前提介绍

1. 之前内容总结

- 1.之前的内容比较零碎
- 2.之前的内容之间关联性不强，或者没有关联性
- 3.之前的内容需要我们理解的不多

2. 面向对象部分内容介绍

- 1.面向对象部分内容 细节 零碎的知识点也很多
- 2.面向对象部分内容 彼此之间关联性很强
- 3.面向对象部分内容 需要我们理解的非常多

3. 学习过程注意事项

- 1.多敲 多练 多记忆
- 2.多思考
- 3.多问

初识面向对象

1. 万物皆对象

> 自然界中的任何事物 我们都可以通过分析其特征和行为 将其描述的具体。

2. 特征---属性

> 特征：是指一类事物共有的信息 使用属性描述
>
> 比如：人 都有 姓名 年龄 身高 体重 等等

3. 行为---方法

> 行为：是指一类事物共有的动作 使用方法描述
>
> 比如：人都可以 吃饭 睡觉 学习 做运动 等等

4. 类和对象的关系

> **类是对象的抽象** ： 类是一个抽象的概念 比如 人类 动物类 电脑类 等等 我们只根据这些类是无法知道具体是哪个事物的 所以 类是抽象的概念
>
> **对象是类的具体** ： 对象是真真实实存在的 看得见 摸得着的一个实体 所以是具体的概念

5. 创建类和对象

> 学生类 类是对象的抽象 对象是类的具体
>
> 描述一类事物 从两个方面：特征和行为
>
> 特征--属性：姓名 年龄 身高 体重 学号 性别 班级 ……
>
> 行为--方法：学习 睡觉 吃饭 玩游戏 做运动
>
> 实例 ： 真真实实存在的一个个例 对象
>
> f-filed : 字段 属性 成员变量 实例属性 实例变量
>
> m-method : 方法 也就是行为
>
> 使用static修饰的方法 称之为 静态方法
>
> 没有使用static修饰的方法 称之为普通方法 也叫 实例方法

```java
package com.atguigu.test1;

/**
 * @author WHD
 * @description TODO
 * @date 2023/6/2 10:24
 *  学生类    类是对象的抽象  对象是类的具体
 *  描述一类事物  从两个方面：特征和行为
 *
 *  特征--属性：姓名   年龄  身高  体重  学号  性别  班级  ……
 *  行为--方法：学习  睡觉  吃饭  玩游戏  做运动
 *
 *  实例 ：  真真实实存在的一个个例   对象
 *  f-filed ： 字段  属性  成员变量  实例属性  实例变量
 *  m-method ： 方法  也就是行为
 *
 *  使用static修饰的方法  称之为  静态方法
 *  没有使用static修饰的方法  称之为普通方法  也叫  实例方法
 */
public class Student {
    String name;
    int age;
    double height;
    double weight;
    String studentNo;
    char sex;
    String gradeName;

    public void study(){
```

```
        System.out.println("学生在学习");
    }

    public void sleep(){
        System.out.println("学生在睡觉");
    }

    public void eat(){
        System.out.println("学生在吃饭");
    }


    public static void main(String[] args) {
        // 创建对象
        // 格式 类名 对象名 =  new 类名();
        Student stu1 = new Student();

        // 访问对象的属性
        // 格式
        // 赋值 对象名.属性名 = 值;
        // 取值 System.out.println(对象名.属性名);
        stu1.name = "赵四";
        stu1.age = 20;
        stu1.studentNo = "sz888";
        stu1.height = 188.5;
        stu1.weight = 75.8;
        stu1.sex = '男';
        stu1.gradeName = "三年二班";

        System.out.println(stu1.name);
        System.out.println(stu1.age);
        System.out.println(stu1.studentNo);
        System.out.println(stu1.height);
        System.out.println(stu1.weight);
        System.out.println(stu1.sex);
        System.out.println(stu1.gradeName);


        // 访问对象的行为
        // 格式 对象名.方法名();
        stu1.study();
        stu1.eat();
        stu1.sleep();
    }



}
```

