JAVA面试题集2

★★★
1.数据库，比如100用户同时来访，要采取什么技术解决；(JDBC)
答：可采用连接池。
2.String，StringBuffer StringBuilder的区别。(Core Java)
答：String是一组不可变的unicode编码的字符序列；
StringBuffer的长度是可变的，如果你对字符串中的内容经常进行操作，特别是内容要修改时，那么使用StringBuffer，如果最后需要String，那么使用StringBuffer的toString()方法；线程安全；
StringBuilder是从 JDK 5
开始，为StringBuffer该类补充了一个单个线程使用的等价类；通常应该优先使用 StringBuilder
类，因为它支持所有相同的操作，但由于它不执行同步，所以速度更快。
3.写一个函数要求输入一个字符串和一个字符长度，对该字符串进行分隔；(Core Java)
答：函数代码如下：

```
public String[] split(String str, int chars){
int n = (str.length()+ chars - 1)/chars;
String ret[] = new String[n];
for(int i=0; i<n; i++){
if(i < n-1){
ret[i] = str.substring(i*chars , (i+1)*chars);
}else{
ret[i] = str.substring(i*chars);
}
}
return ret;
}
```

4.java和javasciprt的区别；(Core Java)
答：javascript与Java是两个公司开发的不同的两个产品。Java是SUN公司推出的新一代面向对象的程序设计语言，特别适合于Internet应用程序开发；而javascript是Netscape公司的产品，其目的是为了扩展Netscape Navigator功能,而开发的一种可以嵌入Web页面中的基于对象和事件驱动的解释性语言，
它的前身是Live Script；而Java的前身是Oak语言。下面对两种语言间的异同作如下比较：
1）基于对象和面向对象：
Java是一种真正的面向对象的语言，即使是开发简单的程序，必须设计对象。
javascript是种脚本语言，它可以用来制作与网络无关的，与用户交互作用的复杂软件。它是一种基于对象（Object Based）和事件驱动（Event
Driver）的编程语言。因而它本身提供了非常丰富的内部对象供设计人员使用。
2）解释和编译：
Java的源代码在执行之前，必须经过编译；
javascript是一种解释性编程语言，其源代码不需经过编译，由浏览器解释执行。
3）强类型变量和类型弱变量：
Java采用强类型变量检查，即所有变量在编译之前必须作声明；
javascript中变量声明，采用其弱类型。即变量在使用前不需作声明，而是解释器在运行时检查其数据类型。
4）代码格式不一样。
6.以下二条语句返回值为true的有：(Core Java)
答：A B

A:“beijing”==“beijing”;
B:“beijing”.equalsIgnoreCase(new String(“beijing”));

7.类Example A继承 Exception, 类 ExampleB继承Example A;
有如下代码片断:(Core Java)

```
try{
throw new ExampleB(“b”);
}catch(ExampleA  e){
System.out.println(“ExampleA”);
}catch(Exception  e){
System.out.printfln(“Exception”);
}
```

输出的内容应该是:A
A:ExampleA B:Exception C:b D:无
8.java多线程有几种实现方法,都是什么?同步有几种实现方法,都是什么;(Core Java)
答:多线程有两种实现方法,分别是继承Thread类与实现Runnable接口;
同步的实现方面有两种,分别是synchronized,wait与notify
14.定义类A和类B如下:(Core Java)

```
class  A {
int  a=1;
 double  d=2.0;
  void  show(){
     System.out.println("Class A: a="+a +"\td="+d);
  }
}
class B extends A{
  float  a=3.0f;
  String  d="Java program.";
  void  show(){
     super.show( );
     System.out.println("Class B:  a="+a +"\td="+d);
  }
}
```

(1) 若在应用程序的main方法中有以下语句:

```
A  a=new A();
a.show();
```

则输出的结果如何?
答:输出结果为:Class A: a=1d=2.0
(2) 若在应用程序的main方法中定义类B的对象b:

```
A  b=new B();
b.show();
```

则输出的结果如何?
答:输出结果为:Class A: a=1d=2.0
Class B: a=3.0d=Java program.
16.详细描述MVC;(Web)

