附录 2：习题答案

第 1 章：Java 简介 —— 自我检测（答案）

1、填空题

1、 Java源程序文件的后缀是 *.java ，Java字节码文件的后缀名称是 *.class 。
2、 Java程序实现可移值性，依靠的是 JVM 。
3、 Java语言的三个分支是： JAVA SE 、 JAVA ME 、 JAVA EE 。
4、 Java程序由 类 组成，如果Java使用 public class 声明类，则文件名称必须与类名称一致。
5、 Java执行是从 main() 方法开始执行的，此方法的完整定义是 public static void main(String args) 。
6、 Java类名的每个单词的首字母通常要求 大写 。

2、选择题

1、 推出Java语言的公司 B 。
A、 IBM B、 SUN C、 Apple D、 Microsoft
2、 下面的那个环境变量java解释时所需要的 B 。
A、 path B、 classpath C、 JAVA_HOME D、TEMP

3、判断题

1、 Java 语言属于编译型的开发语言。 （ × ）
2、 Java Application 程序不是由 main()方法开始执行的。 （ × ）

4、简答题

1、 简述 Java 实现可移值性的基本原理。
答:Java 属于编译型和解释型的编程语言,所有的*.java 程序必须编译为*.class 文件之后才可以在电脑上执行,而执行*.class 文件的电脑并不是一台真实的电脑，而是利用软件和硬件模拟出来的一台虚拟电脑，称为 Java 虚拟机，而针对于不同的操作系统平台，有不同版本的 Java 虚拟机，即：由 Java 虚拟机去适应不同的操作系统，即：只要 Java 虚拟机的支持没有改变，同一个*.class 可以在不同的平台上运行。
2、 简述 Java 语言的三个程序分支。
答： JAVA SE（Java 标准版）：提供基础的 Java 类库及平台支持：
JAVA ME（Java 微型版）：提供嵌入式开发支持：
JAVA EE（Java 企业版）：提供企业平台支持。
3、 简述 Java 中 path 及 classpath 的作用。
答： PATH：是操作系统的环境属性，指的是可以执行命令的程序路径；
CLASSPATH：是所有*.class 文件的执行路径，java 命令执行的时候将利用此路径加载所需要的*.class 文件。

4、 简述 Java 语言的主要特点。

答： Java 语言的主要的特点如下：

1、 Java 语言是一门面向对象语言，且语法足够简单；

2、 Java 避免了 C/C++之中复杂的指针关系，而使用了更为简单的引用方式来进行内存传递；

3、 Java 是为数不多的支持多线程开发的编程语言；

4、 Java 提供了自动的垃圾收集机制，可以定期释放出无用的垃圾空间；

5、 Java 语言的安全性较高；

6、 Java 最大的特点是具备可移植性，即：同一个程序在不同的操作系统上都可以运行。

5、 详细阐述 Java 中使用 public class 或 class 声明一个类的区别及注意事项。

答： 使用 public class 定义的类，要求文件名称和类名称保持一致，在一个*.java 文件之中只能存在一个 public class；

使用 class 定义的类，文件名称可以和类名称不一致，在一个*.java 文件之中可以同时存在多个 class 定义，编译之后会产生多个*.class 文件。

5、编程题

1、 在屏幕上输出：“我喜欢学习 Java”的信息。

答案：

```
public class TestDemo {
    public static void main(String[] args) {
        System.out.println("我喜欢学习 Java");
    }
}
```

2、 在屏幕上打印出以下的图形：

```
************************************
*********   Java 程序设计    *********
************************************
```

答案：

```
public class TestDemo {
    public static void main(String[] args) {
        System.out.println("************************************");
        System.out.println("*********   Java 程序设计    *********");
        System.out.println("************************************");
    }
}
```

第 2 章 简单 Java 程序 —— 自我检测（答案）

1、填空题

1、 Java中的标识符组成原则：由字母、数字、下划线、$所组成，其中不能以数字开头，不能是Java的关键字。

2、 assert关键字是在JDK 1.4 时加入的， enum关键字是在JDK 1.5 时加入的。

3、 列举出已经知道的 5 个关键字：public、static、void、int、double。

4、 Java注释分为以下三种：___单行注释（//）___、___多行注释（/* …*/）___、___文档注释___。

5、 Java中使用___int___关键字，可以定义一个整型数据。

6、 在一个Java源文件中定义了 3 个类和 15 个方法，编译该Java源文件时会产生___3___个字节码文件，其扩展名是 ___*.class___。

7、 布尔型数据类型的关键字是___boolean___，有___true___和___false___两种取值。

8、 整型数可以采用___byte___、___short___、___int___和___long___四种类型表示。

9、 根据占用内存长度的不同将浮点型分为___float___和___double___两种。

10、 Java程序结构分为：___顺序结构___、___分支结构___、___循环结构___三种。

11、 逻辑表达式：true&&false&&true的结果是___false___。

12、 逻辑表达式：!true||false的结果是___false___。

13、 在方法中可以使用___return___语句来结束方法的执行。

14、 方法中的___void___关键字用来表示方法不返回任何值。

2、选择题

1、 下面那些标识符是正确的___D___。

A、 class　　B、 hello world　　C、 123$temp　　D、 Demo

2、 下面那些关键字是Java中未使用到的关键字___A、B___（多选）。

A、 const　　B、 goto　　C、 int　　D、 assert

3、 public static void main方法的参数描述是：___A___。

A、 String args[]　　B、 int[] args　　C、 Strings args[]　　D、 String args

4、 下面说法正确的是___C___。

A、 Java 程序的源文件名称与主类（puublic class）的名称相同，后缀可以是.java 或.txt 等。

B、 JDK 的编译命令是 java。

C、 一个 java 源文件编译后可能产生多个 class 文件。

D、 在命令行编译好的字节码文件，只需在命令行直接键入程序名即可运行该程序。

5、 下面说法不正确的是___A___。

A、 Java 语言是面向对象的、解释执行的网络编程语言

B、 Java 语言具有可移植性，是与平台无关的编程语言。

C、 Java 语言可对内存垃圾自动收集。

D、 Java 语言执行时需要 Java 的运行环境。

6、 下面___A___不是Java的关键字。

A、 integer　　B、 double　　C、 float　　D、 char

7、 在Java中，字节数据类型的关键字和默认值是___A___。

A、 byte 和 0　　B、 byte 和 1　　C、 boolean 和 true　　D、 boolean 和 false

8、 3.15E2 表示的数据是___C___。

A、 3.15×2　　B、 3.15×10^{-2}　　C、 3.15×10^{2}　　D、 0.315×10^{-2}

9、 程序System.out.println("1 + 1 = " + 1 + 1) ;的输出结果是___C___。

A、 1　　B、 1 + 1 = 2　　C、 1 + 1 = 11　　D、 2

10、 程序System.out.println(10 / 3) ;的输出结果是___B___。

A、 1　　B、 3　　C、 3.3　　D、 3.33333

11、 执行下面的语句后，a、b、c的值分别是___C___。

```
int a = 2 ;
int b = (a++) * 3 ;
int c = (++a) * 3 ;
```

A、 2、6、6　　B、 4、9、9　　C、 4、6、12　　D、 3、9、9

12、 以下的____B____能正确表示Java语言中的一个整型常量。

A、 35.d　　B、 -20　　C、 1,234　　D、 "123"

13、 下面的数据类型____D____是float型

A、 33.8　　B、 129　　C、 89L　　D、 8.6F

14、 下列关于自动类型转换的说法中，正确的一个是____B____。

A、 int 类型数据可以自动转换为 char 类型数据

B、 char 类型数据可以被自动转换为 int 类型数据

C、 boolean 类型数据不可以做自动类型转换，但是可以做强制转换

D、 long 类型数据可以被自动转换为 short 类型数据

15、 一个方法在定义过程中又调用自身，这种方法称为____B____。

A、 构造方法　　B、 递归方法　　C、 成员方法　　D、 抽象方法

3、判断题

1、 变量的内容可以修改，常量的内容不可修改。（ √ ）

2、 goto 是 Java 中未使用到的关键字。（ √ ）

3、 enum 关键字是在 JDK 1.4 版本中增加的。（ × ）

4、 使用 public class 定义的类，文件名称可以与类名称不一致。（ × ）

5、 主方法编写：public void main(String arg)。（ × ）

6、 字符$不能作 Java 标识符的第一个字符。（ × ）

7、 System.out.println()输出后是不加换行的，而 System.out.print()输出后是加换行的。（ × ）

8、 使用 break 语句可以跳出一次循环。（ × ）

9、 byte 的取值范围是：0~255。（ × ）

10、 int 和 double 进行加法操作，int 会自动转换为 double 类型。（ × ）

11、 使用“&”操作时，如果第一个条件是 false，则后续的条件都不再判断。（ × ）

12、 使用“&&”操作时，如果第一个条件是 false，则后续的条件都不再判断。（ √ ）

13、 使用“|”操作时，如果第一个条件是 true，则后续的条件都不再判断。（ × ）

14、 使用“||”操作时，如果第一个条件是 true，则后续的条件都不再判断。（ √ ）

15、 定义多个同名方法时，可以依靠返回值区别同名方法。（ × ）

4、简答题

1、 请解释常量与变量的区别？

答： 常量就是一个固定的数值。是不可改变的，例如：数字 1、2 就是一个整型的常量。

变量是利用声明的方式，将内存中的某个内存块保留下来以供程序使用。可以声明的数据类型为整型、字符型、浮点型或是其他数据类型，作为变量的保存之用。变量在程序语言中扮演了最基本的角色。变量可以用来存放数据，而使用变量之前必须先声明它的数据类型。