6. 实例变量

描述	局部变量	实例变量
定义位置	定义在方法体内	定义在类中
作用范围	离当前变量最近的大括号以内	整个类中
默认值	没有默认值	有默认值，与数组相同
重名问题	重合的作用范围以内，不能重名	可以与局部变量重名，就近原则使用
存储位置	基本数据类型存在栈中 引用数据类型名字在栈，值在堆	全部存在堆中，因为对象保存在堆中
生命周期	随着方法的入栈而生效，随着方法的出栈而死亡	随着对象的创建而生效 随着对象被垃圾回收而死亡。

```java
package com.atguigu.test2;

/**
 * @author WHD
 * @description TODO
 * @date 2023/6/2 10:24
 *  学生类    类是对象的抽象  对象是类的具体
 *  描述一类事物  从两个方面：特征和行为
 *
 *  目前问题：当前使用对象访问方法执行  没有显示对象的  姓名  信息  所以不太直观
 *  我们应该将  对象的  姓名信息进行展示
 *
 *  实例变量
 *  定义位置：定义在类中
 *  作用范围：本类中   离当前变量最近的大括号之内
 *  默认值：有默认值  与数组相同
 *  重名问题：可以与局部变量重名
 *
 *
 *
 */
public class Student {
    String name;
    int age;
    double height;
    double weight;
    String studentNo;
    char sex;
    String gradeName;

    public void study(){
        System.out.println(name + "学生在学习");
    }

    public void sleep(){
        System.out.println(name + "学生在睡觉");
    }
```

```java
public void eat(){
    System.out.println(name + "学生在吃饭");
}


public void printInfo(){
    System.out.println("我的名字是: " + name);
    System.out.println("我的年龄是: " + age);
    System.out.println("我的性别是: " + sex);
    System.out.println("我的身高是: " + height);
    System.out.println("我的体重是: " + weight);
    System.out.println("我的班级是: " + gradeName);
    System.out.println("我的学号是: " + studentNo);

}



public static void main(String[] args) {
    // 创建对象
    // 格式 类名 对象名 =  new 类名();
    Student stu1 = new Student();

    // 访问对象的属性
    // 格式
    // 赋值 对象名.属性名 = 值;
    // 取值 System.out.println(对象名.属性名);
    stu1.name = "赵四";
    stu1.age = 20;
    stu1.studentNo = "sz888";
    stu1.height = 188.5;
    stu1.weight = 75.8;
    stu1.sex = '男';
    stu1.gradeName = "三年二班";

    System.out.println(stu1.name);
    System.out.println(stu1.age);
    System.out.println(stu1.studentNo);
    System.out.println(stu1.height);
    System.out.println(stu1.weight);
    System.out.println(stu1.sex);
    System.out.println(stu1.gradeName);


    // 访问对象的行为
    // 格式 对象名.方法名();
    stu1.study();
    stu1.eat();
    stu1.sleep();

    System.out.println("-----------------------------------------------------");
```

```
        // 创建对象
        // 格式 类名 对象名 =  new 类名();
        Student stu2 = new Student();

        // 访问对象的属性
        // 格式
        // 赋值 对象名.属性名 = 值;
        // 取值 System.out.println(对象名.属性名);
        stu2.name = "广坤";
        stu2.age = 21;
        stu2.studentNo = "sz666";
        stu2.height = 188;
        stu2.weight = 75;
        stu2.sex = '男';
        stu2.gradeName = "三年二班";

        System.out.println(stu2.name);
        System.out.println(stu2.age);
        System.out.println(stu2.studentNo);
        System.out.println(stu2.height);
        System.out.println(stu2.weight);
        System.out.println(stu2.sex);
        System.out.println(stu2.gradeName);


        // 访问对象的行为
        // 格式 对象名.方法名();
        stu2.study();
        stu2.eat();
        stu2.sleep();

        System.out.println("-------------------------------------------------------");
        stu1.printInfo();


        System.out.println("-------------------------------------------------------");
        stu2.printInfo();

        System.out.println("-------------------------------------------------------");

        Student stu3 = new Student();
        stu3.printInfo();

    }

}
```

