## 参加测试人员：齐廉浩 工号：180296

## 选择题（每题4分）

1. 在这段代码中:

class Test {

private int m;

public static void fun() {

// some code…

}

}

如何使成员变量m可以被方法fun 访问? ( C )

(A) Change private int m to protected int m

(B) Change private int m to public int m

(C) Change private int m to static int m

(D) Change private int m to int m

2. 在这段代码中：

String s = "story";

一下描述正确的是（ A ）

A：s += "books";

B：char c = s[1];

C：int len = s.length;

D：byte t = s[0];

3. 如果编译并执行B类的main方法，会有怎样的结果( D ) class A {

int i;

A(int i) { this.i = i \* 2; }

}

class B extends A {

public static void main(String[] args) {

B b = new B(2);

}

B(int x) {

System.out.println(x);

}

}

A. The instance variable i is set to 4

B. The instance variable i is set to 2

C. The instance variable i is set to 0

D. This code will not compile

4.以下代码编译运行结果是? （ A ）

public class MyThread extends Thread

{

public static void main(String argv[])

{

MyThread mt = new MyThread();

mt.run();

}

public void run()

{

for(int i = 0; i < 100; i++)

System.out.println(i);

}

}

A编译成功并输出数字0到99

B编译成功，但没有结果输出。

C编译时出错，因为MyThread是抽象类，不能被实例化.

D编译出错，start方法没有定义。

5. 根据以下代码，选择正确的说法 （ B ）

1. class A extends Integer

2. {

3. int x = 0;

4. }

A. 代码正确编译.

B. 代码不能编译因为类Integer为final类.

C. 代码不能编译因为类A没有构造函数

D. 代码编译但运行时抛出Arithmetic异常

6.关于“死锁”，下列说法中正确的是: （ B ）

A 死锁是操作系统中的问题，数据库系统中不存在。

B 只有出现并发操作时，才有可能出现死锁

C 在数据库操作中防止死锁的方法是禁止两个用户同时操作数据库

D 当两个用户竞争相同的资源时不会发生死锁

7.事务的所有操作操作在数据库中要么全部正确反映出来要么全部不反映，这是事务的哪个特性？（ A ）

A 原子性 B 一致性 C 隔离性 D 持久性

8.在“查询选了课程号是4的，并且成绩在80以上的学生的学号和姓名”的SQL语句，将使用的表是 （ D ）

A学生信息表S

B学生信息表S和课程信息表C

C 学生信息表S 课程信息表C 和 学生选课信息表sc

D学生信息表S和 学生选课信息表sc

9.在并发控制的技术中，最常用的是封锁方法，对于共享锁（S）和排它锁（X）来说，下面列出的相容关系中，哪一个是不正确的( A )

A X/X: true B S/S true C S/X:false D : X/S: false

## 简答题

1. 请说明方法重载和重写的区别，并说明他们与多态之间的联系。（6分）

答：

重写(OverRide)：指子类对父类的允许访问的方法的实现过程进行重新编写, 返回值和形参都不能改变。即就是方法“外貌”不变，重新编写其“本质”。

重载(OverLoad)：指在一个类里面，方法名字相同，而参数不同。返回类型可以相同也可以不同。

联系：方法重载是一个类的多态性表现,而方法重写是子类与父类的一种多态性表现。

2.Collection 和 Collections的区别，并列举JAVA集合（List,Map,Set）几个常用的方法。（6分）

答：

区别：

java.util.Collection 是一个集合接口。它提供对集合对象进行基本操作的通用接口方法。Collection接口在Java 类库中有很多具体的实现。Collection接口的意义是为各种具体的集合提供了最大化的统一操作方式。

java.util.Collections则是一个包装类。它包含有各种有关集合操作的静态多态方法。此类不能实例化，就像一个工具类，服务于Java的Collection框架。

常用方法：

List: add(Object obj) //加入元素，返回是否添加成功

get(int index) //获取指定获取集合大小位置的元素

size() //获取集合大小

Map: put(Object key,Object value) //加元素，返回与此key关联的原有value，不存在则返回null。

containsKey(Object key) //根据key从集合中判断key是否存在

get(Object key) //根据key返回key对应的值

Set： add(Object obj) //加入元素

iterator() //返回Set集合中对元素迭代的迭代器

contains(Object obj) //判断Set集合中是否包含指定元素

3.

