面向对象

回顾

```
1工具类的使用Arrays
    binarySearch() //二分查找
    sort()//排序
    fill()//填充
    copyOf()//复制
    copyOfRange();范围复制
    toString()//把数组转成字符串
    equals();
2 方法的参数传递和返回值
     方法的传递采用传值的方式:
         基本类型传递的是实际数据：相当于把数据复制一份
         引用类类型传递的地址。是同一个数据
     返回值：基本类型返回的是实际数据，引用类型返回的是地址
3 二维数组
    数组中的每个元素还是数组
    定义
     int[][] arr=new int[][]{{1,2},{3,4}};
4 调试技巧
    4.1添加断点
    4.2单步执行  F6单步跳过 F5单步进入 F7单步返回  F8继续
    其他的调试方法：加输入语句
5 文档注释
    生成api文档
    编码中有提示
    /**

    */
    标签:
    @author
    @version
    @since
    @see
    @param
    @return
```

今天任务

1. 面向对象
2. 面向过程
3. 类的创建
4. 对象的创建
5. 对象内存分析
6. 构造方法
7. this关键字

教学目标

1. 掌握面向对象思想
2. 掌握类和对象的创建
3. 掌握对象创建内存分析
4. 掌握构造方法
5. 掌握this关键字

第一节： 面向对象编程思想

1.1 什么是面向对象

一种看待问题的思维方式，着眼于找到一个具有特殊功能的具体个体，然后委托这个个体去做某件事情，我们把这个个体就叫做对象。
是一种更符合人类思考习惯的思想【懒人思想】，可以将复杂的事情简单化，将程序员从执行者转换成了指挥者
使用面向对象进行开发，先要去找具有所需功能的对象来用，如果该对象不存在，那么创建一个具有所需功能的对象

1.2 什么是面向过程

一种看待问题的思维方式，在思考问题的时候，着眼于问题是怎样一步一步解决的，然后亲力亲为的去解决问题

1.3 面向对象和面向过程对比

面向对象是基于万物皆对象这个哲学观点

举例说明：

案例一：我想要吃大盘鸡

面向过程	面向对象
1.自己去买菜	1.委托一个会砍价的人帮忙去买菜
2.自己择菜	2.委托一个临时工帮忙择菜
3.自己做菜	3.委托一个厨师帮忙做菜
4.自己开始吃	4.自己开始吃

案例二：小明是一个电脑小白，想要配一台电脑，买完零件后需要运到家里，组装完成后打开电脑玩游戏

面向过程	面向对象
1.小明补充电脑知识	1.委托一个懂电脑的朋友(老王)去帮忙买零件
2.小明去买零件	2.委托一个能跑腿的人去买零件
3.小明把零件带回家里	3.委托一个快递小哥帮小明送到家里
4.小明组装电脑	4.委托一个会组装电脑的人帮小明组装电脑
5.小明开机玩电脑	5.小明自己打开电脑，开始玩游戏

小结

a.都是看待问题的一种思维方式，都能解决问题
b.面向过程着眼于所有的事情按照步骤来实现
c.面向对象着眼于找到一个具有特殊功能的对象，委托这个对象去做某件事情。
注意：面向对象是一种思想，并不是一门编程语言。

1.4 类的概念

一类具有相同属性和功能的实体的集合，类是Java语言的基本单位

1.5 对象的概念

在一个类中，一个具有特殊功能的实体，能够解决特定的问题，对象也被称为实例。

1.6 类与对象之间的关系

类是对象的抽象（模板）,对象是类的具体体现(实例)

1.7 类的声明

```
语法：
访问权限修饰符 class 类名 {
    //属性
    //方法
}
说明：a.访问权限修饰符：只能是public或省略不写
     b.类名只要是一个合法的标识符即可，但是要求：首字母必须大写，遵循帕斯卡命名法
     c.尽量使用单个或多个有意义的单词连接而成
public class Person{

}
public class Student{

}
```

1.8 类中成员变量的定义

> 生活中：使用特征或状态描述一个事物属性。程序中：使用变量表示，通常叫成员变量
>
> 成员变量：成员变量也被称为属性、分为静态变量和非静态变量。

1.9 类中方法的定义

> 生活中：使用功能、动作描述一个事物行为。程序中：使用方法表示。
>
> 方法分为静态方法和非静态方法。

```
/**
 * 人类:描述人    静态特征：姓名  性别  身高   体重  ....   动态特征：  吃  喝   写代码  睡觉  谈恋爱...
 *  成员变量  ：  描述静态特征
 *  方法：  描述动态特征
 *
 * @author wgy
 * public  访问修饰符  公开的
 * class   表示类
 * Person  类名
 *
```

```
 */
public class Person {
    //成员变量
    String  name;//姓名
    String sex;//性别
    int height;//身高
    int weight;//体重

    //方法
    //吃
    public void  eat() {
        System.out.println(name+" 吃饭....");
    }
    //喝
    public void drink() {
        System.out.println(name+" 喝啤酒");
    }
    //写代码
    public void writeCode() {
        System.out.println(name+" 写java代码");
    }

}
```

