****

**尚学堂Java 初级软件工程师**

**认证考试试卷**

**笔试（A卷）**

考试时间150分钟

总分 100分

姓 名\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

身份证号\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

考试结束后考卷必须收回，否则考试成绩无效

**尚学堂Java 初级软件工程师认证考试试卷**

1. **填空题（共20个题目，总计20分）**
2. 安装JDK后，为了告诉计算机javac.exe和java.exe等执行文件的位置，需要配置的环境变量是\_\_\_\_\_\_\_\_\_\_\_\_\_。
3. Java提供了三种注释类型，分别是单行注释，多行注释和\_\_\_\_\_\_\_\_\_\_\_\_\_。
4. char 类型用来表示在**Unicode**编码表中的字符，长度是\_\_\_\_\_\_个字节。
5. \_\_\_\_\_\_\_\_\_是短路与运算符，如果左侧表达式的计算结果是false，右侧表达式将不再进行计算。
6. \_\_\_\_\_\_\_\_\_\_\_\_\_语句用在循环语句体中，用于终止某次循环过程，即跳过循环体中尚未执行的语句，接着进行下一次是否执行循环的判定。
7. 基本数据类型的类型转换中，要将double类型的常量3.14159赋给为整数类型变量n的语句是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
8. 执行Person p = new Person();语句后，将在\_\_\_\_\_\_\_\_\_\_\_中给Person对象分配空间，并在栈内存中给引用变量p分配空间，存放Person对象的引用。
9. 使用static修饰的变量称为静态变量，静态变量可以有两种访问方式，分别是类名.静态变量名和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
10. 在一个类文件中的关键字package，import，class出现的可能顺序是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
11. 异常是由Java应用程序抛出和处理的非严重错误，比如所需文件没有找到、零作除数，数组下标越界等，可分为两类：Checked异常和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
12. 数组在内存中分配连续的空间，数组的长度固定，一经分配无法改变，可以使用\_\_\_\_\_\_\_\_\_\_\_\_属性获取数组的长度。
13. DateFormat类可以实现字符串和日期类型之间的格式转换，其中将日期类型转换为指定的字符串格式的方法名是\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
14. JDK1.5后提供了\_\_\_\_\_\_\_\_\_\_\_\_关键字，用以定义枚举类。枚举类是一种特殊的类，可以有自己的属性、方法和构造方法。
15. 如果希望将自定义类Student的多个对象放入集合TreeSet，实现所有元素按照某个属性的自然顺序排列，则需要Student类实现\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_接口。
16. 迭代器Iterator为集合而生，专门实现集合遍历，该接口有三个方法，分别是hasNext() 、\_\_\_\_\_\_\_\_\_\_\_\_、remove()。
17. Java IO体系中，\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_是字节输入流，提供了可以存取所有Java基础类型数据（如：int，double 等）和String的方法,但没有提供存取对象的方法。
18. \_\_\_\_\_\_\_\_\_\_\_\_是指将Java对象转换成字节序列，从而可以保存到磁盘上，也可以在网络上传输，使得不同的计算机可以共享对象。
19. Java反射技术中，每个Method对象对应一个方法，获得Method对象后，可以调用其\_\_\_\_\_\_\_\_\_\_\_来调用对应方法。
20. 处于运行状态的线程在某些情况下，如执行了sleep（睡眠）方法，或等待I/O设备等资源，将让出CPU并暂时停止自己的运行，进入\_\_\_\_\_\_\_\_\_状态。
21. 统一资源定位符URL是指向互联网“资源”的指针,由4部分组成：协议、存放资源的主机域名、\_\_\_\_\_\_\_\_\_\_和资源路径和文件名。
22. **选择题（共25个题目，总计25分）**

