JSD1612班第一次月考试题解析

**1.**

(单选)class Card{}下列不属于Card类构造方法的是：（）

* +  A.

Card(){}

* +  B.

public Card(){}

* +  C.

public void Card(){}

* +  D.

private Card(){}

正确答案：C

**解析：构造方法的组成部分：访问控制修饰符、类名、参数列表、方法体**

**若一个类中自己没有写构造函数，编译器会自动添加一个默认的构造器，其访问控制修饰符与该类的访问控制修饰符相同。**

**2.**

(单选)下列关于HashMap的方法描述正确的是:

* +  A.

containsKey(Object key)： 判断集合中是否包含指定的Value

* +  B.

containsValue (Object value)： 判断集合中是否包含指定的Key

* +  C.

get(Object key):返回与参数Key所对应的Value对象，如果不存在则返回null

* +  D.

put(K key, V value)：将Key-Value对存入Map，如果在集合中已经包含该Key， 则操作将替换该Key所对应的Value，返回值为该Key当前所对应的Value（如果没有则返回null）

**正确答案：C**

**解析：**

**A选项：**[**containsKey**](mk:@MSITStore:C:\Users\Administrator\Desktop\JDK_API_1_6_zh_CN.CHM::/java/util/HashMap.html#containsKey(java.lang.Object))**(**[**Object**](mk:@MSITStore:C:\Users\Administrator\Desktop\JDK_API_1_6_zh_CN.CHM::/java/lang/Object.html) **key)   
如果此映射包含对于指定键的映射关系，则返回 true。即是否包含该key**

**B选项：**[**containsValue**](mk:@MSITStore:C:\Users\Administrator\Desktop\JDK_API_1_6_zh_CN.CHM::/java/util/HashMap.html#containsValue(java.lang.Object))**(**[**Object**](mk:@MSITStore:C:\Users\Administrator\Desktop\JDK_API_1_6_zh_CN.CHM::/java/lang/Object.html) **value)   
如果此映射将一个或多个键映射到指定值，则返回 true。即是否包含该value**

**D选项：返回值是被替换的key值所对应的value。**

**3.**

(单选)下面关于interface，叙述错误的是：（）

* +  A.

一个interface可以继承多个interface

* +  B.

接口中的方法可以由private修饰

* +  C.

interface中可以定义static final 常量

* +  D.

interface中可以无任何方法定义

正确答案：B

**解析：一个接口可以继承多个接口（interface）用逗号隔开。**

**接口中的所有方法的都是由public abstract修饰的，接口中只能定义常量，不能定义变量。**

**一个类如果想继承父类的同时又实现一个接口。必须先继承extends父类再实现implements接口。**

**4.**

(单选)Java语言可以跨平台的原因是：

* +  A.

Java面向对象

* +  B.

Java虚拟机

* +  C.

Java垃圾回收机制

* +  D.

Java编译器

正确答案：B

**解析：sun公司根据不同的系统开发出不同的JVM，因为java文件是由JVM编译运行的，所有，JVM是java语言开发的程序可以跨平台的保障。**

**5.**

(单选)Java程序的执行过程中用到一套JDK工具，其中javac.exe是指（）。

* +  A.

Java语言编译器

* +  B.

Java字节码解释器

* +  C.

Java文档生成器

* +  D.

Java类分解器

正确答案：A

**解析：**

**Java编译器：将Java源文件（.java文件）编译成字节码文件（.class文件，是特殊的二进制文件，二进制字节码文件），这种字节码就是JVM的“机器语言”。javac.exe是Java编译器。**

**java解释器：是JVM的一部分。Java解释器用来解释执行Java编译器编译后的程序。java.exe是Java解释器。**

**Java文档生成器:javadoc.exe**

**javap.exe是Java类的分解器，即对．class文件提供字节代码的反汇编，并打印。默认时，打印类的公共域、方法、构造方法、静态初值。注意：javac．exe，Java的编译器，能将源文件编译成字节码文件；jdb．exe，Java调试器．如编译器返回程序代码错误，可用它对程序调试，它是解释器的拷贝；jdbdoc．exe，Java文档生成器，对Java源文件和包以HTML格式产生文档。**

**6.**

(单选)下列关于内存回收的说明正确的是：

* +  A.

程序员必须创建一个线程来释放内存

* +  B.

内存回收程序负责释放无用内存

* +  C.

内存回收程序允许程序员直接释放内存

* +  D.

内存回收程序可以在指定的时间释放内存对象

正确答案：B

**解析：**

**Java垃圾回收机制简称：gc，java中gc是jvm一个清理垃圾的线程，该线程不是由程序员自己来创建的，而是由jvm自己来维护。**

**Java中，gc不受程序员直接控制。**

**gc清理内存垃圾的是不定时的。**

**7.**

(单选)运行下列代码，输出为false的是：（）。

* +  A.

String st1 = "abc"; System.out.println("abc" == st1);

* +  B.

String st2 = "abc"; System.out.println(st2.equals(new String("abc")));

* +  C.

Integer i = 100; System.out.println(100 == i);

* +  D.

