

**尚学堂JavaSE 软件工程师**

**认证考试试卷**

**笔试**

考试时间150分钟

总分 100分

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

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

准考证号\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

考试结束后考卷必须收回，否则考试成绩无效

**尚学堂JavaSE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. 使用Java开发应用程序包括编写源程序，编译源程序，解释并运行三个步骤，其中Java源程序编译后生成的字节码文件的扩展名为 。
3. 在Java中，浮点型变量的类型有float和double两种，对于float型变量，内存分配4个字节，对于double型变量，内存分配 个字节。
4. 在Java JDK1.7之前，switch只能支持byte、short、char、int或者其对应的封装类以及Enum类型。在JDK1.7中又加入了 类型。
5. 在循环结构中，如果想跳出循环体，结束整个循环结构可以使用 语句。
6. 类是组成Java程序的基本要素，类体有两部分构成：一部分是变量的定义，另一部分是 的定义。
7. 是指在一个类中可以有多个方法具有相同的名字，但这些方法的参数类型、个数、顺序必须不同，与方法的返回值和访问修饰符无关。
8. 关键字作为Java源文件的第一条非注释性语句，指明该源文件定义的类所有的包。
9. 在Java中如果一个类中没有显示使用extends关键字标明这个类继承自哪个类，那么这个类就默认是 的子类。
10. 声明数组仅仅是给出了数组的名字和元素的数据类型，要想真正地使用数组还必须使用 关键字在堆内存中分配内存空间。
11. 在Java中有二维数组int [ ] [ ] array={{1,2,3},{4,5}} ，可以使用 得到二维数组中第二维中第一个数组的长度。
12. 在Java中使用java.lang包中的 类来创建一个字符串对象，它代表一个字符序列可变的字符串，可以通过相应的方法改变这个字符串对象的字符序列。
13. 在Java中每个Java基本类型在java.lang包中都在一个相应的包装类，把基本类型数据转换为对象，其中包装类Integer是 的直接子类。
14. 对于一个已经不被任何引用变量引用的对象，当垃圾回收器准备回收该对象所占用的内存时，将自动调用该对象的 方法。
15. Math类提供了许多数学运算的静态方法，其中 方法是用来生成随机数的方法，其返回类型是double类型。
16. 在Java中对于程序可能出现的检查时异常，要么用try…catch语句捕获并处理它，要么使用 语句抛出它，由上一级调用者来处理。
17. read方法从输入流中顺序读取源中的单个字节数据，该方法返回字节值(0-255之间的一个整数)，如果到达源的末尾，该方法返回 。
18. 在Java中编写实现多线程应用有两种途径：一种是继承Thread类创建线程，另一种是实现 接口创建线程。
19. 结构是一种由多个节点组成的数据结构，并且每个节点包含有数据以及指向下一个节点的引用。
20. 在Java Socket网络编程中，开发基于TCP协议的服务器端程序使用的套接字是 。
21. 在Java中使用SAX解析XML文档主要有两个部分，一个是SAX解析器，一个是 。
22. **选择题（共25个题目，总计25分）**

|  |  |  |
| --- | --- | --- |
| **1)** | **在Java中，源文件Test.java中包含如下代码，则程序编译运行的结果是（ ）。（选择一项）** | |
|  | **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  system.out.println("Hello!");  }  } | |
|  |  |  |
|  | **A** | 输出：Hello！ |
|  | **B.** | 编译出错，提示“无法解析system” |
|  | **C.** | 运行正常，但没有输出任何内容 |
|  | **D.** | 运行时出现异常 |

|  |  |  |
| --- | --- | --- |
| **2)** | **在Java中，下面（ ）语句能正确通过编译。（选择两项）** | |
|  |  |  |
|  | **A** | System.*out*.println(1+1); |
|  | **B.** | **char** i =2+'2';  System.*out*.println(i); |
|  | **C.** | String s="on"+'one'; |
|  | **D.** | int b=255.0; |

|  |  |  |
| --- | --- | --- |
| **3)** | **分析如下Java代码，编译运行的输出结果是（ ）。（选择一项）** | |
|  | **public** **static** **void** main(String[ ] args) {  **boolean** a=**true**;  **boolean** b=**false**;  **if** (!(a&&b)) {  System.*out.*print("!(a&&b)");  }**else** **if** (!(a||b)) {  System.*out*.println("!(a||b)");  }**else** {  System.*out*.println("ab");  }  } | |
|  |  |  |
|  | **A** | !(a&&b) |
|  | **B.** | !(a||b) |
|  | **C.** | ab |
|  | **D.** | !(a||b)ab |