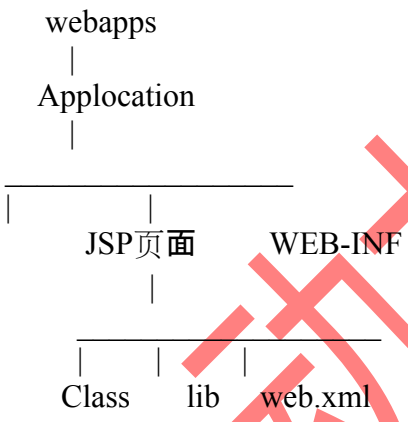
答:基于Java的Web应用系统采用MVC架构模式，即model(模型)、view(视图)、control(控制)分离设计。这是目前WEB应用服务系统的主流设计方向。
Model:即处理业务逻辑的模块，每一种处理一个模块;
View:负责页面显示，显示MODEL处理结果给用户，主要实现数据到页面转换过程;
Control:负责每个请求的分发，把form数据传递给MODEL处理，把处理结果的数据传递给VIEW显示。

17.页面中有一个命名为bank No的下拉列表，写脚本获取当前选项的索引值;(Web)
答:用java或javascript的处理方式分别如下:
Java: request.getParameter("bank No");
javascript: var selectItems = document.getElementsByName("bank No");
selectItems[0].value;

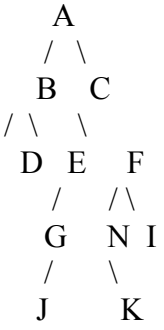
18.javasct常用的方面;(Web)
答:常用于数据输入校验和页面特殊效果等。

19.常用的web容器和开发工具;(Web)
答:最常用的容器包括:tomcat、weblogic;
开发工具有:eclipse,jbuilder

20.请画出Servlet 2.2以上Web Application的基本目录结构(时间2分钟)(Web)
答:目录结构如下图所示:



21.后序遍历下列二叉树，访问结点的顺序是?(数据结构)
答:DJGEBKNIFCA



22.写一种常见排序;(算法)
答:C++中冒泡排序:

```
void swap( int& a, int& b ){
int c=a; a = b; b = c;
}
```

```
void bubble( int* p, int len ){
bool bSwapped;
do {
bSwapped = false;
for( int i=1; i<len; i++ ){
if( p[i-1]>p[i] ){
swap( p[i-1], p[i] );
bSwapped = true;
}
}
}while( bSwapped );
}
```

23.一个byte几个单位。(计算机基础)

答:8bit。

24.常用UNIX命令(Linux的常用命令)(至少10个);(Unix)

答:ls , pwd , mkdir, rm, cp, mv ,cd , ps , ftp, telnet , ping, env, more, echo greap, cat, view, pr , in, chmod, find, chown.

25.写一个一小段程序检查数字是否为质数; (C++)

以上的程序你采用的哪种语言写的.采用该种语言的理由是什么,

答:代码如下:

```
#include <math.h>
bool prime( int n ){
if(n<=0) exit(0);
    for( int i=2; i<=n; i++ )
       for( int j=2; j<=sqrt(i); j++)
          if((n%j==0) && (j!=n))
             return false;
  return true;
}
```

C++, 运行效率高

★★★★

1.张表,学生表S,课程C,学生课程表SC,学生可以选修多门课程,一门课程可以被多个学生选修,通过SC表关联;(SQL)

1)写出建表语句;

答:建表语句如下(mysql数据库):

```
create table s(id integer primary key, name varchar(20));
create table c(id integer primary key, name varchar(20));
create table sc(
sid integer references s(id),
cid integer references c(id),
primary key(sid,cid)
);
```

2)写出SQL语句,查询选修了所有选修课程的学生;

答:SQL语句如下:

```
select stu.id, stu.name from s stu
where (select count(*) from sc where sid=stu.id) =
```

