课后习题答案

习题 1

一、选择题

1. Java 的三大平台不包括（　C　）

A、JavaME

B、JavaEE

C、Android

D、JavaSE

2. Java 的开发工具是（　C　）

A、JRE

B、JVM

C、JDK

D、JavaAPI

3. 下列叙述中正确的是（　A　）

A、在面向对象的程序设计中，各个对象之间相对独立，相互依赖性小

B、在面向对象的程序设计中，各个对象都是公用的

C、在面向对象的程序设计中，各个对象之间相互不独立，它们具有密切的关系

D、以上三种说法都不对

4. JDK 中用于存放 Java 类库文件的文件夹是（　D　）

A、bin

B、include

C、demo

D、lib

5. 下列关于 Java 语言特点的叙述中，错误的是（　C　）

A、Java 是跨平台的编程语言

B、Java 是解释执行的编程语言

C、Java 是面向过程的编程语言

D、Java 是具有健壮性和安全性的编程语言

二、是非题

1. JDK 包括 JRE 及开发工具（ √ ）
2. Java 的运行环境叫 JRE（ √ ）
3. Java 程序编译的结果（class 文件）中包含的是实际机器的 CPU 指令（ × ）
4. Java 有丰富的库供我们调用（ √ ）
5. JRE 包括 JVM 及 API（ √ ）

三、编程题

1. 参照本章的第一个例子，创建一个“Hello，World”程序，在屏幕上简单地显示这句话。注意在自己的类里只需一个方法 main()方法（main()方法会在程序启动时执行）。记住要把 main()方法设为 static 形式。用 javac 编译这个程序，再用 java 运行它。

答案：

```
public class Demo {
    public static void main(String[] args) {
        System.out.println("Hello world!"); //  控制台输出 Hello world!
    }
}
```

javac 编译命令：javac Demo.java
java 解释命令：java Demo

2. 以习题 1 的程序为基础，向其中加入注释文档。利用 javadoc，将这个注释文档提取为一个 HTML 文件，并用 Web 浏览器观看。

答案：略。

习题 2

一、选择题

1. 下列选项中为单精度数的是（ B ）
   A、-56.9
   B、7.2 f
   C、0.6
   D、071

2. 指出正确的表达式（ C ）
   A、byte b = -128;
   B、Boolean = null;
   C、long l = 0xfffL;
   D、float f = 0.63598;

3. 下列语句序列执行后， c 的值是（ C ）

```
public static void main(String[] args) {
    int a = 10, b = 3, c = 5;
    if (a == b)
        c += a;
    else
        c = ++a * c;
    System.out.println("c=" + c);
}
```

A、15
B、50
C、55
D、5

4. 下列语句序列执行后， x 的值是（ D ）

```
public static void main(String[] args) {
    int a = 2, b = 4, x = 5;
    if (a < --b)
        x *= a;
    System.out.println("x=" + x);
}
```

A、5
B、20
C、15
D、10

5. 下列语句序列执行后， num 的值是（ A ）

```
public static void main(String[] args) {
    char ch = '1';
    int num = 10;
    switch (ch + 1) {
    case '1':
        num = num + 3;
    case '2':
        num = num + 5;
    case '3':
        num = num + 6;
        break;
    default:
        num = num + 8;
    }
    System.out.println("num=" + num);
}
```

A、21

B、25
C、26
D、28

二、是非题

1. Java 中没有无符号数（ √ ）
2. Java 的 break 语句只能用来跳出循环（ × ）
3. Java 中非零即真（ × ）
4. 程序的三种基本流程是顺序、分支、循环（ √ ）
5. do.while 循环至少执行一次（ √ ）

三、编程题

1. 定义一个一维整型数组 arr，长度为 8，将数组元素的下标值赋给数组元素，最后打印输出数组中下标为奇数的元素。

```
public class Demo1 {
    public static void main(String[] args) {
        int[] arr = new int[8]; //  数组元素是默认初始化为 0
        for (int i = 0; i < arr.length; i++) {
            arr[i] = i;
        }
        for (int i = 1; i < arr.length ; i = i+2) {
            System.out.println("arr["+ i + "] = " + arr[i]);
        }
    }
}
```

2. 有部分学生的成绩是 29, 90, 56, 90, 52, 95, 83, 45, 60, 43, 78，定义一个一维整型数组，统计成绩不及格的人数。

```
public class Demo2 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int[] scores = {29, 90, 56, 90, 52, 95, 83, 45, 60, 78};
        int sum = 0,i = 0;
        while( i<scores.length){
            if( scores[i] < 60 ){
                sum++;
            }
            i++;
        }
        System.out.println(sum);
    }
}
```

3. 设定一个表示成绩的整型变量 score，当 score 等于 10 分时为冠军，当 score 大于 8 分时亚军，其它情况是季军。

```
public class Demo3 {
    public static void main(String[] args){
        int score = 10;//Scanner sc = new Scanner(System.in);
        if( score == 10){
            System.out.println("冠军");
        }else if(score > 8){
            System.out.println("亚军");
        }else{
            System.out.println("季军");
        }
    }
}
```