|  |  |  |
| --- | --- | --- |
| **4)** | **在Java中有如下代码，则编译运行该类的输出结果是（ ）。（选择一项）** | |
|  | **public** **static** **void** main(String[ ] args) {  **for**(**int** i=0;i<10;i++){  **if** (i%2!=0)  **return**;  System.*out*.print(i);  }  } | |
|  |  |  |
|  | **A** | 13578 |
|  | **B.** | 02468 |
|  | **C.** | 0123456789 |
|  | **D.** | 0 |

|  |  |  |
| --- | --- | --- |
| **5)** | **在Java的程序类中如下方法定义正确的是（ ）。（选择两项）** | |
|  |  |  |
|  | **A** | **public** **int** ufTest(**int** num){  **int** sum=num+100;  **return** sum;  } |
|  | **B.** | **public** String ufTest(**int** num){  **int** sum=num+100;  **return** sum;  } |
|  | **C.** | **public** **void** ufTest(**int** num){  **int** sum=num+100;  **return** sum;  } |
|  | **D.** | **public** **float** ufTest(**int** num){  **int** sum=num+100;  **return** sum;  } |

|  |  |  |
| --- | --- | --- |
| **6)** | **分析下面的Java源程序，编译后的运行结果是（ ）。（选择一项）** | |
|  | **import** java.util.\*;  **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  **int** [ ] numbers=**new** **int**[ ]{1,2,3};  System.*out*.println(Arrays.*binarySearch*(numbers, 2));  }  } | |
|  |  | 输出 |
|  | **A** | 输出：0 |
|  | **B.** | 输出：1 |
|  | **C.** | 输出：2 |
|  | **D.** | 输出：3 |

|  |  |  |
| --- | --- | --- |
| **7)** | **分析如下Java程序的代码所示，则编译运行后的输出结果是（ ）。（选择一项）** | |
|  | **public** **class** Test {  **int** count=9;  **public** **void** count1(){  count=10;  System.*out*.println("count1="+count);  }  **public** **void** count2(){  System.*out*.println("count2="+count);  }  **public** **static** **void** main(String[ ] args) {  Test t=**new** Test();  t.count1();  t.count2();  }  } | |
|  |  |  |
|  | **A** | count1=9;  count2=9; |
|  | **B.** | count1=10;  count2=9; |
|  | **C.** | count1=10;  count2=10; |
|  | **D.** | count1=9;  count2=10; |

|  |  |  |
| --- | --- | --- |
| **8)** | **在Java中，以下定义数组的语句正确的是（ ）。（选择两项）** | |
|  |  |  |
|  | **A** | int t[10]=new int[ ]; |
|  | **B.** | char [ ]a=”hello”; |
|  | **C.** | String [ ] s=new String [10]; |
|  | **D.** | double[ ] d [ ]=new double [4][ ]; |

|  |  |  |
| --- | --- | --- |
| **9)** | **分析如下Java代码，该程序编译后的运行结果是（ ）。（选择一项）** | |
|  | **public** **static** **void** main(String[ ] args) {  String str=**null**;  str.concat("abc");  str.concat("def");  System.*out*.println(str);  } | |
|  |  |  |
|  | **A** | null |
|  | **B.** | abcdef |
|  | **C.** | 编译错误 |
|  | **D.** | 运行时出现NullPointerException异常 |

|  |  |  |
| --- | --- | --- |
| **10)** | **给定如下Java代码，编译运行后，输出结果是（ ）。（选择一项）** | |
|  | **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  StringBuffer a=**new** StringBuffer("A");  StringBuffer b=**new** StringBuffer("B");  *operate*(a, b);  System.*out*.println(a+","+b);  }  **static** **void** operate(StringBuffer x,StringBuffer y){  x=**new** StringBuffer("BBB");  y=x;  }  } | |
|  |  |  |
|  | **A** | BBB,B |
|  | **B.** | BBB,BBB |
|  | **C.** | A,B |
|  | **D.** | A,A |

