目录

[Test 1 2](#_Toc516433901)

[Test 2 2](#_Toc516433902)

[Test 3 3](#_Toc516433903)

[Test 4 4](#_Toc516433904)

[Test 5 5](#_Toc516433905)

[Test 6 5](#_Toc516433906)

[Test 7 7](#_Toc516433907)

[Test 8 8](#_Toc516433908)

[Test 9 8](#_Toc516433909)

[Test 10 9](#_Toc516433910)

[Test 11 10](#_Toc516433911)

[Test 12 11](#_Toc516433912)

[Test 13 12](#_Toc516433913)

[Test 14 13](#_Toc516433914)

[Test 15 14](#_Toc516433915)

[Test 16 14](#_Toc516433916)

[Test 17 15](#_Toc516433917)

[Test 18 16](#_Toc516433918)

[Test 19 17](#_Toc516433919)

[Test 20 18](#_Toc516433920)

[Test 21 19](#_Toc516433921)

[Test 22 20](#_Toc516433922)

[Test 23 21](#_Toc516433923)

[Test 24 23](#_Toc516433924)

[Test 25 24](#_Toc516433925)

[Test 26 25](#_Toc516433926)

[Test 27 26](#_Toc516433927)

[Test 28 27](#_Toc516433928)

[Test 29 29](#_Toc516433929)

[Test 30 30](#_Toc516433930)

[Test 31 31](#_Toc516433931)

# Test 1

**throws** System.***in*** BufferedReader readLine

import java.io.\*;

public class Test1 {

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static void main(String[] args) throws IOException {

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

InputStreamReader reader=new InputStreamReader(System.in);

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

BufferedReader in=new BufferedReader(reader);

System.out.print("请输入学校名称:");

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String name=in.readLine();

System.out.println(name);

}

}

# Test 2

FileInputStream args[0] catch close

import java.io.\*;

public class Test2 {

public static void main(String[] args) throws IOException {

int i=0;

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

FileInputStream fin=null;

try{

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

fin=new FileInputStream (args[0]);

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

}catch (Exception e){

System.out.println("Open File Error");

};

do{

i=fin.read();//Read one byte

if(i!=-1){

System.out.print((char)i);

}

}while(i!=-1);

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

fin.close();//Close the open file

}

}

# Test 3

import java.awt.\*;

import java.io.\*;

import javax.swing.\*;

public class Test3 {

public static void main(String[] args) throws IOException {

int i=0;

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

FileInputStream fin=null;

JFileChooser jf =new JFileChooser();

jf.setCurrentDirectory(new File("."));

int x=jf.showDialog(null, "打开");

String filename = jf.getSelectedFile().getAbsolutePath();

try{

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

fin=new FileInputStream (args[0]);

}catch(Exception e){

System.out.println("Open File Error");

};

do{

i=fin.read();//Read one byte

if(i!=-1){

System.out.print((char)i);

}

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

}while(i!=-1);

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

fin.close();//Close the open file

}

}

# Test 4

import java.applet.Applet;

import java.awt.\*;

import java.awt.event.\*;

public class Test4 extends Applet implements ActionListener{

TextField in1,in2;

Button btn;

int a=0, b=0, max;

public void init(){

in1=new TextField(5);

in2=new TextField(5);

btn=new Button("比较");

add(in1);

add(in2);

add(btn);

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

btn.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

a= Integer.parseInt(in1.getText());

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

b=Integer.parseInt(in2.getText());

if (a>b)

max=a;

else

max=b;

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

btn.setText("两个数中最大值是："+max);

}

}

# Test 5

**import** java.util.Random;

**public** **class** Test5{

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

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

Random random = **new** Random();

**float** p = random.nextFloat();//产生0.0与1.0之间的一个浮点数

**int** n = Math.*round*(10\*p);//构造10以内的一个整数

//\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**long** f =1; //保存阶乘的结果

**int** i = 1 ; //循环变量

**do**{

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

f=f\*i; //在每次i满足条件的情况下累计求积，此空还可以以f\*=i形式填写（第一次执行此语句时并不判断条件）

i++;