|  |  |  |  |
| --- | --- | --- | --- |
| **1)** | **以下选项中是对一个Java源文件进行正确编译的语句是（ ）（选择一项）** | | |
|  |  |  | |
|  | **A.** | javac Test.java | |
|  | **B.** | javac Test | |
|  | **C.** | java Test | |
|  | **D.** | java Test.class | |
| **2)** | **以下选项中属于合法的Java标识符的是（ ）。（选择两项）** | | |
|  |  | |  |
|  | **A.** | | public |
|  | **B.** | | 3num |
|  | **C.** | | name |
|  | **D.** | | \_age |
| **3)** | **以下代码的执行结果是（ ）。（选择一项）** | | |
|  | **boolean** m = **true**;  **if**(m = **false**){  System.*out*.println("false");  }**else**{  System.*out*.println("true");  } | | |
|  |  | |  |
|  | **A.** | | false |
|  | **B.** | | true |
|  | **C.** | | 编译错误 |
|  | **D.** | | 无结果 |
| **4)** | **以下do-while循环代码的执行结果是（ ）。（选择一项）** | | |
|  | **int** a=0;  **int** c=0;  **do**{  --c;  a=a-1;  }**while**(a>0);  System.*out*.println(a+" "+c); | | |
|  |  | |  |
|  | **A.** | | -1 -1 |
|  | **B.** | | 死循环 |
|  | **C.** | | -1 -2 |
|  | **D.** | | -1 0 |
| **5)** | **while循环和do-while循环的区别是（ ）。（选择一项）** | | |
|  |  | |  |
|  | **A.** | | 没有区别，这两个结构在任何情况下效果一样 |
|  | **B.** | | while循环比do-while循环执行效率高 |
|  | **C.** | | while循环是先循环后判断，所以循环体至少被执行一次 |
|  | **D.** | | do-while循环是先循环后判断，所以循环体至少被执行一次 |
| **6)** | **分析下面的Java多重循环代码片段，编译运行后的输出结果是（ ）。（选择一项）** | | |
|  | **for** (**int** i = 0; i < 6; i++) {  **int** k = ++i;  **while** (k < 5) {  System.*out*.print(i);  **break**;  }  } | | |
|  |  | |  |
|  | **A.** | | 024 |
|  | **B.** | | 02 |
|  | **C.** | | 123 |
|  | **D.** | | 13 |
| **7)** | **以下关于方法调用的代码的执行结果是（ ）。（选择一项）** | | |
|  | **public** **class** Test {  **public** **static** **void** main(String args[]) {  **int** i = 99;  *mb\_operate*(i);  System.*out*.print(i + 100);  }  **static** **void** mb\_operate(**int** i) {  i += 100;  }  } | | |
|  |  | |  |
|  | **A.** | | 99 |
|  | **B.** | | 199 |
|  | **C.** | | 299 |
|  | **D.** | | 99100 |
| **8)** | **以下关于继承条件下构造方法执行过程的代码的执行结果是（ ）。（选择一项）** | | |
|  | **class** Person {  **public** Person() {  System.*out*.println("execute Person()");  }  }  **class** Student **extends** Person {  **public** Student() {  System.*out*.println("execute Student() ");  }  }  **class** PostGraduate **extends** Student {  **public** PostGraduate() {  System.*out*.println("execute PostGraduate()");  }  }  **public** **class** TestInherit {  **public** **static** **void** main(String[] args) {  **new** PostGraduate();  }  } | | |
|  |  | |  |
|  | **A.** | | execute Person()  execute Student()  execute PostGraduate() |
|  | **B.** | | execute PostGraduate() |
|  | **C.** | | execute PostGraduate()  execute Student()  execute Person() |
|  | **D.** | | 没有结果输出 |
| **9)** | **以下语句中关于Java构造方法的说法错误的是（ ）。（选择一项）** | | |
|  |  | |  |
|  | **A.** | | 构造方法的作用是为创建对象进行初始化工作，比如给成员变量赋值 |
|  | **B.** | | 一个Java类可以没有构造方法，也可以提供1个或多个构造方法 |
|  | **C.** | | 构造方法与类同名，不能书写返回值类型 |
|  | **D.** | | 构造方法的第一条语句如果是super()，则可以省略，该语句作用是调用父类无参数的构造方法 |
| **10)** | **以下关于this和super关键字的说法错误的是（ ）。（选择二项）** | | |
|  |  | |  |
|  | **A.** | | this关键字指向当前对象自身，super关键字指向当前对象的直接父类 |
|  | **B.** | | 在main方法中可以存在this或super关键字，但不能同时存在。 |
|  | **C.** | | this和super关键字都可以访问成员属性，成员方法和构造方法 |
|  | **D.** | | 在一个类的构造方法中可以同时使用this和super来调用其他构造方法 |
| **11)** | **以下选项中可以使成员变量m 被函数fun()直接访问的是（ ）。（选择一项）** | | |
|  | **class** Test {  **private** **int** m;  **public** **static** **void** fun() {  System.*out*.println(m);  }  } | | |
|  |  | |  |
|  | **A.** | | 将private int m 改为protected int m |
|  | **B.** | | 将private int m 改为 public int m |
|  | **C.** | | 将private int m 改为 static int m |
|  | **D.** | | 将private int m 改为 int m |
| **12)** | **Person类和Test类的代码如下所示，则代码中的错误语句是（ ）。（选择一项）** | | |
|  | **public** **class** Person {  public String name;  **public** Person(String name) {  **this**.name = name;  }  }  **public** **class** Test {  **public** **static** **void** main(String[] args) {  **final** Person person = **new** Person("欧欧");  person.name = "美美";  person = **new** Person("亚亚");  }  } | | |
|  |  | |  |
|  | **A.** | | final Person person = new Person("欧欧"); |
|  | **B.** | | person.name = "美美"; |
|  | **C.** | | person = new Person("亚亚"); |
|  | **D.** | | 没有错误 |
| **13)** | **以下关于异常的代码的执行结果是（ ）。（选择一项）** | | |
|  | **public** **class** Test {  **public** **static** **void** main(String args[]) {  **try** {  System.*out*.println("try");  **return**;  } **catch**(Exception e){  System.*out*.println("catch");  }**finally** {  System.*out*.println("finally");  }  }  } | | |
