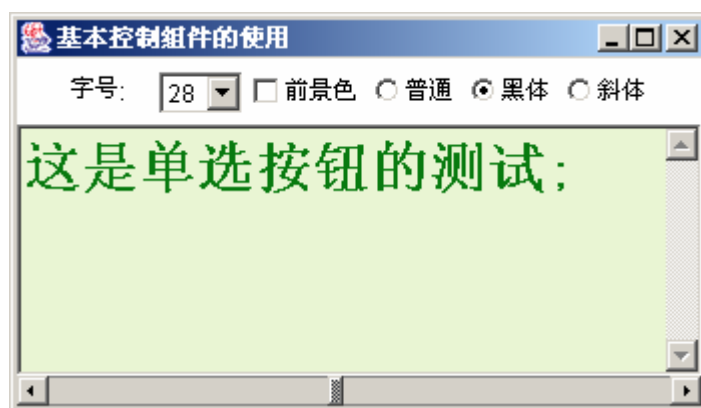


图 8-1

具体要求如下：

- 由下拉列表控制文本区文字的字号；
- 由单选按钮控制文本区文字的字型；
- 当复选框被选中时，滚动条控制文本区的前景色，否则控制文本区的背景色。

##### 实验步骤：

- (1) 开机后，在 java 实验目录下创建 test8 子目录。本阶段的 Java 源程序及编译生成的字节码文件都放在这个子目录中。
- (2) 创建一个 Frame 窗口，用来容纳 GUI 组件。新建一个 Java 文件，输入如下的代码：

```
import java.awt.*;
import java.awt.event.*;
public class TestBasicComponent extends Frame {
    public TestBasicComponent() {
        this.setSize(350,200);
        this.setTitle("基本控制组件的使用");
    }
}
```



```

public static void main(String[] args) {
    TestBasicComponent frm=new TestBasicComponent();
    frm.setVisible(true);
}
}

```

- (3) 将文件命名为 TestBasicComponent.java，保存在 java 实验目录的 test8 子目录下。
- (4) 编译并运行该文件，程序的运行结果如图 8-2 所示。

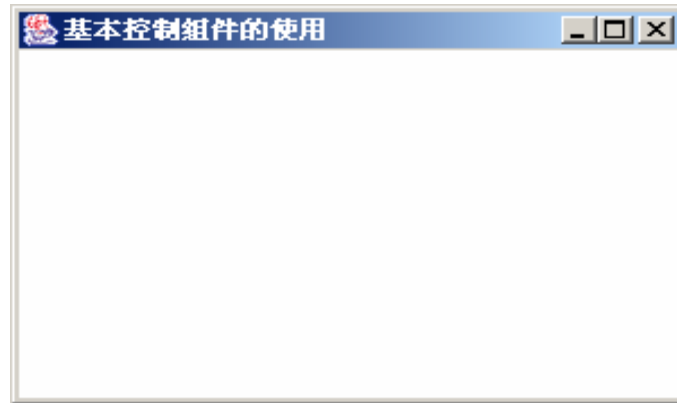


图 8-2

- (5) 为 TestBasicComponent 类添加下列成员属性的定义：

```

Label prompt;
Choice size;
Checkbox forecolor;
CheckboxGroup style;
Checkbox p,b,i;
TextArea dispText;
Scrollbar mySlider;
Panel p1;

```

- (6) 在 TestBasicComponent 类的构造函数中添加如下的代码，以将基本控制组件放在 Frame 窗口中。

```

prompt=new Label("字号:");
size=new Choice();
for(int i=10;i<40;i+=2)
    size.addItem(i+"");
forecolor=new Checkbox("前景色");
style=new CheckboxGroup();
p=new Checkbox("普通",true,style);
b=new Checkbox("黑体",false,style);
i=new Checkbox("斜体",false,style);
dispText=new TextArea("这是单选按钮的测试;",8,50);
mySlider=new
    Scrollbar(Scrollbar.HORIZONTAL,0,1,0,Integer.MAX_VALUE);
mySlider.setUnitIncrement(100);
mySlider.setBlockIncrement(100);
p1=new Panel();

```

```

p1.add(prompt);
p1.add(size);
p1.add(forecolor);
p1.add(p);p1.add(b);p1.add(i);
add("North",p1);
add("Center",dispText);
add("South",mySlider);

```

- (7) 重新编译并运行添加代码后的 TestBasicComponent.java 程序，其运行结果如图 8-3 所示。

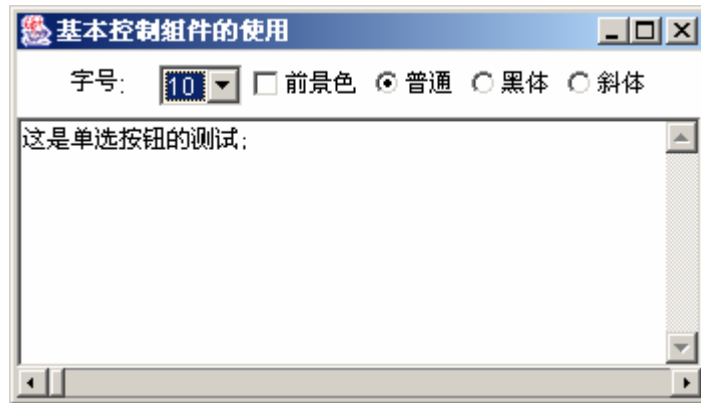


图 8-3

- (8) 在 TestBasicComponent 类中实现事件监听者接口 ItemListener 和 AdjustmentListener，类头的定义改为：

```

public class TestBasicComponent extends Frame
    implements ItemListener,AdjustmentListener

```

- (9) 为接口 ItemListener 的方法 public void itemStateChanged(ItemEvent e) 和接口 AdjustmentListener 的方法 public void adjustmentValueChanged(AdjustmentEvent e) 书写方法体，其代码添加在构造函数之后。两个方法的定义如下：

```

public void itemStateChanged(ItemEvent e) {
    Checkbox temp;
    Choice temp1;
    Font oldF=dispText.getFont();
    if(e.getItemSelectable() instanceof Checkbox) {
        temp=(Checkbox)(e.getItemSelectable());
        if(temp.getLabel()=="普通")
            dispText.setFont(new Font(oldF.getName(),Font.PLAIN,oldF.getSize()));
        if(temp.getLabel()=="黑体")
            dispText.setFont(new Font(oldF.getName(),Font.BOLD,oldF.getSize()));
        if(temp.getLabel()=="斜体")
            dispText.setFont(new Font(oldF.getName(),Font.ITALIC,oldF.getSize()));
    }
    if(e.getItemSelectable() instanceof Choice) {
        temp1=(Choice)(e.getItemSelectable());
        int s=Integer.parseInt(temp1.getSelectedItem());
        dispText.setFont(new Font(oldF.getName(),oldF.getStyle(),s));
    }
}

```

```

    }
}
public void adjustmentValueChanged(AdjustmentEvent e) {
    int value;
    if (e.getSource() == mySlider) {
        value = e.getValue();
        if (forecolor.getState() == true)
            dispText.setForeground(new Color(value));
        else
            dispText.setBackground(new Color(value));
    }
}
}