2、 解释方法重载的概念，并举例说明。

答： 方法重载指的是多个方法的方法名称相同，但是方法中的参数类型及个数不同。

代码说明：

```
public static int add(int x, int y) {
    return x + y;
}
```

```
    public static int add(int x, int y, int z) {
        return x + y + z;
    }
    public static double add(double x, double y) {
        return x + y;
    }
```

5、编程题

1、 打印出 100~1000 范围内的所有 “水仙花数”，所谓“水仙花数”是指一个三位数，其各位数字立方和等于该数本身。例如：153 是一个“水仙花数”，因为 153=1 的三次方＋5 的三次方＋3 的三次方。

```
public class TestDemo {
    public static void main(String[] args) {
        int i, j, k;
        for (int x = 100; x < 1000; x++) {
            i = x / 100;                    // 计算百位数字
            j = (x / 10) % 10;              // 计算十位数字
            k = x % 10;                     // 计算个位数字
            if (x == i * i * i + j * j * j + k * k * k) {
                System.out.print(x + "、");
            }
        }
    }
}
```

程序运行结果：	153、370、371、407、

2、 通过代码完成两个整数内容的交换。

实现一：引用第三方变量	实现二：利用数学计算完成

public class TestDemo {
  public static void main(String[] args) {
  int x = 10;
  int y = 20;
  int temp = x;
  x = y;
  y = temp;
  System.out.println("x = " + x);
  System.out.println("y = " + y);
  }
}
	public class TestDemo {
	  public static void main(String[] args) {
	  int x = 10;
	  int y = 20;
	  x += y ;
	  y = x - y;
	  x = x - y ;
	  System.out.println("x = " + x);
	  System.out.println("y = " + y);
	  }
	}

程序运行结果：	x = 20 y = 10

3、 判断某数能否被 3，5，7 同时整除。

```
public class TestDemo {
    public static void main(String[] args) {
        int data = 105;
```

```
        if (data % 3 == 0 && data % 5 == 0 && data % 7 == 0) {
            System.out.println(data + "可以同时被3、5、7整除。");
        } else {
            System.out.println(data + "不可以同时被3、5、7整除。");
        }
    }
}
```

程序运行结果：	105可以同时被3、5、7整除。

4、 编写程序，分别利用 while 循环、do…while 循环和 for 循环求出 100~200 的累加和。

实现一：使用 while 循环

```
public class TestDemo {
    public static void main(String[] args) {
        int sum = 0;
        int x = 100;
        while (x <= 200) {
            sum += x;
            x++;
        }
        System.out.println("累加结果：" + sum);
    }
}
```

实现二：使用do…while循环

```
public class TestDemo {
    public static void main(String[] args) {
        int sum = 0;
        int x = 100;
        do {
            sum += x;
            x++;
        } while (x <= 200);
        System.out.println("累加结果：" + sum);
    }
}
```

实现三：使用for循环

```
public class TestDemo {
    public static void main(String[] args) {
        int sum = 0;
        for (int x = 100; x <= 200; x++) {
            sum += x;
        }
        System.out.println("累加结果：" + sum);
    }
}
```

第 3 章、面向对象 —— 自我检测（答案）

1、填空题

1、 面向对象有那三大特征：封装、继承、多态。

2、 类由属性和方法组成。

3、 new运算符的作用是根据对象的类型分配内存空间。当对象拥有内存空间时，会自动调用类中的构造方法为对象实例化。

4、 使用private修饰的类成员称为私有成员。私有成员只能在类中使用。

5、 构造方法的名称与类名称相同。

6、 private关键字可以让类中的属性和方法对外部不可见。

7、 this关键字可以调用本类中的属性、方法、构造方法，调用构造方法时必须放在构造方法的首行。

8、 Java中通过extends关键字实现继承。

9、 一个类只能继承一个父类，但能实现多个接口。

10、 Object类是所有类的父类，该类中判断两个对象是否相等的方法是public boolean equals(Object oth)，取得对象完整信息的方法是public String toString()。

11、 Integer类是对int基本数据类型的封装。Float类是对float基本数据类型的封装。Double类是对double基本数据类型的封装。字符类Character是对char基本数据类型的封装。

12、 当子类中定义的方法与父类方法同名且参数类型及个数、返回值类型相同时，称子类方法覆写父类方法，子类默认使用本类已经覆写方法，使用父类的同名方法，必须使用super关键字说明。

13、 当子类定义的成员变量与父类的成员变量同名时，称子类覆盖父类的成员变量，子类默认使用本类属性。使用父类的同名成员变量，必须用super关键字说明。

14、 如果子类定义了构造方法，在创建子类对象时首先默认调用父类无参构造方法，然后再本类的构造方法。

15、 在Java中数组排序的方法是java.util.Arrays.sort()。

2、选择题

1、 如果希望方法直接通过类名称访问，在定义时要使用的修饰符是A。

A、 static　　B、 final　　C、 abstract　　D、 this

2、 如果类中没有定义构造方法，系统会提供一个默认的构造方法。默认构造方法的特点是C。

A、 无参数有操作　B、 有参数无操作　C、 即无参数也无任何操作　D、 有参数有操作

3、 有一个类Demo，对与其构造方法的正确声明是B。

A、 void Demo(int x){…}　　B、 Demo(int x){…}

C、 Demo Demo(int x){…}　　D、 int Demo(){}

4、 以下关于面向对象概念的描述中，不正确的一项是C。

A、 在现实生活中，对象是指客观世界的实体

B、 程序中的对象就是现实生活中的对象

C、 在程序中，对象是通过一种抽象的数据类型来描述的，这种抽象数据类型称为类（class）

D、 在程序中，对象是一组变量和相关方法的集合

5、 下列那一项不属于面向对象程序设计的基本要素？D

A、 类　　B、 对象　　C、 方法　　D、 安全

6、下列程序的执行结果是A

```
public class TestDemo {
    public void fun() {
        static int i = 0;
        i++;
        System.out.println(i);
    }
    public static void main(String args[]) {
        Demo d = new Demo();
        d.fun();
    }
}
```

A、 编译错误　　B、 0　　C、 1　　D、 运行成功，但不输出

7、 顺序执行下列程序语句后，则b的值是___C___。

String str = "Hello" ;

String b = str.substring(0,2) ;

A、 Hello　　B、 hello　　C、 He　　D、 null

8、 不能直接使用new创建对象的类是___B___。

A、 静态类　　B、 抽象类　　C、 最终类　　D、 公有类

9、 为类定义多个名称相同、但参数的类型或个数不同的方法的做法称为___B___。

A、 方法重载　　B、 方法覆写　　C、 方法继承　　D、 方法重用

10、 定义接口的关键字是___C___。

A、 extends　　B、 class　　C、 interface　　D、 public

11、 现在有两个类A、B，以下描述中表示B继承自A的是___D___。

A、 class A extends B　　B、 class B implements A

C、 class A implements　　D、 class B extends A

12、 下面关于子类调用父类构造方法的描述正确的是___C___。

A、 子类定义了自己的构造方法，就不会调用父类的构造方法。

B、 子类必须通过 super 关键字调用父类有参的构造方法。

C、 如果子类的构造方法没有通过 super 调用父类的构造方法，那么子类会先调用父类中无参构造方法，之后再调用子类自己的构造方法。

D、 创建子类对象时，先调用子类自己的构造方法，让后再调用父类的构造方法。

13、 假设类X是类Y的父类，下列声明对象x的语句中不正确的是___D___。

A、 X x = new X() ;　　B、 X x = new Y() ;

C、 Y x = new Y() ;　　D、 Y x = new X() ;

14、 编译并运行下面的程序，结果___B___。

```
public class A {
    public static void main(String args[]) {
        B b = new B();
        b.test();
    }
    void test() {
        System.out.print("A");
    }
}
class B extends A {
```

```
    void test() {
        super.test();
        System.out.println("B");
    }
}
```

A、 产生编译错误　　B、 代码可以编译运行，并输出结果：AB

C、 代码可以编译运行，但没有输出　　D、 编译没有错误，但会运行时会产生异常

15、 编译运行下面的程序，结果是___A___。

```
public class A {
    public static void main(String args[]) {
        B b = new B();
        b.test();
    }
    public void test() {
        System.out.print("A");
    }
}
class B extends A {
    void test() {
        super.test();
        System.out.println("B");
    }
}
```

A、 产生编译错误，因为类 B 覆盖类 A 的方法 test()时，降低了其访问控制的级别。

B、 代码可以编译运行，并输出结果：AB

C、 代码可以编译运行，但没有输出

D、 代码可以编译运行，并输出结果：A

16、 下面___B___修饰符所定义的方法必须被子类所覆写。

A、 final　　B、 abstract　　C、 static　　D、 interface

17、 下面___A___修饰符所定义的方法不能被子类所覆写。

A、 final　　B、 abstract　　D、 static　　D、 interface

18、下面的程序编译运行的结果是___A___

```
public class A implements B {
    public static void main(String args[]) {
        int m, n;
        A a = new A();
        m = a.K;
        n = B.K;
        System.out.println(m + ", " + n);
    }
}
interface B {
    int K = 5;
}
```