ArrayList list = new ArrayList(); System.out.println(list.contains(null));

正确答案：D

**解析：**

**A选项：此时String st2=“adc”;此时当程序执行完这一行代码，会在堆中创建出一个String对象”abc”,并把该对象的内存地址赋值给String类型变量st2，**

**而”==”符号，比较的就是内存地址。所以为true。**

**B选项：同A选项大体相同，不过String类已经重写了从Object类中继承过来的equals()方法，所以其比较的String的内容，所以也为true。**

**C选项：此选项考察的是自动拆装箱：即**

**Integer i =Integer.valueOf(100);**

**System.out.println(100==i.intValue());**

**此时，i.intValue();的返回值就是一个int类型的值100.**

**D选项：ArrayList list = new ArrayList();此行代码执行完毕之后，会创建出一个ArrayList对象list，不过里面没有任何元素（没有元素，并不是list是null）。**

**contains()方判断此集合中是否包含指定的元素，如果包含则返回 true，如果不包含则返回false。如果集合list中包含一个引用类型元素，把该元素赋值为null，此时list.contains(null)的返回值才能为true。反之为false。**

**8.**

(单选)关于String 和 StringBuffer 下面说法正确的是（）。

* +  A.

String操作字符串不改变原有字符串的内容

* +  B.

StringBuffer连接字符串速度没有String快

* +  C.

String可以使用append方法连接字符串

* +  D.

StringBuffer在java.util包中

正确答案：A

**解析：**

**A选项：String类是final的。String对象一经创建，其内容不能改变。**

**B选项：StringBuffer连接字符串速度比String快。因为StringBuffer是直接对字符串进行操作，不会创建新的字符串对象。而String如果连接字符串，则会创建新的String对象，然后把连接好的字符串赋值给新创建的String对象，比StringBuffer多了创建新对象的过程。**

**C选项：String没有append( )方法,StringBuffer和StringBuilder有append( )方法。是追加字符内容。**

**D选项：String、StringBuffer、StringBuilder都在java.lang包下。**

**9.**

(单选)运行下面程序: public static void main(String[] args) { Map<String,String> map=new HashMap<String,String>(); map.put(String.valueOf(System.currentTimeMillis())+"a", "1"); map.put(String.valueOf(System.currentTimeMillis())+"b", "2"); map.put(String.valueOf(System.currentTimeMillis())+"c", "3"); for(Map.Entry<String, String> entry:map.entrySet()){ System.out.println(entry.getValue()); } } 输出的结果是：（）。

* +  A.

123

* +  B.

312

* +  C.

213

* +  D.

123顺序无法确定

正确答案：D

**解析：**

**String.valueOf（Long l）方法：把long类型变量转换成String格式**

**map.entrySet()方法：把map转换成Set<Entry<key,value>>。**

**entry.getValue()方法：得到entry中的value值。**

**map中的元素储存顺序是无序的。所以得到的value值也是无序的。**

**10.**

(单选)请看下列代码：

interface Data {

public void load();

}

abstract class Info {

public abstract void load();

}

下列选项中，能正确使用Data接口和Info类的是（）。

* +  A.

public class Employee extends Info implements Data { public void load() { /\*do something\*/ } }

* +  B.

public class Employee implements Info extends Data { public void load() { /\*do something\*/ } }

* +  C.

public class Employee implements Info extends Data { public void Data.load() { /\*d something \*/ } public void load() { /\*do something \*/ } }

* +  D.

public class Employee extends Info implements Data { public void load() { /\*do something \*/ } public void Info.load() { /\*do something\*/ } }

**正确答案：A**

**解析：**

**一个类如果想继承父类的同时又实现一个接口。必须先继承extends父类再实现implements接口。所以先排除B、C选项。**

**D选项本身就是语法错误，没有这种写法。**

**11.**

(单选)下列关于JVM说法，错误的是（）。

* +  A.

JVM通过专门的线程实现内存的回收。

* +  B.

使用java命令时，可以通过参数来设置分配JVM的内存大小。

* +  C.

JRE包括JVM及Java核心类库。

* +  D.

目前主流版本JVM通过纯解释的方式运行Java字节码。

正确答案：D

解析：

JVM先通过javac进行编译，再将编译后的字节码文件通过java指令进行解释运行，如果纯解释的话，根本无法识别源码。

.java文件是给人看的。.class文件是给电脑看的。

使用解释执行的程序我们一般称为解释程序。它将源语言直接作为源程序输入，解释执行解释一句后就提交计算机执行一句，并不形成目标程序。

**12.**

(单选)下列代码编译和运行的结果是（）。

public class A {

public void start() {

System.out.println("TestA");

}

}

public class B extends A {

public void start() {

System.out.println("TestB");

}

public static void main(String[] args) {

((A) new B()).start();

}

}

* +  A.

输出：TestA

* +  B.

输出：TestB

* +  C.

输出：TestA TestB

* +  D.

编译错误

**正确答案：B**

**解析：**

**方法的运行看对象，B类继承A类，并且重写了A类的start()方法，**

**((A) new B()).start();这行代码的意思是：先创建一个B类的对象，然后把该对象向上造型为A类型。**

**方法的运行看对象，此时，创建的对象是B的对象，所以点出来的start()方法也是点出来的B类中的start()方法。**

**13.**

(单选)下列关于IDE开发环境Eclipse，说法错误的是：（）。

* +  A.

Eclipse可以通过插件（plugin）的方式扩展其功能。

* +  B.

Eclipse联盟是由IBM公司捐资组建的。

* +  C.

Eclipse使用了SWT图形界面技术。

* +  D.

Eclipse的运行不需要有JRE的支持。

**正确答案：D**

**解析：程序的运行需要JRE的支持。**

**JDK：最小的开发环境。**

**JRE：最小的运行环境。**

**JDK：Java虚拟机。**

**14.**

(单选)运行下面的程序：

String[] fileNames = { "abc.txt", "bcd.exe", "cde.exe", "def.dat","efg.exe" };

for (String fileName : fileNames) {

if (fileName.endsWith(".exe")){

System.out.print(fileName.substring(0,fileName .lastIndexOf(".exe"))+" ");

}

}

控制台的输出结果是：（）。

* +  A.

bcd. cde. efg.