4. 有人走台阶若每步走 2 级，则最后剩 1 级。若每步走 3 级则最后剩 2 级。 若每步走 4 级，则最后剩 3 级。若每步走 5 级，则最后剩 4 级，若每步走 6 级，则最后剩 5 级。若每步走 7 级，则刚好不剩。试编制程序求此台阶数。

```
public class Demo4 {
    public static void main(String[] args){
        for( int i = 7; i<=1000; i++){ //7-1000
            // &&:短路：与 ：所有条件都必须 true    ||或：所有条件只要有一个为 true
            if( i % 2 == 1 && i % 3 == 2 && i % 4 == 3 && i % 5 == 4 && i % 6 == 5 && i % 7 == 0){
                System.out.println("i = "+ i);
            }
        }
    }
}
```

5. 求 1+2!+3!+...+20!的和。

```
public class Demo5 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        long fac,sum=0;             // fac:乘积的结果
        for( int i = 1; i <= 20; i++ ){ //i:4
            fac = 1;
            for( int k = 1; k <= i; k++){ //k:3
                fac = fac * k ;    //fac:2
            }
            sum = sum + fac;
        }
        System.out.println("sum = " +sum);
    }
}
```

习题 3

一、是非题

1. 在界面中要达到进度条效果，可以由 JProgressBar 进度条类的对象实现（ √ ）
2. 通常在数据库中创建视图是为了更好的使用基本表的数据以及对表的保护（ √ ）
3. 在 MySQL 中一般都需要进行数据表的创建，因为只有基本表建立起来，才能进行基于表的数据操作（ √ ）
4. 对基本表的数据操作包括增加、删除和查询等（ √ ）
5. MVC 只能用于 Java 编程中（ × ）

二、问答题

1. 什么是视图?

视图是指计算机数据库中的视图，是一个虚拟表，其内容由查询定义。同真实的表一样，视图包含一系列带有名称的列和行数据。但是，视图并不在数据库中以存储的数据值集形式存在。行和列数据来自由定义视图的查询所引用的表，并且在引用视图时动态生成。

2. 什么是 MVC 模式?

MVC 全名是 Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写，一种软件设计典范，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC 被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。

习题 4

一、选择题

1. 下列代码的运行结果是（ E ）

```
public class Demo {
    public static void main(String[] args) {
        Animal animal = new Dog();
        Cat cat = (Cat) animal;
        System.out.println(cat.noise());
    }
}

class Animal {
    public String noise() {
```

```
            return " Animal Call";
        }
    }

    class Dog extends Animal {
        public String noise() {
            return "汪汪";
        }
    }

    class Cat extends Animal {
        public String noise() {
            return "喵喵";
        }
    }
```

A、Animal Call
B、汪汪
C、喵喵
D、编译错误
E、抛出运行时异常

2. 设 Demo 为已定义的类名，下列声明 Demo 类的对象 demo 的语句中正确的是（ D ）

A、float Demo demo;
B、public Demo demo = Demo ( );
C、Demo demo = new int( );
D、static Demo demo = new Demo ( );

3. 设 i , j 为类 Test 中定义的 int 型成员变量，下列 Test 类构造方法中不正确的是（ A ）

A、void Test (int a){ i = a; }
B、Test (int a) { i = a; }
C、Test (int x, int y){ i = x; j = y; }
D、Test ( ) { i = 0; j = 0; }

4. Java 中，一个类可同时定义许多同名的方法，在这些方法的形式参数个数，类型或顺序各不相同，传值也可以各不相同。这种面向对象程序的特性称为（ B ）

A、隐藏
B、重载
C、覆盖
D、Java 不支持此特性

5. 为 Demo 类的一个无形式参数、无返回值的方法 method 书写方法头，使得使用类名 Demo 作为前缀就可以调用它，该方法头的形式为（ A ）

A、static void method()

B、public void method()

C、final void method()

D、abstract void method()

二、是非题

1. 继承可以实现代码重用，提高开发效率和可维护性（ √ ）
2. 子类在构造方法中，可以用 super 来调用父类的构造方法（ √ ）
3. 将字段用 private 修饰，从而更好地将信息进行封装和隐藏（ √ ）
4. 在类的声明中用 implements 子句来表示一个类使用某个接口（ √ ）
5. 如果省略访问控制符，则表示 private（ × ）

三、编程题

1. 定义一个类 Person，包含被封装的数据成员 name、sex、age，表示姓名、性别和年龄，为 Person 类提供一个构造方法，实现三个属性的初始化操作；并提供一个输出方法显示每个成员变量的值。

```
public class Person {
    private String name;
    private String sex;
    private int age;

    // 无参构造方法
    public Person() {
        name = "张三";
        sex = "男";
        age = 29;
    }

    public void output() {
        System.out.println("姓名:" + name + "  性别:" + sex + "  年龄:" + age);
    }

    public static void main(String[] args) {
        Person p1 = new Person();
        p1.output();
    }
}
```