A、 5，5　　B、 0，5　　C、 0，0　　D、 编译程序产生编译结果

19、 下面关于接口的说法中不正确的是___C___。

A、 接口所有的方法都是抽象的

B、 接口所有的方法一定都是 public 类型

C、 用于定义接口的关键字是 implements

D、 接口是 Java 中的特殊类，包含全局常量和抽象方法

20、 下面关于Java的说法不正确的是___A___

A、 abstract 和 final 能同时修饰一个类

B、 抽象类不光可以做父类，也可以做子类

C、 抽象方法不一定声明在抽象类中，也可以在接口中

D、 声明为 final 的方法不能在子类中覆写

3、判断题

1、 没有实例化的对象不能使用。 （ √ ）

2、 不可以为类定义多个构造方法。 （ × ）

3、 使用 static 声明的方法可以调用非 static 声明的方法。 （ × ）

4、 非 static 声明的方法可以调用 static 声明的属性或方法。 （ √ ）

5、 String 对象可以使用==进行内容的比较。 （ × ）

6、 垃圾是指无用的内存空间，会被垃圾收集机制回收。 （ √ ）

7、 构造方法可以有返回值类型的声明。 （ × ）

8、 匿名对象是指使用一次的对象，使用之后将等待被垃圾回收。 （ √ ）

9、 使用 static 定义的内部类就成为外部类。 （ √ ）

10、 多个实例化对象之间不会互相影响，因为保存在不同的内存区域之中。 （ √ ）

11、 final 声明的类可以有子类。 （ × ）

12、 一个类继承了抽象类，则抽象类中的抽象方法需要在其子类中覆写。 （ √ ）

13、 final 类型的变量是常量，其内容不可改变。 （ √ ）

14、 一个类不能即是子类又是父类。 （ √ ）

15、 子类只能继承父类的成员，但不能修改父类成员。 （ × ）

16、 Java 语言只支持单继承，不支持多继承。 （ √ ）

17、 子类可以继承父类的所有成员。 （ √ ）

18、 一个接口可以继承一个抽象类。 （ × ）

19、 一个接口可以同时继承多个接口。 （ √ ）

20、在程序中 this 和 super 调用构造方法时可以同时出现。 （ × ）

4、简答题

1、 String 类的操作特点。

答： String 类的对象有两种实例化方式：

|- 方式一：直接赋值，只开辟一块堆内存空间，并且对象可以入池；

|- 方式二：构造方法，开辟两块堆内存空间，有一块将称为垃圾，不会自动入池，使用 intern()方法手工入池；

String 对象的比较方法：

|- ==：比较的是两个字符串对象的内存地址数值；

|- equals()：字符串内容比较；

字符串对象一旦声明，则内容不可改变，改变的只能是字符串对象的地址指向。

2、 简述垃圾对象的产生。

答： 垃圾指的是一块无用的引用内存，当将变量设置为 null 或者长时间不使用时，就将成为垃圾。

3、 static 方法如何调用？非 static 方法如何调用？

答： static 方法可以使用类名称或实例化对象调用，而非 static 方法只能依靠实例化对象才可以调用。

4、 类与对象的关系是什么？如何创建及使用对象？

答： 类规定了对象所具有的属性及行为（方法），类只有通过产生对象才可以分配属性或者是调用方法，对象的创建依靠关键字 new 创建。

5、 举例说明子类对象的实例化过程。

答： 当通过关键字 new 实例化子类对象时，会默认调用父类的无参构造方法，为父类对象实例化，而后才会调用子类的构造方法，为子类对象实例化。

7、 简述 this 与 super 关键字的区别。

答： this 和 super 都可以调用类中的属性、方法、构造方法，但是 this 调用的是本类操作，而 super 是由子类调用父类操作。

8、 简述方法的重载与覆写的区别。

答： 方法重载是发生在一个类之中，方法名称相同、参数的类型及个数不同，不受权限的限制。而覆写是发生在继承关系之中，子类定义了和父类定义了方法名称相同、参数类型及个数、返回值类型完全相同的方法时所发生的操作，在子类覆写父类方法时，被覆写的方法不能拥有比父类更严格的访问权限。

9、 在已有类的基础上派生新的类有什么好处？

答： 扩充已有类的功能，并且利用方法的覆写扩充已有方法的功能。

10、 如何区分子类和父类？子类可以继承父类的那些内容？

答： 子类使用 extends 继承父类或使用 implements 实现多个接口，子类可以继承父类中的全部内容，但是对于私有操作属于隐式继承，而非私有操作属于显式继承。

11、 什么是多态？实现都态的方法有那些？

答： 多态是面向对象的最后一个主要特征，它本身主要分为两个方面：

- 方法的多态性：重载与覆写
  - |- 重载：同一个方法名称，根据不同的参数类型及个数可以完成不同的功能；
  - |- 覆写：同一个方法，根据操作的子类不同，所完成的功能也不同。
- 对象的多态性：父子类对象的转换。
  - |- 向上转型：子类对象变为父类对象，格式：父类 父类对象 = 子类实例，自动；
  - |- 向下转型：父类对象变为子类对象，格式：子类 子类对象 =(子类) 父类实例，强制；

12、 接口有那些特征？如何定义和实现接口。

答： 接口之中全部由全局常量及抽象方法所组成，一个类可以同时实现多个接口，在 Java 中使用 interface 定义接口，子类使用 implements 实现接口。

13、 接口和抽象类有那些区别？

答： 抽象类及接口区别如下。

No.	区别	抽象类	接口
1	定义关键字	abstract class	interface
2	组成	常量、变量、抽象方法、普通方法、构造方法	全局常量、抽象方法
3	权限	可以使用各种权限	只能是 public
4	关系	一个抽象类可以实现多个接口	接口不能够继承抽象类，却可以继承多接口
5	使用	子类使用 extends 继承抽象类	子类使用 implements 实现接口
		抽象类和接口的对象都是利用对象多态性的向上转型，进行接口或抽象类的实例化操作	
6	设计模式	模板设计模式	工厂设计模式、代理设计模式
7	局限	一个子类只能够继承一个抽象类	一个子类可以实现多个接口

14、 简述基本数据类型的自动装箱及自动拆箱操作。

答： 在 JDK 1.5 之后，基本数据类型可以采用直接赋值的方式为包装类进行对象的实例化操作，而包装类的对象也可以通

过直接赋值的方式变回基本数据类型。

5、编程题

1、 编写并测试一个代表地址的 Address 类，地址信息由：国家，省份，城市，街道，邮编组成，并可以返回完整的地址信息。

```
class Address {
    private String national;
    private String provincial;
    private String city;
    private String street;
    private String zipcode;
    public Address() {
    }
    public Address(String national, String provincial, String city,
            String street, String zipcode) {
        super();
        this.national = national;
        this.provincial = provincial;
        this.city = city;
        this.street = street;
        this.zipcode = zipcode;
    }
    public String toString() {
        return "国家：" + this.national + "，省份：" + this.provincial + "，城市："
                + this.city + "，街道：" + this.street + "，邮政编码：" + this.zipcode;
    }
    // setter、getter略
}
public class TestDemo {
    public static void main(String args[]) {
        Address ad = new Address("中国", "北京", "北京市", "MLDN", "100088");
        System.out.println(ad);
    }
}
```

程序运行结果：	国家：中国，省份：北京，城市：北京市，街道：MLDN，邮政编码：100088

2、 定义并测试一个代表员工的 Employee 类。员工属性包括“编号”、“姓名”、“基本薪水”、“薪水增长额”；还包括 “计算增长后的工资总额”。的操作方法。

```
class Employee {
    private int empno ;                 // 雇员编号
    private String ename ;              // 雇员姓名
    private double sal ;            // 基本工资
    private double rate ;           // 工资增长额
    public Employee() {
```

```
    }
    public Employee(int empno, String ename, double sal, double rate) {
        super();
        this.empno = empno;
        this.ename = ename;
        this.sal = sal;
        this.rate = rate;
    }
    public String toString() {
        return "雇员编号：" + this.empno + "，雇员姓名：" + this.ename + "，基本工资：" + this.sal ;
    }
    public void growthin() {      // 增长薪水
        this.sal = this.sal * this.rate ;
    }
    // setter、getter略
}
public class TestDemo {
    public static void main(String args[]) {
        Employee emp = new Employee(7369, "SMITH", 1000, 1.5);
        emp.growthin() ;          // 工资增长
        System.out.println(emp);
    }
}
```

程序运行结果：	雇员编号：7369，雇员姓名：SMITH，基本工资：1500.0

3、 编写程序在将字符串“want you to know one thing”，统计出字母“n”和字母“o”的出现次数。

```
public class TestDemo {
    public static void main(String args[]) {
        String str = "want you to know one thing" ;    // 定义字符串
        int sum = 0 ;
        while (str.indexOf("n") != -1) {               // 是否还有字母n
            sum ++ ;                                   // 数据统计量增加
            str = str.substring(str.indexOf("n") + 1); // 改变字符串内容
        }
        System.out.println("字母n的出现次数：" + sum);
    }
}
```