* +  B.

bc cd ef

* +  C.

bcd.exe cde.exe efg.exe

* +  D.

bcd cde efg

**正确答案：D**

**解析：**

**fileName.endsWith(".exe")：检测此字符串是否是以指定的后缀结束。如果是，返回true，如果不是，返回false。**

**fileName.substring(0,fileName .lastIndexOf(".exe"))+" ");截取一个新的字符串。第一个参数是开始截取的位置，第二个是结束截取的位置，截取规则是：含头不含尾。**

**15.**

(单选)程序执行的结果是（）。

public class Test {

String name="Tom";

public Test(String name){

name=name;

}

public static void main(String [] args){

Test t = new Test("Jack");

System.out.println(t.name);

}

}

* +  A.

null

* +  B.

Tom

* +  C.

Jack

* +  D.

" "

正确答案：B

**解析：**

**创建对象的过程：使用new关键字，调用该类的构造方法来创建对象，根据参数类型也选择调用哪一个构造方法。**

**此时是传“Jack”到构造方法中，但是构造方法中的参数“name”与成员变量“name”变量名相同，而且此时没有写this.来指明哪一个”name”是成员变量，所以此时根据就近原则，两个”name”都是参数”name”。构造方法的第一行，默认会有一个super（）；来调用父类的构造方法，当执行完父类的构造方法之后，此时程序走到String name="Tom";这一行代码处，并把”Tom”赋值给成员变量“name”。然后执行构造方法中的name=name；也就是Jack=Jack；并不会对成员变量的值造成影响。**

**所以当程序运行完毕之后，该对象的name属性为“Tom”。**

**16.**

(单选)请看下列代码编译和运行的结果是（）。

interface DeclareStuff {

public static final int EASY = 3;

**public abstract** void doStuff(int t);

}

public class TestDeclare implements DeclareStuff {

public static void main(String[] args) {

int x = 5;

new TestDeclare().doStuff(++x);

}

void doStuff(int s) {

s += EASY + ++s; System.out.println("s=" + s);

}

}

* +  A.

s=14

* +  B.

s=16

* +  C.

s=10

* +  D.

编译失败

**正确答案：D**

**解析：**

**接口中的所有方法都是由 public abstract来修饰的。**

**当一个类实现了该接口，那么实现该接口的方法必须是由public 来修饰的，不能使用别的访问控制修饰符（如果没有写访问控制修饰符，那么就是由默认的访问控制修饰符来修饰的。）**

**17.**

(单选)下列代码编译和运行的结果是：

public class Wow {

public static void go(short n){

System.out.println("short");

}

public static void go(Short n){

System.out.println("SHORT");

}

public static void go(Long n) {

System.out.println(" LONG");

}

public static void main(String[] args) {

Short y = 6;

int z = 7;

go(y);

go(z);

}

}

* +  A.

short LONG

* +  B.

SHORT LONG

* +  C.

编译失败

* +  D.

抛出运行时异常

正确答案：C

**解析：**

**因为在自定义类Wow中，没有参数类型为int 的go( )方法，所有会有编译错误。**

**18.**

(单选)List类的对象list中的元素为：[0, 1, 2, 3, 4, 5, 6, 7, 8, 9],现在想返回该list对象的子 集合[5,6,7,8],需要做的操作是：

* +  A.

list.subList(5, 8);

* +  B.

list.subList(5, 9);

* +  C.

list.subList(4, 8);

* +  D.

list.subList(4, 9);

**正确答案：B**

**解析：**

**sublist（int fromIndex,int toIndex）方法：返回集合中指定部分的元素。（含头不含尾）**

**19.**

(单选)下列赋值语句中，正确的是（）。

* +  A.

byte b1 = 10, b2 = 20; byte b=b1+b2;

* +  B.

byte b1 = 10, b2 = 20; byte b=~b1;

* +  C.

byte b1 = 10, b2 = 20; byte b=b1>>1;

* +  D.

byte b1 = 10; byte b=++b1;

正确答案：D

**解析：**

**A、B、C选项：在对byte和short类型的值进行移位运算时，你必须小心。因为Java在对表达式求值时，将自动把这些类型扩大为 int型，而且，表达式的值也是int型 。对byte和short类型的值进行移位运算的结果是int型。**

**D选项是对自身进行操作，不会发生数据类型变化。就相当于一个人胖了二三十斤，他还是他自己一样。**

**20.**

(单选)程序的执行结果是（）。

public class Test {

int x; public static void main(String [] args){

Test t = new Test();

t.x=5;

change(t);

System.out.println(t.x);

}

public static void change(Test t){

t.x=3;

}

}

* +  A.

5

* +  B.

3

* +  C.

0

* +  D.

4

正确答案：B

**解析：**

**change(Test t)方法：是改变传入的Test类型引用的x属性为3。**

**21.**

(单选)请看下列代码编译和运行的结果是：

package packagea;

public class Message {

String getText() {

return "text";

}

}

package packageb;

public class XMLMessage extends packagea.Message{

String getText() {

return "<msg>text</msg>";

}

public static void main(String[] args){

System.out.println(new XMLMessage().getText());

}

}

* +  A.

text

* +  B.

<msg>text</msg>

* +  C.

抛出运行时异常

* +  D.

代码public class XMLMessage extends packagea.Message{行，编译错误

正确答案：B

**解析：**

**方法的调用，看对象。**

**此时创建了一个XMLMessage类，该类继承了Message类，并且重写了从父类中继承过来的getText（）方法。所以输出结果是B选项。**

**22.**

(单选)类A，B的定义如下：

class A {

private int a = 100;

A() {

System.out.print("A()");

System.out.println(a);

}

}

class B extends A {

private int a = 200;

B() {

System.out.print("B()");

System.out.println(a);

}

}

运行下面的代码：

new B();

输出的结果是：（）。

* +  A.