|  |  |  |
| --- | --- | --- |
| **11)** | **给定如下Java程序，Test类中的四个输出语句输出结果依次是（ ）。（选择一项）** | |
|  | **class** Person {  String name="person";  **public** **void** shout(){  System.*out*.println(name);  }  }  **class** Student **extends** Person{  String name="student";  String school="school";  }  **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  Person p=**new** Student();  System.*out*.println(p **instanceof** Student);  System.*out*.println(p **instanceof** Person);  System.*out*.println(p **instanceof** Object);;  System.*out*.println(p **instanceof** System);  }  } | |
|  |  |  |
|  | **A** | true,false,true,false |
|  | **B.** | false,true,false,false |
|  | **C.** | true,true,true,编译错误 |
|  | **D.** | true,true,false,编译错误 |
| **12)** | **在Java中，LinkedList类与ArrayList类同属于集合框架类，下列（ ）选项中是LinkedList类有而ArrayList类没有的方法。（选择两项）** | |
|  |  |  |
|  | **A** | add(Object o) |
|  | **B.** | add(int index,Object o) |
|  | **C.** | getFirst() |
|  | **D.** | removeLast() |

|  |  |  |
| --- | --- | --- |
| **13)** | **如下Java代码，输出的运行结果是（ ）。（选择一项）** | |
|  | **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  List<String> list=**new** ArrayList<String>();  list.add("str1");  list.add(2, "str2");  String s=list.get(1);  System.*out*.println(s);  }  } | |
|  |  |  |
|  | **A** | 运行时出现异常 |
|  | **B.** | 正确运行，输出str1 |
|  | **C.** | 正确运行，输出str2 |
|  | **D.** | 编译时出现异常 |

|  |  |  |
| --- | --- | --- |
| **14)** | **在Java中关于静态方法，以下说法中正确的是（ ）。（选择两项）** | |
|  |  |  |
|  | **A** | 静态方法中不能直接调用非静态方法 |
|  | **B.** | 非静态方法中不能直接调用静态方法 |
|  | **C.** | 静态方法可以用类名直接调用 |
|  | **D.** | 静态方法里可以使用this |

|  |  |  |
| --- | --- | --- |
| **15)** | **如下Java代码运行的结果是（ ）。（选择一项）** | |
|  | **public** **class** Test {  **final** **int** age;  **public** Test(){  age=10;  }  **public** **static** **void** main(String[ ] args) {  System.*out*.println(**new** Test().age);  }  } | |
|  |  |  |
|  | **A** | 输出：0 |
|  | **B.** | 输出：10 |
|  | **C.** | 输出：null |
|  | **D.** | 运行时出现异常 |

|  |  |  |
| --- | --- | --- |
| **16)** | **在Java中关于abstract关键字，以下说法正确的是（ ）。（选择两项）** | |
|  |  |  |
|  | **A** | abstract类中可以没有抽象方法 |
|  | **B.** | abstract类的子类也可以是抽象类 |
|  | **C.** | abstract方法可以有方法体 |
|  | **D.** | abstract类可以创建对象 |

|  |  |  |
| --- | --- | --- |
| **17)** | **在Java中，下列关于读写文件的描述错误的是（ ）。（选择一项）** | |
|  |  |  |
|  | **A** | Reader类的read()方法用来从源中读取一个字符的数据 |
|  | **B.** | Reader类的read(int n )方法用来从源中读取一个字符的数据 |
|  | **C.** | Writer类的write(int n)方法用来向输出流写入单个字符 |
|  | **D.** | Writer类的write(String str)方法用来向输出流写入一个字符串 |

|  |  |  |
| --- | --- | --- |
| **18)** | **分析如下Java代码，有标注的四行代码中，有错误的是第（ ）处。（选择一项）** | |
|  | **import** java.io.FileWriter;  **import** java.io.IOException;  **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  String str = "Hello World";  FileWriter fw = **null**;  **try** {  fw = **new** FileWriter("c:\\hello.txt"); // 1  fw.write(str); // 2  } **catch** (IOException e) {  e.printStackTrace(); // 3  } **finally** {  fw.close(); // 4  }  }  } | |
|  |  |  |
|  | **A** | 1 |
|  | **B.** | 2 |
|  | **C.** | 3 |
|  | **D.** | 4 |