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

}**while**(i<=n);//判断条件，直到n<i为止

System.***out***.println(n+"!= "+f);

}

}

# Test 6

**public** **class** Test6 {

**int** x, y; // 点的坐标

**public** Test6() {

}// 无参数的构造方法

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**public** Test6(**int** x,**int** y) {

**this**.x = x;

**this**.y = y;

}// 带两个参数的构造方法

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**public** Test6(Test6 p) {

x = p.x;

y = p.y;

}// 带一个参数的构造方法

// 以对象的形式返回当前点的位置

**public** Test6 getLocation() {

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

Test6 p = **new** Test6(x, y);// 实例化一个Java\_3对象p

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**return** p;// 返回对象p

}

**public** **int** getX() {

**return** x;

}// 返回点的横坐标

**public** **int** getY() {

**return** y;

}// 返回点的纵坐标

**public** **void** move(**int** x, **int** y) {

**this**.x = x;

**this**.y = y;

}// 把当前点移到新的位置（x，y ）上

**public** String toString() {

**return** "(" + x + "," + y + ")";

}// 以(x，y)的格式返回点的位置

**public** **void** translate(**int** x, **int** y) {

**this**.x += x;

**this**.y += y;

}// 在原有坐标上分别增加 x和y

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

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

Test6 p = **new** Test6(5,5);// 生成一个对象(5,5)

System.***out***.println("x=" + p.x + " y=" + p.y);

System.***out***.println("Location is" + p.toString());// 以(x,y)的方式打印坐标的位置

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

p.translate(3,4);// 在原有位置上增加（3，4）

System.***out***.println("x=" + p.x + " y=" + p.y);// 打印横坐标和纵坐标的值

System.***out***.println("Location is" + p.toString());// 以(x,y)的方式打印坐标的位置

}

}

# Test 7

**import** java.io.\*;

**public** **class** Test7 {

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

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

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

InputStreamReader reader=**new** InputStreamReader(System.***in***);

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

BufferedReader in=**new** BufferedReader(reader);

System.***out***.print("请输入学校名称:");

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String name=in.readLine();

System.***out***.println(name);

}

}

# Test 8

import java.awt.\*;

import java.awt.event.\*;

public class Test8 extends Frame {

public Test8(String s) { // 重写Java\_2的构造函数，并调用父类Frame的构造函数

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

super(s);

}

public static void main(String args[]) {

Test8 fr = new Test8("Testing");

Button b = new Button("Please press me!");

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

b.addActionListener(new HandleButton());//按钮b注册事件

fr.add(b);

fr.pack();

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

fr.addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

System.exit(0);

}

});

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

fr.setVisible(true);;//设置窗体fr可见

}

}

class HandleButton implements ActionListener {

public void actionPerformed(ActionEvent e) {

System.out.println("The button is pressed!");

}

}

# Test 9

import javax.swing.JApplet; // import class JApplet

import java.awt.Graphics; // import class Graphics

public class Test9 extends JApplet

{

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

public void paint(Graphics g) //重写JApplet的方法paint

{

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

g.drawString( "欢迎你参加Java考试!", 25, 25 ); //调用Graphics类的drawString方法输出

}

}

# Test 10

**import** java.io.\*;

**public** **class** Test10 {

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

File inputFile;

File outputFile;

FileInputStream in;

FileOutputStream out;

**int** c;

String path=System.*getProperty*("java.class.path");

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

inputFile = **new** File(path + "\\" + "source.txt"); // 创建文件对象inputFile

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

outputFile = **new** File(path + "\\" + "source.txt"); // 创建文件对象outputFile

in = **new** FileInputStream(inputFile); // 将文件对象inputFile传递给构造函数

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

out = **new** FileOutputStream(outputFile); // 将文件对象outputFile传递给构造函数

**while** ((c = in.read()) != -1)

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

out.write(c); // 调用FileOutputStream的write方法，顺序输出文件

in.close();

out.close();

