## java错题

1. 构造方法用于创建类的实例对象，构造方法名应与类名相同，返回类型为void。

答案：错误

解析：构造函数，名称与类相同，没有返回类型的

1. 一个类的构造器不能调用这个类中的其他构造器。

答案：错误

解析：可以用this

1. 在基本 JAVA 类型中，如果不明确指定，整数型的默认是 \_\_ 类型，带小数的默认是 \_\_ 类型？

答案：int double

解析：整型默认为int，如果需要long，须加l或L。小数默认double，d或D可省略，但如果需要float，须加f或F，例如float = 0.1f。

1. 下面关于Spring的说法中错误的是（D）

A、Spring是一系列轻量级Java EE框架的集合

B、Spring中包含一个“依赖注入”模式的实现

C、使用Spring可以实现声明式事务

D、Spring提供了AOP方式的日志系统

答案：D

解析：Spring并没有为我们提供日志系统，我们需要使用AOP（面向方面编程）的方式，借助Spring与日志系统log4j实现我们自己的日志系统。

1. 如果希望监听TCP端口9000，应该怎样创建socket?

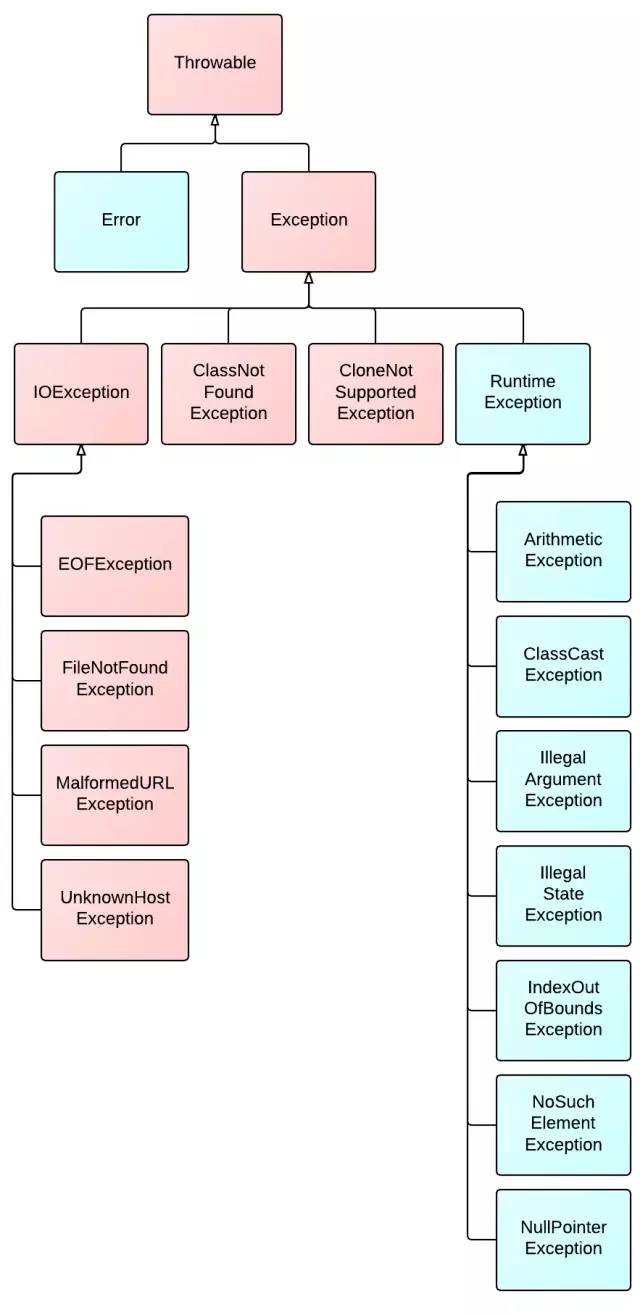
答案：new ServerSocket(9000);

解析：ServerSocket (int port) 创建一个serversocket 绑定在特定的端口

Socket(InetAddress address, int port) 创建一个socket流，连接到特定的端口和ip地址

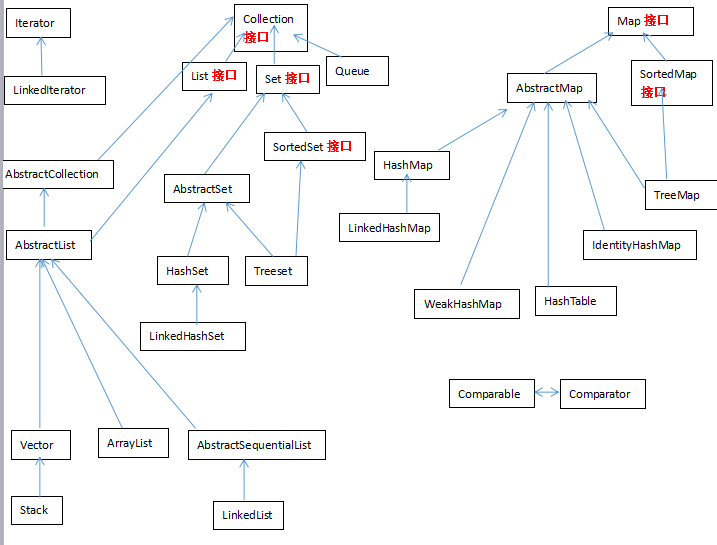
1. 在 java 的异常类的层次组织结构中， 是所有异常类的父类，它是Object类的直接子类。

答案：Throwable

解析：其中绿色方框中的异常是不用手动去抛出或捕捉的异常

1. 以下哪个不是Collection的子接口？
2. List
3. Set
4. SortedSet
5. Map

答案：D

解析：

1. 对于以下代码段，4个输出语句中输出true的个数是( )

class A{}

class B extends A{}

class C extends A{}

class D extends B{}

A obj = new D();

System.out.println(obj instanceof B);

System.out.println(obj instanceof C);

System.out.println(obj instanceof D);

System.out.println(obj instanceof A);

答案：4

解析：这个是用来判断是不是某个类的实例。反过来说，就是这个实例是不是属于某个类。注意，这里说的“类”可以形象为日常生活中的类：比如：泰迪属于狗，狗属于动物。那么泰迪类的实例：一只可爱的泰迪叫小旺，小旺是泰迪，小旺也是狗，小旺也是动物。这样就清楚了。