String str 1= “aaa” ;

String str2= “aaa” ;

String str3 = new String(“aaa”);

String str4= new String(“aaa”);

请写出以下打印的结果是True还是False .

System.out.println(str1.equals(str2)); true

System.out.println(str3.equals(str4)); true

System.out.println(str1.equals(str3)); true

System.out.println(str1==str2); true

System.out.println(str3== str4); false

System.out.println(str1==str3); false

str3= str1;

str4= str2 ;

简要介绍下java中的垃圾回收机制，什么是垃圾？

代码执行到这里，有没有产生垃圾？有的话具体是哪几个对象？ （6分）

答：

“垃圾回收(Garbage Collection)”：是指Java虚拟机(JVM)垃圾回收器提供的一种用于在空闲时间不定时回收无任何对象引用的对象占据的内存空间的一种机制。这里说的是无任何引用的对象占据的内存空间而不是对象本身。换言之，垃圾回收只会负责释放那些对象占有的内存。对象是个抽象的词，包括引用和其占据的内存空间。当对象没有任何引用时其占据的内存空间随即被收回备用，此时对象也就被销毁。但不能说是回收对象，可以理解为一种文字游戏。

“垃圾”：即指无任何对象引用的对象占据的内存空间。

产生了垃圾。

具体对象：通过new关键字在堆中创建的两个字符串对象因为str3、str4两个引用分别指向了str1、str2,所以没有任何对象对它们存在引用关系，所以其之前所开辟出的内存空间会被视作为“垃圾”。

4.写一段jdbc的代码，显示出所有的雇员。（5分）

答：

代码如下：

Connection conn = null;

java.sql.PreparedStatement stat=null;

try {

//1、加载MySQL的JDBC的驱动

Class.forName("com.mysql.jdbc.Driver");

//2、取得连接的URL，能访问MySQL数据库的用户名和密码

String url="jdbc:mysql://localhost:3306/test?useUnicode=true&characterEncoding=utf8";

String username = "root"; //username

String password = "123456"; //userpassword

//3、创建与MySQL数据库的连接类的实例

conn=DriverManager.getConnection(url,username,password);

stat=conn.prepareStatement("select \* from emp");

//执行

ResultSet rs = stat.executeQuery();

//遍历结果集并打印雇员信息

while(rs.next()){

int id = rs.getInt("empno "); //id

String ename = rs.getString("ename"); //name

double salary = rs.getDouble("sal"); //salary

int mgr = rs.getInt("mgr"); //部门经理员工号

String job = rs.getString(("job"); //wok

int deptno = rs.getInt("deptno"); //部门号

}

} catch (Exception e) { //异常处理

e.printStackTrace();

out.print("There are some errors.");

}finally{

if(stat!=null){

try {

stat.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

5

有如下两张表:

雇员表emp和部门表dept表结构如下：

Create table emp(

empno int primary key , --员工号

ename varchar(20), --员工姓名

sal money , --工资

mgr int --部门经理员工号-

job varchar(50) --工作

deptno int --部门号

）；

Create table dept(

deptno int primary key , --部门号

deptname varchar(20) --部门名称

）；

写SQL显示员工人数超过3个的部门名称及部门人数。（5分）

答：

SELECT dept. deptname ,COUNT(\*)

FROM dept

GROUP BY dept. deptname

HAVING COUNT(\*) > 3;

6. 请说明oracle函数nvl, decode的作用, truncate和delete命令有何区别？（5分）

答：

nvl：NVL(a,b)就是判断a是否是NULL，如果不是返回a的值，如果是返回b的值，b可以是一个固定值或者也可以是重新嵌套的函数求出的值。

Decode：DECODE(value, if1, then1, if2,then2, if3,then3, . . . else )，Value 代表某个表的任何类型的任意列或一个通过计算所得的任何结果。当每个value值被测试，如果value的值为if1，Decode 函数的结果是then1；如果value等于if2，Decode函数结果是then2；等等。事实上，可以给出多个if/then 配对。如果value结果不等于给出的任何配对时，Decode 结果就返回else 。需要注意的是，这里的if、then及else 都可以是函数或计算表达式。