思考下面描述包含的类

```
波波坐飞机去泰国旅游
老师在黑板上画圆
列车司机紧急刹车
售货员统计收获小票的金额
张三把门关上
```

上机练习:定义一个学生类:

学生类包含属性：学号、姓名、年龄、专业

学生类包含方法：学习、考试

```
/**
 * 学生类
 *  属性:学号，姓名，年龄，专业
 *  方法:学习 考试
 * @author wgy
 *
 */
public class Student {
    //属性
    //学号
    String stuNo;
    //姓名
    String name;
    //年龄
    int age;
    //专业
    String pro;

    //方法
```

```
    //学习
    public void study() {
        System.out.println(name+"正在好好学习，学号:"+stuNo);
    }
    //考试
    public void exam() {
        System.out.println(name+"要参加"+pro+"考试");
    }
}
```

第二节：对象的创建以及内存分析

2.1 对象的创建

> 对象的创建过程也被称为对象的实例化过程
>
> 语法：类名 对象名 = new 类名();
>
> 调用属性：
>
> 对象名.属性名
>
> 调用方法：
>
> 对象名.方法名(实参);

```
//演示对象的创建，跨类进行调用成员方法以及访问成员变量
//测试类：含有main函数得类被称为测试类
public class TestPerson
{
    public static void main(String[] args)
    {
        //1.创建对象
        //语法:类名 变量名称  = new  类名();
        Person xiaoMing = new Person();

        //2.调用对象属性
        //语法：对象.属性名
        xiaoMing.name = "小明";
        xiaoMing.age = 10;
        xiaoMing.gender = '男';
        //3.调用对象方法
        //语法：对象.方法名()
        xiaoMing.eat();
    }
}

//实体类：表示具有某些特征或者某些行为的类
//描述多个对象所具有的共同特征和共同行为
//需求：人类，具有姓名，年龄，性别等的特征，可以吃东西，可以奔跑，，，，
class Person
{
    //第一部分
    //成员变量：特征【名词】
    //非静态成员变量
    String name;//null
    int age;//0
    char gender;//\u0000
```

```
    //第二部分
    //成员方法：行为【动词】
    //非静态成员方法
    public void eat() {
        System.out.println("eating~~~~");
    }
}
```

2.2 对象内存分配

程序运行时，操作系统会分配三块主要的内存空间

- **栈**：直接存放基本类型数据，和引用类型的地址
  - 栈空间比较小，存取速度相对较快
  - 先进后出
- **堆**：存放引用类型的实际数据部分
  - 堆空间比较大，存取速度相对较慢
- **方法区**：保存类的信息（包括类的名称、方法信息、字段信息）；方法区中有一块空间叫串池（常量池），用来存放字符串常量；另一块空间叫静态区用来存储静态数据。

  在jdk1.7之后常量池、静态区作为堆中一部分，方法区的概念弱化。

2.3 内存分析

>  内存中的对象

> 说明：程序中定义的Person类型的变量实际上是一个引用，它被存放在栈内存中，他指向实际的Person对象存放于堆内存中。

2.4 练习

```
//测试类
public class Test
{
    public static void main(String[] args)
    {
        //需求：开学了，王老师让学生小明，小花，小丽做自我介绍
        //姓名，年龄，爱好，来一段才艺展示
        /*
        老师类
        特征：姓名
        行为：让学生做自我介绍
        */
        //1.创建一个老师的对象
        Teacher wang  = new Teacher();
        wang.name = "王老师";
        wang.knowStudent();
    }
}

//老师类
public class Teacher
{
    String name;
```

```
    //认识学生
    public void knowStudent() {

        Student stu=new Student();
         stu.name = "小明";
        stu.age = 10;
        stu.hobby = "吹牛逼";
        stu.introduce();
        stu.dance();
         stu.sing();
    }
}

//学生类
public class Student
{
    String name;
    int age;
    String hobby;

    public void introduce() {
        System.out.println("我是" + name + "今年" +  age + "爱好：" + hobby);
    }

    public void dance() {
        System.out.println("跳一段广场舞");
    }

    public void sing() {
        System.out.println("来一段freeStyle");
    }
}
```