2. 根据编程题第 1 题，定义 Employee 类继承自 Person 类，增加成员变量 department、positon 存放部门和位置信息。定义一个 5 参的构造方法，重写输出方法用于显示 5 个成员变量的值。定义测试类，完成一个员工对象的创建及信息输出操作。

```
public class Person {
```

```
        String name;
        String sex;
        int age;

        public Person(String name, String sex, int age) {
            super();
            this.name = name;
            this.sex = sex;
            this.age = age;
        }

        public void output() {
            System.out.println("姓名:" + name + "  性别:" + sex + "  年龄:" + age);
        }
        public static void main(String[] args) {
            Employee e = new Employee("zs", "男", 28, "开发部", "经理");
            System.out.println(e);
            e.output();
        }
    }

    class Employee extends Person {
        private String department; //  部门
        private String positon; //  位置

        public Employee(String name, String sex, int age, String department, String positon) {
            super(name, sex, age);
            this.department = department;
            this.positon = positon;
        }

        @Override
        public String toString() {
            return "姓名:" + name + "  性别:" + sex + "  年龄:" + age + "  部门:" + department
+ "  位置:" + positon;
        }

        public void output() {
            System.out.println("姓名:" + name + "  性别:" + sex + "  年龄:" + age + "  部门:"
+ department + "  位置:" + positon);
        }
    }
```

3. 定义一个二维空间的点类（Point），有横、纵坐标属性，计算两点之间距离。

```
public class Demo3 {
	public static void main(String[] args) {
		Point p1 = new Point(1, 4);
		Point p2 = new Point(2, 6);

		System.out.println(p1.distance(p2));
		System.out.println(p1.move(30, 5));
	}
}

class Point {
	private double x;
	private double y;
	public Point(double x, double y) {
		super();
		this.x = x;
		this.y = y;
	}

	public double distance(Point point) {
		return Math.sqrt((point.x - this.x) * (point.x - this.x) + (point.y - this.y) * (point.y -
this.y));
	}
//当前点的横、纵坐标移动一定距离到下一个位置
	public Point move(double x, double y) {
		return new Point(this.x + x, this.y + y);
	}

	@Override
	public String toString() {
		// TODO Auto-generated method stub
		return "(" + x + ", " + y + ")";
	}
}
```

4. 定义一个矩形类(Rectangle)用来表示矩形，定义可以改变矩形坐标位置的方法; 定义可以改变矩形宽高的方法; 定义求矩形面积的方法; 定义计算一个点是否在矩形内的方法。

```
public class Demo4 {
	public static void main(String[] args) {
		Rectangle r = new Rectangle(10, 20, 50, 50);
		r.setLocation(60, 60);
		r.setSize(20, 20);
		System.out.println(r.isInsideOfRectangle(40, 3));
		System.out.println(r.getArea());
```

```
        }
    }

    class Rectangle {
        // top, left  左上角那个顶点的坐标
        // width:  宽
        // heigth:  长
        private double top, left, width, height;

        //  构造函数
        public Rectangle(double top, double left, double width, double height) {
            this.top = top;
            this.left = left;
            this.width = width;
            this.height = height;
        }

        //  改变顶点坐标，即改变矩形坐标位置
        public void setLocation(double top, double left) {
            this.top = top;
            this.left = left;
        }

        //  改变宽，高，即改变矩形宽高
        public void setSize(double width, double height) {
            this.width = width;
            this.height = height;
        }

        //  计算面积，宽×高
        public double getArea() {
            return width * height;
        }

        //  判断某点是否在矩形内
        public boolean isInsideOfRectangle(double x, double y) {
            //  这里采用的是数学上的坐标系，即向上向右为正
            //  如果采用向下向右为正的话，则要改
            // return x > this.left && x < this.left + this.width && y < this.top +
            // this.height && y > this.top;
            //  这里点不包括在边上，如果在边上也算的话，把小于号或大于号改成小于
等于及大于等于
            return x > this.left && x < this.left + this.width && y > this.top - this.height && y
< this.top;
```

```
    }
}
```

5. 按要求编写 Java 程序。

（1）编写一个接口：Calculate，只含有一个方法 int fn(int a)；

（2）编写一个类：ClassA 来实现接口 Calculate，实现 int fn (int n)接口方法时，要求计算 1 到 n 的和；

（3）编写另一个类：ClassB 来实现接口 Calculate，实现 int fn (int n)接口方法时，要求计算 n 的阶乘（n!）；

（4）编写测试类 Test，在测试类的 Test 方法中实现接口。

```
interface Calculate {
    int fn(int n);
}

class ClassA implements Calculate {

    public int fn(int n) { // 1 到 n 的和
        int sum = 0;
        for (int i = 0; i <= n; i++) {
            sum += i;
        }
        return sum;
    }
}

class ClassB implements Calculate {

    public int fn(int n) { // n 的阶乘
        int factorial = 1;
        for (int i = 1; i <= n; i++) {
            factorial *= i;
        }
        return factorial;
    }
}

public class Demo5 {
    public static void main(String[] args) {
        Calculate calculate = new ClassA();
        System.out.println("1 到 100 的和是：" + calculate.fn(100));
        calculate = new ClassB();
        System.out.println("10 的阶乘是：" + calculate.fn(10));
    }
}
```

习题 5

一、选择题

1. 下面那一项是集合 API 中 Set 接口的特点（ D ）

A、不允许重复元素，元素有顺序

B、允许重复元素，元素无顺序

C、允许重复元素，元素有顺序

D、不允许重复元素，元素无顺序

2. 下面那一项是表示键值对概念的接口（ D ）

A、Collection

B、List

C、Set

D、Map

3. 创建一个只能存放 String 的泛型 ArrayList 的语句正确的是（ B ）

A、ArrayList<int> al = new ArrayList<int>( )；

B、ArrayList<String> al = new ArrayList<String>( )

C、ArrayList al = new ArrayList<String>( )；

D、ArrayList<String> al = new List<String>( );

