oop3

抽象类

- 抽象类语法 —— 权限修饰符符 abstract class 类名{ } —— 2个修饰符
- 抽象方法语法 —— 权限修饰符 abstract 方法返回值类型 方法名(形参列表) —— public
- 注意: —— 抽象方法只能放在抽象类里
- 抽象类特点 —— 在普通的类的基础上 可以有抽象方法
- 使用
  - 1.子类是普通类, 必须要实现所有的抽象类中的抽象方法
  - 2.子类还可以是抽象类, 选择性的实现抽象方法

接口

- 接口语法 —— 权限修饰符 interface 接口名{} —— 2个修饰符
- 注意
  - interface表示是个接口, 关键字跟class具有同等的地位
  - 命名要使用大驼峰
  - 某个类实现接口的时候使用implements关键字, 多实现的
  - 接口和接口之间是可以进行继承的 extends, 多继承的
  - 一个类继承了某个父类, 又实现了接口 —— class Son extends Father implements 接口名
- 特点
  - 全局常量
  - 抽象方法
  - 里面接口默认的public, 不用写 —— 抽象方法默认public abstract
  - 默认方法 —— 使用 default —— 作为子类的实现
  - 静态方法
- 使用
  - 1.子类是普通类, 要求子类必须实现接口中所有的抽象方法
  - 2.子类是抽象类, 要求子类选择性实现
  - 3.子类是接口,子类选择性实现

内部类

- 定义 —— 定义在某个类的内部这样的类
- 根据定义的位置分类
  - 成员位置
    - 成员内部类
    - 静态内部类
  - 局部位置
    - 局部内部类
    - 匿名内部类
    - lambda
- 成员内部类
  - 修饰符 —— 4个
  - 成员特点
    - jdk8 不能有static声明
    - JDk17 跟普通类一样
  - 继承和实现 —— 继承和实现外部的类和接口 还能继承成员内部类
  - 访问特点
    - 内部类--->外部类
      - 直接通过成员名 访问 —— 没有权限限制
      - 出现同名的成员
        - this.访问的是内部类的成员
        - Outer.this.访问的是外部类成员
    - 外部类--->内部类
      - 普通方法 —— 创建内部类对象 访问
      - 静态方法 —— 创建外部类对象 再创建内部类对象访问 —— Outer.new Inner()
    - 内部类----> 外部其他类 —— 创建外部其他类对象访问 受权限控制
    - 外部其他类----> 内部类 —— 创建外部类对象,再创建内部类对象访问,受权限控制 —— new Outer.new Inner()
  - 关系 —— 把内部类当做外部类成员
- 静态内部类
  - 修饰符 —— 4个
  - 成员特点 —— 普通类一样
  - 继承和实现 —— 可以继承实现外部的类和接口, 继承静态内部类 不能继承成员内部类
  - 访问特点
    - 内---->外 —— 创建外部类对象访问 不受权限影响
    - 外---> 内 —— 创建内部类对象访问 不受权限影响
    - 内--->外部其他类 —— 创建外部其他类对象访问 受权限控制
    - 外部其他类---> 内 —— 创建内部类对象访问 new Outer.Inner() 受权限控制
  - 关系
    - 2个独立的类
    - 类加载没关系
- 局部内部类
  - 修饰符 —— 没有
  - 成员特点 —— 跟成员内部类一样
  - 继承和实现 —— 可以继承和实现外部类或者接口
  - 访问特点
    - 内-->外
      - 成员方法 —— 直接访问,同名 通过this. 或者Outer.this.区分
      - 静态方法 —— 创建外部类对象访问
    - 外 ----> 内部类 —— 在作用域内, 创建内部 类对象 访问
    - 外部其他类--> 内 —— 没有
    - 内--->外部其他 —— 创建外部其他类对象,访问 受权限控制
  - 关系 —— 把局部内部类当中局部变量
- 匿名内部类
  - 特殊的局部内部类
  - 语法 —— new 类名/接口名() { 方法体 } —— 创建的是类名或者接口名对应的匿名子类对象
  - 基本使用
    - 不用引用接收 —— 直接调用方法 —— 一次性的
    - 使用引用接收 —— 父类或父接口进行接收 —— 多次调用
  - 使用场景
    - 1.方法传参 —— 参数类型是父类或父接口类型 —— 实际参数可以是父类或者父接口对应的匿名子类对象
    - 2.方法返回值 —— 返回值类型可以是父类型或者父接口类型 —— 具体返回的对象可以是父类或父接口对应的匿名子类对象
- lambda
  - 语法 —— (形参列表) -> { 方法体 }
  - 使用前提
    - lambda表达式只能作为接口的匿名子类对象, 也是一种特殊的局部类
    - 接口必须是功能接口 —— @FunctionalInterface 来检查是否是功能接口
    - 功能接口的要求
      - 接口中有且仅有一个要被子类实现的抽象方法
      - 静态方法 默认方法不算
  - 格式与形式
    - 形参列表简化
      - 形参的数据类型可以省略
      - 形参只有1个, ( ) 可以省略掉
      - 如果形参列表为空, () 不可以省略
    - 方法体的简化
      - 如果方法体中只有一行代码, { } 可以省略掉
      - 特殊的, 如果方法体中只有一行代码 并且是返回语句 { } 可以省略 return也可以省略掉
  - 类型推断
    - 1.父接口进行接收
    - 2. ((父接口) lambda) 类型与强制的形式
    - 3.借助于方法完成类型推断
      - 方法的参数
      - 方法的返回值
- 方法引用 —— 能看懂就行

extends VS implements

抽象类VS接口