}

}

# Test 11

**import** javax.swing.JOptionPane;

**public** **class** Test11 {

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

String firstNumber, // 用户输入第1个字符串

secondNumber, // 用户输入第2个字符串

result; // a string containing the output

**int** number1, // 比较的第1个数

number2; // 比较的第2个数

// 读用户输入的第1个字符串 read first number from user as a string

firstNumber = JOptionPane.*showInputDialog*("Enter first integer:");

// 读用户输入的第2个字符串read second number from user as a string

secondNumber = JOptionPane.*showInputDialog*("Enter second integer:");

// 将字符串类型转换成整数类型

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

number1 = Integer.*parseInt*(firstNumber);

number2 = Integer.*parseInt*(secondNumber);

result = "";

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**if** (number1 == number2)

result = number1 + " == " + number2;

**if** (number1 != number2)

result = number1 + " != " + number2;

**if** (number1 < number2)

result = result + "\n" + number1 + " < " + number2;

**if** (number1 > number2)

result = result + "\n" + number1 + " > " + number2;

**if** (number1 <= number2)

result = result + "\n" + number1 + " <= " + number2;

**if** (number1 >= number2)

result = result + "\n" + number1 + " >= " + number2;

// 显示结果

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

JOptionPane.*showMessageDialog*(**null**, result, " 比较结果 ",JOptionPane.***INFORMATION\_MESSAGE***);

// 程序正常退出

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

System.*exit*(0);

}

}

# Test 12

import javax.swing.\*;

import java.text.\*;

public class Test12 {

public static void main(String args[]) {

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

SimpleTime t = new SimpleTime(12, 30, 19); // 创建SimpleTime类的对象t

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

JOptionPane.showMessageDialog(null, t.buildString(),"Demonstrating the \"this\" Reference", JOptionPane.INFORMATION\_MESSAGE);// 显示信息对话框

System.exit(0);

}

}

class SimpleTime {

private int hour, minute, second;

public SimpleTime(int hour, int minute, int second) {

this.hour = hour; // 将局部变量hour的值赋给成员变量hour

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

this.minute = minute; // 将局部变量minute的值赋给成员变量minute

this.second = second; // 将局部变量second的值赋给成员变量second

}

public String buildString() {

// \*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

return "this.toString(): " + this.toString() + "\ntoString(): "

+ toString() + "\nthis (with implicit toString() call): "

+ this;

}

public String toString() {

DecimalFormat twoDigits = new DecimalFormat("00");

return twoDigits.format(this.hour) + ":"

+ twoDigits.format(this.minute) + ":"

+ twoDigits.format(this.second);

}

}

**import** javax.swing.JOptionPane;

# Test 13

**public** **class** Test13 {

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

String firstNumber,secondNumber;//由用户输入的第1,2个字符串

**int** number1,number2,sum;//相加的第1、2个数和和

//输入作为字符串的第1个数值

//\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

firstNumber=JOptionPane.*showInputDialog*("Enter first integer");

secondNumber=JOptionPane.*showInputDialog*("Enter second integer");

//\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

number1=Integer.*parseInt*(firstNumber);

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

number2=Integer.*parseInt*(secondNumber);

sum=number1+number2;

//\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

JOptionPane.*showMessageDialog*(**null**, "The sum is "+ sum,"Result",JOptionPane.***PLAIN\_MESSAGE***);

System.*exit*(0);//结束应用程序

}

}

# Test 14

import java.applet.Applet;

import java.awt.Graphics;

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

public class Test14 extends Applet{

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public void paint(Graphics g) {

for(int row=1;row<=7;row++){

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

int x=220-20\*row;int y=20+20\*row;

for(int column=1;column<2\*row-1;column++)

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

g.drawString("\*", x+20\*(column-1), y);

}

}

}

# Test 15

**public** **class** Test15 {

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

**int** x[]={3,5,9,10,2,7,0,8};

**int** i=0;

**int** max=x[0],min=x[0];

**float** avg=0,s=0;

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

**for**(i=0;i<x.length ;i++){//历遍数组

s=s+x[i];//求各

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

**if**(x[i]>max){

max=x[i];//求最大值

}

**if**(x[i]<min){

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

min=x[i];//求最小值

}

}

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

avg=s/x.length;