4. 以下不属于 ArrayList 的方法的是（ B ）

A、add ( )

B、addFirst ( )

C、size ( )

D、addAll ( )

5. 下列声明语句错误的是（ C ）

A、List list = new ArrayList( );

B、List<String> list = new LinkedList<String> ( );

C、ArrayList al = new List( );

D、Set set = (Set)new ArrayList( );

二、是非题

1. 数组与集合可以相互转换，它们没有什么区别（ × ）

2. 由于 List list = new ArrayList(); 可以知道，List 是 Java 提供的类（ × ）

3. 泛型的本质是参数化类型，也就是说所操作的数据类型被指定为一个参数。这种参数类型可以用在类、接口和方法的创建中，分别称为泛型类、泛型接口、泛型方法（ √ ）

4. Iterator 是对 collection 进行迭代输出的迭代器，它可以遍历并选择序列中的对象（ √ ）

5. Vector 与 ArrayList 一样，Vector 本身也属于 List 接口的子类，所以使用 Vector 和使用 ArrayList 都一样，适用于同样的场合（ × ）

三、编程题

1. 从控制台输入若干个字符串（输入回车结束）放入集合 Stack 中，并将它们输出到控制台，要求最先放进来的最后输出，最后放进来的最先输出。

```
import java.util.*;
public class Demo1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String s = "";
        Stack<String> stack = new Stack<String>();
        while (true) {
            System.out.println("输入字符串");
            s = sc.nextLine();
            if (s.equals("")) {
                break;
            }
            stack.push(s);
        }
        for (int i = stack.size() - 1; i > -1; i--) {
            System.out.println(stack.get(i));
        }
    }
}
```

2. 在 ArrayList 中存储以下元素：yeric、bob、muss、Shirley、nishi binai，将集合中的元素排序，并将排序后的结果输出。

```
import java.util.*;
public class Demo2 {
    public static void main(String[] args) {
        ArrayList<String> arrayList = new ArrayList<String>();
        arrayList.add("yeric");
        arrayList.add("bob");
        arrayList.add("muss");
        arrayList.add("shirley");
        arrayList.add("nishi");
        arrayList.add("binai");
        Collections.sort(arrayList);
        System.out.println("sorted ArrayList: " + arrayList);
    }
}
```

3. 从键盘输入一个字符串，要求去除重复字符后输出。

```
import java.util.*;
public class Demo3 {
	//方法一:集合
	// 方法二：字符串自带方法操作
	public	void str2() {
			Scanner sc = new Scanner(System.in);
			System.out.println("输入字符串");
			String s = sc.nextLine();
			String s1="";
			for(int i=0;i < s.length();i++) {
//				if(!s1.contains(String.valueOf(s.charAt(i)))) {
//					s1+=s.charAt(i);
//				}
				//或者
				if(s1.indexOf(s.charAt(i))==-1) {
					s1+=s.charAt(i);
				}
			}
			System.out.println(s1);
	}
		public static void main(String[] args) {
			Scanner sc = new Scanner(System.in);
			System.out.println("输入字符串");
			String s = sc.nextLine();
			Set<Character> set = new HashSet<>();
			for(int i=0;i < s.length();i++) {
				set.add(s.charAt(i));
			}

			Iterator<Character> it = set.iterator();
			while(it.hasNext()) {
				System.out.print(it.next());
			}

			//str2();
		}
}
```

4. 生成 10 个 1 到 20 之间的不重复的随机数，要求使用 HashSet 和 Iterator。

```
import java.util.*;

public class Demo4 {
	public static void main(String[] args) {
		Set<Integer> set = new HashSet<>();
```

```
            while (true) {
                if (set.size() == 10)
                    break;
                double d = Math.ceil((Math.random()) * 20);
                set.add((int) d);
            }
            Iterator<Integer> it = set.iterator();
            while (it.hasNext()) {
                System.out.println(it.next());
            }
        }
    }
```

5. 不用循环语句，把 List<String>集合中的重复元素去除。

```
import java.util.*;

public class Demo5 {
    public static void main(String[] args) {
        List<String> list = new ArrayList<>();
        list.add("abc");
        list.add("acd");
        list.add("abc");
        list.add("aaa");
        LinkedHashSet<String> set = new LinkedHashSet<>();//插入操作比较多
        set.addAll(list);
        list.clear();
        list.addAll(set);
        System.out.println(list);
    }
}
```

习题 6

一、选择题

1. 下列关于异常的说法正确的是（ D ）

   A、异常是运行时出现的错误

   B、异常是编译时的错误

   C、异常就是程序错误，程序错误就是异常

   D、以上都不对

2. 在 Java 的一个异常处理中，哪个语句块可以有多个（ B ）

A、try
B、catch
C、finally
D、throws

3. 所有异常类的父类是（ C ）
A、Error
B、Exception
C、Throwable
D、IOException

4. 使用 catch(Exception e)的好处是（ D ）
A、执行一些程序
B、忽略一些异常
C、只会捕获个别类型的异常
D、捕获 try 语句块中产生的所有类型的异常

5. 下面这段程序，试判断哪个是正确的结果（ D ）

```
public class Demo {
   public static void main(String args[]) {
        try {
            System.out.print("我在 try 块里");
        } finally {
            System.out.println("我在 finally 块里");
        }
   }
}
```