(select count(*) from c);
3)写出SQL语句,查询选修了至少5门以上的课程的学生。
答:SQL语句如下:
select stu.id, stu.name from s stu
where (select count(*) from sc where sid=stu.id)>=5;
2.数据库表(Test)结构如下:(SQL)
IDNAMEAGEMANAGER(所属主管人ID)
106A30104
109B19104
104C20111
107D35109
112E25120
119F45NULL
要求:列出所有年龄比所属主管年龄大的人的ID和名字?
答:SQL语句如下:
select employee.name from test employee where employee.age>
(select manager.age from test manager where manager.id=employee.manager);
3.有3个表(15分钟):(SQL)
Student 学生表 (学号, 姓名, 性别, 年龄, 组织部门)
Course 课程表 (编号, 课程名称)
Sc 选课表 (学号, 课程编号, 成绩)
表结构如下:
1)写一个SQL语句, 查询选修了'计算机原理'的学生学号和姓名(3分钟)
答:SQL语句如下:
select stu.sno, stu.sname from Student stu
where (select count(*) from sc where sno=stu.sno and cno =
(select cno from Course where cname='计算机原理')) != 0;
2)写一个SQL语句, 查询'周星驰'同学选修了的课程名字(3分钟)
答:SQL语句如下:
select cname from Course where cno in (select cno from sc where sno=(select sno from Student where sname='周星驰'));
3)写一个SQL语句, 查询选修了5门课程的学生学号和姓名(9分钟)
答:SQL语句如下:
select stu.sno, stu.sname from student stu
where (select count(*) from sc where sno=stu.sno) = 5;
5. 写一个方法,输入一个文件名和一个字符串,统计这个字符串在这个文件中出现的次数。(Core Java)
答:代码如下:
public int countWords(String file, String find) throws Exception {
int count = 0;
Reader in = new FileReader(file);
int c;
while ((c = in.read()) != -1) {
while (c == find.charAt(0)) {
for (int i = 1; i < find.length(); i++) {
c = in.read();
if (c != find.charAt(i)) break;
if (i == find.length() - 1) count++;

```
}
}
}
return count;
```

8.用程序给出随便大小的10个数，序号为1-10，按从小到大顺序输出，并输出相应的序号。(Core Java)

答：代码如下：

```
package test;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Iterator;
import java.util.List;
import java.util.Random;
public class RandomSort {
public static void printRandomBySort() {
Random random = new Random(); // 创建随机数生成器
List list = new ArrayList();
for (int i = 0; i < 10; i++) { // 生成10个随机数，并放在集合list中
list.add(random.nextInt(1000));
}
Collections.sort(list); // 对集合中的元素进行排序
Iterator it = list.iterator();
int count = 0;
while (it.hasNext()) { // 顺序输出排序后集合中的元素
System.out.println(++count + ":" + it.next());
}
}
public static void main(String[] args) {
printRandomBySort();
}
}
```

9.写一个函数，2个参数，1个字符串，1个字节数，返回截取的字符串，要求字符串中的中文不能出现乱码：如（“我ABC”，4）应该截为“我AB”，输入（“我ABC汉DEF”，6）应该输出为“我ABC”而不是“我ABC+汉的半个”。(Core Java)

答：代码如下：

```
public String subString(String str, int subBytes) {
int bytes = 0; // 用来存储字符串的总字节数
for (int i = 0; i < str.length(); i++) {
if (bytes == subBytes) {
return str.substring(0, i);
}
char c = str.charAt(i);
if (c < 256) {
bytes += 1; // 英文字符的字节数看作1
} else {
bytes += 2; // 中文字符的字节数看作2
if(bytes - subBytes == 1){
return str.substring(0, i);
}
```

```
}
}
return str;
}
```

14.struts的入口类;(Web)

答:ActionServlet是Struts的入口类,所有的struts请求都经由该类转发处理

16.servlet的生命周期以及与其相关的类接口和相关方法;(Web)

答:servlet有良好的生存期的定义,包括加载和实例化、初始化、处理请求以及服务结束。这个生存期由javax.servlet.Servlet接口的init,service和destroy方法表达。

17.关于hibernate:

1)在hibernate中,在配置文件呈标题一对多,多对多的标签是什么;

2)Hibernate的二级缓存是什么;

3)Hibernate是如何处理事务的;

答:1)一对多的标签为:<one-to-many> ;多对多的标签为:<many-to-many>;

2)sessionFactory的缓存为hibernate的二级缓存;

3)Hibernate的事务实际上是底层的JDBC Transaction的封装或者是JTA Transaction的封装;默认情况下使用JDBCTransaction。

18.JSP中两种 include方式的区别;(Web)

答:动态include用jsp:include动作实现,它总是会检查所含文件中的变化,适合用于包含动态页面,并且可以带参数;静态include用<%@include %>指令实现,适用于包含静态页面

19.javascipt的优缺点和内置对象;(Web)

答:关于javascipt:

1)优点:简单易用,与Java有类似的语法,可以使用任何文本编辑工具编写,只需要浏览器就可执行程序,并且事先不用编译,逐行执行,无需进行严格的变量声明,而且内置大量现成对象,编写少量程序可以完成目标;

2)缺点:不适合开发大型应用程序

3)javascript有11种内置对象:

Array,String,Date,Math,Boolean,Number,Function,Global,Error,RegExp,Object

20.get和post的区别?(Web)

答:form中的get和post方法,在数据传输过程中分别对应了HTTP协议中的GET和POST方法。二者主要区别如下:

1)Get是用来从服务器上获得数据,而Post是用来向服务器上传递数据;

2)Get将表单中数据的按照variable=value的形式,添加到action所指向的URL后面,并且两者使用"?"连接,而各个变量之间使用"&"连接;Post是将表单中的数据放在form的数据体中,按照变量和值相对应的方式,传递到action所指向URL;

3)Get是不安全的,因为在传输过程,数据被放在请求的URL中,Post的所有操作对用户来说都是不可见的;

4)Get传输的数据量小,这主要是因为受URL长度限制,而Post可以传输大量的数据,所以在上传文件只能使用Post;

5)Get限制form表单的数据集必须为ASCII字符,而Post支持整个ISO10646字符集;

6)Get是form的默认方法。

22.写出你熟悉的开源框架以及各自的作用。

答:框价:hibernate,spring,struts.

Hibernate主要用于数据持久化;
Spring 的控制反转能起到解耦合的作用;
Struts 主要用于流程控制;
服务器类:Apache Tomcat Jboss等
25.JSP标签的作用?如何定义;(Web)
答:作用:分离jsp页面的内容和逻辑;
业务逻辑开发者可以创建自定义标签;
封装业务逻辑;
可重用并且易维护;
易于手工修改、易于工具维护;
提供简洁的语法;
定义:
写标签处理器;
写tld文件;
讲标签处理器和tld文件放到同一个包里面;
把jsp页面和标签库配置部署在一起。
26.写一个自定义标签;(Web)
答:代码如下:

```
import javax.servlet.jsp.tagext.*;
import javax.servlet.jsp.*;
import java.io.*;
public class TimeTag extends SimpleTagSupport{
private boolean isServer = true;
public void setServer(boolean isServer){
this.isServer = isServer;
}
public void doTag() throws JspException, IOException{
JspWriter out = getJspContext().getOut();
if(isServer) {
out.println(new java.util.Date());
}else{
out.println("<script language=\"javascript\">");
out.println("document.write(new Date());");
out.println("</script>");
}
}
}
```

27.写出熟悉的JSTL标签;(Web)
答:如下:

```
<c:if>
<c:forEach>
<c:set>
<c:import>
<c:redirect>
```

28.说出struts中的标签;(Web)
答:如下:

```
<bean:message  />          <html:errors />
```

<bean:include /> <html:messages />
<bean:define /> <html:html>
<bean:write /> <html:form>
<bean:resource /> <html:link>
<bean:cokkie /> <html:text>
<bean:heder /> <logic:present />
<bean:parameter /> <logic:equal />
<bean:size />

29.如何从form表单中得取checkbox的值;(Web)

答:可在页面把checkbox的name属性取同一个，value属性取每个条目的id,后台用getParamter(“name”)能取到checkbox的一组值。

30.简述HttpSession的作用、使用方法，可以用代码说明。(时间3分钟)(Web)

答:HttpSession中可以跟踪并储存用户信息，把值设置到属性中，有2个方法:setAttribute(),getAttrribute();

例如:在一个方法中用session.setAttribute(“student”,student);在session中设置一个属性名为student,值为一个名为student的对象。而后可在同一session范围内用getAttribute(“student”)取出该属性，得到student对象。

31.找出程序中的问题，并写出理由(5分钟)(Web)

```
import javax.servlet.*;
import javax.servlet.http.*;
import javax.sql.*;
import javax.naming.*;
import java.sql.*;
import java.io.IOException;
public class TestServlet extends HttpServlet{
private Connection conn;
protected void doGet(HttpServletRequest req,HttpServletResponse res)
throws IOException,ServletException{
try{
Class.forName(“COM.ibm.db2.jdbc.app.DB2Driver”);
conn = DriverManager.getConnection(“jdbc:db2:mydb”,”db2”,”db2”);
PreparedStatement stmt = conn.prepareStatement("delete from mytb where id=?");
for (int i=0;i<5;i++) {
stmt.setInt(1,i);
stmt.executeUpdate();
}
conn.close();
}
catch (Exception ex)
{
ex.printStackTrace();
}
}
}
```