|  |  |  |
| --- | --- | --- |
| **19)** | **在Java的IO操作中，（ ）方法可以用来刷新流的缓冲。（选择两项）** | |
|  |  |  |
|  | **A** | void release() |
|  | **B.** | void close() |
|  | **C.** | void remove() |
|  | **D.** | void flush() |

|  |  |  |
| --- | --- | --- |
| **20)** | **下列关于线程的优先级说法中，正确的是（ ）。（选择两项）** | |
|  |  |  |
|  | **A** | 线程的优先级是不能改变的 |
|  | **B.** | 线程的优先级是在创建线程时设置的 |
|  | **C.** | 在创建线程后的任何时候都可以重新设置 |
|  | **D.** | 线程的优先级的范围在1-100之间 |

|  |  |  |
| --- | --- | --- |
| **21)** | **下列关于Java线程的说法正确的是（ ）。（选择一项）** | |
|  |  |  |
|  | **A** | 每一个Java线程可以看成由代码、一个真实的CPU以及数据三部分组成 |
|  | **B.** | 创建线程的两种方法中，从Thread类中继承的创建方式可以防止出现多父类的问题 |
|  | **C.** | Thread类属于java.util程序包 |
|  | **D.** | 使用new Thread(new X()).run();方法启动一个线程 |

|  |  |  |
| --- | --- | --- |
| **22)** | **在Java网络编程中，使用客户端套接字Socket创建对象时，需要指定（ ）。（选择一项）** | |
|  |  |  |
|  | **A** | 服务器主机名称和端口 |
|  | **B.** | 服务器端口和文件 |
|  | **C.** | 服务器名称和文件 |
|  | **D.** | 服务器地址和文件 |

|  |  |  |
| --- | --- | --- |
| **23)** | **在Java的反射中关于代理说法正确的是（ ）。（选择两项）** | |
|  |  |  |
|  | **A** | 动态代理类与静态代理类一样，必须由开发人员编写源代码，并进行编译 |
|  | **B.** | 代理类与被代理类具有同样的接口 |
|  | **C.** | 动态代理是public、final和abstract类型的 |
|  | **D.** | java.lang.reflect包中的Proxy类提供了创建动态代理类的方法 |

|  |  |  |
| --- | --- | --- |
| **24)** | **在编写XML 文件时，下列关于属性的声明说法正确的是（ ）。（选择两项）** | |
|  |  |  |
|  | **A** | 标记中可以使用多个属性 |
|  | **B.** | 属性可以出现在起始标记或结束标记中 |
|  | **C.** | 属性是不区分大小写的 |
|  | **D.** | 属性值必须用引号引上 |

|  |  |  |
| --- | --- | --- |
| **25)** | **在使用UDP套接字通信时，常用（ ）类把要发送的信息打包。（选择一项）** | |
|  |  |  |
|  | **A** | String |
|  | **B.** | DatagramSocket |
|  | **C.** | MulticastSocket |
|  | **D.** | DatagramPacket |