A、无法编译，因为没有指定异常
B、无法编译，因为没有 catch 子句
C、我在 try 块里
D、我在 try 块里我在 finally 块里

二、是非题

1. 捕获异常，使用关键字 catch（ √ ）
2. finally 语句是指没有异常出现时要执行的语句（ × ）
3. 逻辑错可以由编译器发现（ × ）
4. 若父类中的方法声明了 throws 异常，则子类 Override 时一定也要 throws 异常（ √ ）
5. catch 多个异常时，子类异常要排在父类异常的后面（ × ）

三、编程题

1. 输出一个数组的所有元素，捕获数组下标越界异常。

```
import java.util.*;
```

```
public class Demo1 {
    public static void main(String[] args) {
        String[] str = { "1", "2" };
        try {
            for (int i = 0; i < 3; i++) {
                System.out.println(str[i]);
            }
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("数组越界!");
        }
    }
}
```

2. 编写一个方法，接收两个数并做除法，并捕获除数为 0 异常。

```
import java.util.*;
public class Demo2 {
    public static void main(String[] args) {
        try {
            System.out.println(ResultImpl.getArith(Double.parseDouble("1"),
Double.parseDouble("0")));
        } catch (Exception e) {
            System.out.println("操作是：除以 0 的除法");
            e.printStackTrace();
        }
    }
}

class ResultImpl {
    public static double getArith(double a, double b) throws ArithmeticException {
        ArithmeticException ex = new ArithmeticException("对不起，除数为 0");
        if (b == 0) {
            throw ex;
        }
        return a / b;
    }
}
```

3. 判断一个方法的参数是否是数字，并捕获可能出现的异常。

```
import java.util.*;
import javax.swing.JOptionPane;

public class Demo {
    public static void main(String[] args) {
        int a = 0;
        try {
```

```
                a = Integer.parseInt(JOptionPane.showInputDialog(null, "输入一个整数：
"));
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(null, "非法数值输入,请输入一个整数");
            }
            fn(a);
        }

        public static void fn(int n) {
            System.out.println(n);
        }
    }
```

4. 从控制台输入 5 个整数，并输出。如果输入的不是整数，需要捕获异常，显示：“请输入整数”。

```
    import java.util.*;

    public class Demo4 {
        public static void main(String[] args) {
            Scanner num=new Scanner(System.in);
            System.out.println("请输入 5 个整数");
            int [] ale = new int [5];
            try{
                for(int i=0;i<5;i++){
                    ale[i]=num.nextInt();
                }
                for(int i=0;i<5;i++){
                    System.out.println(ale[i]);
                }

            }
            catch (ArrayIndexOutOfBoundsException e) {
                // TODO: handle exception
                System.out.println("请输入 5 个整数");
                e.getMessage();
                e.printStackTrace();

            }
            catch (InputMismatchException e) {
                // TODO: handle exception
                System.out.println("请输入整数");
                e.printStackTrace();
            }

        }
```

```
}
```

5. 编写一个方法，它有三个参数：a、b、c，判断三个参数能否构成三角形，如果不能则抛出异常。如果可以构成三角形，则计算此三角形的周长。

```
import java.util.*;

public class Demo5 {
    public static void main(String[] args) {
        int a[] = new int[3];
        System.out.println("输入三个整数");
        for (int i = 0; i < 3; i++) {
            Scanner sc = new Scanner(System.in);
            a[i] = sc.nextInt();
        }
        Arrays.sort(a);//  数组默认的从小到大排序

        try {
            ExceptionTest.triangle(a[0], a[1], a[2]);
        } catch (IllegalArgumentException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

class ExceptionTest {
    public static void triangle(int a, int b, int c) throws Exception {
        if (a + b > c && c - a < b) {
            System.out.println("能构成三角形");
            System.out.println("a=" + a + "," + "b=" + b + "," + "c=" + c);
            System.out.println("此三角形的周长:" + (a+b+c));
        } else {
            throw new IllegalArgumentException("a=" + a + "," + "b=" + b + "," + "c=" +
c + "  不能构成三角形");
        }
    }
}
```

习题 7

一、选择题

1. java.awt 包提供了基本的 Java 程序的 GUI 设计工具，包含控件、容器和（ A ）

A、布局管理器

B、数据传送器

C、图形和图像工具

D、用户界面构建

2. 下面说法正确的是（ B ）

A、BorderLayout 是 JPanel 面板的缺省布局管理器

B、FlowLayout 是 JPanel 面板的缺省布局管理器

C、一个面板不能被加入到另一个面板中

D、在 BorderLayout 中，添加到 NORTH 区的两个按钮将并排显示

3. 下列哪种 Java 组件作为容器组件（ D ）

A、JList 列表框

B、JButton 按钮

C、JMenuItem 菜单项

D、JPanel 面板

4. 下列方法中，可以改变容器布局的方法是（ A ）

A、setLayout(layoutManager)

B、addLayout(layoutManager)

C、setLayoutManager (layoutManager)

D、addLayoutManager (layoutManager)