1. Java程序中的类名称必须与存放该类的文件名相同。

答案：错误

解析：声明为public类型的类名必须与文件名相同，默认权限的可以不同并且内部类的类名一般与文件名不同

1. Java 中的集合类包括 ArrayList 、 LinkedList 、 HashMap 等，下列关于集合类描述错误的是？
2. ArrayList和LinkedList均实现了List接口
3. ArrayList访问速度比LinkedList快
4. 随机添加和删除元素时，ArrayList的表现更加快速
5. HashMap实现Map接口，它允许任何类型的键和值对象，并允许将NULL用作键或值

答案：C

解析：ArrayList和LinkedList都是List的子接口，A正确；

ArrayList由数组实现，LinkedList由链表实现，数组的访问速度比链表快，B正确；随机添加删除元素时，链表的效率要高于数组，C错误；

HashMap允许将null用作键或值，D正确

1. 覆盖与重载的关系是（）

答案：覆盖只有发生在父类与子类之间，而重载可以发生在同一个类中

解析：方法的重载是指在一个类中存在这样的一些方法（包含从父类继承而来的方法），方法名称一样，但是参数的个数不同，或者参数个数相同，但对应的参数类型不同，这些同名的方法为重载；

而覆盖是指一个子类对其父类中的某个可见方法进行了重新的定义，即重新定义的方法满足名称相同，参数相同以及对应的参数相同的条件。这样，调用时将会调用子类中的方法，而不会调用父类中被覆盖的方法。

1. 如果int x=20, y=5，则语句System.out.println(x+y +""+(x+y)+y);  的输出结果是（）

答案：25255

解析：x,y都是整型数，然后输出两个相加之和，但是中间出现了“”，代码默认将“”后面的变成了字符串相加，也就出现了25255

1. 以下声明合法的是
2. default String s
3. public final static native int w()
4. abstract double d
5. abstract final double hyperbolicCosine()

答案：B

解析：A：java的访问权限有public、protected、private和default的，default不能修饰变量

B：native修饰方法，native修饰的方法简单来说就是：一个Java方法调用了一个非Java代码的接口。

定义navtive方法时，并不提供实现体，因为其实现体是用非Java语言在外面实现的。native可以和任何修饰符连用，abstract除外。因为native暗示这个方法时有实现体的，而abstract却显式指明了这个方法没有实现体。

C：普通变量不能用abstract修饰，abstract一般修饰方法和类

D：被定义为abstract的类需要被子类继承，但是被修饰为final的类是不能被继承和改写的

1. 下列整型常量 i 的定义中，正确的是(  )
2. final i;
3. static int i;
4. static final int i=234;
5. final float i=3.14f;

答案：C

解析：因为是整型，所以要申明 int，所以A D不对，又因为是常量，所以 要用final限制。

1. 线程安全的map在JDK 1.5及其更高版本环境 有哪几种方法可以实现?
2. Map map = new HashMap()
3. Map map = new TreeMap()
4. Map map = new ConcurrentHashMap();
5. Map map = Collections.synchronizedMap(new HashMap());

答案：C D

解析：1. HashMap,TreeMap 未进行同步考虑，是线程不安全的。

2. HashTable 和 ConcurrentHashMap 都是线程安全的。区别在于他们对加锁的范围不同，HashTable 对整张Hash表进行加锁，而ConcurrentHashMap将Hash表分为16桶(segment)，每次只对需要的桶进行加锁。

3. Collections 类提供了synchronizedXxx()方法，可以将指定的集合包装成线程同步的集合。比如，

List list = Collections.synchronizedList(new ArrayList());

Set set = Collections.synchronizedSet(new HashSet());

1. 下列哪个选项是错误的。（）
2. 一个文件中只能有一个public class。
3. 一个文件中可以有多个类。
4. 一个类中可以有两个main方法。
5. 若类中只含一个main方法，则必须是public的。

答案：A

解析：一个类中只允许有一个public static void main，但是允许有两个main方法，因为可以重载

1. 下列关于包（package）的描述，正确的是（）
2. 包（package）是Java中描述操作系统对多个源代码文件组织的一种方式。
3. import语句将所对应的Java源文件拷贝到此处执行。
4. 包（package）是Eclipse组织Java项目特有的一种方式。
5. 定义在同一个包（package）内的类可以不经过import而直接相互使用。

答案：D

解析：1、为了更好地组织类，Java提供了包机制。包是类的容器，用于分隔类名空间。如果没有指定包名，所有的示例都属于一个默认的无名包。Java中的包一般均包含相关的类，java是跨平台的，所以java中的包和操作系统没有任何关系，java的包是用来组织文件的一种虚拟文件系统。A错

2、import语句并没有将对应的java源文件拷贝到此处仅仅是引入，告诉编译器有使用外部文件，编译的时候要去读取这个外部文件。B错

3、Java提供的包机制与IDE没有关系。C错

4、定义在同一个包（package）内的类可以不经过import而直接相互使用。

1. 系统在调用重载函数时，能作为确定哪个重载函数被调用的依据是（）
2. 函数返回类型
3. 参数个数
4. 函数名称
5. 参数名称

答案：B

解析：应该是参数类型和个数一起决定的

1. String str1 = “abc”，“abc”分配在内存哪个区域？
2. 堆
3. 栈
4. 字符串常量区
5. 寄存器

答案：C

解析：用new创建的对象在堆区

函数中的临时变量在栈去

java中的字符串在字符串常量区

1. HashMap中是用哪些方法来解决哈希冲突的？
2. 开放地址法
3. 二次哈希法
4. 链地址法
5. 建立一个公共溢出区

答案：C

解析：以上方法都是解决哈希冲突的策略，但是在java.util.HashMap中，总体来说是使用的链地址法来解决冲突的。

使用链地址法会导致get的效率从o（1）降至o（n），所以在Java8中，使用的是平衡树来解决提高效率的。

1. 下面有关java内存模型的描述，说法错误的是？
2. JMM通过控制主内存与每个线程的本地内存之间的交互，来为java程序员提供内存可见性保证
3. “synchronized” — 保证在块开始时都同步主内存的值到工作内存，而块结束时将变量同步回主内存
4. “volatile” — 保证修饰后在对变量读写前都会与主内存更新。
5. 如果在一个线程构造了一个不可变对象之后（对象仅包含final字段），就可以保证了这个对象被其他线程正确的查看