A() 100 B() 200

* +  B.

A() 200 B() 200

* +  C.

B() 200 A() 100

* +  D.

B() 200 A() 200

**正确答案：A**

**解析：**

**子类的构造方法中，第一行有默认的super();，调用父类的构造方法，  
执行完父类的构造方法之后，才能给当前类的成员变量赋值。  
当父类执行父类自己的构造方法时，会先执行其父类Object的构造方法。**

**执行完毕之后，父类的成员变量会有初始值。  
Object是所有类的父类。**

**23.**

(单选)请看下列代码：

public static void main(String[] args) {

Calendar c = Calendar.getInstance();

c.set(Calendar.YEAR, 2013);

c.set(Calendar.MONTH, Calendar.FEBRUARY);

c.set(Calendar.DATE, 28);

<插入代码>

}

在<插入代码>处填入将Calendar表示的日期转换为Date表示的日期：

* +  A.

Date d=c.getDate();

* +  B.

Date d=c.getCalendar();

* +  C.

Date d=c.getNow();

* +  D.

Date d=c.getTime();

**正确答案：D**

**解析：**

**Calendar转换成Date：**

**Date d = Calendar.getInstance().getTime();**

**Date转换成Calendar：**

**Date date=new Date();**

**Calendar cal=Calendar.getInstance();**

**cal.setTime(date);**

**24.**

(单选)请看下列代码

public class Member{

private Long userId;

private String nickName;

//以下是getter和sett方法 ……

}

Main方法中的代码：

Member m1=new Member();

m1.setUserId(new Long(100001));

m1.setNickName("mick");

Member m2=new Member();

m2.setUserId(new Long(100001));

m2.setNickName("mick");

System.out.println(m1==m2);

System.out.println(m1.equals(m2));

控制台的输出结果是：

* +  A.

true false

* +  B.

false true

* +  C.

false false

* +  D.

true ture

正确答案：C

解析：

因为Member类没有重写equals方法，所以其equals（）方法只是从Object类中继承过来的那个原生equals方法，其内部就是”==”，比较的对象在内存中的内存地址。

**25.**

(单选)实现Point类的equals方法，具体逻辑为：“成员变量x和y分别相等的Point对象被视为相等”。

public class Point {

private int x;

private int y;

...

public boolean equals(Object obj) {

《填入代码》

}

}

《插入代码》处应填入的代码正确的是:

* +  A.

if(obj.x == this.x || obj.y == this.y){

return true;

}

return false;

* +  B.

if(obj.x == this.x && obj.y == this.y){

return true;

}

return false;

* +  C.

if(!(obj instanceof Point)) {

return false;

}

if(((Point)obj).x == ((Point)obj).y && this.x == this.y){

return true;

}

return false;

* +  D.

if(!(obj instanceof Point)){

return false;

}

if(((Point)obj).x == this.x && ((Point)obj).y == this.y){

return true;

}

return false;

**正确答案：D**

**解析：**

**equals( )方法的重写。在javaSE\_01，Unit02中有ppt详细介绍。此处不作说明。**

**26.**

(单选)请看下列代码：

public class Plant {

private String name;

public Plant(String name) {

this.name = name;

}

public String getName() {

return name;

}

}

class Tree extends Plant {

public void growFruit() { }

public void dropLeaves() { }

}

下列说法正确的是：

* +  A.

在Tree类中添加代码：public Tree() { Plant(); }，编译将通过

* +  B.

在Plant类中添加代码：public Plant() { Tree(); }，编译将通过

* +  C.

在Plant类中添加代码：public Plant() { this(”fern”); }，编译将通过

* +  D.

在Plant类中添加代码：public Plant() { Plant(”fern”); }，编译将通过

正确答案：C

**解析：**

**子类Tree继承父类Plant，父类中有一个有参构造器，所以此时编译器将不再往Plant类中添加无参构造器。**

**子类没有自己写构造器，所以编译器将会在子类中添加默认的无参构造器：**

**Tree( ){**

**super( );**

**}**

**此时super( ),会默认调用父类的无参构造器，父类中没有无参构造器，所以程序会报错。**

**A选项：方法体中，默认第一行是super（）；调用父类无参构造方法，父类中没有无参构造方法，所以会报错，并且是使用super关键字来调用父类的构造方法，此时在方法体中直接调用Plant( );属于语法错误。**

**B选项：方法体中，默认第一行是super（）；并且是使用super关键字来调用父类的构造方法，不能调用子类的构造方法。属于语法错误。**

**C选项：this（）——根据参数，来调用本类中的构造方法。**

**D选项：同B选项，语法错误。**

**27.**

(单选)在Java语言中，字符串“Java程序员”在内存中所占用的字节数是：（）。

* +  A.

10

* +  B.

7

* +  C.

13

* +  D.

14

**正确答案：D**

**解析：**

**String：字符串**

**char：字符（占两个字节）**

**可以简单的把String看成char数组，“Java程序员”是7个char组成，所以在内存中总体占14个字节（byte）的空间。**

**此处不考虑编码集的问题。**

**28.**

(单选)下列代码的输出结果是：

public class Yikes {

public static void go(Long n) {

System.out.println("Long");

}

public static void go(Short n) {

System.out.println("Short");

}

public static void go(int n) {

System.out.println("int");

}

public static void main(String[] args) {

short y = 6;

long z = 7;

go(y);

go(z);

}

}

* +  A.