5. 下列叙述中，正确的是（ C ）

A、类 JTextComponent 继承了类 JTextArea

B、类 JTextArea 继承了 JTextField

C、类 JTextField 继承了类 JTextComponent

D、类 JTextComponent 继承了类 JTextField

二、是非题

1. JFrame 是 Frame 的子类（ √ ）

2. JFrame 的默认布局是 BorderLayout（ √ ）

3. 在使用 BorderLayout 时，最多可以放入 5 个组件（ √ ）

4. JOptionPane 能实现弹出消息对话框的功能（ √ ）

5. 容器是用来组织其它界面成分和元素的单元，它不能嵌套其它容器（ × ）

三、编程题

1. 编写程序，创建一个窗口，在窗口底部设置两个并排的按钮，按钮上分别标注“1”、“2”字样。

```
import java.awt.*;
import javax.swing.*;

public class Demo1 extends JFrame {
```

```
    public Demo1() {
        JButton jbt1 =new JButton("1");
        JButton jbt2 =new JButton("2");
        // 创建面板 jp1,jp2
        JPanel jp1 = new JPanel();
        JPanel jp2 = new JPanel();

        // 往面板对象中添加按钮组件
        jp1.add(jbt1);
        jp1.add(jbt2);

        this.add("South",jp1);
        this.add(jp2);
        this.setTitle("my java window");
        this.setSize(300, 400);// 设置窗体大小
        this.setBounds(300, 200, 500, 400); // 设置窗体的位置和大小(x,y,width,height)
//      this.pack();
        this.setVisible(true); // 设置窗体可见
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        new Demo1(); // 创建窗体
    }
}
```

2. 编写程序，创建一个窗口，在窗口中部有三个组件，一个是标签，写着“请输入正整数”，一个是文本框，一个是“确定”按钮。要求使用网格布局把这三个组件设置为 3 行 1 列布局。

```
import java.awt.*;
import javax.swing.*;

public class Demo2 extends JFrame {
    public Demo2() {
        JLabel jlb = new JLabel("请输入正整数");
        JTextField jtf = new JTextField(10);
        JButton jbt1 = new JButton("确定");
        // 创建面板 jp1,jp2,jp3
        JPanel jp1 = new JPanel();
        JPanel jp2 = new JPanel();
        JPanel jp3 = new JPanel();

        // 往面板对象中添加按钮组件
        jp1.add(jlb);
        jp2.add(jtf);
```

```
        jp3.add(jbt1);

        this.add(jp1);
        this.add(jp2);
        this.add(jp3);
        this.getContentPane().setLayout(new GridLayout(3, 1, 3, 3));
        this.setTitle("my java window");
        this.setSize(300, 400);// 设置窗体大小
        this.setBounds(300, 200, 500, 400); // 设置窗体的位置和大小(x,y,width,height)
        // this.pack();
        this.setVisible(true); // 设置窗体可见
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        new Demo2(); // 创建窗体
    }
}
```

3. 编写程序，创建一个简洁的计算器界面。

```
import java.awt.event.* ;
import java.util.Arrays;
import java.awt.* ;
import javax.swing.* ;

public class Demo3 extends JFrame {

    JTextField txtResult;
    boolean firstDigit = true;// 用于判断是否是数字
    String operator = "=";// 先初始化为等号，等到执行相应运算时再更改
    boolean operateValidFlag = true;// 判断除数是否为 0
    double resultNum = 0.0;// 可以暂存目前的最终结果

    public Demo3() {
        setTitle("计算器");
        setSize(240, 270);
        setResizable(false);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        Container contentPane = this.getContentPane();
        contentPane.setLayout(new BorderLayout(1, 5));
        JPanel pnlNorth = new JPanel();
        JPanel pnlCenter = new JPanel();
```

```
        pnlNorth.setLayout(new BorderLayout());
        pnlCenter.setLayout(new GridLayout(4, 4, 3, 3));

        Font font = new Font("Times Roman", Font.BOLD, 20);

        contentPane.add(BorderLayout.NORTH, pnlNorth);
        contentPane.add(BorderLayout.CENTER, pnlCenter);

        txtResult = new JTextField();
        txtResult.setFont(font);
        txtResult.setEnabled(false);
        JButton btnClear = new JButton("C");
        btnClear.setFont(font);
        pnlNorth.add(BorderLayout.CENTER, txtResult);
        pnlNorth.add(BorderLayout.EAST, btnClear);

        String[] captions = { "7", "8", "9", "+", "4", "5", "6", "-", "1", "2", "3", "*", "0", ".",
"/", "=", };
        for (int i = 0; i < captions.length; i++) {
            JButton btn = new JButton(captions[i]);
            btn.setFont(font);
            pnlCenter.add(btn);

        }
    }

    public static void main(String[] args) {
        JFrame frame = new Demo3();
        frame.setVisible(true);
    }
}
```

4. 编写程序，创建一个窗口，在窗口中部创建一个下拉菜单项，菜单项包括“添加”和“删除”，窗口上部设置一个文本框，在窗口底部设置两个并排的按钮，两个按钮分别标注“添加”和“删除”。

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Demo4 extends JFrame {
    private String[] arr = { "添加", "删除" };
    private JTextField jtf = new JTextField(10);
    private JComboBox jcb = new JComboBox();
    private JButton jbt1 = new JButton("添加");
    private JButton jbt2 = new JButton("删除");
```