答:其中doGet方法的访问限制修饰符应该为public，因为protected修饰的方法只能被同包中的类或其子类才可访问，这将导致web容器无法调用该方法。

34.UML分析设计中常用的几种图;(UML)

答:用例图，静态图(包括类图、对象图、和包图)、行为图、交互图(顺序图、合作图)，实现图。

35.你对软件开发中迭代的含义的理解;(UML)

答:软件开发中，各个开发阶段不是顺序执行的，应该是并行执行,也就是跌代的意思。这样对于开发中的需求变化，及人员变动都能得到更好的适应。

36.统计一篇文章中单词个数(c++);

答:代码如下:

```
#include<iostream>
#include<fstream>
using namespace std;
int main(){
ifstream fin("t.txt");
if(!fin){
cout<<"can't open file"<<endl;
return -1;
}
int count = 0;
char buf[256];
memset(buf, 0, 256);
while(1){
fin2>>buf;
if(fin2.eof())
break;
count++;
}
cout<<"The number of the words is : "<<count<<endl;
fin2.close();
return 0;
}
```

1.线程中为什么不推荐使用stop;(Core Java)

答:此方法已过时。该方法具有固有的不安全性。用 Thread.stop 来终止线程将释放它已经锁定的所有监视器(作为沿堆栈向上传播的未检查 ThreadDeath 异常的一个自然后果)。如果以前受这些监视器保护的任何对象都处于一种不一致的状态，则损坏的对象将对其他线程可见，这有可能导致任意的行为。stop 的许多使用都应由只修改某些变量以指示目标线程应该停止运行的代码来取代。目标线程应定期检查该变量，并且如果该变量指示它要停止运行，则从其运行方法依次返回。如果目标线程等待很长时间(例如基于一个条件变量)，则应使用 interrupt 方法来中断该等待。

2.编程题:设有n个人依围成一圈，从第1个人开始报数，数到第m个人出列，然后从出列的下一个人开始报数，数到第m个人又出列，…，如此反复到所有的人全部出列为止。设n个人的编号分别为1，2，…，n，打印出出列的顺序;要求用java实现。(Core Java)

答:代码如下:

```
package test;
public class CountGame {
private static boolean same(int[] p,int l,int n){
for(int i=0;i<l;i++){
if(p[i]==n){
return true;
}
```

```
}
return false;
}
public static void play(int playerNum, int step){
int[] p=new int[playerNum];
int counter = 1;
while(true){
if(counter > playerNum*step){
break;
}
for(int i=1;i<playerNum+1;i++){
while(true){
if(same(p,playerNum,i)==false) break;
else i=i+1;
}
if(i > playerNum)break;
if(counter%step==0){
System.out.print(i + " ");
p[counter/step-1]=i;
}
counter+=1;
}
}
System.out.println();
}
public static void main(String[] args) {
play(10, 7);
}
}
```

3.抽象方法可以是native？可以是同步的吗？(Core Java)

答:都不能。

4.JDBC，Hibernate分页怎样实现？(JDBC)

答:方法分别为:

1) Hibernate的分页:

```
Query query = session.createQuery("from Student");
query.setFirstResult(firstResult);//设置每页开始的记录号
query.setMaxResults(resultNumber);//设置每页显示的记录数
Collection students = query.list();
```

2) JDBC的分页:根据不同的数据库采用不同的sql分页语句

例如: Oracle中的sql语句为: "SELECT * FROM (SELECT a.*, rownum r FROM TB_STUDENT) WHERE r between 2 and 10" 查询从记录号2到记录号10之间的所有记录

6.请写出spring中I0C的三种实现机制。(Spring)

答:三种机制为:通过setter方法注入、通过构造方法注入和接口注入。

7.EJB的事务是如何实现的？何时进行回滚;(EJB)

答:是通过使用容器或Bean自身管理事务的;当产生一个系统异常时容器就自动回滚事务。

8.写一个方法1000的阶乘。(C++)