答案：D

解析：Java线程之间的通信由Java内存模型（简称为JMM）控制，JMM决定一个线程对共享变量的写入何时对另一个线程可见。从抽象的角度来看，JMM定义了线程和主内存之间的抽象关系：线程之间的共享变量存储在主内存（main memory）中，每个线程都有一个私有的本地内存（local memory），本地内存中存储了该线程以读/写共享变量的副本。本地内存是JMM的一个抽象概念，并不真实存在。它涵盖了缓存，写缓冲区，寄存器以及其他的硬件和编译器优化

volatile变量的写-读可以实现线程之间的通信。

从内存语义的角度来说，volatile与监视器锁有相同的效果：volatile写和监视器的释放有相同的内存语义；volatile读与监视器的获取有相同的内存语义。

1. Java程序的种类有（ ）
2. 类（Class）
3. Applet
4. Application
5. Servlet

答案：B C D

解析：Java程序的种类有：

（a）内嵌于Web文件中，由浏览器来观看的\_Applet

（b）可独立运行的 Application

（c）服务器端的 Servlets

1. 下面属于java包装类的是？
2. String
3. Long
4. Character
5. Short

答案：B C D

解析：Java 语言是一个面向对象的语言，但是Java中的基本数据类型却是不面向对象的，这在实际使用时存在很多的不便，为了解决这个不足，在设计类时为每个基本数据类型设计了一个对应的类进行代表，即包装类。对应的基本类型和包装类如下表：

|  |  |
| --- | --- |
| 基本数据类型 | 包装类 |
| byte | Byte |
| boolean | Boolean |
| short | Short |
| char | Character |
| int | Integer |
| long | Long |
| float | Float |
| double | Double |

包装类是针对基本数据类型的。

1. 以下哪些类是线程安全的（）
2. Vector
3. HashMap
4. ArrayList
5. StringBuffer
6. Properties

答案：A D E

解析：A，Vector相当于一个线程安全的List

B，HashMap是非线程安全的，其对应的线程安全类是HashTable

C，Arraylist是非线程安全的，其对应的线程安全类是Vector

D，StringBuffer是线程安全的，相当于一个线程安全的StringBuilder

E，Properties实现了Map接口，是线程安全的

1. 在使用super 和this关键字时，以下描述正确的是（）
2. 在子类构造方法中使用super（）显示调用父类的构造方法，super（）必须写在子类构造方法的第一行，否则编译不通过
3. super（）和this（）不一定要放在构造方法内第一行
4. this（）和super（）可以同时出现在一个构造函数中
5. this（）和super（）可以在static环境中使用，包括static方法和static语句块

答案：A

解析：使用super()或者this()方法是必须放在构造函数的第一行

由于this函数指向的构造函数默认有super()方法，所以规定this()和super()不能同时出现在一个构造函数中。

因为staic方法或者语句块没有实例时可以使用，而此时不需要构造实例，所以不能用this()和super()。

1. 下列代码的输出结果是\_\_\_\_\_

boolean b=true?false:true==true?false:true;

System.out.println(b);

1. true
2. false
3. null
4. 空字符串

答案：B

解析：== 优先级高于 三目运算符，先判断 true == true，此时返回为 true,

这时表达式为 boolean b = true?false:true?false:true

此时三目运算符从右向左执行,true?false:true，返回false

这时表达式为 boolean b = true?false:false;

结果为：boolean b = false ;

1. class A {}  
   class B extends A {}  
   class C extends A {}  
   class D extends B {}  
   Which four statements are true ?
2. The type List<A>is assignable to List.
3. The type List<B>is assignable to List<A>.
4. The type List<Object>is assignable to List<?>.
5. The type List<D>is assignable to List<?extends B>.
6. The type List<?extends A>is assignable to List<A>.
7. The type List<Object>is assignable to any List reference.
8. The type List<?extends B>is assignable to List<?extends A>.

答案：A C D G

解析：1. 只看尖括号里边的！！明确点和范围两个概念

2. 如果尖括号里的是一个类，那么尖括号里的就是一个点，比如List<A>,List<B>,List<Object>

3. 如果尖括号里面带有问号，那么代表一个范围，<? extends A> 代表小于等于A的范围，<? super A>代表大于等于A的范围，<?>代表全部范围

4. 尖括号里的所有点之间互相赋值都是错，除非是俩相同的点

5. 尖括号小范围赋值给大范围，对，大范围赋值给小范围，错。如果某点包含在某个范围里，那么可以赋值，否则，不能赋值

6. List<?>和List 是相等的，都代表最大范围

----------------------------------------------------------------------------------

7.补充：List既是点也是范围，当表示范围时，表示最大范围

1. 检查程序，是否存在问题，如果存在指出问题所在，如果不存在，说明输出结果。

public class HelloB extends HelloA

{

public HelloB()

{

}

{

System.out.println("I’m B class");

}

static

{

System.out.println("static B");

}

public static void main(String[] args)

{

new HelloB();

}

}

class HelloA

{

public HelloA()

{

}

{

System.out.println("I’m A class");

}

static

{

System.out.println("static A");

}

}

1. static A

I’m A class

static B

I’m B class

1. I’m A class

I’m B class

static A

static B

1. static A

static B

I’m A class

I’m B class

1. I’m A class

static A

I’m B class

static B

答案：C

解析：1.静态代码块 2.构造代码块3.构造方法的执行顺序是1>2>3;明白他们是干嘛的就理解了。

1.静态代码块：是在类的加载过程的第三步初始化的时候进行的，主要目的是给类变量赋予初始值。

2.构造代码块：是独立的，必须依附载体才能运行，Java会把构造代码块放到每种构造方法的前面，用于实例化一些共有的实例变量，减少代码量。

3.构造方法：用于实例化变量。

1是类级别的，2、3是实例级别的，自然1要优先23.

