一、判断题（每题 2 分，共 28 分）

1.为了程序更加简洁，我们应该尽量使用下面的方式来赋值：a=b=1；错

2.每个类都需要定义构建器；错

3.使用 ObjectStream 的方法后，调用 release(),释放对象； **错** 调用 reset() ，释放对象

4. 不能用异常来做一般流程处理的方式，不要过多地使用异常，异常的处理效率比条件分支低，而且异常的跳转流程难以预测。对

5、没有被覆盖的友好方法 和 没有子类的友好类应该定义成 final。对

6、简单的类可以通过名字比较两个对象的类，推荐使用 getClass()或者 instanceof()。错

7、不要调用 Thread 类的 resume(), suspend(),sleep(), stop() 方法。错

8、判断方法是否是重载，只关注方法名、参数个数、参数类型，不关注方法返回值；对

9、类注释部分，描述部分说明该类或者接口的功能、作用、使用方法和注意事项，每次修改后增加作者、新版本号和当天的日期，@since 表示从那个版本开始就有这个类或者接口，@deprecated 表示不建议使用该类或者接口。对

10、对于方法内部用 throw 语句抛出的异常，必须在方法的注释中标明；对于所调用的其他方法所抛出的异常，在注释中要求说明所有的异常；对于非 RuntimeException，即 throws 子句声明会抛出的异常，必须在方法的注释中标明。对