Long Long

* +  B.

Short Long

* +  C.

int Long

* +  D.

int int

正确答案：C

**解析：**

**参数的选择有优先级**

**如果是整数类型数据，参数列表中是byte或者short数据，优先级是**

**byte（short）——>int——>Byte(Short)**

**29.**

(单选)运行下列程序：

String str = "\*\*java\*\*\*java\*\*\*\*\*java\*";

String str1 = "java";

int index = 0;

while ((index = str.indexOf(str1, index)) != -1) {

System.out.print(index+””);

index += str1.length();

}

控制台输出的结果是：（）。

* +  A.

1 8 17

* +  B.

2 9 18

* +  C.

5 12 21

* +  D.

6 13 22

**正确答案：B**

**解析：**

**indexOf（String str，int fromIndex）方法：**

**返回指定字符串在此字符串中第一次出现处的下标（索引），从指定下标处开始检测。**

**30.**

(单选)下面for语句,存在编译错误的是（）。

* +  A.

for( ; ; ){}

* +  B.

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

* +  C.

for(int i = 0, j=0; ;i++,j++){}

* +  D.

for(int i = 0; i < 10){}

**正确答案：D**

**解析：**

**for（单次**[**表达式**](http://baike.baidu.com/view/420676.htm)**;**[**条件表达式**](http://baike.baidu.com/view/3523810.htm)**;末尾**[**循环体**](http://baike.baidu.com/view/1795994.htm)**）{**

**中间循环体；**

**}**

**其中，表示式皆可以省略，但分号不可省略，因为“;”可以代表一个空语句，省略了之后语句减少，即为语句格式发生变化，则编译器不能识别而无法进行编译。**

**for循环小括号里第一个“;”号前为一个为不参与循环的单次表达式，其可作为某一变量的初始化**[**赋值语句**](http://baike.baidu.com/view/1727818.htm)**, 用来给循环控制变量赋初值；**

**两个“;”号之间的**[**条件表达式**](http://baike.baidu.com/view/3523810.htm)**是一个**[**关系表达式**](http://baike.baidu.com/view/3524120.htm)**, 其为循环的正式开端，当条件表达式成立时执行中间循环体。**

**执行的中间循环体可以为一个语句，也可以为多个语句，当中间循环体只有一个语句时，其大括号{}可以省略，执行完中间循环体后接着执行末尾循环体[2]****。**

**执行末尾循环体后将再次进行条件判断，若条件还成立，则继续重复上述循环，当条件不成立时则跳出当下for循环。**

**A选项：for格式的死循环**

**B选项：常见for循环**

**C选项：死循环，缺少**

**D选项：语句格式错误，缺少末尾循环体**

**31.**

(单选)请看下列代码：

class ClassA {

}

class ClassB extends ClassA {

}

class ClassC extends ClassA {

}

public class Test{

public static void main(String[] args) {

ClassA p0 = new ClassA();

ClassB p1 = new ClassB();

ClassC p2 = new ClassC();

ClassA p3 = new ClassB();

ClassA p4 = new ClassC();

<插入代码>

}

}

可以在<插入代码>处，填入的代码正确的是（）

* +  A.

p0 = p1;

* +  B.

p1 = p2;

* +  C.

p2 = p4;

* +  D.

p2 = (ClassC)p1;

正确答案：A

**解析：**

**A选项: p0 = p1； 向上造型，父类引用指向子类对象 ClassA P0 = new ClassB();**

**向上造型：**

**1.） 能造型的类型有： 父类+所实现的接口**

**2.） 能点出来什么，看引用类型**

**B选项: P1 = P2; ClassB 与 ClassC 无任何关系**

**C选项：P2 = P4； 小转大必须强制类型转换 ClassC P2 =（ClassC）P4；**

**强制类型转换：------------------只有两种情况能够强转**

**1.） 引用所指向的对象，就是该类型**

**2.） 引用所指向的对象，实现了该接口**

**D选项：p2 = (ClassC)p1;ClassB 类与ClassC 类无任何关系，不能互相转化**

**32.**

(单选)以下程序的输出结果是： （）。

public class Super {

public Super() {

System.out.println("Super ");

}

}

public class Sub extends Super{

public Sub() {

System.out.println("Sub");

}

public static void main(String[] args) {

Super fc = new Super();

Sub cc = new Sub();

}

}

* +  A.

Super Super Sub

* +  B.

Super Sub

* +  C.

Sub Super

* +  D.

Super Sub Sub

正确答案：A

**解析：**

**第一步执行Super fc = new Super();**

**然后进入Super的构造器，执行System.out.println("Super ");，输出Super；**

**在执行Sub cc = new Sub();**

**然后进入Sub的构造器，在Sub的构造器中第一步执行super（）；调用父类的额构造器即执行System.out.println("Super ");输出Super；**

**父类的构造器执行完毕回到子类的构造器中执System.out.println("Sub"); 输出Sub。**

**注意：解析省略了进入Object类的构造器.**

**33.**

(单选)运行下列程序： String str = "\*\*oracle\*\*\*oracle\*\*\*\*\*oracle\*\*\*"; String str1 = "oracle"; int index = 0; while ((index = str.indexOf(str1, index)) != -1) { System.out.print(index+””); index += str1.length(); } 控制台输出的结果是：（）。

* +  A.

1 10 21

* +  B.

2 11 22

* +  C.

3 13 23

* +  D.