在就明白一点：对子类得主动使用会导致对其父类得主动使用，所以尽管实例化的是子类，但也会导致父类的初始化和实例化，且优于子类执行。

1. 下列对继承的说法，正确的是（ ）
2. 子类能继承父类的所有方法和状态
3. 子类继承父类的非私有方法和状态
4. 子类只能继承父类的public方法和状态
5. 子类只能继承父类的方法

答案：A

解析：private方法是可以继承的，只是不能调用。子类对象确实拥有父类对象中所有的属性和方法，但是父类对象中的私有属性和方法，子类是无法访问到的，只是拥有，但不能使用。

1. 关于访问权限说法正确的是？（）
2. 类定义前面可以修饰public,protected和private
3. 内部类前面可以修饰public,protected和private
4. 局部内部类前面可以修饰public,protected和private
5. 以上说法都不正确

答案：B

解析：对于外部类来说，只有两种修饰，public和默认（default），因为外部类放在包中，只有两种可能，包可见和包不可见。

对于内部类来说，可以有所有的修饰，因为内部类放在外部类中，与成员变量的地位一致，所以有四种可能。

1. 在（）情况下适宜采用内联函数
2. 函数代码小，频繁调用
3. 函数代码多，频繁调用
4. 函数体含有递归语句
5. 函数体含有循环语句

答案：A

解析：内联函数就是在程序编译时，编译器将程序中出现的内联函数的调用表达式用内联函数的函数体来直接进行替换。显然，这样就不会产生转去转回的问题，但是由于在编译时将函数体中的代码被替代到程序中，因此会增加目标程序代码量，进而增加空间开销，而在时间代销上不象函数调用时那么大，可见它是以目标代码的增加为代价来换取时间的节省。

1. 关于sleep()和wait()，以下描述错误的一项是（ ）
2. sleep是线程类（Thread）的方法，wait是Object类的方法；
3. sleep不释放对象锁，wait放弃对象锁
4. sleep暂停线程、但监控状态仍然保持，结束后会自动恢复
5. wait后进入等待锁定池，只有针对此对象发出notify方法后获得对象锁进入运行状态

答案：D

解析：Java中的多线程是一种抢占式的机制，而不是分时机制。抢占式的机制是有多个线程处于可运行状态，但是只有一个线程在运行。

共同点 ：

1. 他们都是在多线程的环境下，都可以在程序的调用处阻塞指定的毫秒数，并返回。

2. wait()和sleep()都可以通过interrupt()方法打断线程的暂停状态，从而使线程立刻抛出InterruptedException。

如果线程A希望立即结束线程B，则可以对线程B对应的Thread实例调用interrupt方法。如果此刻线程B正在wait/sleep/join，则线程B会立刻抛出InterruptedException，在catch() {} 中直接return即可安全地结束线程。

需要注意的是，InterruptedException是线程自己从内部抛出的，并不是interrupt()方法抛出的。对某一线程调用 interrupt()时，如果该线程正在执行普通的代码，那么该线程根本就不会抛出InterruptedException。但是，一旦该线程进入到 wait()/sleep()/join()后，就会立刻抛出InterruptedException 。

不同点 ：

1.每个对象都有一个锁来控制同步访问。Synchronized关键字可以和对象的锁交互，来实现线程的同步。

sleep方法没有释放锁，而wait方法释放了锁，使得其他线程可以使用同步控制块或者方法。

2.wait，notify和notifyAll只能在同步控制方法或者同步控制块里面使用，而sleep可以在任何地方使用

3.sleep必须捕获异常，而wait，notify和notifyAll不需要捕获异常

4.sleep是线程类（Thread）的方法，导致此线程暂停执行指定时间，给执行机会给其他线程，但是监控状态依然保持，到时后会自动恢复。调用sleep不会释放对象锁。

5.wait是Object类的方法，对此对象调用wait方法导致本线程放弃对象锁，进入等待此对象的等待锁定池，只有针对此对象发出notify方法（或notifyAll）后本线程才进入对象锁定池准备获得对象锁进入运行状态。

1. 说明输出结果

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | package test;  import java.util.Date;  public class SuperTest extends Date{      private static final long serialVersionUID = 1L;      private void test(){         System.out.println(super.getClass().getName());      }        public static void main(String[]args){         new SuperTest().test();      }  } |

1. SuperTest
2. SuperTest.class
3. test.SuperTest
4. test.SuperTest.class

答案：C

解析：TestSuper和Date的getClass都没有重写，他们都是调用Object的getClass，而Object的getClass作用是返回的是运行时的类的名字。这个运行时的类就是当前类，所以

super.getClass().getName()

返回的是test.SuperTest，与Date类无关

要返回Date类的名字需要写super.getClass().getSuperclass()。

1. 下列有关Servlet的生命周期，说法不正确的是？
2. 在创建自己的Servlet时候，应该在初始化方法init()方法中创建Servlet实例
3. 在Servlet生命周期的服务阶段，执行service()方法，根据用户请求的方法，执行相应的doGet()或是doPost()方法
4. 在销毁阶段，执行destroy()方法后会释放Servlet 占用的资源
5. destroy()方法仅执行一次，即在服务器停止且卸载Servlet时执行该方法

答案：A

解析：创建Servlet的实例是由Servlet容器来完成的，且创建Servlet实例是在初始化方法init()之前

Servlet的生命周期分为5个阶段：加载、创建、初始化、处理客户请求、卸载。

(1)加载：容器通过类加载器使用servlet类对应的文件加载servlet

(2)创建：通过调用servlet构造函数创建一个servlet对象

(3)初始化：调用init方法初始化

(4)处理客户请求：每当有一个客户请求，容器会创建一个线程来处理客户请求

(5)卸载：调用destroy方法让servlet自己释放其占用的资源

1. 给出以下代码,请给出结果.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | class Two{      Byte x;  }  class PassO{      public static void main(String[] args){          PassO p=new PassO();          p.start();      }      void start(){          Two t=new Two();          System.out.print(t.x+””);          Two t2=fix(t);          System.out.print(t.x+” ” +t2.x);      }      Two fix(Two tt){          tt.x=42;          return tt;      }  } |

1. null null 42
2. null 42 42
3. 0 0 42
4. 0 42 42
5. An exception is thrown at runtime
6. Compilation

答案：B

解析：注意x是Byte类型，也就是byte的包装类型，属于引用类型。实例该类对象时，如果成员变量没有显示初始化那么Java默认初始化为null.