11.相对独立的程序块之间、变量说明之后必须加行空行； 对

12.任何时候都不要使接口可以序列化；对

13.减小单个方法的复杂度，使用的 if,while,for,swith 语句要在 10 个以内； 对

14.main()方法的定义是 public static void main(String args[])；错

二、单选题（每题 2 分，共 36 分）

1 下列错误使用异常的做法是（D）

A. 在程序中使用异常处理还是使用错误返回码处理，根据是否有利于程序结构来确定，并且异常和错误码不应该混合使用，推荐使用异常。

B. 一个方法不应抛出太多类型的异常。throws/exception 子句标明的异常最好不要超过三个。

C. 异常捕获尽量不要直接 catch (Exception ex)，应该把异常细分处理。

D. 程序内抛出的异常本身就可说明异常的类型、抛出条件，可不填写详细的描述信息。捕获异常后用 exception.toString()取到详细信息后保存。

2 下列说法错误的是：C

A. 段代码各语句之间有实质性关联并且是完成同一件功能的，那么可考虑把此段代码构造成一个新的方法。

B. 源程序中关系较为紧密的代码应尽可能相邻。

C. 程序中可同时使用错误码和异常进行处理，推荐使用异常。

D. 方法参数建议不超过 5 个。

3 下面对类、方法、属性的说法不符合编程规范的有：C

A. 不要覆盖父类的私有方法。

B. 类中不要使用非私有的非静态属性。

C.

```
类定义
{
    类的私有属性定义
    类的公有属性定义
    类的保护属性定义
    类的私有方法定义
    类的公有方法定义
    类的保护方法定义
}
```

D. 类私有方法的最大规模建议为 15 个

4 下面的程序片断符合 JTest 规范的是（B）

A.

```
for(int i = 0; i < MAX_INDEX; i++ )
{
    Apple apple = array[i];
}
```

B.

```
public interface ServiceConst
{
    int MAX_BLACK_SIZE = 100;
}
```

C.

```
For (int i = 0; i < MAX_INDEX; i++ )
{
    list.add(FruitFactory.getInstance().createApple());
}
```

D. String log = message + “Y”;

5.排版时，代码缩进应该采用的方式是（C）

A Tab 缩进

B 2 个空格缩进

C 4 个空格缩进

D 8 个空格缩进

6.关于复杂度，下面那句话是错误的：A

A 继承层次建议不要超过 5 层

B 方法行数建议在 10—50 行

C 方法参数建议不要超过 5 个

D 类的行数不要超过 1000 行

7 下列说法错误的是（D）

A. 尽可能的使用局部变量进行运算。

B. 不要使用静态集合，其内存占用增长没有边界。

C. 一个只有 abstract 方法、final static 属性的类应该定义成接口。

D. 使用 while(), sleep()代替 wait(), notify()。

8 下面说法错误的是（D）

A. 属性名不能与方法名相同。

B. 方法重载的时候，一定要注意方法名相同。

C. 方法的参数名不要和类中的方法名相同。

D. 使用 equals() 比较两个类是否相同。

9 下列关于注释说法正确的是 C

A 包注释可有可无，一般大家都是看类注释和方法注释

B 可以把一个类的类注释改为它的文件注释

C 类注释应该放在 package 关键字之后，class 或者 interface 关键字之前

D 文件注释应该使用 javadoc 定义的方式注释，保证能够被收集并形成 doc 文档

10 关于安全，下面那句话是正确的：D

A 任何时候都不要使用内部类

B 任何时候都不要使类可以克隆

C 任何时候不要使接口可以序列化

D 为方法、属性和类定义明确的存取控制，并且尽量不要使用友好方法、属性和类

11 于说法正确的是：D

A. 使用 StringBuffer 的时候设置初始容量，推荐设置为 1024。

B. 使用 StringBuffer 代替 String

C. 在国际化相关的处理逻辑，不要使用 String。

D. 不要通过名字比较两个对象的类，应该使用 instanceof()

E. 类调用方法的最大规模建议不超过 20 个。

12 列关于 finalize（）的描述错误的有：D

A. 在 finalize() 方法中一定要调用 super.finalize() 方法

B. 在 finalize() 方法中的 finally 中调用 super.finalize() 方法;

C. 不要在 finalize() 方法中删除监听器（Listeners);

D. 可以在 finalize() 方法中删除监听器（Listeners);

13 下列错误使用异常的做法是（D）

A. 在程序中使用异常处理还是使用错误返回码处理，根据是否有利于程序结构来确定，并且异常和错误码不应该混合使用，推荐使用异常。

B. 一个方法不应抛出太多类型的异常。throws/exception 子句标明的异常最好不要超过三个。

C. 异常捕获尽量不要直接 catch (Exception ex)，应该把异常细分处理。

D. 程序内抛出的异常本身就可说明异常的类型、抛出条件，可不填写详细的描述信息。捕获异常后用 exception.toString()取到详细信息后保存。

14、下列说法错误的是：

A. 段代码各语句之间有实质性关联并且是完成同一件功能的，那么可考虑把此段代码构造成一个新的方法。

B. 源程序中关系较为紧密的代码应尽可能相邻。

C. 程序中可同时使用错误码和异常进行处理，推荐使用异常。

D. 方法参数建议不超过 5 个。

15 下面的选项与公司的排版规范不相符的是

A. 如果语句已足够清晰则括号内侧(即左括号后面和右括号前面)不需要加空格，多重括号间不必加空格，因为在 Java 语言中括号已是最清晰的标志了。

B.

```
DatabaseKey servicekey = null;
key = getServiceKey();
currentEventsCount = getCurrentEventsCount();
 if (currentEventsCount > 0 )
 {
     //...program code
 }
```

C.

```
if ( writeToFile )
{
    writeFileThread.interrupt();
}
```

D.

```
if  ((a >= b) &&  (c > d))
{
    //program code
}
```

E. 在长语句中，如果需要加的空格非常多，那么应该保持整体清晰，而在局部不加空格。给操作符留空格时不要连续留两个以上空格

16 下面说法或者语句不符合公司编程规范的排版要求的是：

A. 逗号、分号只在后面加空格；比较操作符，赋值操作符"="、 "+="，算术操作符"+"、"%"，逻辑操作符"&&"、"&"，位域操作符"<<"、"^"等双目操作符的前后加空格；"!"、"~"、"++"、"--"、"&"（地址运算符）等单目操作符前后不加空格；