|  |  | |  |
|  | **A.** | | try  catch  finally |
|  | **B.** | | catch  finally |
|  | **C.** | | try  finally |
|  | **D.** | | try |
| **14)** | **以下关于String类的代码的执行结果是（ ）。（选择一项）** | | |
|  | **public** **class** Test2 {  **public** **static** **void** main(String args[]) {  String s1 = **new** String("bjsxt");  String s2 = **new** String("bjsxt");  **if** (s1 == s2)  System.*out*.println("s1 == s2");  **if** (s1.equals(s2))  System.*out*.println("s1.equals(s2)");  }  } | | |
|  |  | |  |
|  | **A.** | | s1 == s2 |
|  | **B.** | | s1.equals(s2) |
|  | **C.** | | s1 == s2  s1.equals(s2) |
|  | **D.** | | 以上都不对 |
| **15)** | **以下关于StringBuffer类的代码的执行结果是（ ）。（选择一项）** | | |
|  | **public** **class** **TestStringBuffer {**  **public** **static** **void** main(String args[]) {  StringBuffer a = **new** StringBuffer("A");  StringBuffer b = **new** StringBuffer("B");  *mb\_operate*(a, b);  System.*out*.println(a + "." + b);  }  **static** **void** mb\_operate(StringBuffer x, StringBuffer y) {  x.append(y);  y = x;  }  } | | |
|  |  | |  |
|  | **A.** | | A.B |
|  | **B.** | | A.A |
|  | **C.** | | AB.AB |
|  | **D.** | | AB.B |
| **16)** | **以下选项中能够正确创建一个数组的是（ ）。（选择二项）** | | |
|  |  | |  |
|  | **A.** | | float []f[] = new float[6][6]; |
|  | **B.** | | float f[][] = new float[][]; |
|  | **C.** | | float [6][]f = new float[6][6]; |
|  | **D.** | | float [][]f = new float[6][]; |
| **17)** | **以下选项中关于int和Integer的说法错误的是（ ）。（选择二项）** | | |
|  |  | |  |
|  | **A.** | | int是基本数据类型，Integer是int的包装类，是引用数据类型 |
|  | **B.** | | int的默认值是0，Integer的默认值也是0 |
|  | **C.** | | Integer可以封装了属性和方法提供更多的功能 |
|  | **D.** | | Integer i=5;该语句在JDK1.5之后可以正确执行，使用了自动拆箱功能 |
| **18)** | **以下选项中关于Java集合的说法错误的是（ ）。（选择二项）** | | |
|  |  | |  |
|  | **A.** | | List接口和Set接口是Collections接口有两个子接口 |
|  | **B.** | | List接口中存放的元素具有有序，不唯一的特点 |
|  | **C.** | | Set接口中存放的元素具有无序，不唯一的特点 |
|  | **D.** | | Map接口存放的是映射信息，每个元素都是一个键值对 |
| **19)** | **以下代码的执行结果是（ ）。（选择一项）** | | |
|  | Set<String> s=**new** HashSet<String>();  s.add("abc");  s.add("abc");  s.add("abc");  s.add("abcd");  System.*out*.println(s.size()); | | |
|  |  | |  |
|  | **A.** | | 1 |
|  | **B.** | | 2 |
|  | **C.** | | 3 |
|  | **D.** | | 4 |
| **20)** | **下面集合类中属于非线程安全，且结构采用了哈希表的是（ ）。（选择一项）** | | |
|  |  | |  |
|  | **A.** | | Vector |
|  | **B.** | | ArrayList |
|  | **C.** | | HashMap |
|  | **D.** | | Hashtable |
| **21)** | **以下选项中关于如下代码的说法正确的是（ ）。（选择二项）** | | |
|  | **public** **class** TestBuffered {  **public** **static** **void** main(String[] args) **throws** IOException {  BufferedReader br =  **new** BufferedReader(**new** FileReader("d:/bjsxt1.txt"));  BufferedWriter bw =  **new** BufferedWriter(**new** FileWriter("d:/bjsxt2.txt"));  String str = br.readLine();  **while**(str !=**null**){  bw.write(str);  bw.newLine();  str = br.readLine();  }  br.close();  bw.close();  }  } | | |
|  |  | |  |
|  | **A.** | | 该类使用字符流实现了文件复制，将d:/bjsxt1.txt复制为d:/bjsxt2.txt |
|  | **B.** | | FileReader和FileWriter是处理流，直接从文件读写数据 |
|  | **C.** | | BufferedReader和BufferedWriter是节点流，提供缓冲区功能，提高读写效率 |
|  | **D.** | | readLine()可以读取一行数据，返回值是字符串类型，简化了操作 |
| **22)** | **InputStreamReader是转换流，可以将字节流转换成字符流，是字符流与字节流之间的桥梁。它的实现使用的设计模式是（ ）。（选择一项）** | | |
|  |  | |  |
|  | **A.** | | 工厂模式 |
|  | **B.** | | 装饰模式 |
|  | **C.** | | 适配器模式 |
|  | **D.** | | 代理模式 |
| **23)** | **以下选项中可以填写到横线处，让代码正确编译和运行的是（ ）。（选择一项）** | | |
|  | **public** **class** Test **implements** Runnable {  **public** **static** **void** main(String[] args) {  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  t.start();  System.*out*.println("main");  }  **public** **void** run() {  System.*out*.println("thread1!");  }  } | | |
|  |  | |  |
|  | **A.** | | Thread t = new Thread(new Test()); |
|  | **B.** | | Test t = new Test(); |
|  | **C.** | | Thread t = new Test(); |
|  | **D.** | | Thread t = new Thread(); |
| **24)** | **在多个线程访问同一个资源时，可以使用（ ）关键字来实现线程同步，保证对资源安全访问。（选择一项）** | | |
|  |  | |  |
|  | **A.** | | synchronized |
|  | **B.** | | transient |
|  | **C.** | | static |
|  | **D.** | | yield |
| **25)** | **以下说法中关于UDP协议的说法正确的是（ ）。（选择二项）** | | |
|  |  | |  |
|  | **A.** | | 发送不管对方是否准备好，接收方收到也不确认 |
|  | **B.** | | 面向连接 |
|  | **C.** | | 占用系统资源多、效率低 |
|  | **D.** | | 非常简单的协议，可以广播发送 |