程序运行结果：	字母n的出现次数：4

4 设计一个 Dog 类，有名字、颜色、年龄等属性，定义构造方法来初始化类的这些属性，定义方法输出 Dog 信息。编写应用程序使用 Dog 类。

```
class Dog {
    private String name ;
    private String color ;
    private int age ;
    public Dog() {
    }
```

```
    public Dog(String name, String color, int age) {
        super();
        this.name = name;
        this.color = color;
        this.age = age;
    }
    public String toString() {
        return "狗的名字：" + this.name + "，狗的颜色：" + this.color + "，狗的年龄：" + this.age ;
    }
    // setter、getter略
}
public class TestDemo {
    public static void main(String args[]) {
        Dog dog = new Dog("金毛", "金黄色", 3);
        System.out.println(dog);
    }
}
```

程序运行结果：	狗的名字：金毛，狗的颜色：金黄色，狗的年龄：3

5、 字符串操作：

- 从字符串“MLDN 中心 Java 技术学习班 20130214”中提取开班日期。

```
public class TestDemo {
    public static void main(String args[]) {
        String str = "MLDN中心Java技术学习班20130214" ;
        System.out.println(str.substring(str.indexOf("20130214")));
    }
}
```

程序运行结果：	20130214

- 将“MLDN JAVA 高端技术培训”字符串中的“Java”替换为“JAVA EE”。

```
public class TestDemo {
    public static void main(String args[]) {
        String str = "MLDN JAVA高端技术培训" ;
        System.out.println(str.replaceAll("Java".toUpperCase(), "JAVA EE"));
    }
}
```

程序运行结果：	MLDN JAVA EE高端技术培训

- 取出“Java 技术学习班 20130214”中的第八个字符。

```
public class TestDemo {
    public static void main(String args[]) {
        String str = "Java技术学习班20130214" ;
        System.out.println(str.charAt(8));
    }
}
```

程序运行结果：	班

- 清除“Java 技术学习班 20130214”中的所有‘0’。

```
public class TestDemo {
```

```
    public static void main(String args[]) {
        String str = "Java技术学习班20130214" ;
        System.out.println(str.replaceAll("0", ""));
    }
}
```

程序运行结果：	Java技术学习班213214

- 从任意给定的身份证号码中提取此人的出生日期。

```
public class TestDemo {
    public static void main(String args[]) {
        String str = "11010519760919001900520" ;
        System.out.println(str.substring(6,14));
    }
}
```

程序运行结果：	19760919

6、 编写一个银行帐户类,类的构成包括：

- 数据成员：
  - |- 用户的帐户名称、用户的帐户余额；
- 方法包括：
  - |- 开户（设置帐户名称，及余额），利用构造方法完成
- 查询余额

```
class Account {
    private String name ;
    private double balance ;
    public Account() {
    }
    public Account(String name, double balance) {
        super();
        this.name = name;
        this.balance = balance;
    }
    public String toString() {
        return "账户名称：" + this.name + "，余额：" + this.balance;
    }
    public double getBalance() {
        return balance;
    }
    // setter、getter略
}
public class TestDemo {
    public static void main(String args[]) {
        Account acc = new Account("张三", 5000.0);
        System.out.println(acc);
        System.out.println("账户余额：" + acc.getBalance());
    }
}
```

程序运行结果：	账户名称：张三，余额：5000.0 账户余额：5000.0

7、 定义一个 ClassName 接口，接口中只有一个抽象方法 getClassName()。设计一个类 Company，该类实现接口 ClassName 中的方法 getClassName()，功能是获取该类的类名称。编写应用程序使用 Company 类。

```
interface ClassName {
    public String getClassName() ;
}
class Company implements ClassName {
    public String getClassName() {
        return "Company";
    }
}
public class TestDemo {
    public static void main(String args[]) {
        ClassName name = new Company() ;
        System.out.println(name.getClassName());
    }
}
```

程序运行结果：	Company

8、 建立一个人类（Person）和学生类（Student）功能要求：

A、 Person 中包含 4 个保护型的数据成员 name、address、sex、age 分别为字符串，字符串，字符及整型。表示：姓名、地址、性别和年龄。一个四参构造方法，一个无参构造方法，及一个输出方法用于显示四种属性。

B、 Student 继承 Person，并增加输出成员 math、english 存放数学和英语成绩。一个六参构造方法，一个两参构造方法，一个无参构造方法，重写输出方法用于显示全部六种种属性.

```
class Person {
    private String name ;
    private String address ;
    private char sex ;
    private int age ;
    public Person() {
    }
    public Person(String name, String address, char sex, int age) {
        super();
        this.name = name;
        this.address = address;
        this.sex = sex;
        this.age = age;
    }
    public String toString() {
        return "姓名：" + this.name + "，地址：" + this.address + "，性别：" + this.sex
                + "，年龄：" + this.age;
    }
    // setter、getter略
}
class Student extends Person {
```

```
    private double math ;
    private double english ;
    public Student() {
    }
    public Student(String name, String address, char sex, int age, double math,
            double english) {
        super(name, address, sex, age);
        this.math = math;
        this.english = english;
    }
    public String toString() {
        return super.toString() + ", 数学成绩: " + this.math + ", 英语成绩: " + this.english;
    }
    // setter、getter略
}
public class TestDemo {
    public static void main(String args[]) {
        Student stu = new Student("张三", "北京西城区甲11号德外大街德胜科技园美江大厦 A座 - 6层", '
男', 25, 90.0, 99.0);
        System.out.println(stu);
    }
}
```

程序运行结果：	姓名：张三，地址：北京西城区甲11号德外大街德胜科技园美江大厦 A座 - 6层，性别：男，年龄：25，数学成绩：90.0，英语成绩：99.0

9、 定义员工类，具有姓名，年龄，性别属性，并具有构造方法，显示数据方法，定义管理层类，继承员工类，并有自己的属性：职务，年薪。定义职员类，继承员工类，并有自己的属性：所属部门，月薪。

```
class Employee {
    private String name ;
    private int age ;
    private char sex ;
    public Employee() {
    }
    public Employee(String name, int age, char sex) {
        super();
        this.name = name;
        this.age = age;
        this.sex = sex;
    }
    public String toString() {
        return "雇员姓名: " + this.name + ", 年龄: " + this.age + ", 性别: " + this.sex;
    }
    // setter、getter略
}
class Manager extends Employee {
```

```
    private String job ;
    private double income ;
    public Manager() {
    }
    public Manager(String name, int age, char sex, String job, double income) {
        super(name, age, sex);
        this.job = job;
        this.income = income;
    }
    public String toString() {
        return super.toString() + ", 职位: " + this.job + ", 年薪: " + this.income ;
    }
    // setter、getter略
}
class Staff extends Employee {
    private String dept ;
    private double salary ;
    public Staff() {
    }
    public Staff(String name, int age, char sex, String dept, double salary) {
        super(name, age, sex);
        this.dept = dept;
        this.salary = salary;
    }
    public String toString() {
        return super.toString() + ", 部门: " + this.dept + ", 月薪: " + this.salary ;
    }
    // setter、getter略
}
public class TestDemo {
    public static void main(String args[]) {
        Employee ea = new Manager("张三", 30, '男', "总监", 200000.0);
        Employee eb = new Staff("李四", 25, '女', "业务部", 1500.0);
        System.out.println(ea);
        System.out.println(eb);

    }
}
```

程序运行结果：	雇员姓名：张三，年龄：30，性别：男，职位：总监，年薪：200000.0 雇员姓名：李四，年龄：25，性别：女，部门：业务部，月薪：1500.0

10、 定义类 Shape 表示一般二维图形。Shape 具有抽象方法 area 和 perimeter，分别计算形状的面积和周长。试定义一些二维形状类（如矩形、三角形、圆形等），这些类均为 Shape 类的子类。

```
abstract class Shape {
    public abstract double area();
    public abstract double perimeter();
```

```
}
class Rectangle extends Shape {            // 矩形
    private double wide ;              // 宽
    private double longs ;                 // 长
    public Rectangle() {
    }
    public Rectangle(double wide, double longs) {
        super();
        this.wide = wide;
        this.longs = longs;
    }
    public double area() {
        return this.longs * this.wide ;
    }
    public double perimeter() {
        return (this.longs + this.wide) * 2;
    }
}
class Triangle extends Shape {                    // 三角形
    private double edgea ;                        // 边长
    private double edgeb ;                        // 边长
    private double edgec ;                        // 边长
    public Triangle() {
    }
    public Triangle(double edgea, double edgeb, double edgec) {
        super();
        this.edgea = edgea;
        this.edgeb = edgeb;
        this.edgec = edgec;
    }
    public double area() {
        return this.edgea * this.edgeb / 2 ;
    }
    public double perimeter() {
        return this.edgea + this.edgeb + this.edgec ;
    }
}
class Round extends Shape {                       // 圆形
    private double radius ;                       // 半径
    public Round() {
    }
    public Round(double radius) {
        super();
        this.radius = radius;
    }
```

```
    public double area() {
        return this.radius * this.radius * Math.PI;
    }
    public double perimeter() {
        return this.radius * 2 * Math.PI;
    }
}
public class TestDemo {
    public static void main(String args[]) {
        Shape rectangle = new Rectangle(10.5, 20.6);
        Shape triangle = new Triangle(10.1, 20.2, 30.3);
        Shape round = new Round(30.3) ;
        System.out.println("矩形面积：" + rectangle.area() + "，矩形周长：" + rectangle.perimeter());
        System.out.println("三角形面积： " + triangle.area() + "， 三角形周长： " +
triangle.perimeter());
        System.out.println("圆形面积： " + round.area() + "， 圆形周长： " + round.perimeter());
    }
}
```