该题中引用类型t作为形参进行传递，形参的改变会改变实参的值，所以再次打印t.x时已经变为42了

1. 下面哪一项不属于优化Hibernate所鼓励的？
2. 使用单向一对多关联，不使用双向一对多
3. 不用一对一，用多对一取代
4. 配置对象缓存，不使用集合缓存
5. 继承类使用显式多态

答案：A

解析：Hibernate鼓励使用双向一对多关联，不使用单向一对多关联。单向一对多关联映射是在one端维护关系的，必须先保存many端后才可以保存one端，所以在保存many端时该端不知道one端是否存在相应的数据，所以只能将维护的关系字段设置为null，如果为非空则无法保存。因为是one端维护关系，所以在保存one端时，会发出多余的update语句维护many端的外键关系。

1. 根据以下代码段，下列说法中正确的是(    )。

public class Parent {

    private void m1(){}

    void m2(){}

    protected void m3(){}

    public static void m4(){}

}

1. 子类中一定能够继承和覆盖Parent类的m1方法
2. 子类中一定能够继承和覆盖Parent类的m2方法
3. 子类中一定能够继承和覆盖Parent类的m3方法
4. 子类中一定能够继承和覆盖Parent类的m4方法

答案：C

解析：



static修饰的方法，在类创建的时候就写到了内存中，不能被修改 ，public和protected都可以作用于子类，但在多态情况下，静态函数调用时编译和运行看左边，所以子父类存在同名静态函数访问的是父类，子类并不能覆盖父类的方法，所以选C。

1. 下面不属于HttpServletRequest接口完成功能的是？
2. 读取cookie
3. 读取HTTP头
4. 设定响应的content类型
5. 读取路径信息

答案：C

解析：HttpServletRequest类主要处理：

1.读取和写入HTTP头标

2.取得和设置cookies

3.取得路径信息

4.标识HTTP会话

1. 下面代码的输出结果是什么？

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | public class ZeroTest {      public static void main(String[] args) {       try{         int i = 100 / 0;         System.out.print(i);    }catch(Exception e){         System.out.print(1);         throw new RuntimeException();    }finally{         System.out.print(2);    }        System.out.print(3);   }   } |

1. 3
2. 123
3. 1
4. 12

答案：D

解析：还是需要理解Try...catch...finally与直接throw的区别：try catch是直接处理，处理完成之后程序继续往下执行，throw则是将异常抛给它的上一级处理，程序便不往下执行了。本题的catch语句块里面，打印完1之后，又抛出了一个RuntimeException，程序并没有处理它，而是直接抛出，因此执行完finally语句块之后，程序终止了。

1. ResultSet中记录行的第一列索引为？
2. -1
3. 0
4. 1
5. 以上都不是

答案：C

解析：ResultSet跟普通的数组不同，索引从1开始而不是从0开始

1. 下列程序的运行结果

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | public void getCustomerInfo() {          try {              // do something that may cause an Exception          } catch (java.io.FileNotFoundException ex) {              System.out.print("FileNotFoundException!");          } catch (java.io.IOException ex) {              System.out.print("IOException!");          } catch (java.lang.Exception ex) {              System.out.print("Exception!");          }      } |

1. IOException!
2. IOException!Exception!
3. FileNotFoundException!IOException!
4. FileNotFoundException!IOException!Exception!

答案：A

解析：题目说抛出一个异常，但是没说具体是什么异常，那么就要分情况了：

1.如果抛出一个FileNotFoundException(或其子类)，那么最终结果就打印FileNotFoundException

2.如果抛出一个IOException,或者IOException的子类(不包含FileNotFoundException及其子类)，那么最终结果就打印IOException

3.如果抛出一个Exception(不包含IOException及其子类),那么最终结果就打印Exception.

以上，所以3个皆有可能.但是，不管是哪一种情况，只会输出其中之一。

从答案上来看，B,C，D的输出情况是不存在的。因此选A

1. 下列在Java语言中关于数据类型和包装类的说法，正确的是（）
2. 基本（简单）数据类型是包装类的简写形式，可以用包装类替代基本（简单）数据类型
3. long和double都占了64位（64bit）的存储空间。
4. 默认的整数数据类型是int，默认的浮点数据类型是float。
5. 和包装类一样，基本（简单）数据类型声明的变量中也具有静态方法，用来完成进制转化等。

答案：B

解析：A基本类型和包装类是两回事，不是同一个概念。包装类像普通的类一样，封装了一系列的方法

C默认的浮点数据类型是float

D基本数据类型没有静态方法的，包装类却可以，因为包装类像普通的类一样，封装了一系列的方法

1. 在异常处理中，若try中的代码可能产生多种异常则可以对应多个catch语句，若catch中的参数类型有父类子类关系，此时应该将父类放在后面，子类放在前面。

答案：正确

解析：若 catch 中的参数类型有父类子类关系，此时应该将父类放在后面，子类放在前面

1. JSP分页代码中，哪个步骤次序是正确的？
2. 先取总记录数，得到总页数，再取所有的记录，最后显示本页的数据。
3. 先取所有的记录，得到总页数，再取总记录数，最后显示本页的数据。
4. 先取总页数，得到总记录数，再取所有的记录，最后显示本页的数据。
5. 先取本页的数据，得到总页数，再取总记录数，最后显示所有的记录。

答案：A

解析：1、count(\*)得到总记录数

2、计算总页数

3、获取所有记录(个人感觉这一步不需要，可以直接获取指定页数数据)

4、过滤显示本页数据

1. 下面有关JVM内存，说法错误的是？
2. 程序计数器是一个比较小的内存区域，用于指示当前线程所执行的字节码执行到了第几行，是线程隔离的
3. Java方法执行内存模型，用于存储局部变量，操作数栈，动态链接，方法出口等信息，是线程隔离的
4. 方法区用于存储JVM加载的类信息、常量、静态变量、即使编译器编译后的代码等数据，是线程隔离的
5. 原则上讲，所有的对象都在堆区上分配内存，是线程之间共享的

答案：C

解析：运行时数据区包括：虚拟机栈区，堆区，方法区，本地方法栈，程序计数器

虚拟机栈区 ：也就是我们常说的栈区，线程私有，存放基本类型，对象的引用和 returnAddress ，在编译期间完成分配。

堆区 ， JAVA 堆，也称 GC 堆，所有线程共享，存放对象的实例和数组， JAVA 堆是垃圾收集器管理的主要区域。

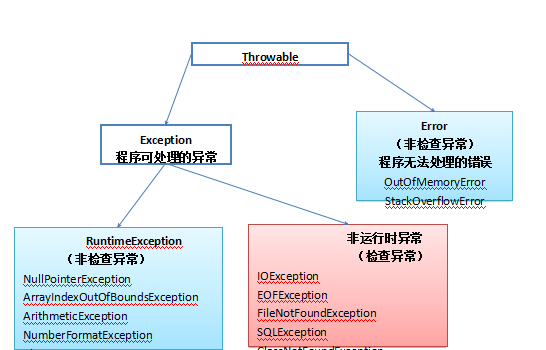
方法区 ：所有线程共享，存储已被虚拟机加载的类信息，常量，静态变量，即时编译器编译后的代码等数据。这个区域的内存回收目标主要是针对常量池的对象的回收和对类型的卸载。

