Java

**一.简介**

Java用于计算机开发程序，为了让计算机去执行某些操作，而编写的一系列有序指令的集合，这就是计算机程序，Java就是编写程序的一种工具，Java是目前企业中开发运用最广泛的

作者是詹姆斯.高斯林 James.Gosling

必须要使用JDK工具包

**option+/ 内容提示**

**command+option+方向下 复制当前行**

**command+shift+o 快速导包**

**command+shift+f 格式化代码**

**IDE的使用：**

**新建的是Module**

**二.Java平台的三个版本**

（1）JavaSE

Java Platform Standard Edition标准版，包含java语言的基础核心，用于开发桌面应用程序，如QQ，记事本，计算器等

（2）JavaEE

Java Platform Enterprise Edition 企业版，包含了JavaSE，用于开发企业应用程序，如web网站，分布式应用

（3）JavaME

Java Platform Micro Edition 微缩版/移动版，用于开发移动设备和嵌入式设备，比如指纹机和手机游戏

**三.特点**

（1）跨平台，各种平台（mac windows等）都可以运行

（2）安全

（3）简单，语法简洁

（4）健壮，可靠性高

（5）面向对象，面向对象的程序设计语言

**四.JDK、JRE、JVM**

（1）JDK（Java开发工具包） Java Development Kit

（2）JRE（Java运行时的环境）Java Runtime Environment

（3）JVM（Java虚拟机）Java Virtual Machine

Java的跨平台就是通过在不同的平台安装相应的JVM来实现的

（4）三者之间的关系JDK包含JRE，JRE包含JVM

**五.工作原理**

源文件.java---编译器（compiler）---字节码文件.class---解释器（interpreter）---010010101...---计算机显示

**六.环境配置**

java -version

javac -version

**七.使用记事本开发**

第一步：编写代码文件 demo.java

public class Demo01 //这里要和文件名一致

{

public static void main(String[]args)

{

System.out.println("welcom");

}

}

第二步：在终端进入文件所在的文件夹，将源代码为字节码文件demo.class

hufei@macdeMacBook-Pro-2 java % **javac demo01.java**

此时会在当前文件夹产生一个同名的.class文件

第三步：运行字节码文件

hufei@macdeMacBook-Pro-2 java % java demo01

在终端打印输出welcom

**八.基本用法**

（1）输出信息

System.out.println(); 打印输出后换行

System.out.print();打印输出后不换行

（2）文件取名大驼峰式命名

HelloWord.java

（3）基本语法

public class Demo01 //这里要和文件名一致

{

public static void main(String[]args)

{

System.out.println();

}

}

**九.转义符 反斜杠 \**

由于某些特殊字符没办法打印输出，所以必须通过转义,反斜杠与后面相邻的字符组成一个新含义的字符

System.out.println("TOM say:\"welcom\""); 转义双引号这样就能打印双引号

\n 表示换行

\t表示制表符Tab（多个空格）

\\表示反斜杠\

**十.注释**

不会被执行，有三种注释

（1）单行注释//单行注释（2）多行注释/\* 多行注释 \*/（3）文档注释写在开头用于项目说明/\*\* 文档注释 \*/

文档注释可以提取

javadoc命令可以自动提取文档注释内容，生成文档

javadoc Demo01.java

生成一个index.html

**十一.编码规范**

（1）**类名必须要和文件名相同，通常采用所有单词首字母大写**

（2）语句必须以分号结尾

（3）所有标点符号必须是英文符号

（4）Java是区分大小写的

（5）所有命名使用中文、空格、特殊符号

（6）命名可以使用数字但不能以数字开头

（7）一行写一条语句（分号表示一句写完

（8）合理的代码缩进

**十二.项目创建Eclipse**

（1）创建JAVA项目

File--New--Project--Java Project--取项目名字Project name

（2）创建包

右击src--New--Package--取包的名Package name

（3）创建源代码文件并编写

右击包--New--Class--取类的名字Class name

（4）运行程序（会自动将源代码编译成字节码存在bin目录下）

空白处右击选择Run As--Java Application

**十二.容量是单位**

bit(b)

1byte(B)=8b

1KB=1024B

MB、GB、TB、PB、EB

**十三.字符集**

计算机底层使用二进制01，所有数据在计算机底层都是以二进制保存

ASCII

**十四.变量和数据类型**

（1）变量

变量的三要素：1.变量名2.变量类型3.变量值

（2）变量基本使用方法

1.定义变量，指定变量名和类型

**int** age;

2.变量赋值

age=18;

3.调用变量

调用变量名即可

int age;

age=18;

System.out.println("年龄是"+age);

（3）可是同时定义多个变量

int a=0,b;