程序运行结果：	矩形面积：216.3，矩形周长：62.2 三角形面积：102.00999999999999，三角形周长：60.599999999999994 圆形面积：2884.2647993342534，圆形周长：190.38051480754146

第 4 章、异常的捕获及处理 —— 自我检测（答案）

1、填空题

1、 Throwable下有那两个子类：___Error___、___Exception___。

2、 ArthmeticException类表示___算术___异常，ArraysIndexOutOfBoundsException表示___数组越界___异常。

3、 一个try代码后面必须跟着若干个___catch___代码段或者一个___finally___代码段。

4、 如果一个方法使用了___throws___，则编译器会强制在使用此方法时进行异常的处理。

5、 异常处理中使用___finally___作为异常的统一出口。

2、选择题

1、 使用那个关键字可以在程序中手工抛出异常___B___。

A、 throws　　B、 throw　　C、 assert　　D、 class

2、 下面___A___关键字可以用在方法的声明处？

A、 throws　　B、 assert　　C、 class　　D、 interface

3、 为了捕获一个异常，代码必须放在下面___A___语句块中。

A、 try 块　　B、 catch 块　　C、 throws 块　　D、 finally 块

4、 下面关于try块的说法正确的是___C___。

A、 try 块后至少应有一个 catch 块　　B、 try 块后必须有 finally 块

C、 可能抛出异常的方法应放在 try 块中　　D、 对抛出的异常的处理应放在 try 块中

5、 finally块中的代码将____A____。

A、 总是被执行

B、 如果 try 块后面没有 catch 块时，finally 块中的代码才会执行

C、 异常发生时才被执行

D、 异常没有发生时才执行

6、 一个异常将终止____A____。

A、 整个程序　　B、 只终止抛出异常的方法

C、 产生异常的 try 块　　D、 上面的说法都不对

7、 所有异常的共同父类是___B___。

A、 Error　　B、 Exception　　C、 Throwable　　D、 RuntimeException

3、判断题

1、 一个 try 语句后有多个 catch 时，捕获范围大的异常要放在捕获范围小的异常之后。 （ √ ）

2、 finally 语句可以根据需要有选择的添加。 （ √ ）

4、简答题

1、 简述 RuntimeException 和 Exception 的区别。

答： 异常（Exception）表示程序运行过程中可能出现的非正常状态，运行时异常（RuntimeException）表示虚拟机的通常操作中可能遇到的异常，是一种常见运行错误。java 编译器要求方法必须声明抛出可能发生的非运行时异常，但是并不要求必须声明抛出未被捕获的运行时异常，即：Exception 定义了必须处理的异常，而 RuntimeException 定义的异常可以选择性的进行处理。

RuntimeException 是 Exception 的子类；

2、 try、catch、finally 三种语句的功能是什么？

答： try 语句负责捕获程序之中产生的异常；

catch 负责匹配异常类型，并且对指定的异常进行处理；

finally 作为异常处理的统一出口，不管是否发生异常，都会执行本程序。

3、 简述 Java 中的异常处理机制。

答： （1）、如果程序之中产生了异常，那么会自动的由 JVM 根据异常的类型，实例化一个指定异常类的对象；

（2）、如果这个时候程序之中没有任何的异常处理操作，则这个异常类的实例化对象将交给 JVM 进行处理，而 JVM 的默认处理方式就是进行异常信息的输出，而后中断程序执行；

（3）、如果程序之中存在了异常处理，则会由 try 语句捕获产生的异常类对象；

（4）、与 try 之后的每一个 catch 进行匹配，如果匹配成功，则使用指定的 catch 进行处理，如果没有匹配成功，则向后面的 catch 继续匹配，如果没有任何的 catch 匹配成功，则这个时候将交给 JVM 执行默认处理；

（5）、不管是否有异常都会执行 finally 程序，如果此时没有异常，执行完 finally，则会继续执行程序之中的其他代码，如果此时有异常没有能够处理（没有一个 catch 可以满足），那么也会执行 finally，但是执行完 finally 之后，将默认交给 JVM 进行异常的信息输出，并且程序中断；

4、 简述 Error 和 Exception 的区别。

答： Error：指的是 JVM 错误，这个时候的程序并没有执行，无法处理；

Exception：指的是程序之中出现的错误信息，可以进行异常处理，主要关心 Exception。

5、 列举三个常见的 RuntimeException 子类。

答： NumberFormatException、ClassCastException、NullPointerException

5、编程题

1、 编写应用程序，从命令行输入两个小数参数，求它们的商。要求程序中捕获 NumberFormatException 异常和 ArithmeticException 异常。

```
class MyMath {
    public int div(String x, String y) throws NumberFormatException,
            ArithmeticException {                          // 出现异常要交给被调用处出
        int result = 0;
        try {
            int numa = Integer.parseInt(x) ;
            int numb = Integer.parseInt(y) ;
            result = numa / numb;                          // 除法计算
        } catch (Exception e) {
            throw e;                                       // 向上抛
        }
        return result;
    }
}
public class TestDemo {
    public static void main(String args[]) {
        if (args.length != 2) {
            System.out.println("程序运行出错！");
            System.exit(1) ;                               // 程序退出
        }
        try {
            MyMath mm = new MyMath() ;
            System.out.println(mm.div(args[0], args[1])); // 被调用处处理异常
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

第 5 章、包及访问控制权限 —— 自我检测（答案）

1、填空题

1、 package 关键字可以定义一个包，import 关键字可以导入包。

2、 Java中存在四种访问权限：private 、default 、protected 、public 。

3、 Java中可以使用 import static 包.类.* 导入一个类的全部静态方法。

4、 jar 命令可以将全部的class打成一个压缩包。

5、 Java中 java.lang 包是自动导入的。

6、 Java程序分为两种形式： Applet 、Application 。

2、选择题

1、 String和Object类在 A 包中定义的。

A、 java.lang B、 java.util C、 java.net D、 java.sql

2、 C 权限是同一包可以访问，不同包的子类可以访问，不同包的非子类不可以访问。

A、 private B、 default C、 protected D、 public

3、 下列说法正确的一项是 C 。

A、 java.lang.Integer 是接口

B、 String 定义在 java.util 包中

C、 Double 类在 java.lang 包中

D、 Double 类在 java.lang.Object 包中

4、 下列关于包、类和源文件的描述中，不正确的一项是 C 。

A、 一个包可以包含多个类

B、 一个源文件中，只能有一个 public class

C、 属于同一个包的类在默认情况不可以互相访问，必须使用 import 导入

D、 系统不会为源文件创建默认的包

5、 定义类时不可能用到的关键字是 C 。

A、 final B、 public C、 protected D、 static

3、判断题

1、 java.lang 包必须由用户手工导入，否则无法使用。 （ × ）

2、 定义包后类的完整名称是：包.类名称。 （ √ ）

4、简答题

1、 简述包的作用及使用。

答： 使用包可以将功能近似的类统一管理，同时可以避免项目之中同名类名称冲突问题；

2、 简述 Java 的四种访问权限的区别。

答： Java 一共有四种访问控制权限：

- private：只可以在一个类中访问，其他类不允许访问；
- default：可以在一个包中访问，但是不同包不允许访问；
- protected：可以在同一个包中和不同包的子类所访问，其他包的类不允许访问；
- public：可以被所有包的类所访问。

第 6 章、JDK 1.5 新特性 —— 自我检测（答案）

1、填空题

1、 在使用泛型类时，没有指定泛型的类型，则编译会出现 不安全的警告 信息，程序在使用时会使用 Object 类型进行接收。

2、 通过 ? extends 类 指定泛型的上限，通过 ? super 类 指定泛型的下限。

3、 使用 ? 通配符可以接收全部的泛型类型实例，但却不可修改泛型属性内容。

4、 Java中通过 enum 关键字定义一个枚举，使用此关键字实际上就相当于一个类继承 java.lang.Enum 。

5、 枚举中通过 values() 方法取得枚举的全部内容。

6、 Java提供的三个内建的Annotation是： @Override 、 @Deprecated 、 @SuppressWarnings 。

2、判断题

1、 在枚举类中可以定义抽象方法，而抽象方法只需要实现一次即可。 （ × ）

2、 枚举中可以定义构造方法，但要求每个枚举对象都必须调用此构造方法。 （ √ ）

3、 枚举中定义的构造方法可以使用 public 权限声明。 （ × ）

3、简答题

1、 简述泛型的作用

答： 泛型的主要作用是解决对象向下转型所带来的 ClassCastException，使用泛型之后，类之中的属性或方法中的参数类型就由调用处来决定，而如果调用处不设置泛型，为了保证程序不出错，则会使用 Object 类型进行定义。

2、 简述枚举的作用及实现特点。

答： 枚举定义出了一个类所能使用的若干几个实例化对象，枚举可以直接采用“枚举类型.对象”的方式取得类中的实例化对象进行操作。

3、 简述 JAVA SE 中三个内建的 Annotation 的作用。

答： 在 JAVA SE 之中定义了三种 Annotation：

- @Override：准确的方法覆写；
- @Deprecated：某个结构（类、方法、属性）不再建议被用户所使用；
- @SuppressWarnings：压制编译时所产生的警告信息。

4、编程题

1、 定义一个品牌电脑的枚举类，里面只有固定的几个电脑品牌，例如：Lenovo、HP、Dell、Apple、Acer。