```

(10) 将基本组件注册给事件监听者。在类 TestBasicComponent 的构造函数中添加如下的代码：

```

size.addItemListener(this);
p.addItemListener(this);
b.addItemListener(this);
i.addItemListener(this);
mySlider.addAdjustmentListener(this);

```

(11) 重新编译运行程序，调整滚动条滑块的位置，并进行其它项的选择，即可得到图 8-1 所示的结果。

## 2. 练习思考

### 练习内容：

完善基本指导部分的程序。要求如下：

- (1) 修改程序，以包含窗口关闭事件。  
提示：从 WindowAdapter 类扩展类，覆盖 WindowClosing()方法。
- (2) 在窗口的滚动条的底部添加一文本域，提供给用户当前文本区前景色和背景色的值是多少？

### 思考问题：

- (1) 如果不用 TestBasicComponent 充当事件监听者，而创建一个新类来实现事件监听者接口 ItemListener 和 AdjustmentListener, 程序应如何修改？你觉得哪种实现方法更好一些。
- (2) 图 8-1 中显示了垂直滚动条和水平滚动条，它们是如何产生的？

## 3. 上机作业

编写一个 Application 程序输入学生的有关信息，用 Checkbox 表示学生是否注册，用 CheckboxGroup 表示学生的性别，用 List 表示学生的年级，用 Choice 表示学生的系别。程序还包括一个按钮，用户点击按钮时，程序读取当前所有组件中的选择并显示在一个 TextArea 中。

## 实验九 菜单及 Swing 组件

### 一、实验目的

- (1) 掌握菜单的创建及使用方法；
- (2) 掌握弹出式菜单的创建及使用方法；
- (3) 熟悉常用的 Swing 组件及其方法；
- (4) 掌握使用 Swing 组件的一般步骤。

### 二、实验内容

#### 1. 基本指导

##### 指导内容 1:

菜单、弹出式菜单的使用。具体要求如下:

- 创建一个 Frame 窗口，窗口中包括一个菜单和一个 TextArea。程序监听 ActionEvent 事件，每当用户选择一个菜单项时，TextArea 中将显示这个菜单项的名称。菜单中设置一个“退出”项，当用户选择“退出”时，关闭 Frame 并退出整个程序的执行；
- 创建弹出式菜单，该弹出式菜单含有两个菜单项。当用户右击文本区时，弹出该弹出式菜单。同样选择弹出式菜单的菜单项时，TextArea 中也将显示这个菜单项的名称；

##### 实验步骤:

- (1) 开机后，在 java 实验目录下创建 test9 子目录。本阶段的 Java 源程序及编译生成的字节码文件都放在这个子目录中。
- (2) 创建一个 Frame 窗口，用来容纳 GUI 组件。新建一个 Java 文件，输入如下的代码：

```
import java.awt.*;
import java.awt.event.*;
public class UseMenu extends Frame {
    UseMenu(){
        setTitle("菜单、对话框、弹出式菜单的使用");
    }
    public static void main(String[] args) {
        UseMenu frm=new UseMenu();
        frm.setSize(new Dimension(350,200));
        frm.setVisible(true);
    }
}
```

- (3) 将文件命名为 UseMenu.java，保存在 java 实验目录的 test9 子目录下。
- (4) 编译并运行该文件，查看程序的运行结果。其结果如图 9-1 所示。

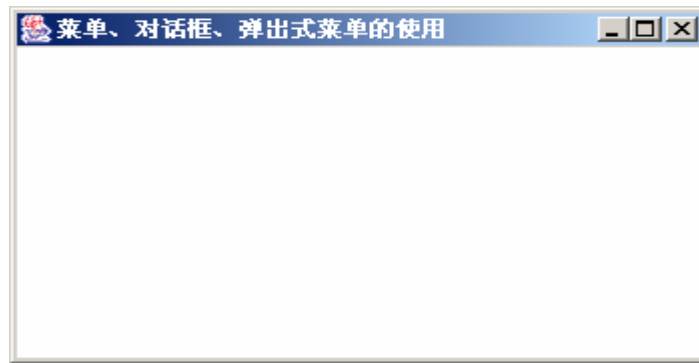


图 9-1

(5) 为 TestBasicComponent 类添加菜单及文本区组件，代码如下：

■ 在类 UseMenu 属性声明处加入如下的代码：

```
TextArea ta;
MenuBar mb;
Menu menuFile,menuEdit;
MenuItem File_Open,File_Close,File_Exit;
MenuItem Edit_Copy,Edit_Paste,Edit_Cut;
```

■ 在类 UseMenu 的构造函数中添加代码：

```
ta=new TextArea("\n\n\n\n\n\n\n\n\t\t 没有选项",5,20);
add("Center",ta);
//创建 MenuBar 对象
mb=new MenuBar();
//创建 Menu 对象
menuFile=new Menu("文件");
menuEdit=new Menu("编辑");
//创建 MenuItem 对象
File_Open=new MenuItem("打开");
File_Close=new MenuItem("关闭");
File_Exit=new MenuItem("退出");
Edit_Copy=new MenuItem("复制");
Edit_Cut=new MenuItem("剪切");
Edit_Paste=new MenuItem("粘贴");
//将 MenuItem 对象加入 Menu 对象中
menuFile.add(File_Open);
menuFile.add(File_Close);
menuFile.addSeparator() ;
menuFile.add(File_Exit);
menuEdit.add(Edit_Copy);
menuEdit.add(Edit_Cut);
menuEdit.add(Edit_Paste);
//将 Menu 对象加入 MenuBar 对象中
mb.add(menuFile);
mb.add(menuEdit);
//将菜单添加到窗口中
```

```
this.setMenuBar(mb);
```

- (6) 重新编译并运行程序，程序的运行结果如图 9-2 所示。



图 9-2

- (7) 在 UseMenu 类中实现事件监听者接口 ActionListener，类头的定义改为：

```
public class UseMenu extends Frame implements ActionListener
```

- (8) 为接口 ActionListener 的方法 public void actionPerformed(ActionEvent e) 书写方法体，其代码添加在构造函数之后。方法的定义如下：

```
public void actionPerformed(ActionEvent e){  
    if(e.getActionCommand()=="退出"){  
        dispose();  
        System.exit(0);  
    }  
    else  
        ta.setText("\n\n\n\n\t\t 你选择了: "+e.getActionCommand());  
}
```

- (9) 将基本组件注册给事件监听者。在类 TestBasicComponent 的构造函数中添加如下的代码：

```
size.addItemListener(this);  
p.addItemListener(this);  
b.addItemListener(this);  
i.addItemListener(this);  
mySlider.addAdjustmentListener(this);
```