1. **判断题（共20个题目，总计10分，正确√，错误×）**
2. Java语言是一种面向对象的语言，具有较好的安全性和可移植性及与平台无关等特性。（ ）
3. 假设有一个Java源程序文件，它只定义了一个具有public属性的类Hello，那么编译该文件的命令是“javac Hello”。（ ）
4. 在编译Java源程序时，计算机根本不会去识别各个变量名的具体含义，因此命名规范对编写Java程序而言不是必要的，而且有可能会降低编写Java源程序的效率。（ ）
5. 在Java中任何一个子类的构造方法都必须调用其父类的构造方法（包括隐式调用），并且调用父类的构造方法的语句必须是子类构造方法的第一条语句。（ ）
6. Java语言为所有的Java程序自动导入包“java.lang”，因此Java程序可以直接用“java.lang”中的类和接口。（ ）
7. Java中的类不允许多重继承，但接口支持多重继承。（ ）
8. 父类Person中定义了一个private void show()的方法，那么子类要重写这个方法时，方法的访问修饰符可以是默认的，protected或public。（ ）
9. 抽象方法不能含有方法体，并且必须定义在抽象类中。（ ）
10. 构造方法用于创建对象，一般不允许有任何返回值，因此需要在构造方法返回类型处标注为void。（ ）
11. 在Java中对象可以赋值，只要使用赋值运算符即可，相当于生成了一个各属性与赋值对象相同的新对象。（ ）
12. Java允许创建不规则数组，即Java多维数组中各行的列数可以不同。（ ）
13. 在Java异常处理中可以使用多个catch子句，此时应将异常类的父类Exception放到第一个catch子中。（ ）
14. 与未加访问控制符的缺省情况相比，public和protected修饰符扩大了类及其属性和方法的被访问范围，private修饰符则缩小了这种范围。（ ）
15. 泛型是JavaSE1.7的新特性，泛型的本质是参数化类型，也就是说所操作的数据类型被指定为一个参数。Java语言引入泛型的好处是安全简单。（ ）
16. List是有序的Collection，使用此接口能够精确的控制每个元素插入的位置。用户能够使用索引来访问List中的无素，这类似于Java的数组。（ ）
17. 在使用File类中的delete()方法时，删除时可能文件不存在，所以我们最好先判断一下是否存在，不然会出现NullPointerException异常。（ ）
18. 当一个线程进入一个对象的一个synchronized方法后，其它线程不可以再进入该对象的其它方法执行。（ ）
19. TCP/IP传输控制协议是Internet的主要协议，定义了计算机和外设进行通信的规则。TCP/IP网络参考模型包括七个层次：应用层、会话层、表示层、传输层、网络层、链路层和物理层。（ ）
20. Java反射机制是指：Java程序可以加载一个运行时才得知名称的类，获悉其完整结构信息（但不包括methods定义），并生成其对象实体、或对其fields设值、或调用其methods。（ ）
21. XML在不同的语言里解析方式都是一样的，只不过实现的语法不同而已。基本的解析方式有两种，一种叫SAX，另一种叫DOM。SAX是基于事件流的解析，DOM是基于XML文档树结构的解析。（ ）
22. **简答题（共5个题目，总计25分）**
23. Java异常处理中，关键字try、catch、finally、throw、throws分别代表什么含义？（5分）
24. 请你简述sleep( )和wait( )有什么区别？（5分）
25. 请你简述HashMap和Hashtable的区别？（5分）
26. 什么是设计模式？设计模式的作用？（5分）
27. TCP/IP协议栈中，TCP协议和UDP协议的联系和区别？（5分）
28. **编码题（共3个题目，总计20分）**
29. **编写两个线程,一个线程打印1-52的整数，另一个线程打印字母A-Z。打印顺序为12A34B56C….5152Z。即按照整数和字母的顺序从小到大打印，并且每打印两个整数后，打印一个字母，交替循环打印，直到打印到整数52和字母Z结束。（8分）**

要求：

1. 编写打印类Printer，声明私有属性index，初始值为1，用来表示是第几次打印。
2. 在打印类Printer中编写打印数字的方法print(int i)，3的倍数就使用wait()方法等待，否则就输出i，使用notifyAll()进行唤醒其它线程。
3. 在打印类Printer中编写打印字母的方法print(char c)，不是3的倍数就等待，否则就打印输出字母c，使用notifyAll()进行唤醒其它线程。
4. 编写打印数字的线程NumberPrinter继承Thread类，声明私有属性private Printer p;在构造方法中进行赋值，实现父类的run方法，调用Printer类中的输出数字的方法。
5. 编写打印字母的线程LetterPrinter继承Thread类，声明私有属性private Printer p;在构造方法中进行赋值，实现父类的run方法，调用Printer类中的输出字母的方法。
6. 编写测试类Test，创建打印类对象，创建两个线程类对象，启动线程。
7. **使用IO包中的类读取D盘上exam.txt文本文件的内容，每次读取一行内容，将每行作为一个输入放入ArrayList的泛型集合中并将集合中的内容使用加强for进行输出显示。（5分）**
8. **使用反射机制完成学生对象的创建并输出学生信息。（7分）**

要求：

1. 定义一个学生类Student，其中包含姓名(String)、年龄(int)、成绩(float)的属性。
2. 编写带参与无参构造方法。
3. 重写父类的toString()方法用于输出学生的信息。
4. 编写测试类TestStudent，从键盘录入学生的信息格式为(姓名:年龄:成绩)一次性录入使用“:”分隔，举例(mary:20:90)。
5. 使用String类的split方法按照“:”进行分隔。
6. 调用Constructor的newInstance()方法并用分隔后的信息初始化学生对象。
7. 调用重写父类的toString()方法将学生信息进入输出显示。