华为的JAVA面试题

**QUESTION NO: 1**  
  
publicclass Test1 {  
  
    publicstaticvoid changeStr(String str){  
  
        str="welcome";  
  
    }  
  
    publicstaticvoid main(String[] args) {  
  
        String str="1234";  
  
        changeStr(str);  
  
        System.out.println(str);  
  
    }  
  
}  
  
//输出结果：1234  
  
//这里虽然是一个静态方法，但是里面的变量是一个局部变量，  
  
//所以这里不因为是静态方法，就误认为里面的变量也是静态变量了  
  
**QUESTION NO:2**  
  
publicclass Test2 {  
  
    staticboolean foo(char c) {  
  
       System.out.print(c);  
  
       returntrue;  
  
    }  
  
    publicstaticvoid main(String[] argv) {  
  
       int i = 0;  
  
       //for(65;88&&(i<2);67)  
  
       for (foo('A'); foo('B') && (i < 2); foo('C')) {  
  
           i++;  
  
           foo('D');  
  
       }  
  
    }  
  
}  
  
/\*  
  
What is the result?  
  
A. ABDCBDCB  
  
B. ABCDABCD  
  
C. Compilation fails.  
  
D. An exception is thrown at runtime.  
  
//输出结果是：ABDCBDCB  
  
分析：FOR循环里面讲究的条件要为真，与你的判断式是什么没有关系  
  
就像这里，虽然是打印的字母，但是却不是false，所以可以执行  
  
第一次进行循环：  
  
foo('A')打印字母A，（注：这里不是false条件就默认为true条件）  
  
foo('B')打印字母B，i=0,比较(i < 2)，条件为true，进行循环体，foo('D')打印D  
  
foo('C')打印字母C  
  
第二次循环：  
  
foo('B')打印B，i=1,比较(i < 2)为true，进行循环体，foo('D')打印D  
  
foo('C')打印字母C  
  
第三次循环：  
  
foo('B')打印字母B，i=2，比较(i < 2)为false，退出循环，得结果  
  
\*/  
  
**QUESTION NO: 3**  
  
1. class A {  
  
2. protected int method1(int a, int b) { return 0; }  
  
3. }  
  
Which two are valid in a class that extends class A? (Choose two)  
  
A. public int method1(int a, int b) { return 0; }  
  
B. private int method1(int a, int b) { return 0; }  
  
C. private int method1(int a, long b) { return 0; }  
  
D. public short method1(int a, int b) { return 0; }  
  
E. static protected int method1(int a, int b) { return 0; }  
  
publicclass B extends A{  
  
    /\*\*  
  
     \*@paramargs  
  
     \*/  
  
    //can not reduce the visibility of the inherited method from A  
  
    //即不能够使从类A中继续来的方法的可见性降低    
  
    //private int method1(int a, int b) { return 0; }  
  
    //This static method cannot hide the instance method from A  
  
    //静态方法不能够隐藏继承于A的实例  
  
    //static protected int method1(int a, int b) { return 0; }  
  
    //返回类型与A中的该方法不一致  
  
    //public short method1(int a, int b) { return 0; }  
  
    /\*\*  
  
     \*总结：类的继承中，如果要想重载父类的方法，必须要和父类中的返回类型、可见性等等都要操作一致  
  
     \*否则，程序就会报错。一定遵守子类要遵从于父类的原则  
  
     \*而我选择的答案居然是privateintmethod1和staticprotectedint  
  
     \*我选择第一个的错误理由是：因为原来为保护的，如果我这里设为public，那么就扩展了其原来的可见性  
  
     \*本来原来就是对包外不可见的，现在变成对包外可见的了，所以就选择的是private  
  
     \*选择第二个的错误理由是：都是保护的，这里只是变成了静态的而已  
  
     \*/  
  
    //这里是写了一个重载方法，因为参数类型不一致，不会报错  
  
    privateint method1(int a, long b) { return 0; }  
  
    //可见性可以增大，但是不能够缩小，正确  
  
    publicint method1(int a, int b) { return 0; }  
  
    publicstaticvoid main(String[] args) {  
  
       // TODO Auto-generated method stub  
  
    }  
  
}  
  
**QUESTION NO: 4**  
  
1. public class Outer{  
  
2. public void someOuterMethod() {  
  
3. // Line 3  
  
4. }  
  
5. public class Inner{}  
  
6. public static void main( String[]argv ) {  
  
7. Outer o = new Outer();  
  
8. // Line 8  
  
9. }  
  
10. }  
  
Which instantiates an instance of Inner?  
  
A. new Inner(); // At line 3  
  
B. new Inner(); // At line 8  
  
C. new o.Inner(); // At line 8  
  
D. new Outer.Inner(); // At line 8//new Outer().new Inner()  
  
答案如下：  
  
publicclass Outer {  
  
    publicvoid someOuterMethod() {  
  
       // Line 3  
  
       new Inner();//放在这里不出错  
  
    }  
  
    publicclass Inner {  
  
    }  
  
    publicstaticvoid main(String[] argv) {  
  
       Outer o= new Outer();  
  
       // Line 8  
  
       //o不能够被解释成为一种类型，出错  
  
       //new o.Inner();  
  
       /\*\*  
  
        \*下面两种用法，都报下面的错误：  
  
        \*NoenclosinginstanceoftypeOuterisaccessible.  
  
        \*Mustqualifytheallocationwithanenclosinginstance  
  
        \*oftypeOuter(e.g.x.newA()wherexisaninstanceofOuter)  
  
        \*/     
  
       //new Outer.Inner();  
  
       //new Inner();        
  
    }  
  
}  
  
**QUESTION NO: 5**  
  
Which method is used by a servlet to place its session ID in a URL that is written to the servlet’s response output stream?  
  
（译：那个方法是servlet用于将其session ID入在一个URL中，该URL写入servlet的响应输出流）  
  
A. The encodeURL method of the HttpServletRequest interface.  
  
B. The encodeURL method of the HttpServletResponse interface.  
  
C. The rewriteURL method of the HttpServletRequest interface.  
  
D. The rewriteURL method of the HttpServletResponse interface.  
  
**QUESTION NO: 6**  
  
Which two are equivalent? (Choose two)  
  
A. <%= YoshiBean.size%>  
  
B. <%= YoshiBean.getSize()%>  
  
C. <%= YoshiBean.getProperty("size")%>  
  
D. <jsp:getProperty id="YoshiBean" param="size"/>  
  
E. <jsp:getProperty name="YoshiBean" param="size"/>  
  
F. <jsp:getProperty id="YoshiBean" property="size"/>  
  
G. <jsp:getProperty name="YoshiBean" property="size"/>  
  
**QUESTION NO: 7**  
  
Which of the following statements regarding the lifecycle of a session bean are correct?  
  
1. java.lang.IllegalStateException is thrown if SessionContext.getEJBObject() is invoked when a stateful session bean instance is passivated.  
  
2. SessionContext.getRollbackOnly() does not throw an exception when a session bean with bean-managed transaction demarcation is activated.  
  
3. An exception is not thrown when SessionContext.getUserTransaction() is called in the afterBegin method of a bean with container-managed transactions.  
  
4. JNDI access to java:comp/env is permitted in all the SessionSynchronization methods of a stateful session bean with container-managed transaction demarcation.  
  
5. Accessing resource managers in the SessionSynchronization.afterBegin method of a stateful session bean with bean-managed transaction does not throw an exception.