第三节：构造方法的定义

3.1 构造方法的定义

> 构造方法也叫构造器，是指当实例化一个对象（创建一个对象）的时候，第一个被调用的方法
>
> 语法：
>
> ```
> 访问权限修饰符 类名(参数列表) {
>             //方法体
> }
> ```
>
> 普通方法：
>
> ```
> 访问权限修饰符 其他的修饰符 返回值类型 方法名（参数列表） {
>
> }
> ```
>
> 注意：a.构造方法没有返回值类型,构造方法是在实例化对象的过程中自动调用的。
>
> b.如果不写构造方法，系统会默认为我们提供一个无参的构造方法，如果添加了构造方法，系统不再提供默认的构造方法。

3.2 构造方法的调用

```
//演示构造方法的使用
public class Test
{
    public static void main(String[] args)
    {
        //创建动物
        Animal dog=new Animal();
        //使用属性
        dog.color="黄色";
        dog.nickname="旺财";
        dog.age=3;
        //使用方法
        dog.eat();
    }
}

public class Animal {
    //颜色
    String color;
    //昵称
    String nickname;
    //年龄
    int age;
    //默认构造方法
    public Animal() {

    }

    //吃
    public void eat() {
        System.out.println(nickname+"大口大口吃东西");
    }
}
```

3.3 构造方法和普通方法的区别

> a.构造方法是在创建对象的过程中自动调用的，普通方法只能手动进行调用
> b.构造方法没有返回值类型【注意区别返回值void】，普通方法的返回值类型要么是确定的类型，要么为void
> c.系统会默认为我们提供一个无参的构造方法,普通方法只能手动添加
> d.构造方法的方法名称必须和对应的类名保持一致
> e.构造方法在创建对象的过程中就会执行，而且每个对象只执行一次，对于普通方法而言，只有在需要使用的时候才被执行，并且一个对象可以调用多次

3.4 构造方法重载

方法重载：同一个类中，方法名相同，参数列表不同

参数列表不同：个数不同，类型不同，顺序不同

和返回值修饰符无关

构造方法重载：同一个类中，构造方法名相同，参数列表不同

参数列表不同：个数不同，类型不同，顺序不同

方法名和类名相同。

默认构造方法：只能创建对象，不能做任何初始化操作，如果实现创建对象时初始化属性，需要添加带参的构造方法，初始化对象的属性。如果一个类中有多个构造方法，这就是构造方法重载。

```
//演示构造方法的重载
//测试类
public class Test
{
    public static void main(String[] args)
    {
        //直接赋值
        /*
        Dog maomao = new Dog();
        maomao.name = "毛毛";
        maomao.age = 3;

        maomao.lookHome();
        */

        //通过构造方法赋值
        Dog dahuang = new Dog("大黄",5);
        dahuang.lookHome();
    }
}

//实体类
public class Dog
{
    //成员变量
    String name;
    int age;

    //无参构造方法
    public Dog() {}
    //带参的构造方法，参数一般设置为和成员变量有关的参数
    public Dog(String n,int a) {
        //给成员变量赋值
        name = n;
        age = a;
    }
    /*
    public Dog(String n) {
        name = n;
    }
    */

    //成员方法
    public void lookHome() {
        System.out.println(name + "看家");
    }
}
```

3.5 练习

```
//测试类
public class Test
{
```

```
    public static void main(String[] args)
    {
        //场景：富二代王思聪开着新买的白色宝马在马路上奔跑，很自豪的向他的新女友炫耀车
        /*
        富二代类
        特征：姓名  钱的数量
        行为：开车，炫耀

        汽车类
        特征：颜色，品牌
        行为：奔跑

        女友类
        特征:姓名、年龄、颜值
        */
        //1.创建一个富二代的对象
        RichMan wang = new RichMan("王思聪",true);


        wang.drive();
        wang.show();
    }
}

/*
富二代类
        特征：姓名  钱
        行为：开车，炫耀车
*/
public class RichMan
{
    //成员变量
    String name;
    double money;

    //默认构造方法
    public RichMan() {

    }
    //带参构造方法
    public RichMan(String n,double m) {
        name = n;
        money = m;
    }

    //成员方法
    public void drive() {
        Car c=new Car();
        c.brand="宝马";
        System.out.println(name + "开着豪车" + c.brand);
    }
    //展示
    public void show() {
        Car c=new Car();
        c.brand="宝马";
        GirlFriend gf=new GirlFriend();
        gf.name="凤姐";
        System.out.println(name + "向" + gf.name + "炫耀豪车" + c.brand);
```

```
    }
}

/*
汽车类
        特征：颜色，品牌
        行为：奔跑
*/
public class Car
{
    //成员变量
    String color;
    String brand;

    //构造方法
    public Car() {}
    public Car(String c,String b) {
        color = c;
        brand = b;
    }

    //成员方法
    public void run() {
        System.out.println("一辆" + color + "的" + brand + "在奔跑");
    }
}

/*
女友类
        特征:姓名
*/
public class GirlFriend
{
    //成员变量
    String name;

    //构造方法
    public GirlFriend(){}
    public GirlFriend(String n) {
        name = n;
    }
}
```