- (10) 重新编译运行程序，选择菜单“编辑”下的“复制”菜单项，可得到图 9-3 所示的结果。

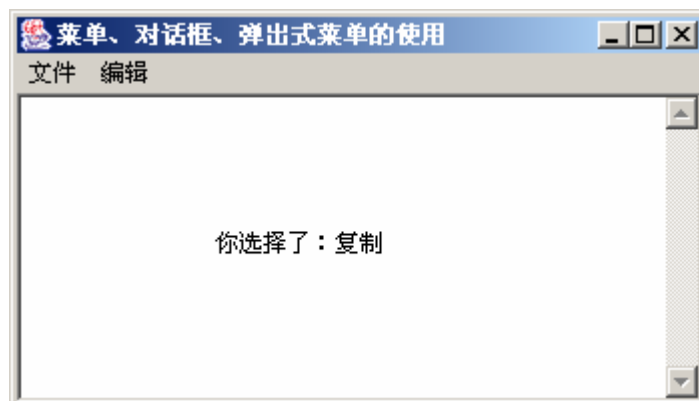


图 9-3

(11) 创建弹出式菜单，在上面的程序中继续加入代码

■ 在类 UseMenu 属性声明处加入如下的代码：

```
PopupMenu popM;
MenuItem popItem1, popItem2;
```

■ 添加类 HandleMouse 的定义，用以监听鼠标事件。代码如下：

```
class HandleMouse extends MouseAdapter{
    UseMenu m_Parent;
    HandleMouse(UseMenu mf){
        m_Parent=mf;
    }
    public void mouseReleased(MouseEvent e) {
        if(e.isPopupTrigger())
            m_Parent.popM.show((Component)e.getSource() ,e.getX(),e.getY());
    }
}
```

■ 在类 UseMenu 的构造函数中添加代码，用以创建弹出式菜单并注册事件监听者。代码如下：

```
popM=new PopupMenu();
popItem1=new MenuItem("弹出项 1");
popItem2=new MenuItem("弹出项 2");
popM.add(popItem1);
popM.add(popItem2);
ta.add(popM);
popItem1.addActionListener(this);
popItem2.addActionListener(this);
ta.addMouseListener(new HandleMouse(this));
```

(12) 完成 (11) 步骤如有的代码添加后，编译运行程序。即可得到如图 9-4 所示的运行效果。

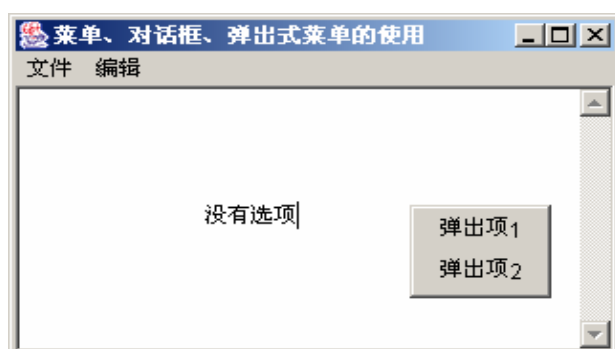


图 9-4

## 2. 练习思考

### 练习内容：

修改基本指导部分的程序，使用 Swing 组件构建程序中窗口、菜单和文本区。

### 程序代码：

```

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class UseJMenu extends JFrame implements ActionListener{
    JTextArea ta;
    JMenuBar mb;
    JMenu menuFile,menuEdit;
    JMenuItem File_Open,File_Close,File_Exit;
    JMenuItem Edit_Copy,Edit_Paste,Edit_Cut;
    JPanel p;
    UseJMenu(){
        setTitle("菜单、对话框、弹出式菜单的使用");
        ta=new JTextArea("\n\n\n\n\t 没有选项",5,20);
        //创建 MenuBar 对象
        mb=new JMenuBar();
        //创建 Menu 对象
        menuFile=new JMenu("文件");
        menuEdit=new JMenu("编辑");
        //创建 MenuItem 对象
        File_Open=new JMenuItem("打开");
        File_Close=new JMenuItem("关闭");
        File_Exit=new JMenuItem("退出");
        Edit_Copy=new JMenuItem("复制");
        Edit_Cut=new JMenuItem("剪切");
        Edit_Paste=new JMenuItem("粘贴");
        //将 MenuItem 对象加入 Menu 对象中
        menuFile.add(File_Open);
        menuFile.add(File_Close);
        menuFile.addSeparator();
        menuFile.add(File_Exit);
        menuEdit.add(Edit_Copy);
        menuEdit.add(Edit_Cut);
        menuEdit.add(Edit_Paste);
        //将 Menu 对象加入 MenuBar 对象中
        mb.add(menuFile);
        mb.add(menuEdit);
        //获得一个容器
        Container contentPane=getContentPane();
        p=new JPanel(new BorderLayout());
        p.add("Center",ta);
        contentPane.add(p);
        setJMenuBar(mb);
        //将菜单项注册给事件监听者
        File_Open.addActionListener(this);
    }
}

```



```

        File_Close.addActionListener(this);
        File_Exit.addActionListener(this);
        Edit_Copy.addActionListener(this);
        Edit_Cut.addActionListener(this);
        Edit_Paste.addActionListener(this);
    }
    public void actionPerformed(ActionEvent e)    {
        if(e.getActionCommand()=="退出"){
            dispose();
            System.exit(0);
        }
        else
            ta.setText("\n\n\n\n\t 你选择了: "+e.getActionCommand());
    }
    public static void main(String[] args) {
        UseJMenu frm=new UseJMenu();
        frm.setSize(new Dimension(350,200));
        frm.setVisible(true);
    }
}

```

#### 思考问题:

- (1) 用 JFrame 类和 Frame 类构建窗口容器有什么不同?
- (2) 用 JMenu 类和 Menu 类构建菜单有什么不同?
- (3) 用 JTextArea 类和 TextArea 类构建的文本区有什么不同?
- (4) 用 JPopupMenu 类实现基本指导部分的弹出式菜单。
- (5) 用 Swing 组件重新构建实验八中基本指导部分程序的图形用户界面。

#### 3. 上机作业

扩展基本指导部分的程序代码,当用户要关闭 Frame 时,弹出一个 Dialog 向用户确认关闭操作。Dialog 包括一个包含文字提示的 Label 和两个按钮,用户点击按钮“确认”则关闭 Frame 和整个程序,否则关闭 Dialog,返回原来的 Frame。

## 实验十 多媒体编程

### 一、实验目的

- (1) 理解 Java Applet 的工作原理;
- (2) 掌握 Java Applet 的生命周期方法;
- (3) 掌握 Graphics 类绘制各种图形的方法;
- (4) 掌握字体、颜色、图像、动画和声音的控制方法。

## 二、实验内容

### 1. 基本指导

#### 指导内容:

编写一个程序,说明 Applet 如何工作以及启动 Applet 时调用 init()、start()和 paint()方法的顺序。

#### 实验步骤:

- (1) 开机后,在 java 实验目录下创建 test10 子目录。本阶段的 Java 源程序及编译生成的字节码文件都放在这个子目录中。
- (2) 定义类 AppletDemo,此类为 Java 中 Applet 类的子类;声明三个类型为 String 类型的类变量,并定义 Applet 类的 init 方法,代码如下:

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class AppletDemo extends Applet {
    String stringMsg1,stringMsg2,stringMsg3;
    public void init(){
        setBackground(Color.yellow);
        setForeground(Color.black);
        stringMsg1="已执行 init()方法";
    }
}
```

- (3) 在类 Applet 类中定义 start 方法,其代码如下:

```
public void start(){
    stringMsg2="已执行 start()";
}
```

- (4) 在类 Applet 类中定义 paint 方法,其代码如下:

```
public void paint(Graphics graphics){
    stringMsg3="已执行 paint()方法";
    graphics.drawString(stringMsg1,10,30);
    graphics.drawString(stringMsg2,10,60);
    graphics.drawString(stringMsg3,10,90);
}
```

- (5) 将文件命名为 AppletDemo.java,保存在本次实验目录下并编译该文件。
- (6) 新建一个文件,输入如下的代码:

```
<html>
  <body>
    <applet code=" AppletDemo" width=300 height=200>
  </applet>
</body>
</html>
```

- (7) 将文件命名为 AppletDemo.html,保存在本次实验目录下。
- (8) 通过 Applet 查看器执行该 HTML 文件,命令如下:  
appletviewer AppletDemo.html

程序的输出结果如图 10-1 所示。

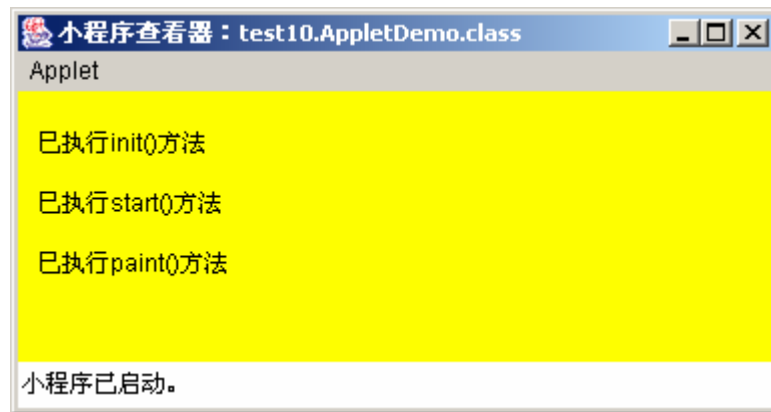


图 10-1

## 2. 练习思考

### 练习内容 1:

在鼠标单击的两点间绘制直线，可以连续绘制直线且线段的颜色为红色。

### 程序代码:

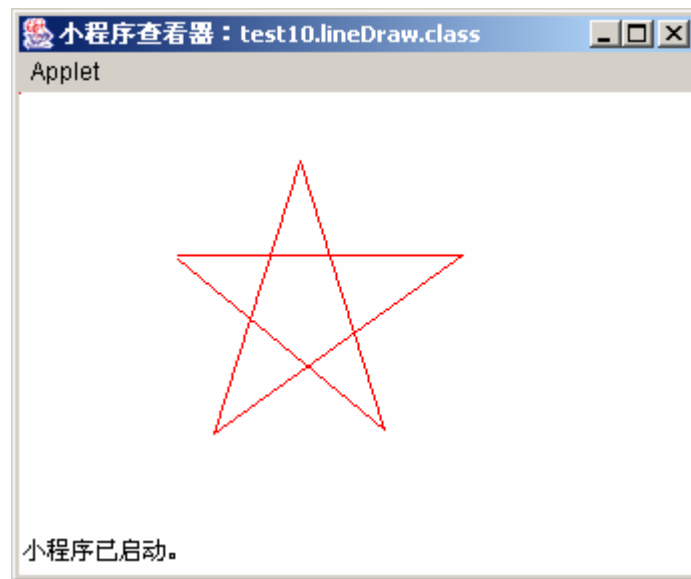
```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class lineDraw extends Applet {
    int x1=-1,y1=-1;
    boolean flag=true;
    int x2,y2;
    public void init(){
        this.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mousePressed(MouseEvent e) {
                this_mousePressed(e); }
        });
    }
    void this_mousePressed(MouseEvent e) {
        flag=!flag;
        if(flag==true){
            x1=e.getX();
            y1=e.getY();
        }
        else {
            x2=e.getX();
            y2=e.getY();
        }
        if(x1!=-1 && y1!=-1)
            repaint();
    }
    public void update(Graphics g){
        paint(g);
    }
}
```

```

    }
    public void paint(Graphics g){
        g.setColor(Color.red);
        g.drawLine(x1,y1,x2,y2);
    }
}

```

运行上面的程序，程序的运行界面如图 10-2 所示。



#### 思考问题：

- (1) 本程序是如何定位直线两端点的坐标的？
- (2) 本程序是如何处理鼠标事件的？
- (3) 程序中 update()方法的作用是什么？

#### 练习内容 2:

用 Applet 动画实现一个简单的 Applet 影集。

#### 程序代码：

```

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class ImageType extends Applet {
    int num=5;
    Image imgs[];
    public void init(){
        imgs=new Image[num];
        for(int i=0; i<num;i++)
        { imgs[i]=getImage(getDocumentBase(),"images/"+i+(i+1)+".gif");

```

```

    }
    this.setBackground(Color.white);
}
public void paint(Graphics g){
    while(true){
        for(int i=0;i<num;i++){
            g.drawImage(imgs[i],0,0,this);
            try{
                Thread.sleep(2000);
            } catch (InterruptedException e){
                e.printStackTrace();
            }
            g.clearRect(0,0,getBounds().width,getBounds().height);
        }
    }
}
}
}

```

#### 思考问题：

- (1) 这部影集里可以放几张照片？
- (2) 要使该程序正常运行，照片对应的图片文件名需怎样命名，应将它们放在什么目录下？
- (3) 在本程序中每张照片播放的时间相隔是多长？

#### 3. 上机作业

编写 Applet 程序，实现下面的功能：

- 接受用户输入指定的字号、字体和字体风格，在 Applet 上显示一段指定字体的文字；
- 接受用户输入的 R、G、B 三种颜色的分量，配置页面的背景颜色。

## 实验十一 异常处理

### 一、实验目的

- (1) 掌握异常的概念及异常处理的机制；
- (2) 掌握 try-catch-finally 异常处理语句的使用；
- (3) 熟悉用户自定义异常及处理用户自定义异常的方法。

## 二、实验内容

### 1. 基本指导

#### 指导内容:

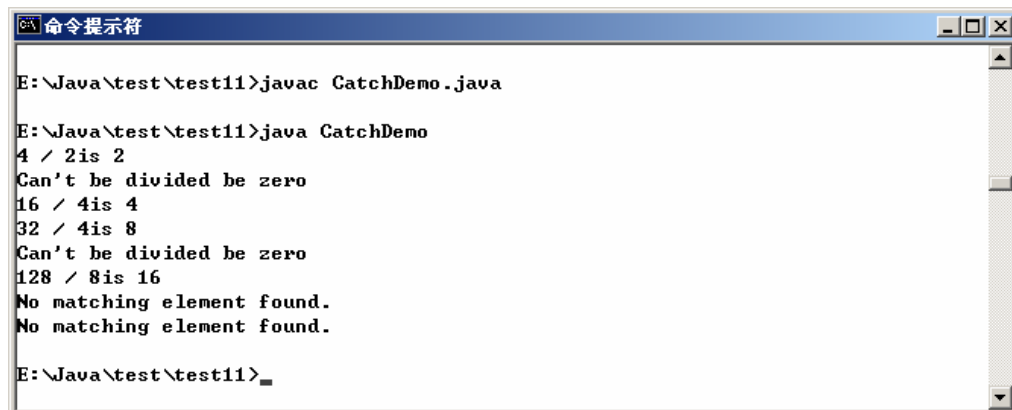
编写一个程序，同时捕获数组越界和被 0 除的异常，说明异常处理语句 try-catch-finally 的处理机制。

#### 实验步骤:

- (1) 开机后，在 java 实验目录下创建 test11 子目录。本阶段的 Java 源程序及编译生成的字节码文件都放在这个子目录中。
- (2) 新建一个 Java 文件，输入如下的程序代码：