truncate和delete区别：

1. 在功能上，truncate是清空一个表的内容，它相当于delete from table\_name。
2. delete是dml操作，truncate是ddl操作；因此，用delete删除整个表的数据时，会产生大量的roolback，占用很多的rollback segments， 而truncate不会。
3. 在内存中，用delete删除数据，表空间中其被删除数据的表占用的空间还在，便于以后的使用，另外它是“假相”的删除，相当于windows中用delete删除数据是把数据放到回收站中，还可以恢复，当然如果这个时候重新启动系统（OS或者RDBMS），它也就不能恢复了！而用truncate清除数据，内存中表空间中其被删除数据的表占用的空间会被立即释放，相当windows中用shift+delete删除数据，不能够恢复！
4. truncate 调整high water mark 而delete不；truncate之后，TABLE的HWM退回到 INITIAL和NEXT的位置（默认）delete 则不可以。
5. truncate 只能对TABLE，delete 可以是table,view,synonym。
6. TRUNCATE TABLE 的对象必须是本模式下的，或者有drop any table的权限 而 DELETE 则是对象必须是本模式下的，或被授予 DELETE ON SCHEMA.TABLE 或DELETE ANY TABLE的权限。
7. 在外层中，truncate或者delete后，其占用的空间都将释放。
8. truncate和delete只删除数据，而drop则删除整个表（结构和数据）。

7.什么是WebServer,为什么我们要把servlet部署到WebServer上？（5分）

答：

WebServer：所谓webserver 就是 B/S结构的server。其一般指网站服务器，是指驻留于因特网上某种类型计算机的程序，可以向浏览器等Web客户端提供文档，也可以放置网站文件，让全世界浏览；可以放置数据文件，让全世界下载。

原因：

1. Servlet是J2EE的一套规范，也就是说其中的一部分的类抽象成了接口，实际过程中Servlet运行在web服务器中，具体的实现要有web服务器来实现，这个和JDBC的驱动一个道理。
2. Servlet是用来处理请求和响应的，Servlet为我们提供了request和response2个接口，web服务器就是实现了这2个接口用来处理请求和响应，我们要是自己想开发一套web服务器，其实最核心的也就是实现这2个接口。
3. Servlet程序必须通过Servlet容器来启动运行，并且储存目录有特殊要求，通需要存储在<WEB应用程序目录>\WEB-INF\classes\目录中。 Servlet程序必须在WEB应用程序的web.xml文件中进行注册和映射其访问路径，才可以被Servlet引擎加载和被外界访问。一个<servlet>元素用于注册一个Servlet，它包含有两个主要的子元素：<servlet-name>和<servlet-class>，分别用于设置Servlet的注册名称和Servlet的完整类名。 一个<servlet-mapping>元素用于映射一个已注册的Servlet的一个对外访问路径，它包含有两个子元素：<servlet-name>和<url-pattern>，分别用于指定Servlet的注册名称和Servlet的对外访问路径。

8.**Hibernate中get和load有什么不同之处?** （5分）

答：

区别：

从返回结果上对比：

load方式检索不到的话会抛出org.hibernate.ObjectNotFoundException异常

get方法检索不到的话会返回null

从检索执行机制上对比：

get方法和find方法都是直接从数据库中检索 而load方法的执行则比较复杂首先查找session的persistent Context中是否有缓存，如果有则直接返回 如果没有则判断是否是lazy，如果不是直接访问数据库检索，查到记录返回，查不到抛出异常 如果是lazy则需要建立代理对象，对象的initialized属性为false，target属性为null 在访问获得的代理对象的属性时,检索数据库，如果找到记录则把该记录的对象复制到代理对象的target上，并将initialized=true，如果找不到就抛出异常。