1. **判断题（共20个题目，总计10分）**
2. 一个Java源文件中可以定义多个类，但是至多只能有一个public类。( )
3. Java的字节码文件的扩展名是class，是一种平台独立的非二进制文件。( )
4. ^ 是异或位运算符，运算规则是如果两个操作数相同，结果是0，否则结果是1。( )
5. 赋值运算符优先级别低于条件运算符，条件运算符优先级别低于算术运算符。( )
6. switch选择语句是多分支选择语句，只能处理等值条件判断的情况，表达式可以是int类型、char类型，但不能是double,float类型。( )
7. 在多重循环中，外层循环变量变化一遍，内层循环变量要变化一次 。( )
8. 程序调用自身的编程技巧称为递归。递归的特点是编程简单并且节省内存。( )
9. 类可以看成一类对象的模板，对象可以看成该类的一个具体实例。( )
10. 使用public修饰的成员属性和方法可以被当前项目中所有包的所有类访问。( )
11. 将子类对象赋给父类引用变量，称为向下转型，将无法访问子类特有的方法。( )
12. 继承是多态的基础，没有继承就没有多态。( )
13. 内部类作为外部类成员，权限修饰符和其他成员一样，可声明为private、默认、protected或public。( )
14. 对于物理连接，比如数据库连接、输入流输出流、Socket连接等，垃圾回收机制无能为力，必须手动关闭才可以。( )
15. 数组和集合中的元素可以是任何数据类型，包括基本类型和引用类型。( )
16. 增强的for循环可以遍历数组和集合，语法简单 不需要知道数组的长度，不能实现与下标相关的操作。( )
17. LinkedHashMap是一种有序的HashMap，查询速度快，便于进行添加删除操作。( )
18. Collection类是专门用来操作集合的工具类，提供一系列静态方法实现对各种集合的操作。( )
19. 一个File对象可以代表一个文件或目录，它可以获取文件和目录属性，也可以访问文件内容。( )
20. 进程是线程Thread内部的一个执行单元，它是程序中一个单一顺序控制流程。( )
21. Socket是传输层供给应用层的编程接口，是应用层与传输层之间的桥梁 。( )
22. **简答题（共5个题目，总计25分）**
    1. 方法重载和方法重写（覆盖）的区别。（4分）
    2. 写出java.lang.Object类的六个常用方法的声明并说明其作用。（6分）
    3. 接口和抽象类的联系和区别。（5分）
    4. Vector和ArrayList、ArrayList和LinkedList的区别和联系。（6分）
    5. 面向对象设计原则有哪些。（4分）
23. **编码题（共3个题目，总计20分）**
    1. 写出饿汉式单例模式、懒汉式单例模式、多线程情况下懒汉式双重判断单例模式的示例代码。（7分）
    2. 写出使用JDBC访问数据库表t\_student中所有数据，并将结果集内容封装成List返回的代码。（6分）
       * 1. 数据库表已经创建，表名称及字段信息如下