```
public class Catch Demo {  
    public static void main(String[] args) {  
        int number[]={4,8,16,32,64,128,256,512};  
        int denom[]={2,0,4,4,0,8};  
        for(int i=0;i<number.length;i++){  
            try {  
                System.out.println(number[i] + " / " + denom[i] + "is " +  
                                    number[i] / denom[i]);  
            } catch(ArithmeticException exc){  
                System.out.println("Can't be divided be zero");  
            }  
            catch(ArrayIndexOutOfBoundsException exc){  
                System.out.println("No matching element found.");  
            }  
        }  
    }  
}
```

- (3) 将文件命名为 CatchDemo.java，保存在本次实验目录下并编译并运行该程序，程序的运行结果如图 11-1 所示：



```
命令提示符  
E:\Java\test\test11>javac CatchDemo.java  
E:\Java\test\test11>java CatchDemo  
4 / 2is 2  
Can't be divided be zero  
16 / 4is 4  
32 / 4is 8  
Can't be divided be zero  
128 / 8is 16  
No matching element found.  
No matching element found.  
E:\Java\test\test11>
```

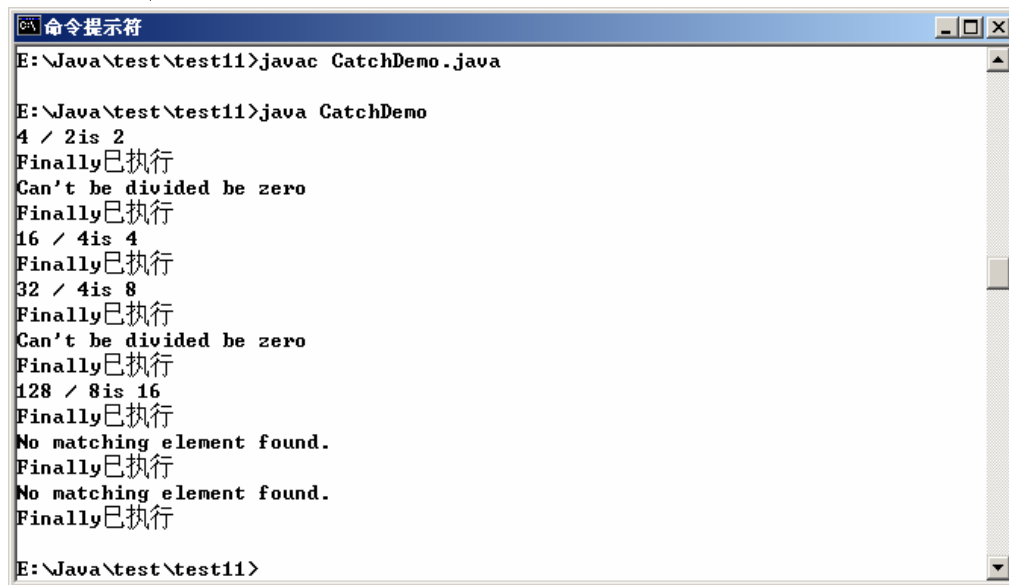
图 11-1

- (4) 为上述的异常处理添加 finally 块，其代码如下：

```
finally {  
    System.out.println("Finally 已执行");  
}
```

- (5) 重新编译运行该 java 程序，程序的运行结果如图 11-2 所示：

- (6) 试一试：如果没有异常处理，直接输出两个数组对应元素相除的结果，会出现什么样的结果，分析其原因。



```
命令提示符  
E:\Java\test\test11>javac CatchDemo.java  
  
E:\Java\test\test11>java CatchDemo  
4 / 2 is 2  
Finally已执行  
Can't be divided be zero  
Finally已执行  
16 / 4 is 4  
Finally已执行  
32 / 4 is 8  
Finally已执行  
Can't be divided be zero  
Finally已执行  
128 / 8 is 16  
Finally已执行  
No matching element found.  
Finally已执行  
No matching element found.  
Finally已执行  
E:\Java\test\test11>
```

图 11-2

## 2. 练习思考

### 练习内容：

创建用户自定义异常，用于描述数据取值范围的错误信息。

### 程序代码：

```
class UserException extends Exception {  
    private int idnumber;  
    public UserException(String message,int id){  
        super(message);  
        this.idnumber=id;}  
    public int getId(){  
        return idnumber;  
    }  
}  
  
public class TestException {  
    public void regist(int num) throws UserException {  
        if(num<0){  
            throw new UserException("人数为负值，不合理",3);  
        }  
        System.out.println("登记人数: "+num);  
    }  
    public void manager(){
```

```

        try {
            regist(-100);
        } catch (UserException e) {
            System.out.println("登记出错，类别：" + e.getId());
        }
        System.out.println("本次登记操作结束");
    }