```
    private int count = 0;
    private JPanel jp1 = new JPanel();
    private JPanel jp2 = new JPanel();
    private JPanel jp3 = new JPanel();

    public Demo4() {
        for (int i = 0; i < arr.length; i++)
            jcb.addItem(arr[count++]);
        jtf.setEditable(false);
        jcb.setEditable(true);
        jp1.add(jbt1);
        jp1.add(jbt2);
        jp2.add(jcb);
        jp3.add(jtf);

        this.add(jp3);
        this.add(jp2);
        this.add(jp1);
        this.getContentPane().setLayout(new GridLayout(3, 1, 3, 3));
        this.setTitle("my java window");
        this.setSize(300, 400);
        this.setBounds(300, 200, 500, 400);
        this.setVisible(true);
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        new Demo4();
    }
}
```

5. 编写程序，利用标签、文本框、复选框和单选按钮等，制作一个填写学生信息的表单，包括学生学号、学生姓名、学生年龄、学生性别、兴趣爱好等。

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class Demo5 extends JFrame {
    private String[] arr = { "足球", "跳绳", "跑步" };
    private JLabel jLabelId = new JLabel("学生学号"); //  学生学号
    private JLabel jLabelName = new JLabel("学生姓名"); //  学生姓名
    private JLabel jLabelAge = new JLabel("学生年龄"); //  学生年龄
    private JLabel jLabelSex = new JLabel("学生性别"); //  学生性别
    private JLabel jLabelHobby = new JLabel("兴趣爱好"); //  兴趣爱好
    private JTextField jtfId = new JTextField(10);
```

```
private JTextField jtfName = new JTextField(10);
private JTextField jtfAge = new JTextField(10);
private JComboBox jcb = new JComboBox(arr);
private JRadioButton jrb1 = new JRadioButton("男");
private JRadioButton jrb2 = new JRadioButton("女");

private JButton jbt1 = new JButton("添加");
private JButton jbt2 = new JButton("清空");
private int count = 0;
private JPanel jp1 = new JPanel();
private JPanel jp2 = new JPanel();
private JPanel jp3 = new JPanel();
private JPanel jp4 = new JPanel();
private JPanel jp5 = new JPanel();
private JPanel jp6 = new JPanel();
public Demo5() {
	jp1.add(jLabelId);
	jp1.add(jtfId);
	jp2.add(jLabelName);
	jp2.add(jtfName);
	jp3.add(jLabelAge);
	jp3.add(jtfAge);
	jp4.add(jLabelSex);
	jp4.add(jrb1);
	jp4.add(jrb2);
	jp5.add(jLabelHobby);
	jp5.add(jcb);
	jp6.add(jbt1);
	jp6.add(jbt2);
	this.add(jp1);
	this.add(jp2);
	this.add(jp3);
	this.add(jp4);
	this.add(jp5);
	this.add(jp6);
	this.getContentPane().setLayout(new GridLayout(6, 2, 3, 3));
	this.setTitle("学生信息表单");
	this.setSize(300, 400);
	this.setBounds(300, 200, 500, 300);
	this.setVisible(true);
	this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}

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

```
            new Demo5();
        }
    }
```

习题 8

一、选择题

1. 下面哪项不是 GUI 事件模型的组成元素（ C ）

A、事件

B、事件处理器

C、GUI 容器

D、事件源

2. 编写 JButton 组件的事件处理器类时，需实现哪个接口（ B ）

A、ItemListener

B、ActionListener

C、ButtonListener

D、WindowListener

3. 下面哪项不是事件适配器类的作用是（ B ）

A、为编写事件侦听器程序提供简便手段

B、创建一种全新的事件侦听机制

C、是由相应的事件侦听器接口继承而来

D、定义在 java.awt.event 中

4. 按下键盘上的按键会产生下面哪种事件（ A ）

A、KeyEvent

B、MouseEvent

C、ItemEvent

D、ActionEvent

5. 对象 myListener 的类实现了 ActionListener，下面哪条语句可以使 myListener 对象接受组件 jbutton 产生的 actionEvent （ C ）

A、jbutton.add(myListener)

B、jbutton.addListener (myListener)

C、jbutton.addActionListener (myListener)

D、jbutton.setActionListener (myListener)

二、是非题

1. 所有组件都是事件源（ × ）

2. 事件监听者除了得知事件的发生外，还应调用相应的方法处理事件（ √ ）

3. 事件参数中包含有事件发生的详情（ √ ）

4. 在 GUI 的事件处理机制中，每个事件类对应一个事件监听器接口，每一个监听器接口都有相对应的适配器（ √ ）

5. 事件监听器是一些接口，其中含有一些方法（ √ ）

三、编程题

1. 编写程序实现：一个窗口，窗口底部有两个按钮，点击按钮在窗口中分别出现“1”、“2”字样。

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Demo1 extends JFrame {
    JTextField jtf = new JTextField(10);

    public Demo1() {
        jtf.setEditable(false);
        JButton jbt1 = new JButton("1");
        jbt1.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {
                jtf.setText("1");
            }
        });
        JButton jbt2 = new JButton("2");
        jbt2.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {
                jtf.setText("2");
            }
        });
        // 创建面板 jp1,jp2
        JPanel jp1 = new JPanel();
        JPanel jp2 = new JPanel();

        // 往面板对象中添加按钮组件
        jp1.add(jbt1);
        jp1.add(jbt2);
        jp2.add(jtf);
        this.add("South", jp1);
        this.add(jp2);
```

```
        this.setTitle("my java window");
        this.setSize(300, 400);// 设置窗体大小
        this.setBounds(300, 200, 500, 400); // 设置窗体的位置和大小(x,y,width,height)
        // this.pack();
        this.setVisible(true); // 设置窗体可见
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        new Demo1(); // 创建窗体
    }
}
```