B. a *= 2;

C. x = y&z;

D. key--;

17 有关各种注释内容，描述错误的是：

A. 成员变量注释内容：成员变量的意义、目的、功能，可能被用到的地方。

B. 公有和保护方法注释内容：列出方法的一句话功能简述、功能详细描述、作者、输入参数、输出参数、返回值、违例等。

C. 类和接口的注释内容：类的注释主要是一句话功能简述、功能详细描述，可根据需要列出：版本号、生成日期、作者、内容、功能、与其它类的关系等。 如果一个类存在 Bug，请如实说明这些 Bug。

D. 文件注释内容有：文件名、版权说明、描述信息、生成日期、修改历史。

E. 包的注释内容：简述本包的作用、详细描述本包的内容、产品模块名称和版本、公司版权。

18 下面说法正确的是：

A. 编写代码边注释，修改代码同时修改相应的注释，以保证注释与代码的一致性。不再有用的注释不要删除，使用@deprecated 表示此注释无效。

B. 避免在注释中使用缩写，特别是不常用缩写；但是，注释也是可使用缩写，在使用缩写时或之前，应对缩写进行必要的说明。

C. 在程序块的结束行下方加注释标记，以表明某程序块的结束。

D. 注释应考虑程序易读及外观排版的因素，使用的语言若是中、英兼有的，根据公司国际化的趋势，建议多使用英文。

三、多选题（每题 3 分，共 36 分）

1、下面的程序片断不符合编码规范的有：

A.

```
private final static int TRUNK_BUSY = 1;
private final static int TRUNK_UNKNOWN = -1;

public int writeToDatabase()
{
    ...// program code
    if (state == TRUNK_IDLE)
    {
        state = TRUNK_BUSY;
        ...  // program code
        return 0;
```

```
            }
            else
            {
                state = TRUNK_UNKNOWN;
                return -1;
            }
        }
```

B.

```
    private void initializePool(int count) throws Exception
    {
        // program code
        try
        {
            // program code
        }
        catch (OutOfMemoryError ex)
        {
            throw new Error(ex.toString());
        }
    }
```

C. if ((a | b) && (a & c))

D.

```
    rect.length = 10;
    context.phoneNumber = callData.getPhoneNumber();
    rect.width = 5;
```

2、下面描述中符合公司编程规范的说法有：

A. 不要使用空的 for 、if 、while 语句。

B. 在 switch 中每个 case 语句都应该包含 break 或者 return。

C. 在运算中允许减小数据的精度，在赋值过程要进行强制转型操作。

D. switch 语句中的 case 关键字要和后面的常量保持一个空格，switch 语句中不要定义 case 之外的

无用标签。

E. 尽量显式初始化所有的静态属性，但是对于 int、char 等等非 Object 属性，都有默认值，可以不进行初始化。

3.对包的命名，下面正确的是：

A com.huawei.产品名.模块名称

B com.huawei.开发组名称.项目名称

C com.huawei.部门名称.模块名称

D com.huawei.部门名称.项目名称

4.关于 String 和 StringBuffer，下面哪些是正确的

A 常量字符串使用 String,非常量字符串使用 StringBuffer

B 使用 StringBuffer 的时候设置初始容量

C 尽量使用 StringTokenizer 代替 indexOf()和 substring()

C 尽量不要使用 StringBuffer,StringTokenizer 类

5、下列使用异常的错误的是

A. 程序发生了致命的错误，抛出一个 ERROR 错误通知虚拟机。

B. 程序必须足够健壮，在有可能抛出 ERROR 错误的地方，将其捕获处理，以免错误扩散

C. 运行期异常是程序在运行过程中本身考虑不周导致的异常，程序设计之初考虑不周是难免的，设计时应该定义 RuntimeException 的子类表示这种异常。

D. 方法内可能抛出的异常必须在方法声明上加 throws 子句。

6、下面的做法符合公司的编程规范要求的有：

A. 明确方法功能，精确或近似地实现方法设计。一个函数仅完成一件功能，即使简单功能也应该编写方法实现。

B. 应明确规定对接口方法参数的合法性检查应由接口方法本身负责还是由方法的调用者负责，缺省是由后者负责。

C. 注释的原则是有助于对程序的阅读理解，如果一个类存在 Bug，要如实说明这些 Bug。

D. 父类如果实现了比较合理的 toString() ，子类可以继承不必再重写 toString()。

E.数据库操作、IO 操作等需要使用结束 close()的对象必须在 try-catch-finally 的 finally 中 close()。

7、下列程序片断符合编码规则的有：

A.

```
try
{
  // ...程序
}
catch( NullPointException ex)
{
    Log.doLog(ex.getMessage());
}
```

B.

```
public void subscribe(int id)
{
    // program code
    System.out.println("Result: " + id + " subscribe succeed")
    // program code……
}
```

C.

```
public void subscribe(int id)
{
    // program code
    LogManager.info("Result: " + id + " subscribe is succeed")
}
```