System.***out***.println("最大值:"+max + ",最小值:" + min + "，平列值:" + avg);

}

}

# Test 16

**public** **class** Test16 {

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

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

**int** x[]=**new** **int**[5];

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

**for**(i=0;i<5;i++){

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

x[i]=(**int**)(Math.*random*()\*100) + 1;

}

System.***out***.println("排序之前的数为：");

**for**(i=0;i<5;i++){

System.***out***.print(x[i]+ ",");

}

**for**(i=0;i<5;i++){

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

**for**(j=0;j<i;j++){

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

**if**(x[j]<x[j+1]){

t=x[j];x[j]=x[j+1];x[j+1]=t;

}

}

}

System.***out***.println("\n排序之后的数为：");

**for**(i=0;i<5;i++){

System.***out***.print(x[i]+ ",");

}

}

}

# Test 17

**public** **class** Test17 {

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

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

**int** [][] x=**new** **int**[9][];

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

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

**for**(i=0;i<x.length ;i=i+1){

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

x[i]=**new** **int**[i+1];

}

**for**(i=0;i<x.length ;i++ ){

**for**(j=0;j<x[i].length ;j++){

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

x[i][j]=(i+1)\*(j+1);

System.***out***.print((i+1) + "X" + (j+1) + "=" + x[i][j] + " ");

}

System.***out***.println();//打印换行

}

}

}

# Test 18

**public** **class** Test18 {

**private** String name;

**private** String sex;

**private** **int** age;

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

Test18(String name){

setName(name);

}

Test18(String name,String sex,**int** age){

setName(name);

setSex(sex);

setAge(age);

}

**public** **void** setName(String name){

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

**this**.name =name;

}

**public** **void** setSex(String sex){

**this**.sex=sex;

}

**public** **void** setAge(**int** age){

**this**.age =age;

}

**public** String getName(){

**return** name;

}

**public** String getSex(){

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

**return** sex;

}

**public** **int** getAge(){

**return** age;

}

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

Test18 p1=**new** Test18("Z3","Men",28);

Test18 p2=**new** Test18("L4");

p2.setSex("Wormen");

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*//

p2.setAge(20);

System.***out***.println("第1个人：姓名:"+p1.getName()+",性别:" + p1.getSex()+",年龄:"+ p1.getAge());

System.***out***.println("第2个人：姓名:"+p2.getName()+",性别:" + p2.getSex()+",年龄:"+ p2.getAge());

}

}

# Test 19

**public** **class** Test19 {

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

**int**[] a = { 8, 3, 7, 88, 9, 23 };

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

LeastNumb ln = **new** LeastNumb();//定义一个LeastNumb类的对象

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ln.least(a);//将一维数组a传入least()方法

}

}

**class** LeastNumb {

**public** **void** least(**int**[] array) {

**int** temp = array[0];

**for** (**int** i = 1; i < array.length; i++) {

**if** (array[i] < temp) {

//\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

temp=array[i];//较小的值赋给临时变量temp

}

}

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

System.***out***.println("最小值为:" + temp);//输出最小值

}

}

# Test 20

public class Test20 {

void modify(int i){

i++;

}

void modify(int[] arr){

for(int i=0;i<arr.length;i++){

arr[i]=1;

}

}

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

void modify(SimpleClass s){ //定义一个成员方法s,形式参数为SimpleClass类

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

s.field =1;//为成员变量赋值

}

public static void main(String[] args) {

Test20 v=new Test20();

int i=0;

v.modify(i);

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

int[] intArr=new int[1];

intArr[0]=100;

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

v.modify(intArr);//调用对象v的modify方法，实际参数为intArr

System.out.println("intArr[0]="+ intArr[0]);

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

SimpleClass s=new SimpleClass();//创建s对象并初始化

v.modify(s);

System.out.println(s.field);

}

}

class SimpleClass{

int field;

}

# Test 21

**class** Cylinder {

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

**private** **static** **int** *num*=0;

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

**private** **static** **double** *pi*=3.14;//定义静态成员变量

**private** **double** radius;

**private** **int** height;

**public** Cylinder(**double** r,**int** h){

**this**.radius =r;

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

**this**.height =h;

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

*num*++;//num值增加1

}

**public** **void** count(){

System.***out***.println("创建了" + *num* +"个对象:");