    public static void main(String[] args) {
        TestException t = new TestException();
        t.manager();
    }
}

```

运行上面的程序，程序的运行结果如图 11-3 所示。

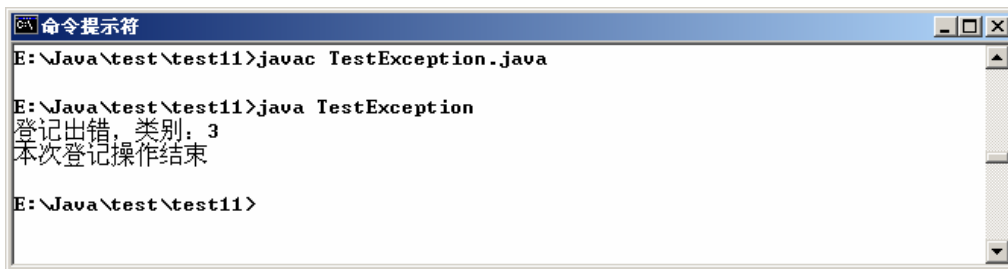


图 11-3

#### 思考问题：

- (1) 本程序中 throws 和 throw 语句的作用是什么？
- (2) 本程序中是如何定义用户自定义异常的？
- (3) 本程序是如何处理程序产生的用户自定义异常的？
- (4) 如果将程序中的 “public void regist(int num) throws MyException” 改为 “public void regist(int num) ”，会出现什么样的情况？

#### 3. 上机作业

- (1) 编写一个程序，将字符串转换成数字。请使用 try-catch-finally 语句处理转换过程中可能出现的异常。
- (2) 创建一个类 Area，用来计算长方形或正方形的面积。用于计算面积的方法是一个重载的方法，如果该方法带一个参数，则应计算正方形的面积；如果带两个参数，则应计算长方形的面积。创建一个带有 main 方法的主类，来测试 Area 类。如果传入的参数个数不对，则应通过异常处理的方法显示相应的错误信息。



## 实验十二 输入输出与文件处理

### 一、实验目的

- (1) 理解流式输入输出的基本原理；
- (2) 掌握 DataInputStream 和 DataOutputStream 类的使用方法；
- (3) 掌握 File、FileInputStream、FileOutputStream 类的使用方法；
- (4) 掌握 RandomAccessFile 类的使用方法。

### 二、实验内容

#### 1. 基本指导

##### 指导内容 1:

使用 FileInputStream 和 FileOutputStream 类将文件 Input.txt 文件中的内容复制到 output.txt 文件中。

##### 实验步骤:

- (1) 开机后，在 java 实验目录下创建 test12 子目录。本阶段的 Java 源程序及编译生成的字节码文件都放在这个子目录中。
- (2) 在本实验的目录下新建一个文本文件 input.txt，并输入几行文字。
- (3) 新建一个 Java 文件，输入如下的程序代码：

```
import java.io.*;
public class CopyFile {
    public static void main(String[] args) {
        try {
            FileInputStream fis=new FileInputStream("Input.txt");
            FileOutputStream fos=new FileOutputStream("Output.txt");
            int read=fis.read();
            while(read!=-1){
                fos.write(read);
                read=fis.read();
            }
            fis.close();
            fos.close();
        } catch(IOException e){
            System.out.println(e);
        }
    }
}
```

- (4) 将文件命名为 CopyFile.java，保存在本次实验目录下并编译并运行该程序。
- (5) 查看本实验目录下是否存在 Output.txt 文件，如果存在，查看该文件的内容是否与 Input.txt 的文件内容相同。相同则说明文件复制成功。

##### 指导内容 2:

使用 `DataOutputStream` 将 Java 基本类型数据写出到一个输出流,然后再使用 `DataInputStream` 输入流读取这些数据。

### 实验步骤:

- (1) 新建一个 Java 文件,输入如下的程序代码:

```
import java.io.*;

public class DataStream {
    public static void main(String[] args) {
        try {
            FileOutputStream fos;
            DataOutputStream dos;
            FileInputStream fis;
            DataInputStream dis;
            fos=new FileOutputStream("DataStream.txt");
            dos=new DataOutputStream(fos);
            dos.writeUTF("Java 程序设计");
            dos.writeInt(90);
            dos.close();
            fis=new FileInputStream("DataStream.txt");
            dis=new DataInputStream(fis);
            System.out.println("课程:"+dis.readUTF());
            System.out.println("分数:"+dis.readInt());
        } catch (IOException e) {
            System.out.println(e);
        }
    }
}
```

- (2) 将文件命名为 `DataStream.java`, 保存在本次实验目录下。

- (3) 编译并运行该文件,运行结果如图 12-1 所示。

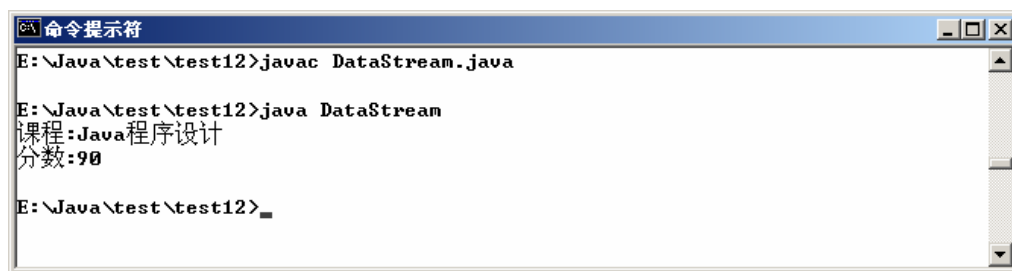


图 12-1

- (4) 查看实验目录下 `DataStream.txt` 文件的内容,分析 `DataInputStream` 和 `DataOutputStream` 的作用。

## 2. 练习思考

### 练习内容:

创建包含一个 `TextArea`、一个打开按钮和一个关闭按钮的 `Application` 程序。当用户按打开按钮时,弹出一个 `FileDialog` 以帮助用户选择要查看的文件名称,然后使用 `RandomAccessFile` 类读取选定的文件并将其显示在文本区中。

### 程序代码:

```

import java.io.*;
import java.awt.*;
import java.awt.event.*;
public class RandomAccessFileDemo {
    public static void main(String[] args) {
        new FileFrame();
    }
}
class FileFrame extends Frame implements ActionListener{
    TextArea ta;
    Button open,quit;
    FileDialog fd;
    FileFrame(){
        super("获取并显示文本文件");
        ta=new TextArea(10,45);
        open=new Button("打开");
        quit=new Button("关闭");
        open.addActionListener(this);
        quit.addActionListener(this);
        setLayout(new FlowLayout());
        add(ta);
        add(open);
        add(quit);
        setSize(350,300);
        show();
    }
    public void actionPerformed(ActionEvent e){
        if(e.getActionCommand()=="打开"){
            fd=new FileDialog(this,"打开文件",FileDialog.LOAD);
            fd.setDirectory(".");
            fd.show();
            try {
                File myfile = new File(fd.getDirectory(), fd.getFile());
                RandomAccessFile raf=new RandomAccessFile(myfile,"r");
                while(raf.getFilePointer()<raf.length()){
                    ta.append(raf.readLine()+"\n");
                }
            } catch(IOException ioe){
                System.err.println(ioe.toString());
            }
        }
        if(e.getActionCommand()=="关闭"){
            dispose();
            System.exit(0);
        }
    }
}