答:C++的代码实现如下:

```
#include <iostream>
#include <iomanip>
#include <vector>
using namespace std;
class longint {
private:
vector<int> iv;
public:
longint(void) { iv.push_back(1); }
longint& multiply(const int &);
friend ostream& operator<<(ostream &, const longint &);
};
ostream& operator<<(ostream &os, const longint &v) {
vector<int>::const_reverse_iterator iv_iter = v.iv.rbegin();
os << *iv_iter++;
for ( ; iv_iter < v.iv.rend(); ++iv_iter) {
os << setfill('0') << setw(4) << *iv_iter;
}
return os;
}
longint& longint::multiply(const int &rv) {
vector<int>::iterator iv_iter = iv.begin();
int overflow = 0, product = 0;
for ( ; iv_iter < iv.end(); ++iv_iter) {
product = (*iv_iter) * rv;
product += overflow;
overflow = 0;
if (product > 10000) {
overflow = product / 10000;
product -= overflow * 10000;
}
*iv_iter = product;
}
if (0 != overflow) {
iv.push_back(overflow);
}
return *this;
}
int main(int argc, char **argv) {
longint result;
int l = 0;
if(argc==1){
cout << "like: multiply 1000" << endl;
exit(0);
}
sscanf(argv[1], "%d", &l);
for (int i = 2; i <= l; ++i) {
result.multiply(i);
}
cout << result << endl;
```

```
return 0;
}
```

9.怎么处理XML的中文问题;(XML)

答:示例代码如下:

```
package xml;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
public class DOMTest {
private String outFile = "c:\\test.xml";
public static void main(String args[]) {
DOMTest test = new DOMTest();
try {
DocumentBuilder builder = DocumentBuilderFactory.newInstance()
.newDocumentBuilder();
Document doc = builder.newDocument();
Element root = doc.createElement("Tarena");
Element zhang = doc.createElement("张");
zhang.appendChild(doc.createTextNode("我是张丽芳"));
root.appendChild(zhang);
doc.appendChild(root);
Transformer transformer = TransformerFactory.newInstance()
.newTransformer();
//设置xml的编码
transformer.setOutputProperty(OutputKeys.ENCODING, "gb2312");
//设置缩近格式
transformer.setOutputProperty(OutputKeys.INDENT, "yes");
transformer.transform(new DOMSource(doc),
new StreamResult(test.outFile));
} catch (Exception e) {
System.out.println(e.getMessage());
}
}
}
```

在main(String[] args)方法内是否可以调用一个非静态方法?

答案:不能

1□ 同一个文件里是否可以有两个public类?

答案:不能

2□ 方法名是否可以与构造器的名字相同? 答案:可以。

```
public class Test
{
    public Test(String iceboy)
```

```
        {
                System.out.println(iceboy);
        }
    public void Test(String iceboy)
    {
                System.out.println(iceboy);
    }
    public static void main(String[] args)
    {
                Test a = new Test("abc");//输出“abc”
                a.Test("iceboy");//输出“iceboy”
    }
}
```

4. 初始化了一个没有run()方法的线程类,是否会出错？

答案:不会。

第一种方法:直接继承Thread类。

```
public class Test
{public static void main(String[] args)
    {
        ThreadClass t = new ThreadClass();
        t.start();
        System.out.println("end");//输出“end”
    }
}
class ThreadClass extends Thread  //Thread类已经实现了空的run()方法。
{
}
```

第二种方法:实现Runnable接口

```
public class Test
{public static void main(String[] args)
    {ThreadClass t = new ThreadClass();
        Thread thread = new Thread(t);
        thread.start();
        System.out.println("end");
    }
}
class ThreadClass implements Runnable
{  ublic void run()  //必须有此方法否则编译报错。它是Runnable接口中的抽象方法。
    {System.out.println("Threads");
    }
}
```

3□ 局部内部类是否可以访问非final变量？

答案:不能访问局部的，可以访问成员变量(全局的)。

```
class Out
{private String name = "out.name";
    void print()
    {       final String work = "out.local.work";//若不是final的则不能被Animal 使用.
            int age=10;
```

```
            class Animal
            //定义一个局部内部类.只能在print()方法中使用.
        //局部类中不能使用外部的非final的局部变量.全局的可以.
            {
                public void eat()
                {
                    System.out.println(work);//ok
                    //age=20;error not final
                    System.out.println(name);//ok.
                }
            }
            Animal local = new Animal();
            local.eat();
        }
    }
```

