1. 类的初始化顺序：

父类静态初始化块

子类静态初始化块

父类初始化块

调用了父类无参构造器

子类初始化块

调用子类的构造器

1. 访问修饰符：



1. 重载和重写：

|  |  |  |
| --- | --- | --- |
|  | 重写（Overriding） | 重载（Overloading） |
| 类的数量 | 父子类、接口与实现类 | 本类 |
| 方法名称 | 一致 | 一致 |
| 参数列表 | 一定不能修改 | 必须修改 |
| 返回类型 | 一定不能修改 | 可以修改 |
| 异常 | 可以减少或删除，但不能扩展 | 可以修改 |

重写：两同两小一大原则

方法名相同，参数类型相同

子类返回类型小于等于父类方法返回类型，

子类抛出异常小于等于父类方法抛出异常，

子类访问权限大于等于父类方法访问权限。

重载：（1）参数个数不同

（2）参数类型不同

（3）参数的多类型顺序不同

与下列因素无关

（1）参数名称

（2）方法的返回值类型

（3）访问修饰符

1. 接口和抽象类：

接口是对动作的抽象，抽象类是对根源的抽象。抽象类表示的是，这个对象是什么。接口表示的是，这个对象能做什么。

1.接口是抽象类的变体，接口中所有的方法都是抽象的。而抽象类是声明方法的存在而不去实现它的类。  
 2.接口可以多继承，抽象类不行  
 3.接口定义方法，不能实现，而抽象类可以实现部分方法。  
 4.接口中基本数据类型为static 而抽类象不是的。

5.接口中没有构造方法，抽象类有构造方法。

当你关注一个事物的本质的时候，用抽象类；当你关注一个操作的时候，用接口。

抽象类：

*用abstract修饰的类叫做抽象类。*

1.abstract修饰的类为抽象类，此类不能有对象，（无法对此类进行实例化，说白了就是不能new）；

2.abstract修饰的方法为抽象方法，此方法不能有方法体（就是什么内容不能有）；

关于抽象类的使用特点：

1.抽象类不能有对象，（不能用new此关键字来创建抽象类的对象）；

2.有抽象方法的类一定是抽象类，但是抽象类中不一定有抽象方法；

3.抽象类中的抽象方法必须在子类中被重写。

接口：

1.接口中的所有属性默认为：**public static final** \*\*\*\*；

2.接口中的所有方法默认为：**public abstract**\*\*\*\*；

5.final类：

final类中的成员变量可以根据需要设为final，但是要注意final类中的所有成员方法都会被隐式地指定为final方法。

1. static关键字：

（1）A，静态对象的数据在全局是唯一的，一改都改。如果你想要处理的东西是整个程序中唯一的，弄成静态是个好方法。 非静态的东西你修改以后只是修改了他自己的数据，但是不会影响其他同类对象的数据。   
B，引用方便。**直接用 类名.静态方法名  或者  类名.静态变量名**就可引用并且直接可以修改其属性值，不用get和set方法。  
C，保持数据的唯一性。此数据全局都是唯一的，修改他的任何一处地方，在程序所有使用到的地方都将会体现到这些数据的修改。有效减少多余的浪费。

D，static final用来修饰成员变量和成员方法，可简单理解为“全局常量”。对于变量，表示一旦给值就不可修改；对于方法，表示不可覆盖。

（2）静态变量、静态方法和静态块

A，它们仅能调用其他的static 方法

B，它们只能访问static数据

C，它们不能以任何方式引用this 或super(this涉及到对象，super 与继承有关）