```
enum ComputerBrand {
    LENOVO("联想"), HP("惠普"), DELL("戴尔"), APPLE("苹果"), ACER("宏基");
    private String title;
    private ComputerBrand(String title) {
        this.title = title;
    }
    @Override
    public String toString() {
        return this.title ;
    }
}
```

第 7 章、多线程 —— 自我检测（答案）

1、填空题

1、 Java多线程可以依靠 继承Thread类 和 实现Runnable接口 两种方式实现。

2、 多个线程操作同一资源的时候需要注意 同步 ，依靠 synchronized 关键字实现，实现手段是： 同步代码块 和 同步方法 ，过多的使用，则会出现 死锁 问题。

3、 Java程序运行时，至少启动 两个 个线程，分别是： main线程 和 gc线程 。

4、 main线程的优先级是 中等优先级 。

5、 线程在生命周期中要经历五种状态，分别是 创建 状态、 就绪 状态、 运行 状态、 堵塞 状态、 销毁 状态。

6、 Object类提供的 wait() 、 notify() 、 notifyAll() 三个方法可以控制线程。

2、选择题

1、 线程的启动方法是 B 。

A、 run()　　B、 start()　　C、 begin()　　D、 accept()

2、 Thread类提供表示线程优先级的静态常量，代表普通优先级的静态常量是 D 。

A、 MAX_PRIORITY　　B、 MIN_PRIORITY

C、 NORMAL_PRIORITY　　D、 NORM_PRIORITY

3、 设置线程优先级的方法是 A 。

A、 setPriority()　　B、 getPriority()　　C、 getName()　　D、 setName()

4、 Thread类的 D 方法是不建议使用的？

A、 stop()　　B、 suspend()　　C、 resume()　　D、 全部都是

5、 下列 C 关键字通常用赖对对象加锁，从而似的对对象的访问是排他的。

A、 serialize　　B、 transient　　C、 synchronized　　D、 static

3、判断题

1、 Java 中直接调用 Thread 类中的 run()方法可以启动一个线程。 （ × ）

2、 进程是在线程的基础之上的进一步划分。 （ √ ）

3、 Java 是多线程的编程语言。 （ √ ）

4、简答题

1、 简述线程两种实现方式及区别？

答： 多线程的两种实现方式都需要一个线程的主类，而这个类可以实现 Runnable 接口或继承 Thread 类，不管使用何种方式都必须在子类之中覆写 run()方法，此方法为线程的主方法；

Thread 类是 Runnable 接口的子类，而且使用 Runnable 接口可以避免单继承局限，以及更加方便的实现数据共享的概念。

2、 简述死锁的产生。

答： 当多个线程访问某一共享资源时，为保证数据的正确性，需要使用同步进行控制，线程同步指的是某一线程要等待

其他线程对象操作完成之后才可以进行操作，但是在程序之中过多的线程等待就会出现死锁。

5、编程题

1、 设计四个线程对象，两个线程执行减操作，两个线程执行加操作。

```
class Message {
    private int data = 10 ;                         // 初始值
    private boolean flag = true;
    public synchronized void add() {            // 加法操作
        if (this.flag == false) {                   // 已经生产过了，不能生产
            try {
                super.wait();                       // 等待
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        try {
            Thread.sleep(200);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println("加法操作：" + this.data ++);
        this.flag = false;                          // 已经生产完成，修改标志位
        super.notify();                             // 唤醒等待线程
    }
    public synchronized void subtract() {       // 减法操作
        if (this.flag == true) {                // 未生产，不能取走
            try {
                super.wait();                       // 等待
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        try {
            Thread.sleep(100);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println("减法操作：" + this.data --);
        this.flag = true;                           // 已经取走了，可以继续生产
        super.notify();                             // 唤醒等待线程
    }
    // setter、getter略
}
class Addition implements Runnable {                // 定义生产者
```

```
    private Message msg = null ;
    public Addition(Message msg) {
        this.msg = msg ;
    }
    @Override
    public void run() {
        for (int x = 0; x < 50; x++) {                  // 加法执行50次
            this.msg.add() ;                          // 加法操作
        }
    }
}
class Subtraction implements Runnable {                 // 定义消费者
    private Message msg = null ;
    public Subtraction (Message msg) {
        this.msg = msg ;
    }
    @Override
    public void run() {
        for (int x = 0; x < 50; x++) {                  // 减法执行50次
            this.msg.subtract() ;                     // 执行减法
        }
    }
}
public class TestDemo {
    public static void main(String args[]) {
        Message msg = new Message();
        new Thread(new Addition(msg), "加法对象A").start();          // 启动线程
        new Thread(new Addition(msg), "家访对象B").start();          // 启动者线程
        new Thread(new Subtraction(msg), "减法对象A").start();  // 取得线程
        new Thread(new Subtraction(msg), "减法对象B").start();  // 取得线程
    }
}
```

第 8 章、常用类库 —— 自我检测（答案）

1、填空题

1、 在java.lang包中提供了两个字符串类，分别是 String 和 StringBuffer。这两个类都是 CharSequence 接口的子类，字符串类提供的求字符串长度的方法是 public int length() 。

2、 Java中提供的两个大数操作类是： BigInteger 和 BigDecimal 。

3、 对象克隆方法是 Object 类提供的，方法名称是 public Object clone () thorws CloneNotSupportException ，对象所在的类必须实现 java.lang.Cloneable 接口。

4、 String类的 matches() 、 split() 、 replaceAll() 、 replaceFirst() 四个方法可以使用正则。

5、 通过Object类中的 public Class<?> getClass() 方法可以取得一个类的Class对象。

6、 Constructor类定义在 java.lang.reflect 包中。

7、 Class类对象的三种实例化方式： getClass()方法 、 类.class 、 forName(className) 方法 。

2、选择题

1、 使用Runtime类的 D 方法，可以释放垃圾内存。

A、 exec()　　B、 run()　　C、 invoke()　　D、 gc()

2、 Object类中的 B 方法不能被覆写？

A、 toString()　　B、 getClass()　　C、 clone()　　D、 finalize()

3、 如果要为对象回收做收尾操作，则应该覆写Object类中的 D 方法。

A、 toString()　　B、 getClass()　　C、 clone()　　D、 finalize()

3、判断题

1、 任何类的对象数组都可以使用 Arrays.sort()方法进行排序操作。 （ × ）

2、 Random 类存放在 java.lang 包中。 （ √ ）

3、 Runtime 类的对象可以直接通过构造方法实例化。 （ × ）

4、 Class 类的对象可以通过关键字 new 进行实例化操作。 （ × ）

5、 可以通过 Class 实例化一个类的对象，但是要求此类必须存在无参构造。 （ √ ）

4、简答题

1、 String 类和 StringBuffer 类的区别是什么？StringBuffer 类提供了那些独特的方法？

答： String 类的内容一旦声明则不可修改，而 StringBuffer 类的内容定义之后可以修改。StringBuffer 类使用 append()方法可以完成字符串的连接操作，而 String 类使用“+”完成；

特殊方法：insert()、reverse()、replace()。

2、 简述 final、finally、finalize 的区别及作用？

答： final 表示终结器，用于定义不能被继承的父类，不能被覆写的方法，常量；

finally 是异常处理的出口；

finalize()是 Object 类定义的一个方法，用于执行对象回收前的收尾操作。

3、 解释 Comparable 和 Comparator 的区别。

答： java.lang.Comparable 是在一个类定义的时候默认实现好的接口，里面只有一个 compareTo()方法；

java.util.Comparator 是需要单独定义一个比较的规则类，里面有两个方法；compare()、equals()。

5、编程题

1、 定义一个 StringBuffer 类对象，然后通过 append()方法向对象里添加 26 个小写字母，要求每次只添加一次，共添加 26 次。

```
public class TestDemo {
    public static void main(String args[]) {
        StringBuffer buf = new StringBuffer() ;
        for (char c = 'a' ; c <='z' ; c++){
            buf.append(c) ;                          // 连接字符串
```

```
        }
        System.out.println(buf);
    }
}
```

程序运行结果：	abcdefghijklmnopqrstuvwxyz

2、 利用 Random 类产生 5 个 1~30 之间（包括 1 和 30）的随机整数。

```
import java.util.Random;
public class TestDemo {
    public static void main(String args[]) {
        Random rand = new Random();
        for (int x = 0; x < 5; x++) {
            System.out.print(rand.nextInt(31) + "、");
        }
    }
}
```

程序运行结果	23、2、20、0、30、

3、 输入一个 email 地址，之后使用正则表达式验证该 email 地址是否正确。