5 13 22

正确答案：B

**解析：**

**方法：**

**int indexOf(String str, int fromIndex) -----返回指定子字符串在此字符串中第一次出现处的索引，从指定的索引开始。若当前字符串不含有给定的字符串则返回-1**

**Int length() -------------------- 返回此字符串的长度。**

**在执行到while循环中时，第一次index = 2不等于-1；**

**然后执行System.out.print(index+””);**

**index += str1.length(); 即：输出 2**

**然后 index = 2 + 6；**

**第二次循环index = 11不等于-1；**

**然后执行System.out.print(index+””); index += str1.length();**

**即：输出 11**

**然后 index = 11 + 6；**

**第三次循环index = 22不等于-1；**

**然后执行System.out.print(index+””); index += str1.length();**

**即：输出 22 然后 index = 22 + 6；**

**第四次循环index = -1；**

**循环结束。**

**34.**

(单选)下列代码编译和运行的结果是：

class SuperCalc {

protected static int multiply(int a, int b) {

return a \* b;

}

}

class SubCalc extends SuperCalc {

public static int multiply(int a, int b) {

int c = super.multiply(a, b);

return c;

}

}

public class TestSuper {

public static void main(String[] args) {

SubCalc sc = new SubCalc(); System.out.println(sc.multiply(3, 4)); System.out.println(SubCalc.multiply(2, 2));

}

}

* +  A.

运行代码，但是没有输出

* +  B.

代码public static int multiply(int a, int b) {行，编译错误

* +  C.

代码int c = super.multiply(a, b);行，编译错误

* +  D.

代码System.out.println(sc.multiply(3, 4));行，编译错误

**正确答案：C**

**解析：**

**super不能用在static修饰的方法里面。**

**解析：**

**super：**

**1. 指代当前对象的父类对象**

**2. 用法：**

**1）. super.成员变量名----------访问父类的成员变量**

**2）. super.方法名()------------调用父类的方法**

**3）. super()-------------------调用父类的构造方法**

**static1：**

**修饰方法：静态方法**

**1. 由static修饰，**

**2. 属于类的，存在方法区，只有一份**

**3. 常常通过类名点来访问**

**4. 静态方法没有隐藏的this传递，没有this意味着没有对象，而实例变量必须通过对象来访问，所以静态方法不能直接访问实例变量（特点），方法的操作仅与参数相关于对象无关。**

**因为没有对象所以不能通过super调用父类的方法。C出现编译错误**

**35.**

(单选)下列语句创建对象的总个数是：（）。

String s=”a”+”b”+”c”+”d”+”e”;

* +  A.

1

* +  B.

2

* +  C.

3

* +  D.

4

**正确答案：A**

**解析：**

**当几个字符串直接量相加,在编译器就会直接给出相加后的结果**

**当其中有一个是变量时, java编译器会创建一个StringBuilder对象，后面的拼接，实际上是调用StringBuilder对象的append方法。**

**36.**

(多选)下列关于HashMap的描述正确的是：

* +  A.

HashMap的Key和Value是以链表的方式存入对应的bucket

* +  B.

HashMap的查找方式是获取Key的hashCode值，通过hash算法确定存储的bucket， 调用equals方法依次与bucket中的Key进行比较

* +  C.

放入HashMap集合中的元素按照key的自然顺序排序

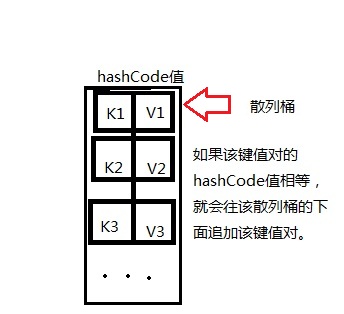
* +  D.

HashMap中的key是不可以的重复的

**正确答案：ABD**

**解析：**

**HashMap中的存储结构：每一个bucket（散列桶）对应一个hashCode值，如果该键值对的hashCode值相等，就会往该bucket下面追加该键值对。**

****

**37.**

(多选)请看下列代码：

public abstract class Shape {

int x; int y; public abstract void draw();

public void setAnchor(int x, int y) {

this.x = x; this.y = y;

}

}

下列选项中能正确使用Shape类的是：

* +  A.

public class Circle implements Shape { private int radius; }

* +  B.

public abstract class Circle extends Shape { private int radius; }

* +  C.

public class Circle extends Shape { private int radius; public void draw(); }

* +  D.

public class Circle extends Shape { private int radius; public void draw() {/\* code here \*/} }

**正确答案：BD**

**解析：**

**A选项：父子类中如果发生继承关系，应该是子类extends父类，而不是implements，implements关键字是类实现接口的。**

**C选项：Circle类继承Shape类后，要重写Shape类中的抽象方法draw（）；方法的五个组成部分：访问控制修饰符、返回值类型、方法名、参数列表、方法体。此处省略了方法体，所以错误。**

**38.**

(多选)请看下列代码 public class Foo { public void method(String str,int age){} } 和Foo类中method方法重载的方法是:

* +  A.

public int method(String str,int age){}

* +  B.

public void method(String s,int year){}

* +  C.

public void method(int year,String s){}

* +  D.

public int method(int year,String s){}

**正确答案：CD**

**解析：**

**方法的重载：**

**1）发生在一个类中，方法名称相同，参数列表不同（与返回值类型无关，与参数的名称也无关）；**

**2）编译器在编译时会自动根据签名来绑定调用不同的方法。**

**AB中的参数列表与题目中的参数列表中的参数类型、个数、顺序都相同，所以AB错误。**

**CD中的参数列表与题目中的参数列表中的参数顺序不同，所以CD正确。**

**39.**

(多选)在Java语言中，下列说法正确的是（）。

* +  A.

一个接口可以继承多个接口

* +  B.