根本区别说明

如果你使用load方法，hibernate认为该id对应的对象（数据库记录）在数据库中是一定存在的，所以它可以放心的使用，它可以放心的使用代理来 延迟加载该对象。在用到对象中的其他属性数据时才查询数据库，但是万一数据库中不存在该记录，那没办法，只能抛异常。所说的load方法抛异常是指在使用 该对象的数据时，数据库中不存在该数据时抛异常，而不是在创建这个对象时(注意：这就是由于“延迟加载”在作怪)。由于session中的缓存对于hibernate来说是个相当廉价的资源，所以在load时会先查一下session缓存看看该id对应的对象是否存在，不存在则创建代理。所以如果你知道该id在数据库中一定有对应记录存在就可以使用load方法来实现延迟加载。

对于get方法，hibernate会确认一下该id对应的数据是否存在，首先在session缓存中查找，然后在二级缓存中查找，还没有就查数据库，数据库中没有就返回null。对于load和get方法返回类型：虽然好多书中都这么说：“get()永远只返回实体类”，但实际上这是不正确的，get方法如果在 session缓存中找到了该id对应的对象，如果刚好该对象前面是被代理过的，如被load方法使用过，或者被其他关联对象延迟加载过，那么返回的还是 原先的代理对象，而不是实体类对象，如果该代理对象还没有加载实体数据（就是id以外的其他属性数据），那么它会查询二级缓存或者数据库来加载数据，但是 返回的还是代理对象，只不过已经加载了实体数据。get方法首先查询session缓存，没有的话查询二级缓存，最后查询数据库；反而load方法创建时首先查询session缓存，没有就创建代理，实际使用数据时才查询二级缓存和数据库。

简单总结

总之对于get和load的根本区别，一句话，hibernate对于load方法认为该数据在数据库中一定存在，可以放心的使用代理来延迟加载，如果在使用过程中发现了问题，只能抛异常；而对于get方法，hibernate一定要获取到真实的数据，否则返回null。

9.什么是基于Java的Spring注解配置? Spring中常用的注解有哪些? （5分）

答：

基于Java的Spring注解配置：

Spring框架是控制反转 (IOC) 或依赖性注入 (DI) 模式的推动因素，而这种推动是通过基于容器的配置实现的。过去，Spring 允许开发人员使用基于 XML 的配置，通过利用应用程序上下文 XML 文件来管理 bean 依赖性。此文件处于应用程序的外部，包含 bean 及其与该应用程序的依赖项的定义。尽管使用 XML 配置较为简单和便捷，但仍有另外一种方法可定义bean 及其依赖项。这种方法也称为基于 Java 的配置。不同于 XML，基于 Java 的配置能够以编程方式管理bean。这可通过运用多种注释来实现。

Spring中常用的注解：

@Configuration 的注解类表示这个类可以使用 Spring IoC 容器作为 bean 定义的来源。

@Bean 注解告诉 Spring，一个带有 @Bean 的注解方法将返回一个对象，该对象应该被注册为在Spring 应用程序上下文中的 bean。

@Value：用于修饰一个Field，用于为该Field配置一个值，相当于配置一个变量。

@Import：修饰一个Java配置类，用于向当前Java配置类中导入其他Java配置类

@Scope：用于修饰一个方法，指定该方法对应的Bean的生命域

@Lazy：用于修饰一个方法，指定该方法对应的Bean是否需要延迟初始化

@DependsOn：用于修饰一个方法，指定在初始化该方法对应的Bean之前初始化指定的Bean

@ImportResource注：用于修饰Java配置类，用于导入指定的XML配置文件

10.HTTP method有哪些？（4分）

答：

有GET，POST， PATCH， PUT ，DELETE。

11.position的absolute与fixed共同点与不同点。 （4分）

答：

共同点：

1、改变行内元素的呈现方式，将display置为inline-block

2、使元素脱离普通文档流，不再占据文档物理空间

3、覆盖非定位文档元素

不同点：

1、abuselute与fixed的根元素不同，abuselute的根元素可以设置，fixed根元素是浏览器。

滚动网页，fixed与浏览器的距离是不变的。

12.如何使用jQuery设置一个属性值? （4分）

答：利用JQuery的attr() 方法设置或返回被选元素的属性值。

13. $(document).ready() 是个什么函数？ （4分）

答：

因为jquery是对DOM进行操作的，也就是说需要 DOM 结构加载完成之后，jquery才能工作，那么 $(document).ready() 或者 $(function(){}) 的作用就是等待DOM 结构加载完成，然后才会去执行里面具体的jquery操作。