}

**double** area(){

**return** *pi*\*radius\*radius;

}

**double** volume(){//求体积

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

**return** area()\*height;

}

}

**public** **class** Test21{

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

Cylinder v1=**new** Cylinder(3.5,7);

v1.count();

System.***out***.println("圆柱1的体积="+ v1.volume());

Cylinder v2=**new** Cylinder(2.0,3);

v1.count();

System.***out***.println("圆柱2的体积="+ v2.volume());

}

}

# Test 22

**public** **class** Test22 {

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

Student st=**new** Student();

Student st2=**new** Student("张文秀",20,"信息系");

}

}

**class** Person{

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

String name; //定义字符串成员变量name表示姓名

**int** age;

**public** Person(){

System.***out***.println("调用了无参数的构造方法Person()");

}

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** Person(String name,**int** age){//定义构造函数

System.***out***.println("调用了Person类的有参数构造方法");

**this**.name=name;

**this**.age=age;

}

**public** **void** show(){

System.***out***.println("姓名:" + name + "年龄：" + age);

}

}

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

**class** Student **extends** Person{//定义子类Student继承父类Person

**private** String department;

**public** Student(){

System.***out***.println("调用了Student类的无参数构造方法");

}

**public** Student(String name,**int** age,String dep){

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

**super**(name,age);//使用super方法调用父类的构造方法，参数是name和age

**this**.department=dep;

System.***out***.println("我是" + department + "的学生");

System.***out***.println("调用了学生类的有参数构造方法Student");

}

}

# Test 23

**import** java.text.DecimalFormat;

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

**interface** Shape{ //定义接口Shape

**public** **abstract** **double** area();

}

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

**class** Circle **implements** Shape{//定义类Circle实现接口Shape

**protected** **double** radius;

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** Circle(){ //定义类Circle的构造函数

setRadius(0);

}

**public** Circle(**double** r){

radius=r;

}

**public** **void** setRadius(**double** r){

**this**.radius=(r>=0?r:0);

}

**public** **double** getRadius(){

**return** **this**.radius;

}

**public** **double** area() {

**return** Math.***PI***\*radius;

}

}

**class** Triangle **implements** Shape{

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**protected** **double** x,y;//定义保护的成员变量x,y，类型为double

**public** Triangle(){

setxy(0,0);

}

**public** Triangle(**double** a,**double** b){

setxy(a,b);

}

**public** **void** setxy(**double** x,**double** y){

**this**.x=x;

**this**.y=y;

}

**public** **double** getx(){

**return** x;

}

**public** **double** gety(){

**return** y;

}

**public** **double** area() {

**return** x\*y/2;

}

}

**public** **class** Test23 {

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

Circle c=**new** Circle(7);

Triangle t=**new** Triangle(3,4);

String output="";

DecimalFormat p2=**new** DecimalFormat("0.00");

output+="\n 半径为" + c.getRadius() + "圆的面积：" + p2.format(c.area());

output+="\n 底为" + t.getx() + "高为" + t.gety() + "的三角形面积：" + p2.format(t.area());

System.***out***.println(output);

}

}

# Test 24

import java.awt.Container;

import javax.swing.JApplet;

import javax.swing.JTextArea;

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

public class Test24 extends JApplet{//继承JApplet类

public void init(){

JTextArea out=new JTextArea(2,20);

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

Container c=getContentPane();

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

c.add(out); //将out对象加到容器中

out.setText("边长为7的正方形面积为:" + square(7) + "\n边长为7.5的正方形面积为：" + square(7.5));

}

public int square(int x){

return x\*x;

}

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

public double square(double x){//定义方法square，形式参数为双精度类型

return x\*x;

}

}

# Test 25

**import** java.util.ArrayList;

**public** **class** Test25 {

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ArrayList<String> Notes=**new** ArrayList();//泛型类集合对象，比数组功能更强