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 表名 | t\_student | | 中文表名称 | 学生信息表 | |
| 序号 | 字段名称 | 字段说明 | 类型 | 长度 | 备注 |
| 1 | id | 学生编号 | number | 6 | 主键 |
| 2 | name | 学生姓名 | varchar2 | 8 | 非空 |
| 3 | age | 学生年龄 | number | 3 | 非空 |
| 4 | score | 学生分数 | number | 5,1 | 非空 |
| 5 | enterdate | 入学时间 | date |  | 非空 |

* + - 1. 实体类Student已经提供，和数据库表字段对应，有id,name,age,score,enterDate五个属性及相应的getter和setter方法，提供了各种参数的构造方法。
      2. 使用JDBC访问数据库四个连接参数分别为
         1. driver:oracle.jdbc.driver.OracleDriver
         2. url:jdbc:oracle:thin:@localhost:1521:orcl
         3. username:bjsxt
         4. password:bjsxt
      3. 方法的声明为public List<Student> findAll(){ }，在方法体中书写JDBC代码完成功能即可。
  1. 使用TCP网络编程完成用户登录功能：客户端输入用户名和密码，向服务器发出登录请求；服务器接收数据并进行判断，如果用户名和密码均是bjsxt，则登录成功，否则登录失败，返回相应响应信息；客户端接收响应信息并输出登录结果。（7分）
     + 1. 用户User类已提供构造方法 public User(String username,String password)。
       2. 客户端采用ObjectOutputStream发送封装了用户名和密码的User对象。
       3. 服务器端和客户端的类名和方法声明如下所示。

**public** **class** LoginServer {

**public** **static** **void** main(String[] args) **throws** Exception {}

}

**public** **class** LoginClient {

**public** **static** **void** main(String[] args) **throws** Exception {}

}