第四节：this关键字

this：表示当前对象的引用。

4.1 this.属性

> **访问本类的成员变量**
>
> 作用：为了区分成员变量和形参变量名一样的情况
>
> 成员变量默认值:
>
> 引用类型：null
>
> 基本类型：byte short int ： 0

```
        long : 0L
    float: 0.0f
    double:0.0
        char :0
    boolean:false
```

4.2 this.方法

> 访问本类的其他方法

4.3 练习

```
//演示this的使用
public class Test
{
    public static void main(String[] args)
    {
        //
        Cat maomao = new Cat("毛毛",10);
        maomao.show1();


    }
}

public class Cat
{
    String name;//昵称
    int age;//年龄
    String color;//颜色

    //3.this()
    public Cat() {
        System.out.println("无参的构造方法被调用");
    }
    //1.this.属性
    public Cat(String name,int age,String color) {
        this.color = color;
        this.name=name;
         this.age=age;
        System.out.println("带参2构造方法被调用");
    }
    //2.this.方法
    //普通方法
    public void show1() {
        //在本类中调用方法时，this可以省略
        this.show2();
    }

    public void show2() {
    }
}
```

4.4 this()（了解）

> 访问本类中的其他构造方法
>
> 注意:
>
> (1) this(参数)只能用在构造方法中，必须是第一条语句
>
> (2) this(参数)只能调用一次

```
public class Test
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
public class Dog
{
     String name;
     int age;
     int num;
     String hobby;

    //提高代码的可读性，可维护性
    //构造方法
    public Dog() {
    }
    public Dog(String name) {
        this.name = name;
    }
    public Dog(int age) {
        this.age = age;
    }

    public Dog(String name,int age) {
        this.name = name;
        this.age = age;
    }
    public Dog(String name,int age,int num,String hobby) {
        this(name,age);
        this.num= num;
        this.hobby = hobby;
    }
}
```

总结

1 面向对象思想： 一种思维方式，着眼于找到一个具有特殊功能的具体个体，然后委托这个个体去做某件事情。

面向过程思想：一种思维方式，着眼于解决问题的过程和步骤。

2 类的概念 ： 一类具有相同属性和功能的实体的集合。

3 对象：一个具体特殊功能的个体。

4 类和对象的关系：类是对象的抽象或模板，对象是类的具体或实例。

5 定义类

```
public   class  Person{
    //成员变量 :   属性
    //方法    : 行为功能
}
```

6 创建对象

```
类名  对象名=new 类名();
//使用对象的属性和方法
对象名.属性名
对象名.方法名();
```

对象的内存分配

栈：比较小，存取速度块，特点：先进后出

堆：空间比较大，存储速度较慢。

方法区 (静态区，串池(常量池)，代码段)

7 构造方法：创建对象时调用的方法。

```
public  类名(){
    }
```

默认构造方法

带参构造方法

构造方法重载

8 this关键字 ：表示当前对象的引用

```
this.属性
this.方法
this();//调用其他的构造方法
```

默写

> 1.什么是方法?
>
> 2.定义方法的语法?

3.什么是方法重载?

作业

一：根据要求描述对象

1.写出厨房里面至少三个类，并创建对象运行。

2.写出教室里至少三个类，并创建对象运行。

3.写出动物园里至少三个动物类，并创建对象运行。

二：利用面向对象的思想描述下面的场景

1.小美在朝阳公园溜旺财【注：旺财是狗】

2.小明穿着白色的特步运动鞋在奥林匹克公园跑步

3.赵老师在讲台上讲课，小刚认真的听课做笔记

4.张阿姨和李阿姨在物美超市买红富士。

面试题

1.什么是面向对象？面向对象和面向过程的区别是什么

2.构造方法与普通方法之间的区别

3.this关键字的作用以及使用

4.成员变量和局部变量的区别：

1.定义的位置不同
    成员变量：定义于类中，作用于整个类
    局部变量：定义于方法或者语句中，作用于该方法或者该语句。
2.内存中出现的时间和位置不同
    成员变量：当对象被创建时，出现在堆内存当中。
    局部变量：所属的区间被运算时，出现在栈内存当中。
3.生命周期不同
    成员变量：随着对象的出现而出现，随着对象的消失而消失。
    局部变量：随着所属区间运算结束，它就被释放。
4.初始化值不同
    成员变量：成员变量因为在堆内存当中，所以他有默认的初始值
    局部变量：没有默认的初始值
5. 成员变量和局部变量名字相同，局部变量优先级高。就近原则