**void** add(String s){//增加1条

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

Notes.add(s);

}

**int** getSize(){ //获取总记录数

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**return** Notes.size();

}

String getNote(**int** index){//获取指定的记录

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**return** Notes.get(index);

}

**void** delNote(**int** index){//删除指定的位置的记录

//\*\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Notes.delNote(index);

}

**void** list(){

**for**(**int** i=0;i<Notes.size();i++){

System.***out***.println(**this**.getNote(i));

}

}

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

Test25 Nb=**new** Test25();

Nb.add("第1条"); Nb.add("第2条");Nb.add("第3条");

Nb.list();

Nb.delNote(1);//删除1条记录

System.***out***.println("........");

Nb.list();

}

}

# Test 26

import java.applet.Applet;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

public class Test26 extends Applet implements ActionListener{//继承父类Applet，实现接口ActionListener

Button b1;

TextField tf1,tf2;

public void init(){

Label t1,t2,t3;

Choice c1;

this.setBackground(Color.white);

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

this.setLayout(new FlowLayout(FlowLayout.LEFT));//设置布局

t1=new Label("邮箱");

tf1=new TextField("用户名",10);

t2=new Label("@");

t2.setFont(new Font("宋体",0,18));

c1=new Choice();

c1.addItem("263.net");

t3=new Label("密码");

tf2=new TextField("\*\*\*\*\*",10);

b1=new Button("登录");

//为按钮b1注册事件监听程序

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

b1.addActionListener(this);

tf2.addActionListener(this);

this.add(t1);

this.add(tf1);

this.add(t2);

this.add(c1);

this.add(t3);

this.add(tf2);

this.add(b1);

}

public void actionPerformed(ActionEvent e) {

//单击按钮或文本行中按Enter键时

if(e.getSource()==b1 || e.getSource()==tf2){

//设置按钮b1显示“已登录”

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

b1.setLabel("已登录");

}

}

}

# Test 27

import java.awt.FlowLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.\*;

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*

class WinVarial extends JFrame{//继承父类JFrame

JTextField text;

TextListener listener;

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

public WinVarial(){//创建无参数的构造函数

init();

this.setBounds(100, 100, 150, 150);

this.setVisible(true);

this.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

}

void init(){

this.setLayout(new FlowLayout());

text=new JTextField(10);

listener=new TextListener();

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

text.addActionListener(listener);//文本框注册监听事件

this.add(text);//将文本框添加到窗口中

}

}

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

class TextListener implements ActionListener{

public void actionPerformed(ActionEvent e) {

int n=0,m=0;

String str=e.getActionCommand();

try{

n=Integer.parseInt(str);

m=n\*n;

System.out.println(n+"的平方是：" + m);

}

//\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

catch(Exception ee){

System.out.println(ee.toString());

}

}

}

public class Test27 {

public static void main(String[] args) {

new WinVarial();

}

}

# Test 28

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

public class Test28 extends JFrame implements ActionListener,ItemListener{

JTextArea output;

JScrollPane sp;

String nl="\n";

public Test28(){

JMenuBar mnb;

JMenu mn;

JMenuItem mni;

JRadioButtonMenuItem rbmni;

JCheckBoxMenuItem cbmni;

this.addWindowListener(new WindowAdapter (){

public void windowClosing(WindowEvent e){

System.exit(0);

}

});

Container c=this.getContentPane();

output =new JTextArea(6,20);

sp=new JScrollPane(output);//文本区域中放滚动条

c.add(sp,BorderLayout.CENTER);

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

mnb=new JMenuBar();//定义菜单栏

this.setJMenuBar(mnb);

mn=new JMenu("文件");

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

mnb.add(mn);//菜单栏中加入文件菜单

mni=new JMenuItem("关闭");

//\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mn.add(mni);//菜单中加入菜单项

mni.addActionListener(this);

//\*\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

mn.addSeparator();//菜单中加入分隔线

ButtonGroup gp=new ButtonGroup();//定义按钮组

rbmni=new JRadioButtonMenuItem("单选按钮菜单1");

gp.add(rbmni);

rbmni.addActionListener(this);

mn.add(rbmni);