D.

```
try
{
    //.... ...
}
catch (ServiceException ioe)
{
    LogManager.warn(ioe);
}
```

8 下面哪些符合公司的编程规范的注释要求的有：

A. 文件注释：

```
/*
 * 文件名：LogManager.java
 * 描述： WIN V200R002 WEBSMAP 通用日志系统
 * 修改人： 张三
 * 修改时间：2001-02-16
 * 修改内容：新增
*/
```

B. 类注释：

```
/**
 * LogManager 类集中控制对日志读写的操作。
 * 全部为静态变量和静态方法，对外提供统一接口。分配对应日志类型的读写器，
 * 读取或写入符合条件的日志纪录。
 * @author      张三，李四，王五
 * @version     1.2, 2001-03-25
 * @see         LogIteraotor
 * @see         BasicLog
 * @since        CommonLog1.0
 */
```

C.

```
<... 省略了文件注释、包语句、类的注释…>
public class KeyManager
{
     private int key = 0; //key 属性记录关键事件 ID

    /**
    * 设置关键事件 ID
    * 函数功能：呼叫过程，设置（记录）关键事件的 ID
    * @param [key|int]     呼叫过程的关键事件 ID
    * @return  [void]     返回空
    */
```

```
        public void setKey( int key )
        {
            this.key = key;
        }
    }
```

D.

```
/**
 * 根据日志类型和时间读取日志。
 * 分配对应日志类型的 LogReader，反复器缓冲数，
 * 读取日志记录。查询条件为 null 或 0 的表示没限制，
 * 反复器缓冲数为 0 读不到日志。
 * @param logTypeName  日志类型名（在配置文件中定义的）
 * @param startTime     查询日志的开始时间
 * @param bufferNum     日志反复器缓冲记录数
 * @return  结果集，日志反复器
 * @since  CommonLog1.0
 */
 public static LogIterator read(String logType, Date startTime,
                                int bufferNum)
                                throws Exception
{
    if (null == logType )
    {
        //如果日志类型没有设置，抛出“日志类型为空”异常
        throw new LogTypeException("Log Type is null!");
    }
    ...//program code
}
```

E.

```
 public void example( )
```

```
{
    // 注释
    CodeBlock One

    // 注释
    CodeBlock Two
}
```

9 下面说法正确的是

A 没有子类的友好类应该定义成 final

B 没有被覆盖在友好方法应该定义成 final

C 不定义在包中没有被用到的友好属性、方法和类

D 不要定义不会被用到的局部变量、类私有属性、类私有方法和方法参数

10 下面说法正确的有：

A. 对于方法内部用 throw 语句抛出的异常，必须在抛出异常的语句上一行注释标明抛出异常的含义、抛出条件等。

B. 通过对函数或过程、变量、结构等正确的命名以及合理地组织代码的结构，使代码成为自注释的。对保护方法以清晰准确的函数命名，可增加代码可读性，并减少不必要的注释，甚至可以不注释。

C. 异常的注释必须说明该异常的含义及什么条件下抛出该异常。

D. 当代码段较长，特别是多重嵌套时，在程序块的结束行右方加注释标记，以表明某程序块的结束，这样做可以使代码更清晰，更便于阅读。

E. 调试程序的时候可以方便的使用 /* 。。。*/ 注释掉一长段程序。

11 下面程序片断不符合公司命名规范的有：

A.

```
public class Car
{
    private static int itemCount = 0;

    public void setItemCount( int cout )
    {
        itemCount = cout;
```

```
        }
    }
```

B.

```
    public final static int DEFAULT-START-KEY = 0;
```

C.

```
    public class LogManager
    {
        private int size = 0;
        public boolean writeFile( String value )
        {
            int size = 0;
            boolean isRight = false;
            size = LogUtility.getSize( value );
            isRight = LogUtility.write( size, value );
            return isRight;
        }
    }
```

D. `protected abstract boolean getServiceConfigurationData( int serviceKey );`

12、下面描述错误的有：

A. 一个方法不应抛出太多类型的异常，throws/exception 子句标明的异常最好不要超过五个。

B. 运行期异常必须有 throws 子句标出，不标出或者调用者不捕获该类型异常都会导致编译失败，从而防止程序员本身疏忽。

C. 抛出非运行期异常的目的是防止异常扩散，导致定位困难。

D. 在部门内部应该规划好包名的范围，防止产生冲突