```
package cn.mldn.demo;
public class TestDemo {
    public static void main(String[] args) throws Exception {
        String str = "mldnqa@163.net";
        String regex = "[a-zA-Z_][a-zA-Z_0-9\\.]*@[a-zA-Z_0-9\\.]+\\.(com|cn|net)";
        if (str.matches(regex)) {                          // 符合于验证要求
            System.out.println("TRUE，EMAIL输入合法。");
        } else {
            System.out.println("FLASE，EMAIL输入非法！");
        }
    }
}
```

程序运行结果：	TRUE，EMAIL输入合法。

4、 编写正则表达式，判断给定的是否是一个合法的 ip 地址。

```
public class TestDemo {
    public static void main(String[] args) throws Exception {
        String str = "192.168.1.3";
        String regex = "\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}";
        if (str.matches(regex)) {                          // 符合于验证要求
            System.out.println("TRUE，IP地址输入合法。");
        } else {
            System.out.println("FLASE，IP地址输入非法！");
        }
    }
}
```

程序运行结果：	TRUE，IP地址输入合法。

5、 编写程序，将字符串“1981-09-19 09:07:27.727”变为 Date 型数据。

```
import java.text.SimpleDateFormat;
```

```
import java.util.Date;
public class TestDemo {
    public static void main(String[] args) throws Exception {
        String str = "1981-09-19 09:07:27.727";        // 字符串
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss.SSS");
        Date date = sdf.parse(str);                    // 将字符串变为日期
        System.out.println(date);
    }
}
```

程序运行结果：	Sat Sep 19 09:07:27 CST 1981

第 9 章、Java IO 操作 —— 自我检测（答案）

1、填空题

1、 IO操作的所有类都保存在 java.io 包中。

2、 文件输入流是 FileInputStream、FileReader 、文件输出流 FileOutputStream、FileWriter 。

3、 IO操作中字节流的操作类是 InputStream 、 OutputStream ，字符流的操作类是 Reader 和 Writer 。

4、 System类中提供那个三个常量是对IO有所支持的： out 、 err 、 in 。

5、 序列化对象使用 ObjectOutputStream 、 ObjectInputStream 类，对象所在的类必须实现 Serializable 接口，才可以自动序列化所有的内容。

6、 transient 关键字可以让类中的属性不被序列化下来。

2、选择题

1、 File类提供了许多管理磁盘的方法。其中，建立目录的方法是 B 。

A、 delete() B、 mkdirs() C、 makedir() D、 exists()

2、 提供println()方法和print()方法的类是 A 。

A、 PrintStream B、 System C、 InputStream D、 DataOutputStream

3、 不同的操作系统使用不同的路径分隔符。静态常量separator表示路径分隔符，它属于的类是 C 。

A、 FileInputStream B、 FileOutputStream C、 File D、 InputStream

4、 下面的说法不正确的是 D 。

A、 InputStream 与 OutputStream 类通常是用来处理字节流，也就是二进制文件

B、 Reader 与 Writer 类则是用来处理字符流，也就是纯文本文件

C、 Java 中 IO 流的处理通常分为输入和输出两个部分

D、 File 类是输入/输出流类的子类

5、 下面的说法正确的是 A 。

A、 InputStream 与 OutputStream 都是抽象类

B、 Reader 与 Writer 不是抽象类

C、 RandomAccessFile 是抽象类

D、 File 类是抽象类

6、 与InputStream相对应的Java系统的标准输入对象是 A 。

A、 System.in B、 System.out C、 System.err D、 System.exit()

7、 FileOutputStream类的父类是___C___。

A、 File　　B、 FileOutput　　C、 OutputStream　　D、 InputStream

8、 InputStreamReader类提供的功能是___D___。

A、 数据校验　　B、 文本行计数　　C、 压缩　　D、 将字节流变为字符流

3、判断题

1、 字节流操作时使用到了缓冲区，字符流操作时没有使用到缓冲区。（ × ）

2、 File 类用于管理本地磁盘的文件和目录。（ √ ）

3、 通过 read()方法可以从字节输入流读出各种类型的数据。（ √ ）

4、简答题

1、 简述字节流与字符流操作的区别。

答： 字节流没有使用到缓冲区，而字符流使用了；

处理各种数据都可以通过字节流完成，而在处理中文的时候使用字符流会更好。

2、 简述对象序列化的主要作用。

答： 对象序列化主要是指将内存之中保存的对象以二进制数据流的方式进行传输，要实现序列化的对象所在类必须实现 java.io.Seriliazable 接口。

5、编程题

1、 编写 Java 程序，输入 3 个整数，并求出三个整数的最大值、最小值。

```
import java.util.Scanner;
public class TestDemo {
    public static void main(String[] args) throws Exception {
        Scanner scan = new Scanner(System.in);
        int data[] = new int[3]; // 接收三个输入数据
        for (int x = 0; x < data.length;) {
            System.out.print("请输入第" + (x + 1) + "个数字：");
            if (scan.hasNextInt()) { // 输入的是数字
                data[x] = scan.nextInt(); // 接收数字
                x ++ ;
            }
        }
        int max = data[0] ;  // 假设第1个数据为最大值
        int min = data[0] ;  // 假设第1个数据为最小值
        for (int x = 1; x < data.length; x++) {
            if (min > data[x]) {
                min = data[x];
            }
            if (max < data[x]) {
                max = data[x] ;
```

```
            }
        }
        System.out.println("最大值：" + max);
        System.out.println("最小值：" + min);
    }
}
```

程序运行结果：	请输入第1个数字：10 请输入第2个数字：20 请输入第3个数字：30 最大值：30 最小值：10

2、 从键盘输入文件的内容和要保存的文件名称，之后根据输入的名称创建文件，并将内容保存到文件之中。

```
import java.io.File;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.util.Scanner;
public class TestDemo {
    public static void main(String[] args) throws Exception {
        Scanner scan = new Scanner(System.in);
        String fileName = null ;                // 保存文件名称
        String fileContent = null ;                 // 保存文件内容
        System.out.print("请输入文件名称：");          // 提示信息
        scan.useDelimiter("\n") ;               // 设置分隔符
        if (scan.hasNext()) {                   // 有输入内容
            fileName = scan.next().trim() ;             // 接收数据
            System.out.print("请输入文件内容：");
            if (scan.hasNext()) {
                fileContent = scan.next().trim() ;      // 接收数据
                File file = new File(fileName) ;
                if (!file.getParentFile().exists()) { // 文件目录不存在
                    file.getParentFile().mkdirs() ;         // 创建目录
                }
                PrintStream out = new PrintStream(new FileOutputStream(file));
                out.print(fileContent) ;
                out.close() ;
            }
        }
    }
}
```

程序运行结果：	请输入文件名称：d:\mldnjava\lxh.txt 请输入文件内容：www.mldnjava.cn，北京魔乐科技软件学院！

3、 编写程序，程序运行后，根据屏幕提示输入一个数字字符串，输入后统计有多少个偶数数字和奇数数字。