4□ 选择语句case中，允许使用的值有哪些？ 答案：int,short,char,byte(都在int范围之内，且是整数)

5□ Math,String是不可继承的。(final类)

Instanceof 后面跟的应该是OBJECT。

构造器可以是私有的。(private)

=与==意义是完全不同的。一个是赋值，一个是等于。

全局变量可以不进行初始化，如果使用一个局部变量，则这个局部变量要被初始化。

6□ 在try-catch-final块中的退出语句。

```
public class Test
{   public static void main(String[] args)
    {
        int a=1;
        try
        {a=a/0;
        }catch(Exception e)
        {       System.out.println("catch");
                return;//当return时，finally中的语句会执行。
                //System.exit(0);//若用上这句，finally中的语句不会执行。直接返回，退出程序。
        }
        finally //当没有System.exit(0);时，无论是否发生异常它都会执行。
        {System.out.println("finally");
        }
    }
}
```

注： try-catch-final块的顺序不能调换。

7□ 下面都是正确的main方法签名。

```
public static void main(String[] args)
public static final void main(String[] args)
static public void main(String[] args)
static public synchronized void main(String[] args)
static public abstract void main(String[] args)//错误
```

8□ if(-0.0 == 0.0)是相等还是不等？

答案：相等。

10. 一个抽象类是否可以没有抽象方法？
答案：可以。
11. RandomAccessFile 类继承Object，并且实现了DataInput和DataOutput接口。
答案：正确
13. class Child extends Parents{}
class Parents{}
是否可以这样声明类，在一个文件中？
答案：可以。无所谓的。
14. 数组，无论是当前的，还是类等级的，都会被初始化。
String 是被初始化为 null,不是空字符。
null,“”,” “,都是不同的。
“continue”只能在一个循环里(如for,do,while)，它不能在case语句中出现。
Primitive(int,char,long等)数组是不能互相赋值的，即使它们本身可以。
一个Constructor可以抛出任何异常。
初始化块是按照声明的顺序执行的。
所有关于 NaN(Not a Number) 和 non-NaN 的比较，都返回false. 这条很重要。
==会产生编译错误，如果两边类型不匹配的话。
equals() 返回 false 如果对象类型不同，但不产生 编译错误。
15. Java成员变量默认初始化的值。
成员变量类型　　取值
byte-0　short-0　int-0　long-0L
char-'\u0000'　float-0.0F　double-0.0D　boolean-false
所有引用类型　　null
16. integer和long 操作 /和% 的话，可能会抛出ArithmeticException，比如除0。但是 float与double不会，即使是除以0。
double a=0;a=a/0;则a等于NaN。
17. 普通内部类不可以拥有静态变量，但静态内部类可以。
File类没有 任何 处理文件内容的方法。
InputStream 和 OutputStream 是 抽象类，DataInput 和 DataOutput是接口。DataInputStream实现了DataInput接口。
25. 数组转换问题。

```
Object[] object = new Person[2];
Person [] person = new Person [3];
person = (Person [])object;//可以转换
int[] i = new int[2];
long[] l = new int[3];
i = (long[])l;//不可以转换
```

26. 用socket通讯写出客户端和服务器端的通讯，要求客户发送数据后能够回显相同的数据。
Server.java:源代码

```
import java.net.*;
import java.io.*;
class  Server
{   public Server()
        {BufferedReader br = null;
            PrintWriter pw = null;
```

```
                try
                {       ServerSocket server = new ServerSocket(8888);//建立服务器端
                        Socket socket = server.accept();//监听客户端
                        //得到该连接的输入流
                        br = new BufferedReader(new InputStreamReader(socket.getInputStream()));
                        //得到该连接的输出流
                        pw = new PrintWriter(socket.getOutputStream(),true);
                        //先读后写
                        String data = br.readLine();
                        System.out.println(data);//输出到控制台
                        pw.println(data);//转发给客户端
                }catch(Exception e)
                {e.printStackTrace();
                }
                finally
                {try{//关闭读写流
                                br.close();
                                pw.close();
                        }catch(Exception e)
                        {}
                }
        }
        public static void main(String[] args)
        {Server server = new Server();
        }}
```

