1. 0==0.clone() 与0.equals(0) 的区别

答：浅度复制是不会把要复制的那个对象的引用对象重新开辟一个新的引用空间，当我们需要深度复制的时候，这个时候我们就要重写clone()方法。

因为Object的equal方法默认是两个对象的引用的比较，意思就是指向同一内存,地址则相等，否则不相等；如果你现在需要利用对象里面的值来判断是否相等，则重载equal方法。

1. 浅拷贝，深拷贝和完全拷贝的区别

答：如果一个类拥有资源，当这个类的对象发生复制过程的时候，资源重新分配，这个过程就是深拷贝，反之，没有重新分配资源，就是浅拷贝。

浅拷贝:拷贝所考虑对象，不拷贝所引用对象；

深拷贝：所引用的都拷贝；

完全拷贝：所有的都拷贝。

1. 单例模式与序列化的比较

答：单例模式的目的是使得类的一个对象成为系统中的唯一实例。要实现这一点，可以从客户端对其进行实例化开始。因此需要用一种只允许生成对象类的唯一实例的机制，“阻止”所有想要生成对象的访问。使用工厂方法来限制实例化过程。这个方法应该是静态方法；

序列化将对象的状态信息转换为可以存储或传输的形式的过程。在序列化期间，对象将其当前状态写入到临时或持久性存储区。以后，可以通过从存储区中读取或反序列化对象的状态，重新创建该对象。

序列化使其他代码可以查看或修改那些不序列化便无法访问的对象实例数据。通常，对象实例的所有字段都会被序列化，这意味着数据会被表示为实例的序列化数据。这样，能够解释该格式的代码有可能能够确定这些数据的值，而不依赖于该成员的可访问性。类似地，反序列化从序列化的表示形式中提取数据，并直接设置对象状态，这也与可访问性规则无关。

对于任何可能包含重要的安全性数据的对象，如果可能，应该使该对象不可序列化。如果它必须为可序列化的，请尝试生成特定字段来保存不可序列化的重要数据。如果无法实现这一点，则应注意该数据会被公开给任何拥有序列化权限的代码，并确保不让任何恶意代码获得该权限。