```
import java.util.Arrays;
import java.util.Scanner;
public class TestDemo {
    public static void main(String[] args) throws Exception {
        Scanner scan = new Scanner(System.in);
        scan.useDelimiter("\n") ;
        String data = null ;                // 接收数据
        boolean flag = true ;               // 循环标记
        while(flag) {
            System.out.print("请输入一串数字：");
            if (scan.hasNext()) {
                data = scan.next().trim() ;   // 接收数据
                if (data.matches("\\d+")) {   // 由数字所组成
                    flag = false ;            // 循环结束
                } else {
                    System.out.println("输入数据不是数字，请重新输入！");
                }
            }
        }
        int oddCount = 0 ;                    // 奇数个数
        int evenCount = 0 ;                   // 偶数个数
        String result [] = data.split("") ;   // 逐个拆分
        for (int x = 1 ; x < result.length ; x ++) {
            int temp = Integer.parseInt(result[x]) ;  // 取得每一个数字
            if (temp % 2 == 0) {          // 是偶数
                evenCount ++ ;
            } else {
                oddCount ++ ;
            }
        }
        System.out.println("奇数个数：" + oddCount);
        System.out.println("偶数个数：" + evenCount);
    }
}
```

程序运行结果：	请输入一串数字：123456789 奇数个数：5 偶数个数：4

第 10 章、Java 网络编程 —— 自我检测（答案）

1、选择题

1、 Socket的工作流程是：___C___。

①、打开连接到 Socket 的输入/输出

②、按照某个协议对 Socket 进行的读/写操作

③、创建 Socket

④、关闭 Socket

A、 ①③②④ B、 ②①③④ C、③①②④ D、 ①②③④

2、判断题

1、 java.net 包为网络通讯包。 （ √ ）

2、 ServerSocket 类和 Socket 类主要完成 TCP 程序设计。 （ √ ）

第 11 章、Java 类集框架 —— 自我检测（答案）

1、填空题

1、 在类集中存放单值的最大父接口是 Collection ，存放一对值的最大父接口是 Map 。

2、 Set 接口保存的数据是不允许重复的，并且 TreeSet 子类是可以排序的，根据 Comparable接口 排序。

3、 Java类集可以使用的输出方式是： Iterator 、 ListIterator 、 Enumeration 、 foreach 。

4、 在Java中实现栈操作的类是 Stack 。

2、选择题

1、 下面那个类不是Collection的子类 C 。

A、 ArrayList B、 Vector C、 HashMap D、 TreeSet

2、 HashSet子类依靠 C 方法区分重复元素。

A、 toString()、equals() B、 clone()、equals() C、 hashCode()、equals() D、 getClass()、clone()

3、判断题

1、 List 接口中的内容是不能重复的。 （ √ ）

2、 TreeSet 是排序类。 （ √ ）

3、 Set 接口的内容可以使用 Enumeration 接口进行输出。 （ × ）

4、 Map 接口的内容可以使用 ListIterator 接口进行输出。 （ √ ）

4、简答题

1、 简述 ArrayList 和 Vector 的区别。

答：

No.	区别	ArrayList	Vector
1	推出时间	JDK 1.2	JDK 1.0
2	性能	采用异步处理方式，性能更高	采用同步处理方式，性能相对较低
3	安全性	非线程安全	线程安全

4	输出	Iterator、ListIterator、foreach	Iterator、ListIterator、foreach、Enumeration

2、 简述 HashMap 及 Hashtable 的区别。

答：

No.	区别	HashMap	Hashtable
1	推出时间	JDK 1.2	JDK 1.0
2	性能	采用异步处理方式，性能更高	采用同步处理方式，性能相对较低
3	安全性	非线程安全	线程安全
4	设置 null	允许将 key 或 value 设置为 null	不允许出现 null，否则出现空指向异常

3、 Set 集合中的内容是不允许重复的，Java 依靠什么来判断重复对象？

答： Java 依靠 Object 类中的 hashCode()和 equals()方法来判断重复对象。

4、 TreeSet 类是允许排序的，Java 依靠什么进行对象的排序操作？

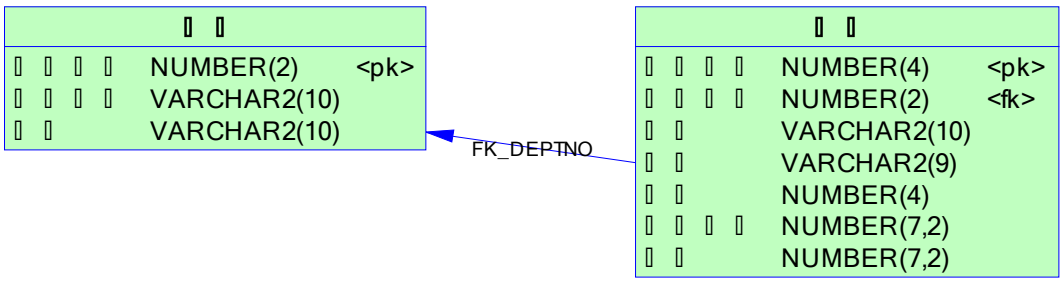
答： 在 TreeSet 类中依靠 Comparable 接口来进行排序。

5、 简述 Collection 和 Collections 的区别。

答： Collection 是一个接口，用于定义集合操作的标准、Collections 是一个工具类，可以操作任意的集合对象。

5、编程题

1、 使用类集实现以下数据表和简单 Java 类的映射实现：




```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
class Emp {                                 // emp表映射类
    private int empno;
    private String ename;
    private String job;
    private double sal;
    private double comm;
    private Emp mgr;                    // 雇员领导
    private Dept dept;                      // 雇员所属部门
    public Emp() {                          // 无参构造
    }
    public Emp(int empno, String ename, String job, double sal, double comm) {
        this.empno = empno;
        this.ename = ename;
        this.job = job;
        this.sal = sal;
        this.comm = comm;
```

```
    }
    // 部分setter、getter略
    public void setDept(Dept dept) {  // 设置雇员所在部门
        this.dept = dept;
    }
    public Dept getDept() {                // 取得雇员所在部门
        return this.dept;
    }
    public void setMgr(Emp mgr) {      // 设置雇员领导
        this.mgr = mgr;
    }
    public Emp getMgr() {              // 取得雇员领导
        return this.mgr;
    }
    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        long temp;
        temp = Double.doubleToLongBits(comm);
        result = prime * result + (int) (temp ^ (temp >>> 32));
        result = prime * result + ((dept == null) ? 0 : dept.hashCode());
        result = prime * result + empno;
        result = prime * result + ((ename == null) ? 0 : ename.hashCode());
        result = prime * result + ((job == null) ? 0 : job.hashCode());
        result = prime * result + ((mgr == null) ? 0 : mgr.hashCode());
        temp = Double.doubleToLongBits(sal);
        result = prime * result + (int) (temp ^ (temp >>> 32));
        return result;
    }
    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Emp other = (Emp) obj;
        if (Double.doubleToLongBits(comm) != Double
                .doubleToLongBits(other.comm))
            return false;
        if (dept == null) {
            if (other.dept != null)
                return false;
```

```
        } else if (!dept.equals(other.dept))
            return false;
        if (empno != other.empno)
            return false;
        if (ename == null) {
            if (other.ename != null)
                return false;
        } else if (!ename.equals(other.ename))
            return false;
        if (job == null) {
            if (other.job != null)
                return false;
        } else if (!job.equals(other.job))
            return false;
        if (mgr == null) {
            if (other.mgr != null)
                return false;
        } else if (!mgr.equals(other.mgr))
            return false;
        if (Double.doubleToLongBits(sal) != Double.doubleToLongBits(other.sal))
            return false;
        return true;
    }
    @Override
    public String toString() {        // 取得雇员信息
        return "雇员编号：" + this.empno + "，姓名：" + this.ename + "，职位：" + this.job
                + "，工资：" + this.sal + "，佣金：" + this.comm;
    }
}
class Dept {                          // dept表映射类
    private int deptno;
    private String dname;
    private String loc;
    private List<Emp> emps;           // 多个雇员
    public Dept() {                       // 无参构造
        this.emps = new ArrayList<Emp>() ;
    }
    public Dept(int deptno, String dname, String loc) {
        this() ;                      // 调用无参构造
        this.deptno = deptno;
        this.dname = dname;
        this.loc = loc;
    }
    // 部分setter、getter略
    public void setEmps(List<Emp> emps) {
```

```
        this.emps = emps;
    }
    public List<Emp> getEmps() {
        return emps;
    }
    @Override
    public String toString() {              // 取得部门信息
        return "部门编号：" + this.deptno + "，部门名称：" + this.dname + "，位置："
                + this.loc;
    }
}
public class TestDemo {
    public static void main(String args[]) {
        // 1、第一层配置关系
        Dept dept = new Dept(10, "ACCOUNTING", "New Yrok");
        Emp empa = new Emp(7369, "SMITH", "CLERK", 800.0, 0.0);
        Emp empb = new Emp(7566, "ALLEN", "MANAGER", 2450.0, 0.0);
        Emp empc = new Emp(7839, "KING", "PRESIDENT", 5000.0, 0.0);
        empa.setMgr(empb);                  // 设置雇员和领导的关系
        empb.setMgr(empc);                  // 设置雇员和领导的关系
        empa.setDept(dept);                 // 每个雇员属于一个部门
        empb.setDept(dept);                 // 每个雇员属于一个部门
        empc.setDept(dept);                 // 每个雇员属于一个部门
        // 每一个部门有多个雇员，通过对象数组表示多个雇员
        dept.getEmps().add(empa) ;
        dept.getEmps().add(empb) ;
        dept.getEmps().add(empc) ;
        // 2、第二层取得关系
        System.out.println(dept);
        Iterator<Emp> iter = dept.getEmps().iterator() ;
        while (iter.hasNext()) {
            Emp emp = iter.next() ;
            System.out.println(emp);
            if (emp.getMgr() != null) { // 有领导
                System.out.println("\t" + emp.getMgr());
            }
            System.out.println("------------------------------------------------------");
        }
    }
}
```

程序运行结果：

```
部门编号：10，部门名称：ACCOUNTING，位置：New Yrok
雇员编号：7369，姓名：SMITH，职位：CLERK，工资：800.0，佣金：0.0
    雇员编号：7566，姓名：ALLEN，职位：MANAGER，工资：2450.0，佣金：0.0
------------------------------------------------------
雇员编号：7566，姓名：ALLEN，职位：MANAGER，工资：2450.0，佣金：0.0
```

	雇员编号：7839，姓名：KING，职位：PRESIDENT，工资：5000.0，佣金：0.0 ---------- 雇员编号：7839，姓名：KING，职位：PRESIDENT，工资：5000.0，佣金：0.0 ----------

第 12 章、Java 数据库操作 —— 自我检测（答案）

1、填空题

1、 要想执行数据库更新的操作接口是：___Statement___和___PreparedStatement___。

3、 数据库查询结果使用___ResultSet___接口保存。

4、 JDBC中通过___Class___类加载数据库驱动程序。

2、选择题

1、 下列___C___不是getConnection()方法的参数？

A、 数据库用户名　　B、 数据库的访问密码

C、 JDBC 驱动器的版本　　D、 连接数据库的 URL

2、 Statement接口中的executeQuery(String sql)方法返回的数据类型是___D___。

A、 Statement 接口实例　　B、 Connection 接口实例

C、 DatabaseMetaData 类的对象　　D、 ResultSet 类的对象

3、 下列不属于更新数据库操作的步骤的一项是___C___

A、 加载 JDBC 驱动程序　　B、 定义连接的 URL

C、 执行查询操作　　D、 执行更新操作

3、判断题

1、 JDBC 的驱动程序要在 classpath 中进行配置。（ √ ）

2、 PreparedStatement 是 Statement 的子接口，使用 PreparedStatement 要比使用 Statement 性能更高。（ √ ）