2. 编写程序，向文本框里输入正整数，当输入整数为负数时，显示错误对话框，提示“你输入的是负数”。

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class Demo2 extends JFrame {
    public Demo2() {
        JLabel jlb = new JLabel("请输入正整数");
        JTextField jtf = new JTextField(10);
        JButton jbt1 = new JButton("确定");
        jbt1.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                if (Integer.parseInt(jtf.getText()) <= 0)
                    JOptionPane.showMessageDialog(null, "你输入的是负数!", "出错啦", JOptionPane.ERROR_MESSAGE);
            }
        });
        // 创建面板 jp1,jp2,jp3
        JPanel jp1 = new JPanel();
        JPanel jp2 = new JPanel();
        JPanel jp3 = new JPanel();

        // 往面板对象中添加按钮组件
        jp1.add(jlb);
        jp2.add(jtf);
        jp3.add(jbt1);

        this.add(jp1);
        this.add(jp2);
```

```
        this.add(jp3);
        this.getContentPane().setLayout(new GridLayout(3, 1, 3, 3));
        this.setTitle("my java window");
        this.setSize(300, 400);// 设置窗体大小
        this.setBounds(300, 200, 500, 400); // 设置窗体的位置和大小(x,y,width,height)
        // this.pack();
        this.setVisible(true); // 设置窗体可见
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        new Demo2(); // 创建窗体
    }
}
```

3. 编写一个计算器，能实现加减功能。

```
import java.awt.event.* ;
import java.util.Arrays;
import java.awt.* ;
import javax.swing.* ;

public class Demo3 extends JFrame implements ActionListener {

    JTextField txtResult;
    boolean firstDigit = true;// 用于判断是否是数字
    String operator = "=";// 先初始化为等号，等到执行相应运算时再更改
    boolean operateValidFlag = true;// 判断除数是否为 0
    double resultNum = 0.0;// 可以暂存目前的最终结果

    public Demo3() {
        setTitle("计算器");
        setSize(240, 270);
        setResizable(false);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(EXIT_ON_CLOSE);

        Container contentPane = this.getContentPane();
        contentPane.setLayout(new BorderLayout(1, 5));
        JPanel pnlNorth = new JPanel();
        JPanel pnlCenter = new JPanel();

        pnlNorth.setLayout(new BorderLayout());
        pnlCenter.setLayout(new GridLayout(4, 4, 3, 3));

        Font font = new Font("Times Roman", Font.BOLD, 20);
```

```
        contentPane.add(BorderLayout.NORTH, pnlNorth);
        contentPane.add(BorderLayout.CENTER, pnlCenter);

        txtResult = new JTextField();
        txtResult.setFont(font);
        txtResult.setEnabled(false);
        JButton btnClear = new JButton("C");
        btnClear.setFont(font);
        btnClear.addActionListener(this);

        pnlNorth.add(BorderLayout.CENTER, txtResult);
        pnlNorth.add(BorderLayout.EAST, btnClear);

        String[] captions = { "7", "8", "9", "+", "4", "5", "6", "-", "1", "2", "3", "*", "0", ".",
"/", "=", };
        for (int i = 0; i < captions.length; i++) {
            JButton btn = new JButton(captions[i]);
            btn.setFont(font);
            pnlCenter.add(btn);
            btn.addActionListener(this);
        }
    }

    public static void main(String[] args) {
        JFrame frame = new Demo3();
        frame.setVisible(true);
    }

    // 对按钮进行的反应
    @Override
    public void actionPerformed(ActionEvent event) {
        String label = event.getActionCommand();
        if (label.equals("C")) {
            handleC();
        } else if ("0123456789.".indexOf(label) >= 0) {
            // 无论整数还是小数都一起提取出来
            handleNumber(label);
        } else {
            // 将当前要执行的运算的运算符赋给 operator
            handleOperator(label);
        }
    }
```

```
// 提取数字
void handleNumber(String key) {
    if (firstDigit) {
        txtResult.setText(key);// 在文本框中显示数字的字符串
    } else if ((key.equals(".")) && (txtResult.getText().indexOf(".") < 0)) {

        txtResult.setText(txtResult.getText() + ".");// 在文本框中显示整数数字的字
符串
    } else if (!key.equals(".")) {

        txtResult.setText(txtResult.getText() + key);// 在文本框中显示整数数字的
字符串
    }
    firstDigit = false;// 当数字显示完之后，即可重置为 false
}

// 实现清零
void handleC() {

    txtResult.setText("0");
    firstDigit = true;
    operator = "=";
}

// 进行运算
void handleOperator(String key) {
    if (operator.equals("/")) {
        // 判断除数是否为 0
        if (getNumberFromText() == 0.0) {
            // 以下代码很关键，不做多余说明，每次看都有不同的理解
            operateValidFlag = false;
            txtResult.setText("除数不能为零");
        } else {
            resultNum /= getNumberFromText();
        }
    } else if (operator.equals("+")) {

        resultNum += getNumberFromText();
    } else if (operator.equals("-")) {

        resultNum -= getNumberFromText();
    } else if (operator.equals("*")) {
```

```
resultNum *= getNumberFromText();
```