一个类可以继承多个类

* +  C.

一个类可以实现多个接口

* +  D.

一个类可以有多个子类

**正确答案：ACD**

**解析：**

**A．一个接口可以继承多个接口，多个接口之间用逗号隔开。**

**B．一个类只能继承一个类。**

**一个子类只能有一个父类，称为单一继承**

**C．一个类可以实现多个接口，多个接口之间用逗号隔开**

**当既有实现又有继承时，应先继承后实现**

**D．一个父类可以有多个子类**

**40.**

(多选)题目代码的功能为：输出每个字符在一个字符串中出现的次数(不区分大小写)。 String str = "ewrwqFrewqfsadfdsfdsfs"; str=str.toLowerCase(); int max\_length = 0; while (str.length() > 0) { 《插入代码》 }

* +  A.

int length = str.length();

char first=str.charAt(0);

String strNew = str.replaceAll(String.valueOf(first), "");

if (length>strNew.length()) {

max\_length = length - strNew.length(); System.out.println(first+":"+max\_length);

}

* +  B.

int length = str.length();

char first=str.charAt(0);

String strNew = str.replaceAll(String.valueOf(first), "");

if (length>strNew.length()) {

max\_length = length - strNew.length();

str = strNew;

System.out.println(first+":"+max\_length);

}

* +  C.

int length = str.length();

String first = str.substring(0, 1);

String strNew = str.replaceAll(first, "");

if (length>strNew.length()) {

max\_length = length - strNew.length();

str = strNew;

System.out.println(first+":"+max\_length);

}

* +  D.

int length = str.length();

String first = str.substring(0, 1);

String strNew = str.replaceAll(first, "");

if (length>strNew.length()) {

max\_length = length - strNew.length(); System.out.println(first+":"+max\_length);

}

**正确答案：BC**

**解析：**

**int length = str.length(); 获取字符串的长度**

**char first=str.charAt(0); 获取字符串下标为“0”的字符**

**String strNew = str.replaceAll(String.valueOf(first), ""); 将下标为0的的字符转换为字符串格式，将str字符串中与之相同的字符全部替换为""空.返回修改后的字符串。**

**if (length>strNew.length()) { 判断str的长度是否大于steNew的长度**

**max\_length = length - strNew.length(); 得到下标为0的字符的个数：替换之前的长度-替换之后的长度**

**str = strNew; 将修改后的字符串重新复制给str准备下次循环。**

**System.out.println(first+":"+max\_length);输出下标为0的字符以及其个数**

**}**

**int length = str.length();**

**String first = str.substring(0, 1); 截取下标为0的字符（含首不含尾）**

**String strNew = str.replaceAll(first, "");**

**if (length>strNew.length()) {**

**max\_length = length - strNew.length();**

**str = strNew;**

**str = strNew; System.out.println(first+":"+max\_length);**

**}**

**考察：将修改后的字符串重新赋值给str，每次循环都是对修改后的字符串进行操作，**

**否则每次都是对原字符串进行操作，将一直输出e以及e的个数，造成死循环。**

**41.**

(多选)请看下列代码：

public class Old {

public static Object get(List list) {

return list.get(0);

}

}

以下选项调用get方法，能编译通过的是：

* +  A.

Object o = Old.get(new LinkedList());

* +  B.

Object o = Old.get(new LinkedList<?>());

* +  C.

String s = Old.get(new LinkedList<String>());

* +  D.

String s = (String)Old.get(new LinkedList<String>());

**正确答案：AD**

**解析：**

**A选项：static修饰的方法用类名调用，调用之后new LinkedList用Object接收，所以正确**

**B选项：在使用过程中必须指明泛型的类型，由于为？没有指定泛型的数据类型，所以错误**

**C. 在使用过程中，指定泛型类型之后需要强制类型转换，它没有，所以错误**

**D. 标准的泛型使用方法。**

**42.**

(多选)查看如下代码： public class Foo { public void method(String str,int age){} } 下列选项中，和 Foo 类中 method 方法重载的方法是（）。

* +  A.

public int method(String str,int age){}

* +  B.

public void method(int year,String s){}

* +  C.

public int method(int year,String s){}

* +  D.

public int method(String str){}

**正确答案：BCD**

**解析：**

**本题考查的是方法的重载：**

**重载要求方法名相同，参数列表不同,与返回值无关。（发生在本类中）**

**A选项参数列表数据类型相同，所以错误。**

**BCD满足方法重载的定义**

**43.**

(多选)请看下列代码：

public class Key {

private long id1;

private long 1d2;

// class Key methods

}

程序员开发Key类，作为java.util.HashMap的key，那么Key应该覆盖的两个方法是：

* +  A.

public int hashCode()

* +  B.

public boolean equals(Key k)

* +  C.

public int compareTo(Object o)

* +  D.

public boolean equals(Object o)

**正确答案：AD**

**解析：**

**HashMap类是根据hashCode散列值找到对应的散列桶bucket，如果该散列桶中有多个键值对（entry），就会使用equals方法，遍历key值，然后找到对应的键值对。**

**44.**

(多选)下面的方法属于StringBuffer的是：（）。

* +  A.

size

* +  B.

insert

* +  C.

delete

* +  D.

length

**正确答案：BCD**

**解析：**

**查询相关API手册。**

**45.**

(多选)在<插入代码>处，填入下列代码编译正确的是：

public void foo(int[] x) {

<插入代码>

}

* +  A.

foreach(int z : x) System.out.println(z);

* +  B.

for(int z : x) System.out.println(z);

* +  C.

while( x.hasNext()) System.out.println( x.next());

* +  D.

for( int i=0; i< x.length; i++ ) System.out.println(x[i]);

**正确答案：BD**

**解析：**

**A、B选项：foreach语句是java5的新特征之一。**

**foreach语句是for语句的特殊简化版本，也就是增强型for循环。但是foreach语句并不能完全取代for语句，然而，任何的foreach语句都可以改写为for语句版本。**

**foreach的语句格式：**

**for(元素类型t 元素变量x : 遍历对象obj){**

**引用了x的java语句;**

**}**

**C选项：hasNext( )方法是迭代器Iterator独有的方法。迭代器是为了遍历集合。不支持遍历数组。**

**D选项：标准for循环。**

**46.**

(多选)在Java语言中，下列说法正确的是：（）。

* +  A.