程序计数器 ：线程私有，每个线程都有自己独立的程序计数器，用来指示下一条指令的地址。

1. 下面有关 JAVA 异常类的描述,说法正确的有()
2. 异常的继承结构:基类为 Throwable,Error 和 Exception 继承 Throwable,RuntimeException 和 IOException 等继承 Exception
3. 非 RuntimeException 一般是外部错误(不考虑Error的情况下),其必须被 try{}catch 语句块所捕获
4. Error 类体系描述了 Java 运行系统中的内部错误以及资源耗尽的情形,Error 不需要捕捉
5. RuntimeException 体系包括错误的类型转换、数组越界访问和试图访问空指针等等,必须 被 try{}catch 语句块所捕获

答案：A B C

解析：



都是Throwable的子类：

1.Exception（异常） :是程序本身可以处理的异常。

2.Error（错误）: 是程序无法处理的错误。这些错误表示故障发生于虚拟机自身、或者发生在虚拟机试图执行应用时，一般不需要程序处理。

3.检查异常（编译器要求必须处置的异常） ：  除了Error，RuntimeException及其子类以外，其他的Exception类及其子类都属于可查异常。这种异常的特点是Java编译器会检查它，也就是说，当程序中可能出现这类异常，要么用try-catch语句捕获它，要么用throws子句声明抛出它，否则编译不会通过。

4.非检查异常(编译器不要求处置的异常): 包括运行时异常（RuntimeException与其子类）和错误（Error）。

1. 在Jdk1.7中，下述说法中抽象类与接口的区别正确的有哪些？
2. 抽象类中可以有普通成员变量，接口中没有普通成员变量。
3. 抽象类和接口中都可以包含静态成员常量。
4. 一个类可以实现多个接口，但只能继承一个抽象类
5. 抽象类中可以包含非抽象的普通方法，接口中的方法必须是抽象的，不能有非抽象的普通方法。

答案：A B C D

解析：接口（interface）可以说成是抽象类的一种特例，接口中的所有方法都必须是抽象的。接口中的方法定义默认为public abstract类型，接口中的成员变量类型默认为public static final。另外，接口和抽象类在方法上有区别：

1.抽象类可以有构造方法，接口中不能有构造方法。

2.抽象类中可以包含非抽象的普通方法，接口中的所有方法必须都是抽象的，不能有非抽象的普通方法。

3.抽象类中可以有普通成员变量，接口中没有普通成员变量

4. 抽象类中的抽象方法的访问类型可以是public，protected和默认类型

5. 抽象类中可以包含静态方法，接口中不能包含静态方法

6. 抽象类和接口中都可以包含静态成员变量，抽象类中的静态成员变量的访问类型可以任意，但接口中定义的变量只能是public static final类型，并且默认即为public static final类型

7. 一个类可以实现多个接口，但只能继承一个抽象类。二者在应用方面也有一定的区别：接口更多的是在系统架构设计方法发挥作用，主要用于定义模块之间的通信契约。而抽象类在代码实现方面发挥作用，可以实现代码的重用，例如，模板方法设计模式是抽象类的一个典型应用，假设某个项目的所有Servlet类都要用相同的方式进行权限判断、记录访问日志和处理异常，那么就可以定义一个抽象的基类，让所有的Servlet都继承这个抽象基类，在抽象基类的service方法中完成权限判断、记录访问日志和处理异常的代码，在各个子类中只是完成各自的业务逻辑代码。

1. 下列哪些情况可以终止当前线程的运行？
2. 当一个优先级高的线程进入就绪状态时
3. 当该线程调用sleep()方法时
4. 当创建一个新线程时
5. 抛出一个异常时

答案：D

解析：优先级高的并不一定会马上执行。

sleep方法会阻塞一个线程并不会终止

创建一个新的线程时也不会终止另一个线程

当抛出一个异常后程序会结束，所以线程也会被终止

## 数据结构错题

1. 不能将字符串“Good!”存放到数组 s 中的代码是（）
2. char s[8] = {'G','o','o','d','!', '\0'};
3. char s[8];strcpy(s, "Good!");
4. char s[8];s = "Good!";
5. char s[8] = "Good!";

答案：C

解析：char数组只有在初始化的时候才能整体赋值。

1. 串是一种数据对象和操作都特殊的线性表()

答案：对

1. 下面函数的功能是

int fun (char \*s)

{

    char \*p=s;

    while(\*p++);

    return p-s-1;

}

1. 计算字符串的位(bit)数
2. 复制一个字符串
3. 求字符串的长度
4. 求字符串存放的位置

答案：C

解析：while(\*p++); //此句执行结束，p指向的是字符串末尾的"\0"

此时s仍然是指向字符串的开头

所以 p-s-1是求字符串的长度，-1就是去掉"\0"的长度1

1. 若有以下程序

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | main( )  {  char c1,c2;  c1 ='C'+'8'－'3';  c2 ='9'－'0';  printf("%c %d\n",c1,c2);  } |

则程序的输出结果是

1. H'9'
2. H 9
3. F'9'
4. 表达式不合法输出无定值

答案：B

解析：字符变量c1被赋值为'C'+'8'－'3',即ASSCII码的运算,67十54－49=72,即H;字符变量 c2被赋值为'9'－'0',但输出时,需要注意的是c1以字符变量输出,而c2是以十进制整型变量输出。因此B选项正确。

1. 某字符串满足:concat(head(s),head(tail(tail(s))))="ac",(head,tail的定义同广义表),则S=()
2. aabc
3. acba
4. accc
5. acac