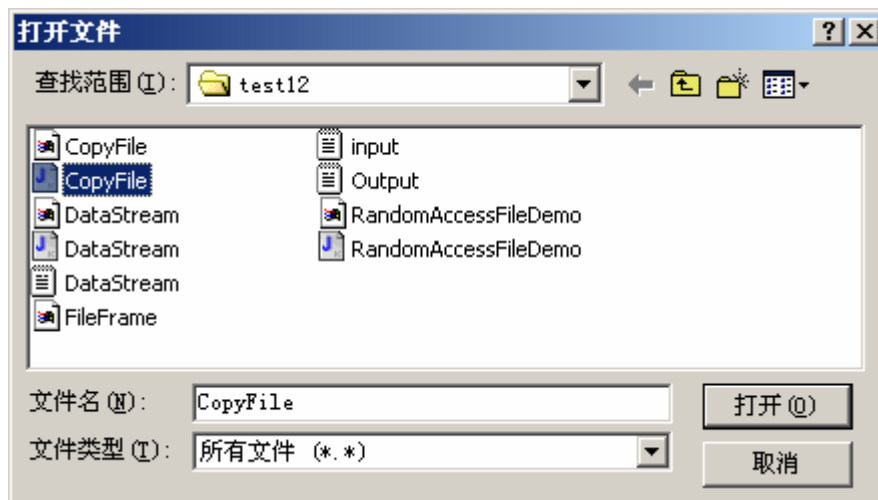
```

```

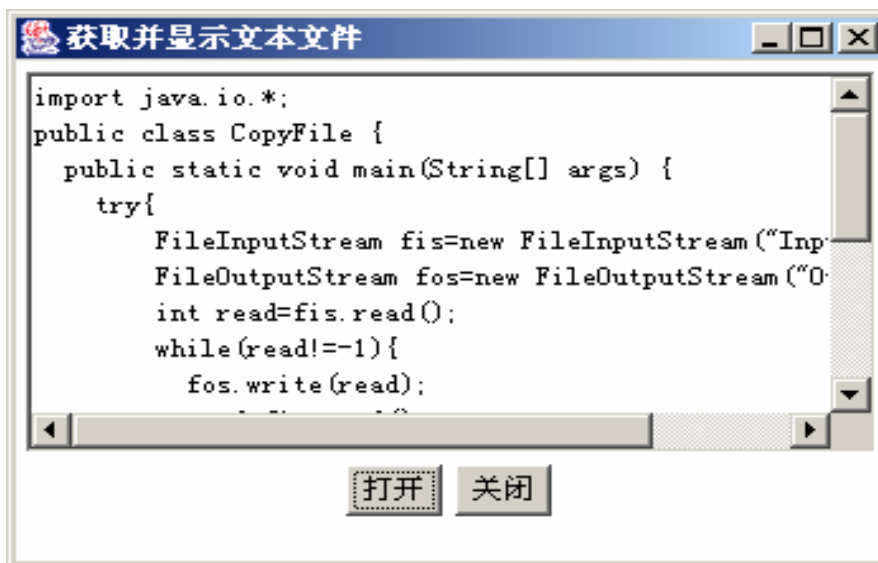
    }
}
}

```

运行上面的程序，程序的运行结果如图 12-2 所示。



(a) 使用 FileDialog 的效果



(b) 获取选定的文件的内容

图 12-2

#### 思考问题：

- (1) 文件对话框打开时的基础目录是什么？
- (2) 本程序中 File 类的作用是什么？
- (3) 如果用该程序读取带有汉字的文件时，会出现乱码，这是为什么？

- (4) 如何将本程序改为以 `FileInputStream` 类读取文本文件的内容?

### 3. 上机作业

- (1) 编写一个图形界面的 `Application` 程序, 包括分别用于输入字符串和浮点数的两个 `TextField`, 以及两个按钮 (一个是“输入”按钮, 一个是“输出”按钮) 和一个 `TextArea`。用户在两个 `TextField` 中输入数据并按“输入”按钮后, 程序利用 `DataOutputStream` 把这两个数据保存入一个文件 `file.dat` 中, 按“输出”按钮, 则把这个文件的内容利用 `DataInputStream` 读出来显示在 `TextArea` 中。
- (2) 改写上面的程序, 利用 `PrintStream` 向文件中输入数据, 则应该怎样读取数据?
- (3) 改写上面的程序, 利用 `FileDialog` 确定保存数据的文件的名称和位置。

## 实验十三 多线程

### 一、实验目的

- (1) 掌握线程与多线程的基本概念;
- (2) 掌握创建线程的两种基本方法;
- (3) 掌握 `Thread` 类的常用方法, 如 `start()`、`run()`、`stop()`、`sleep()` 等的使用;
- (4) 掌握如何编写同步代码的方法。

### 二、实验内容

#### 1. 基本指导

##### 题目内容:

创建两个线程。每个线程均输出“你好”, 接着输出线程名及消息数字, 每个线程输出 5 次“你好”, 可以查看这些消息是如何以交叉方式显示的。请按下面的步骤进行上机。

##### 实验步骤:

- (1) 在 Java 程序编辑器中输入如下的程序代码

```
class RunnableClass implements Runnable { //定义线程类
    private String mName;
    private int mCounter;
    public RunnableClass(String pName) { //定义线程构造函数
        mName = pName;
        mCounter = 0;
    }
    //实现 Runnable 接口所需的 run()方法
    public void run() {
        for(int cnt=0; cnt<5; cnt++){
            mCounter++;
        }
    }
}
```

```

        System.out.println("你好,来自"+mName+" "+mCounter);
    }
}
//定义主类，以实例化线程
public class RunnableThread {
    public static void main(String[] args) {
        Runnable objOne=new RunnableClass("第一个线程");
        System.out.println("在启动第一个线程之前");
        Thread thOne=new Thread(objOne);
        thOne.start();
        Runnable objTwo=new RunnableClass("第二个线程");
        System.out.println("在启动第二个线程之前");
        Thread thTwo=new Thread(objTwo);
        thTwo.start();
    }
}

```

(2) 将文件以 RunnableThread.java 保存，编译后其运行结果为：

```

E:\Java\test\ch10>java RunnableThread
在启动第一个线程之前
在启动第二个线程之前
你好,来自第一个线程 1
你好,来自第一个线程 2
你好,来自第一个线程 3
你好,来自第一个线程 4
你好,来自第一个线程 5
你好,来自第二个线程 1
你好,来自第二个线程 2
你好,来自第二个线程 3
你好,来自第二个线程 4
你好,来自第二个线程 5

```

图 13-1 RunnableThread.java 输出结果

## 2. 练习思考

### 题目内容：

假设某家银行，它可接受顾客的汇款，每做一次汇款，便可计算出汇款的总额。现有两个顾客，每人都分 3 次，每次 100 元将钱存入。试编写一个程序，模拟实际作业。

### 程序代码：

```

public class DespositMoney {
    public static void main(String[] args) {
        CCustomer c1=new CCustomer();
        CCustomer c2=new CCustomer();
        c1.start();
        c2.start();
    }
}

```