StringBuffer和StringBuilder的区别在于：StringBuffer是线程安全的而StringBuilder不是。

* +  B.

String是不可变对象，而StringBuffer中封装的字符串数据是可以动态改变的。

* +  C.

判断两个StringBuilder对象的字符序列是否相同，可以调用其equlas方法进行比较。

* +  D.

String的重写了equals方法，重写的逻辑是：字符序列相同的String对象equals方法返回true。

**正确答案：ABD**

**解析：**

**C选项：**

**StringBuffer类没有重写Object里面的equals()的方法，所以直接用equals()比较的是地址  
 判断两个StringBuffer对象是否值相等，用toString()方法就可以了  
 如： StringBuffer sb1 = new StringBuffer("hello");  
 StringBuffer sb2 = new StringBuffer("hello");  
 判断 sb1.toString().equals(sb2.toString())即可**

**47.**

(多选)请看下列代码：

class One {

public One foo() {

return this;

}

}

class Two extends One {

public One foo() {

return this;

}

}

class Three extends Two {

<插入代码>

}

下列选项中的代码，放置在<插入代码>处无编译错误的是：

* +  A.

public void foo() { }

* +  B.

public Object foo() { return this; }

* +  C.

public Two foo() { return this; }

* +  D.

public One foo() { return this; }

正确答案：CD

**解析：**

**此题考察的是方法重写的“两同两小一大”原则。**

**“两小”之中的：子类的返回值类型小于或等于父类，返回类型为引用类型，小于或等于父类。**

**返回类型为void，则必须相同。**

**返回值如果是基本类型的时候，则必须相同。**

**重写后的方法的权限（访问控制修饰符）必须大于重写前方法的权限。**

**48.**

(多选)查看如下代码： class A { protected int method (int a, int b) { return 0; } ｝ 下列选项中，可以在 A 的子类中使用的是（）。

* +  A.

public int method (int a, int b) { return 0; }

* +  B.

private int method(int a, int b) { return 0; }

* +  C.

private int method(int a, long b) { return 0; }

* +  D.

public short method(int a, int b) { return 0; }

**正确答案：AC**

**解析：**

**A选项：方法签名一样，发生重写。**

**B选项：方法发生重写，要遵循“两同两小一大原则”。“一大”原则：子类的访问权限大于或等于父类。父类的是protected修饰的，子类是private修饰的，所以错误。**

**C选项：此处发生了方法的重载，因为在子类的参数列表中，第二个参数是long值类型，所以方法名相同，参数列表不同，为方法的重载。**

**D选项：**

**此处发生method的重写，应遵循“两同两小一大”原则。“两小”**

**原则之一是：1）子类的返回值类型小于或等于父类。**

**关于这一条，还有三点：**

**➀.返回类型为void，则必须相同。**

**➁.返回类型为基本类型，则必须相同。**

**➂.返回类型为引用类型，小于或等于（父类大，子类小**

**49.**

(多选)下列逻辑表达式，值为false的是（）。

* +  A.

"abc,,,bcd,,def,efg,,".split("[,]+").length == 4;

* +  B.

"1st456".matches("\\d[a-z&&[^et]]{2,8}[0-9]+");

* +  C.

"abcdefghijklmnopqrstuvwxyz".substring(5,26).length() == 20;

* +  D.

"whatisjava".equals(null);

**正确答案：BCD**

**解析：**

**A选项：**

**String split方法，将当前字符串中满足正则表达式的部分进行拆分，并返回所有拆分后的部分。**

**此题，[,]表示一个“，”，加上“+”号，表示所有。所以经过split方法后，将原字符串拆分为四个部分“abc”“bcd”“def”“efg”，所以.length==4。**

**B选项：**

**Matches方法，使用给定的正则表达式验证当前字符串是否满足格式要求，满足则返回true。**

**\\d表示\d，而\d表示任意一个数字字符，相当于[0-9]**

**[a-z&&[^et]]小a到小z，并且不包括e和t,{2,8}去2到8个。因为不包括小t,所以不正确了。**

**C选项：**

**Substring方法，取值范围，含头不含尾。**

**D选项：**

**字符串”whatisjava”有具体内容，不是空（null），所以调用String的equals()方法，比较该字符串与null(空)，结果为false.**

**50.**

(多选)下列赋值语句中，会有编译错误的是（）。

* +  A.

int a = 8888888888;

* +  B.

char b = 1000+300;

* +  C.

byte c = 100+30;

* +  D.

int d = 'a'+'b'+'c';

**正确答案：AC**

**解析：**

**A选项：int值得范围为正负21个亿多，变量a超范围了，所以错误。**

**B选项：**

**对char进行赋值的两种方式：**

**1：字符直接量，如‘中’**

**2：整数类型直接量（十进制、八进制、16进制均可）**

**如10；07；0xff；**

**C选项：byte的范围为-128~127，超范围了，错误。**

**D选项：char本质就是数值，此题正确**