Client.java:源代码

```
import java.net.*;
import java.io.*;
class  Client
{public Client()
    {BufferedReader br = null;
        PrintWriter pw = null;
        try
        {  Socket socket = new Socket("localhost",8888);//与服务器建立连接，服务器要先启
         //得到Socket的输入与输出流
         br = new BufferedReader(new InputStreamReader(socket.getInputStream()));
         pw = new PrintWriter(socket.getOutputStream(),true);
         //先写后读
         pw.println("Client:你好！");
         String data = null;
         while(true)
         {      data = br.readLine();
                if(data!=null) break;
         }
         System.out.println(data);
        }catch(Exception e)
        {
                e.printStackTrace();
```

```
                }
                finally
                {       try
                        {
                                br.close();
                                pw.close();
                        }catch(Exception e)
                        {}
                }
        }
        public static void main(String[] args)
        {       Client c = new Client();
        }
    }
```

40. 数组是不是基本类型？

答案：不是。

43.”==”与equal有何区别？

答案：“==”比较的是内存地址，equal比较的是内容本身。

53. 文件读写的基本类有哪些？

答案：FileInputStream,FileOutputStread,File,IOException等。

题目1：用1、2、2、3、4、5这六个数字，用java写一个main函数，打印出所有不同的排列，如：512234、412345等，要求："4"不能在第三位，"3"与"5"不能相连.

解决思路：强化题目，用1、2、2、3、4、5这六个数字排列“递增”序列。其他要求不变。

算法思路：显然是递归，初始序列122345，先从末两位(45)变化(45,54),然后末三位(345) ...

直到最后六位.怎样解决重复问题？很简单，由于是递增序列，每生成新序列可与前一生成序列比较，如<放弃当前序列。当然有更好效率，如预先预测。代码如下：

```
class test
{
 // 当前固定部分
 private String CurFixPart;
 private String PreGenNum;

public static void main(String[] args)
{
 test t=new test();
 t.GenControll("122345");
}
// 调整字符串s位置pos字符到最前
private String shift(String s, int pos)
{
String newStr;
if (s.length()>pos+1)
  newStr=s.substring(pos, pos+1)
       +s.substring(0, pos)
       +s.substring(pos+1);
else
  newStr=s.substring(pos)
       +s.substring(0, pos);
```

```
return newStr;
}
protected int Validate(String newNum)
{
  String newGenNum=CurFixPart+newNum;
  if (Integer.valueOf(newGenNum)<=Integer.valueOf(PreGenNum))
    return 0;
  if (newGenNum.substring(2,3).equals("4") ||
       (newGenNum.indexOf("35")!=-1) || (newGenNum.indexOf("53")!=-1))
    return 0;
  PreGenNum=newGenNum;
System.out.println(newGenNum);
return 0;
}
public void GenControll(String Base)
{
  PreGenNum="0";
CurFixPart="";
GenNext(Base, 0);
}
void GenNext(String varPart, int curPos)
{
if (varPart.length()==2)
{
  Validate(varPart);
  Validate(shift(varPart, 1));
  return;
 }
// Next Layer
String newGen=shift(varPart, curPos);
String SavedFixPart=CurFixPart;
CurFixPart=CurFixPart+newGen.substring(0,1);
GenNext(newGen.substring(1), 0);
CurFixPart=SavedFixPart;
// 同层递增
if (curPos==varPart.length()-1)
  return;
GenNext(varPart, curPos+1);
}
}
```

1.编写一个四舍五入的函数，要求可以保留到小数后面的任意一位（如3.1415927保留到小数点后第3位后得到3.142）

2.编写一个函数（不需要写出完整的类定义），计算两个日期（java.util.date）对象间相隔的天数，代码量不的超过10行。

```
1.    public String numberDeal(double num,int n){
          NumberFormat nf=NumberFormat.getInstance();
          nf.setMaximumFractionDigits(n);
          nf.setMinimumFractionDigits(n);
          return nf.format(num);
```

```
    }
2.   public long dateDivide(Date begin,Date end){
  long begintime=begin.getTime();
  long endtime=end.getTime();
  long value=endtime-begintime;
  if(value%(1000*60*60*24)==0)
     return value/(1000*60*60*24);
  else
     return (value/(1000*60*60*24))+1;
     }
```