答案：C

解析：概念解释：广义表L=(A,B,C),表头是A，表尾是（B,C)，这是定义。

tail()表示取字符串的尾部，head()表示取字符串的头，concat()表示字符串拼接。

取两次尾部一次头部，然后合并，答案是C。

1. 串是一种特殊的线性表,其特殊性体现在()
2. 数据元素是一个字符
3. 可以顺序存储
4. 数据元素可以是多个字符
5. 可以链接存储

答案：A

解析：串就是字符串,是一种特殊的线性表,它的每个结点仅由一个字符组成。

串值也可用链表来存储,由于串的数据元素是一个字符,它只有8位二进制数, 因此用链表存储时,通常一个结点中存放的不是一个字符,而是一个子串,例如: 在编辑系统中,整个文本编辑区可以看成是一个串,每一行是一个子串,构成一个结点。

1. 用二进制来编码字符串“abcdabaa”，需要能够根据编码，解码回原来的字符串，最少需要()长的二进制字符串？
2. 12
3. 14
4. 18
5. 24

答案：B

解析：这道题需要对abcd进行Huffman编码。首先根据权值建立Huffman树，得到最优编码：

a=0, b=10, c=110, d=111

然后数一下就行了。

1. 字符串′ababaabab′的nextval为()
2. (0,1,0,1,0,4,1,0,1)
3. (0,1,0,1,0,2,1,0,1)
4. (0,1,0,1,0,0,0,1,1)
5. (0,1,0,1,0,1,0,1,1)

答案：A

解析：

i     0  1  2  3  4  5  6  7  8

s     a  b  a  b  a  a  b  a  b

next  -1  0  0  1  2  3  1  2  3

nextval[0] = -1;

s[1]=b != s[next[1]]=s[0]=a; nextval[1]=next[1]=0;

s[2]=a = s[next[2]]=s[0]=a, nextval[2]=nextval[0]=-1;

s[3]=b = s[next[3]]=s[1]=b, nextval[3]=nextval[1]0;

s[4]=a = s[next[4]]=s[2]=a, nextval[4]=nextval[2]=-1;

s[5]=a != s[next[5]]=s[3]=b, nextval[5]=next[5]=3;

s[6]=b = s[next[6]]=s[1]=b, nextval[6]=nextval[1]=0;

s[7]=a = s[next[7]]=s[2]=a, nextval[7]=nextval[2]=-1;

s[8]=b = s[next[8]]=s[3]=b, nextval[8]=nextval[3]=0;

nextval -1 0 -1 0 -1 3 0 -1 0

有的时候下标从1开始即 0 1 0 1 0 4 1 0 1

1. 在一个存放字符串的数组char  p\_str[]，要把字符串中第4个字符的值改为'a'，正确的做法是？
2. p\_str[3]='a'
3. \*(p\_str+3)='a'
4. p\_str[4]='a'
5. \*(p\_str+4)='a'

答案：A D

解析：数组名是指针，p\_str[3] ＝＝ \*(p\_str+3)。

1. 已知一段文本有1382个字符，使用了1382个字节进行存储，这段文本全部是由a、b、c、d、e这5个字符组成，a出现了354次，b出现了483次，c出现了227次，d出现了96次，e出现了232次，对这5个字符使用哈夫曼（Huffman）算法进行编码，则以下哪些说法正确（）
2. 使用哈夫曼算法编码后，用编码值来存储这段文本将花费最少的存储空间
3. 使用哈夫曼算法进行编码，a、b、c、d、e这5个字符对应的编码值是唯一确定的
4. 使用哈夫曼算法进行编码，a、b、c、d、e这5个字符对应的编码值可以有多套，但每个字符编码的位（bit）数是确定的
5. b这个字符的哈夫曼编码值位数应该最短，d这个字符的哈夫曼编码值位数应该最长

答案：A C D

解析：A正确，Huffman树就是求最优解。可以有多套方案，但最终每套方案生成的编码长度都相同且都是最优解。

B错误，我们可以将左子树定为1右子树定为0也可以反之，不同的方案获得的编码值是不同的，但每个字符的编码长度是固定的。

C正确，不同的方案影响的只是通向节点的路径为0还是1，而不会影响Huffman树的层次结构

D正确，生成了Huffman树之后，我们就能看到，出现频率越高的节点越靠近根，深度越小即编码值尾数越短；出现频率越低的节点越远离根，深度越大即编码位数越长。

1. 下列排序算法中，某一趟结束后未必能选出一个元素放在其最终位置上的是（）。
2. 堆排序
3. 冒泡排序
4. 快速排序
5. 直接插入排序

答案：D

解析：A：堆排序每趟总能选出一个最大值或者最小值位于根节点。

B：冒泡排序总是两两比较选出一个最小值位于数组前面。

C：快排选出的枢轴在一趟排序中就位于了它最终的位置

D：直接插入排序不一定会位于最终的位置，因为不确定后面插入的元素对于前面的元素是否产生影响。

1. 拓扑排序算法把一个无向图中的顶点排成一个有序序列。（ ）

答案：错误

解析：拓扑排序算法把一个无向图中的顶点排成一个线性序列

1. 拓扑排序算法仅能适用于有向无环图。（ ）

答案：错误：

解析：对有向无环图，拓扑排序成功；否则，图中有环，不能说算法不适合。

1. 在堆排序算法中我们用一个数组A来模拟二叉树T，如果该A[0]存放的是T的根节点，那么A[K](K>0)的父亲节点是
2. (K-1)/2
3. K/2
4. (K+1)/2
5. 都不对

答案：A

解析：一般堆排序的话我们都把A[0]作为一个哨兵，不存储实际数据，此时（父节点=子节点/2），而题目要求是A[0]存放的是根节点，则我们可以用相同的转换（父节点=（子节点-1）/2）。

1. 序列{2,1,4,9,8,10,6,20}是某排序算法第二轮排序的结果，则该算法只能是
2. 快速排序
3. 冒泡排序
4. 选择排序
5. 插入排序

答案：A

解析：冒泡排序特征第一趟排序之后最大值会在最后面，第二趟排序会在次后面

选择排序特征第一趟排序之后最小值会在最前面，第二趟排序会在次前面

插入排序特征第一趟排序范围0~1,前一个数比后一个数小，第二趟排序范围0~2，前三个数小大排列

快速排序以一个值为分界点

1. 向一个有 127 个元素的顺序表中插入一个新元素并保持原来顺序不变，平均要移动的元素数是（）
2. 8
3. 63.5
4. 64
5. 7

答案：B

解析：在表首插入要移动127个元素,第二个元素插入移动126....表位插入移动0个元素,总共插入机会有128次,移动的数分别是 127+126+125...+1+0再除以128,就是平均的,也就是63.5