```

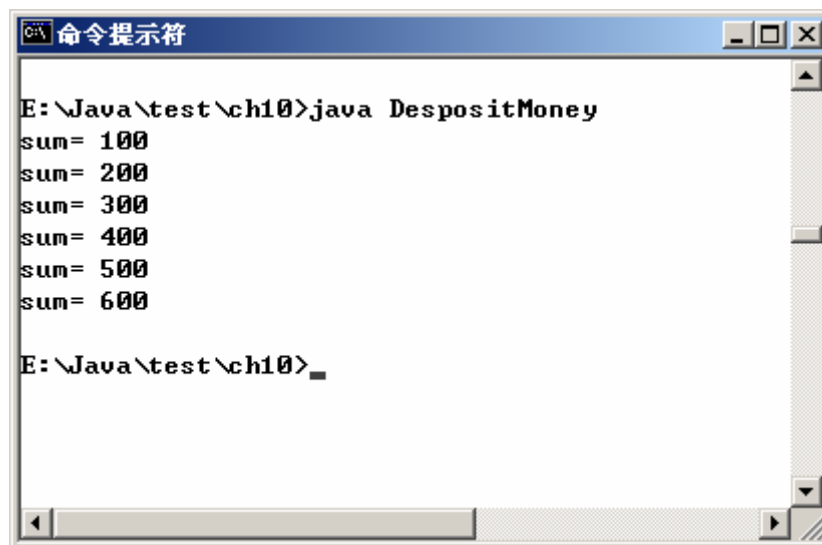
    }
}
class CBank{
    private static int sum=0;
    public static void add(int n){
        int tmp=sum;
        tmp=tmp+n;    // 累加汇款总额
        try{
            Thread.sleep((int)(10000*Math.random()));    // 小睡几秒钟
        }catch(InterruptedException e){}
        sum=tmp;
        System.out.println("sum= "+sum);
    }
}
class CCustomer extends Thread // CCustomer 类，继承自 Thread 类
{
    public void run(){    // run() method
        for(int i=1;i<=3;i++)
            CBank.add(100);    // 将 100 元分三次汇入
    }
}

```

### 思考问题：

运行上面的程序，思考下面的问题：

- (1) 程序运行结果每次是否相同，运行时间是否相同，为什么？
- (2) 要使程序运行结果每次相同，应该怎样修改程序？
- (3) 本程序使用的哪种方法创建线程，它和实验指导部分的程序创建线程的方式有何不同？
- (4) 怎样修改程序，使程序的运行结果能输出图 13-2 所示的结果。



```

命令提示符

E:\Java\test\ch10>java DespositMoney
sum= 100
sum= 200
sum= 300
sum= 400
sum= 500
sum= 600

E:\Java\test\ch10>

```

图 13-2 DespositMoney.java 输出结果

### 3. 上机作业

#### 题目内容：

(1) 编写一个 Applet 程序 ScreenProtect.java, 模拟屏幕保护程序：屏幕上自动出现由小到变换的实心圆，每个圆出现的位置和颜色都是随机的，当圆扩大到 200 像素时将其擦除，重新出现一个新的圆。

#### 运行结果：

图 13-3 为某一时刻的运行结果，圆的大小从 0 扩大到 200 后将自动擦除。



图 13-3 ScreenProtect.java 输出结果

(2) 使屏幕上一次可以显示多于一个圆，圆的个数由 HTML 的参数给定（提示：可以使用一个向量对象保存当前屏幕上所有圆的位置和大小）；

## 实验十四 网络编程基础

### 一、实验目的

- (1) 掌握用 URL 类访问网络资源的方法和步骤；
- (2) 掌握用 URLConnection 类访问网络资源的基本步骤；
- (3) 理解 Socket 通信的概念和机制；
- (4) 掌握流式 Socket 服务器和客户机的建立与通信的编程方法

### 二、实验内容

#### 1. 基本指导

##### 题目内容：

使用 URL、TextArea 在 Applet 中显示 Internet 上的文件。

##### 实验步骤：



(1) 在Java程序编辑器中输入如下的程序代码

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
import java.io.*;
import java.net.*;

public class showfile extends Applet implements
MouseListener,KeyListener {
    URL fileur;

    TextArea showarea=new TextArea("Please wait a while for get new text",10,70);
    public void mousePressed(MouseEvent e){e.consume();}
    public void mouseReleased(MouseEvent e){e.consume();}
    public void mouseEntered(MouseEvent e){e.consume();}
    public void mouseExited(MouseEvent e){e.consume();}
    public void mouseClicked(MouseEvent e){e.consume();}
    public void keyPressed(KeyEvent e){e.consume();}
    public void keyReleased(KeyEvent e){e.consume();}
    public void keyTyped(KeyEvent e){e.consume();}
    public void init(){
        String url="http://www.sina.com/index.html";
        try{
            fileur=new URL(url);
        }catch(MalformedURLException e){
            System.out.println("Can't get URL:");
        }
        showarea.addMouseListener(this);
        showarea.addKeyListener(this);
        add(showarea);
    }
    public void paint(Graphics g){
        InputStream filecon=null;
        InputStreamReader filedata=null;
        BufferedReader fnewdata=null;
        String fileline;
        try{
```

```

        filecon = fileur.openStream();
        filedata = new InputStreamReader(filecon);
        fnewdata = new BufferedReader(filedata);
        while ( (fileline = fnewdata.readLine()) != null) {
            showarea.append(fileline + "\n");
        }
    } catch (IOException e){
        System.out.println("Error in I/O:"+e.getMessage() );
    }
}
}

```

- (2) 编译上述程序，并编写相应的 HTML 文件
- (3) 运行 HTML 文件，观察并分析程序运行结果。

## 2. 练习思考

### 题目内容 1:

修改基本指导部分的程序，将程序改为使用 `URLConnection` 类获取网络的资源。

### 题目内容 2:

编写流式 Socket 服务器，在某端口建立监听服务。编写流式 Socket 的客户机，与服务器完成一次通信问答。

服务器端程序代码：

```

import java.net.*;
import java.io.*;
public class TcpServer {
    public static void main(String[] args) {
        try{
            ServerSocket s = new ServerSocket(8001);
            Socket s1 = s.accept();
            OutputStream ops = s1.getOutputStream();
            DataOutputStream dos=new DataOutputStream(ops);
            dos.writeUTF("Hello,"+s1.getInetAddress());
            ops.close();
            s.close();
            s.close();
        }
        catch(IOException e){
            System.out.println("程序运行错误:"+e);
        }
    }
}

```

客户端程序代码：

```
import java.net.*;
import java.io.*;
public class TcpClient {
    public static void main(String[] args) {
        try{
            Socket s=new Socket("127.0.0.1",8001);
            InputStream s1=s.getInputStream();
            DataInputStream dis=new DataInputStream(s1);
            System.out.println(dis.readUTF());
            dis.close();
            s.close();
        }
        catch(ConnectException e){
            System.out.println("服务器连接错误");
        }
        catch(IOException e){}
    }
}
```

#### 思考问题：

运行上面的程序，思考下面的问题：

- (1) 先运行服务器端程序，再运行客户端程序，可知客户端程序将读取服务器端发送来的问候语；修改程序，使服务器端同样能收到客户端的问候语。
- (2) 先编译运行 TcdServer.java 程序时，如果 ServerSocket.accept 方法没有发生阻塞，最可能的原因是什么？试一试将端口号由“8001”改为“80”的运行结果。
- (3) 上面的程序中，服务器端的 accept 方法只能接受一次客户端连接。怎样修改程序，才能使 accept 方法接受多个客户端连接？

### 3. 上机作业

编写一 Socket 程序完成下面的功能：

- (1) 第一种 Client 向 Server 端提供一系列的 IP 地址，Server 接受 Client 的输入，将这些 IP 地址记录下来，保存在特定的文件中，形成一个特别控制名单。
- (2) 第二种 Client 在进行网络连接前，先向 Server 询问用户欲连接的 IP 地址是否在特别控制名单之中，若 Server 端回答是则不允许这样的连接，否则协助用户完成网络连接。

2008 年 3 月