7. 构造方法

> 构造方法：用于创建对象的特殊方法
>
> 名称与类名完全相同。

> 没有返回值类型。
>
> 创建对象时，触发构造方法的调用，不可通过句点手动调用。
>
> 格式： 访问修饰符 + 类名(){}

> 如果没有在类中显示定义构造方法，则编译器(JVM)默认提供无参构造方法。
>
> 但是 如果书写了有参构造方法 无参构造将会被覆盖 如需使用 必须 显式 书写
>
> 根据构造方法重载规则 我们在一个类中可以写很多构造方法
>
> 实际开发中 我们针对一个类 只写两个构造方法 一个无参构造 一个全参构造 即可
>
> 构造方法重载：同一个类中的构造方法 参数列表不同

```
package com.atguigu.test3;

/**
 * @author WHD
 * @description TODO
 * @date 2023/6/2 10:24
 *  学生类
 *
 *  构造方法：用于创建对象的特殊方法
 *       名称与类名完全相同。
 *       没有返回值类型。
 *       创建对象时，触发构造方法的调用，不可通过句点手动调用。
 *
 *  格式 ：      访问修饰符 + 类名(){}
 *  普通方法：  访问修饰符 +  返回值类型 + 方法名(){}
 *
 *  如果没有在类中显示定义构造方法，则编译器(JVM)默认提供无参构造方法。
 *  但是 如果书写了有参构造方法 无参构造将会被覆盖 如需使用 必须  显式  书写
 *
 *
 *  构造方法重载：同一个类中的构造方法 参数列表不同
 */
public class Student {
    String name;
    int age;
    double height;
    String studentNo;
    char sex;

    public void study(){
        System.out.println(name + "学生在学习");
    }

    public void sleep(){
        System.out.println(name + "学生在睡觉");
    }

    public void eat(){
```

```
        System.out.println(name + "学生在吃饭");
    }

    public void printInfo(){
        System.out.println("我的名字是: " + name);
        System.out.println("我的年龄是: " + age);
        System.out.println("我的性别是: " + sex);
        System.out.println("我的身高是: " + height);
        System.out.println("我的学号是: " + studentNo);

    }

    public Student(){
        System.out.println("Student类无参构造方法执行了");
    }

    public Student(String n){
        name = n;
    }

    public Student(String n,int a,double h,char s,String sNo){
        name = n;
        age = a;
        height = h;
        sex = s;
        studentNo = sNo;
    }

    public static void main(String[] args) {
        Student stu1 = new Student();
        stu1.printInfo();

        System.out.println("----------------------------------------");

        Student stu2 = new Student("赵四", 20, 175, '男', "sz8956");
        stu2.printInfo();

        System.out.println("----------------------------------------");

        Student stu3 = new Student("广坤");
        stu3.age = 20;
        stu3.height = 185;
        stu3.sex = '男';
        stu3.studentNo = "sz8945";

        stu3.printInfo();



    }
```

```
}
```

8.this关键字

> this关键字 表示当前正在使用的对象
>
> 可以访问本类中的
>
> 属性 this.属性名
>
> 方法 this.方法名(实参列表)
>
> 构造方法 this(实参列表) this访问本类的构造方法 只能在本类构造的第一句

```
package com.atguigu.test4;

/**
 * @author WHD
 * @description TODO
 * @date 2023/6/2 10:24
 *  学生类
 *  this关键字 表示当前正在使用的对象
 *  可以访问本类中的
 *  属性  this.属性名
 *  方法  this.方法名(实参列表)
 *  构造方法 this(实参列表) this访问本类的构造方法 只能在本类构造的第一句
 *
 */
public class Student {
    String name;
    int age;
    double height;
    String studentNo;
    char sex;

    public void printName(){
        System.out.println("我的名字是: " + name);
    }

    public void printAge(){
        System.out.println("我的年龄是: " + age);
    }

    public void printInfo(){
        this.printName(); // printName();
        printAge();
        System.out.println("我的性别是: " + sex);
        System.out.println("我的身高是: " + height);
        System.out.println("我的学号是: " + studentNo);
    }
```

```
    public Student(){
        System.out.println("Student类无参构造方法执行了");
    }
    public Student(String name){
        this.name = name;
    }


    public Student(int age){
        this.age = age;
    }


    public Student(String name, int age){
        this(name);




    }
    public Student(String name, int age, double height){
        this(name,age);
        this.height = height;
    }
    public Student(String name, int age, double height, char sex){
        this(name,age,height);
        this.sex = sex;
    }

    public Student(String name, int age, double height, char sex, String studentNo){
        this(name,age,height,sex);
        this.studentNo = studentNo;
    }

    public static void main(String[] args) {
        Student stu1 = new Student("赵四");

        stu1.printInfo();

        System.out.println("----------------------------------------");

        Student stu2 = new Student("广坤", 20, 185, '男', "sz897845");

        stu2.printInfo();



    }
```

```
}
```