1. 有些排序算法在每趟排序过程中,都会有一个元素被放置在其最终的位置上,下列算法不会出现此情况的是()
2. 希尔排序
3. 堆排序
4. 起泡排序
5. 快速排序

答案：A

解析：希尔排序在稳定性上面来说是不稳定的，不能保证每次排序过程中，都会有一个元素放置在其最终的位置上面。

1. 在一个元素个数为N的数组里，找到升序排在N/5位置的元素的最优算法时间复杂度是
2. O(n)
3. O(n log n)
4. O(n (log n)2)
5. O(n 3/2)

答案：A

解析：利用快排，一开始将数列分为两部分，用时为n，根据n/5然后在左部分或者右部分继续分割，用时n/2，。。。。最后时间总的为n+n/2+n/4+n/8...=O(n)

1. Which of the following statements are true?
2. We can create a binary tree from given inorder and preorder traversal sequences.
3. We can create a binary tree from given preorder and postorder traversal sequences.
4. For an almost sorted array, insertion sort can be more effective than Quicksort.
5. Suppose T(n) is the runtime of resolving a problem with n elements, T(n) = Θ(1) if n = 1; T(n) = 2T(n/2) + Θ(n) if > 1; so T(n) is Θ(n log n).

答案：A C D

解析：必须要有中序才能得到一颗二叉树的正确顺序

D 和Merge-sort类似

令O(n) --> f(n) 设n = 2^k

T(n) = T(2^k) = 2T(2^k-1) + f(2^k)

= 2(2(T(2^k-2) + f(2^k-1)) + f(2^k)

     = 2\*2T(2^k-2) + 2f(2^k-1) + f(2^k)

     = ....

     = 2^kT(1) + 2^(k-1)f(2) + 2^(k-2)f(2\*2) + ... 2^0f(2^k)

设g(1) = T(1) = O(1)     n = 1时

设g(1) = a  f(n) = bn  a、b为常数

T(n) = T(2^k)

     = 2^k\*a + 2^(k-1)\*2b + 2^(k-2)\*2^2b + ....2^0\*2^k\*b

     = 2^k\*a + kb\*2^k

     = an + kbn

     = an + bnlog2n

     = O(nlog2n)

1. 字符串”qiniu”根据顺序不同有多少种排列组合的方式？
2. 96
3. 72
4. 60
5. 24

答案：C

解析：先不看有没有重复的，共5个字母，有5×4×3×2×1 = 120种组合.

然后有两个重复的i，所以应该除以2，120 / 2 = 60,所以答案C

1. 由4个“1”和4个“0”组成的8位二进制补码，能表示的最小整数是：
2. -125
3. -32
4. -121
5. -3

答案：C

解析：最大和最小分别是01111000 10000111,那么10000111的原码是11111001为-121

1. 在下列表述中,()是错误的
2. 含有一个或多个空格字符的串称为空串
3. 对n(n>0)个顶点的网,求出权最小的n-1条边便可构成其最小生成树
4. 选择排序算法是不稳定的
5. 平衡二叉树的左右子树的结点数之差的绝对值不超过1

答案：A B D

解析：A选项：长度为0 的串为空串，即为“” 。由多个空格字符构成的字符串称为空格串

B选项：如果权值最小的n-1条边构成了环的话，就不能构成最小生成树

D选项：平衡二叉树的左右子树高度之差的绝对值不超过1.

C选项：选择排序包括直接选择排序和堆排序，都是不稳定的算法

1. 如果在一个排序算法的执行过程中，没有一对元素被比较过两次或以上，则称该排序算法为节俭排序算法，以下算法中是节俭排序算法的有\_\_\_\_\_\_\_\_。
2. 插入排序
3. 选择排序
4. 堆排序
5. 归并排序

答案：A　D

解析：A。插入排序的思想是对第i+1位置上的数，将其插入前i个有序数组中。插入以后形成新的有序数组，根据排序数组不会在比较的原则，该元素不可能再次比较了。

B。选择排序的思想是对当前第i个位置上的数，那么在后续数组中，选最小的与i对换。说明肯定比较过第二小和第三的数。那么在i+1位置上，上次第二小和第三小的数还需要比较一次选出最小的与i+1交换。那么至少比较了两次。

C。堆排序。堆排序分两步。初始建堆和堆重建。当最大元素与最末尾元素交换后。面临堆重建的问题。那么堆顶元素下层过程中，必然与第二小的元素比较一次。再一次堆重建，假设第二小元素被替换的时候，他们会在比较一次。

D。归并排序思路是对两个已经排好序的数组，同时向后移动。那么每个元素只会与其他数组中的元素比较一次。然后合并在一起。根据同组元素不会比较的原则的，以后两个元素不可能在比较到。

## 数据库错题

**1、查找薪水涨幅超过15次的员工号emp\_no以及其对应的涨幅次数t**  
CREATE TABLE `salaries` (  
`emp\_no` int(11) NOT NULL,  
`salary` int(11) NOT NULL,  
`from\_date` date NOT NULL,  
`to\_date` date NOT NULL,  
PRIMARY KEY (`emp\_no`,`from\_date`));

**输入描述:**

无

**输出描述:**

|  |  |
| --- | --- |
| emp\_no | t |
| 10001 | 17 |
| 10004 | 16 |
| 10009 | 18 |

**答案：**

select emp\_no, count(salary) as t from salaries group by emp\_no having t > 15;

**解析：**

此题应注意以下四点：

1、用COUNT()函数和GROUP BY语句可以统计同一emp\_no值的记录条数

2、根据题意，输出的涨幅次数为t，故用AS语句将COUNT(emp\_no)的值转换为t

3、由于WHERE后不可跟COUNT()函数，故用HAVING语句来限定t>15的条件

4、最后存在一个理解误区，涨幅超过15次，salaries中相应的记录数应该超过16（从第2条记录开始算作第1次涨幅），不过题目为了简单起见，将第1条记录当作第1次涨幅，所以令t>15即可

WHERE语句在GROUP BY语句之前；SQL会在分组之前计算WHERE语句。

HAVING语句在GROUP BY语句之后；SQL会在分组之后计算HAVING语句。