}

public static void main(String[] args) {

Test28 f=new Test28();

f.setTitle("菜单演示");

f.setVisible(true);

f.setSize(400,200);

}

public void itemStateChanged(ItemEvent e) {

JMenuItem source =(JMenuItem)(e.getSource());

String s=nl+"" + source.getText();

output.append(s+"\n");

}

public void actionPerformed(ActionEvent e) {

JMenuItem source =(JMenuItem)(e.getSource());

String s="\n" + source.getText();

output.append(s+"\n");

}

}

# Test 29

**public** **class** Test29 {

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

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

Thread t1=**new** MyThread("thread1");//创建线程MyThread对象t1

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

Thread t2=**new** MyThread("thread2");//创建线程MyThread对象t2

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

t1.start();//启动线程t1

//\*\*\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

t2.start();//启动线程t2

System.***out***.println("主程序Main方法结束");

}

}

**class** MyThread **extends** Thread{

**public** MyThread (String str){

**super**(str);

}

**public** **void** run(){

**for**(**int** i=0;i<3;i++){

System.***out***.println(**this**.getName() + "在运行");

**try**{

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*sleep*(1000);//当前线程休眠1000毫秒

}**catch**(Exception e){}

}

System.***out***.println(**this**.getName() + "已结束");

}

}

# Test 30

**public** **class** Test30 {

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

MyThread m1=**new** MyThread("thread1");

MyThread m2=**new** MyThread("thread2");

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

Thread t1=**new** Thread(m1);//创建Thread对象t1，以m1为参数绑定

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

Thread t2=**new** Thread(m2);//创建Thread对象t1，以m2为参数绑定

t1.start();

t2.start();

}

}

//\*\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**class** MyThread **implements** Runnable{//定义类MyThread，实现接口Runnable

String name;

**public** MyThread(String str){

//\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

name=str;//把参数str赋给成员变量name

}

//\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*

**public** **void** Run() {//实现接口中的Run方法

**for**(**int** i=0;i<3;i++){

System.***out***.println(name + "在运行");

**try**{

Thread.*sleep*(1000);

}**catch**(Exception e){}

}

System.***out***.println(name + "已结束");

}

}

# Test 31

**public** **class** Test31 {

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

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

ThreadSale t =**new** ThreadSale();//创建一个实现接口的售票对象t

//用此对象t作为参数创建3个线程，第2个参数为线程名称

Thread t1=**new** Thread(t,"第一个售票窗口");

Thread t2=**new** Thread(t,"第二个售票窗口");

Thread t3=**new** Thread(t,"第三个售票窗口");

t1.start();

t2.start();

t3.start();

}

}

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**class** ThreadSale **implements** Runnable{//创建类ThreadSale实现接口Runnable

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

**private** **int** tickets=10;//创建私有成员变量tickets，类型为基本整型

**public** **void** run() {

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

**if**(tickets>0)//如果有票可售

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

System.***out***.println(Thread.*currentThread*().getName() + "出售车票第" + tickets-- + "张");

**else**

System.*exit*(0);

}

}

}

31.

**public** **class** Test31 {

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

//\*\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*

ThreadSale t =**new** ThreadSale();//创建一个实现接口的售票对象t

//用此对象t作为参数创建3个线程，第2个参数为线程名称

Thread t1=**new** Thread(t,"第一个售票窗口");

Thread t2=**new** Thread(t,"第二个售票窗口");

Thread t3=**new** Thread(t,"第三个售票窗口");

t1.start();

t2.start();

t3.start();

}

}

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*

**class** ThreadSale **implements** Runnable{//创建类ThreadSale实现接口Runnable

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*\*\*\*\*

**private** **int** tickets=10;//创建私有成员变量tickets，类型为基本整型

**public** **void** run() {

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

**if**(tickets>0)//如果有票可售

//\*\*\*\*\*\*\*Found\*\*\*\*\*\*\*\*\*

System.***out***.println(Thread.*currentThread*().getName() + "出售车票第" + tickets-- + "张");

**else**

System.*exit*(0);

}

}

}