（4）变量的命名规则

1.只能由数字、字母、下划线、$组成

2.不能以数字开头

3.使用英文单词

4.小驼峰式J

5.严格区分大小写

6.不能使用Java关键字

**十五.常用的八种基本数据类型和引用数据类型String**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| byte | 整数 | 占1个字节（共8位） | 字节型 | 取值范围[-128,127] | byte num1=5; |
| short | 整数 | 占2个字节 | 短整型 | 取值范围[-32728,32767] | short num2=26; |
| int | 整数 | 占4个字节 | 整数型 | 取值范围约21亿（最常用） | int num3=12; |
| long | 整数 | 占8个字节 | 长整型 | 定义时必须以l或者L结尾 | long num4=666l; |
| float | 小数 | 占4个字节（单精度） |  | 需要以f或F结尾 | float height=128.1f; |
| double | 小数 | 占8个字节（双精度） |  | 默认将会小数作为double类型处理（常用） | double weight=120.3; |
| char | 字符 | 占2个字节 |  | 表示单个字符需要使用单引号 | char sex='男'; |
| boolean | 布尔 | 占1个字节 |  | 表示真还是假（true或false） | boolean isOK=false; |
| String | 字符串 | 长度不确定 |  | 表示一串字符，需要用双引号 | String name="tom"; |

**十六.类型转换**

（1）自动类型转换（隐式转换）

1.数据类型要兼容

int a=3;

double b=a;

2.目标类型要大于源类型如int转化long

（2）强制类型转换（显式转换）

类型兼容

语法：(目标类型)数据

double e=12.6;

int f=(int)e;

可能让数据发生丢失进而丢失数据精度

（3）char与int类型的转换

char转换为int是可以的，结果是char字符对应的ASCII码

但是int是不能隐式转换为char，可以强制转化，结果就是相对ASCII码对应的char值

！char z=97;是不会报错的（整数和整型是不同概念）

**十七.获取用户的输入Scanner（扫描仪）**

（1）需要导入Scanner

(写在public class上面)

**import java.util.Scanner;**

（2）创建一个Scanner对象名字叫input

**Scanner input=new Scanner(System.*in*);**

（3）使用Scanner获取输入的数据，使用next()方式获取字符串数据，存到字符串变量name中

**String name=input.next();**

**如下：**

**package** second;

**import** java.util.Scanner;

**public** **class** demo04 {

**public** **static** **void** main(String[]args) {

Scanner input =**new** Scanner(System.***in***);

System.***out***.print("请输入名字：");

String name=input.next();

System.***out***.println("名字是"+name);

}

}

注：使用next()获取字符串数据

使用nextInt()获取整数

使用nextDouble()获取小数

System.***out***.print("请输入年龄：");

**int** age=input.nextInt();

System.***out***.println("年龄是"+age);

System.***out***.print("请输入身高：");

**double** height=input.nextDouble();

System.***out***.println("身高是"+height);

**十八.运算符**

包括算数运算符、关系运算符、赋值运算符、逻辑运算符、条件运算符、位运算符

（1）算数运算符

|  |  |
| --- | --- |
| 运算符 | 含义 |
| + | 加(字符串也可以相加，作为拼接) |
| - | 减 |
| \* | 乘 |
| / | 除 |
| % | 求余数 |
| ++ | 自增1 |
| -- | 自减1 |

（2）比较运算符，结果是boolean

|  |
| --- |
| == |
| < |
| > |
| <= |
| >= |
| != |

也可以用于字符比较，比较ASCII码

（3）赋值运算符

|  |
| --- |
| = |
| += |
| -= |
| /= |
| \*= |
| %= |

（4）逻辑运算符

|  |
| --- |
| && |
| || |
| ! |

逻辑运算是一种短路运算

（5）条件运算符（三目运算符）

条件?表达式1:表达式2

**十九.进制**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 进制种类 | 单词 | 范围 | 引导符号 | 示例 |
| 十进制 | decimal | 0-9 | 无 | int a = 10 |
| 二进制 | binary | 0-1 | 0b或0B | int a = 0b10 |
| 八进制 | octonary | 0-7 | 0 | int a = 010 |
| 十六进制 | hex | 0-9、a-f或A-F | 0x或0X | int a =0x10 |

**二十.二进制**

（1）二进制分两种

无符号二进制，只能存储正整数

有符号二进制，可以存正数也可以存负数，最高位为符号位，0表示正数，1表示负数

正数5表示为 00000000 00000000 00000000 00000101

负数5表示为 10000000 00000000 00000000 00000101

（2）原码、补码、反码

1.原码

原始的二进制值

2.反码

正数：反码与原码相同，如5的反码00000000 00000000 00000000 00000101

负数：符号位不变，其余各自取反，如-5的反码11111111 11111111 11111111 11111010

3.补码

正数：补码与原码相同，如5的补码00000000 00000000 00000000 00000101

负数：反码+1，如-5的补码11111111 11111111 11111111 11111011

**计算机的内部用到是补码来储存和表示值**

System.out.println(0b11111111111111111111111111111011); 输出为-5

**二十一.位运算（与逻辑不一样）**

针对二进制的位进行运算，其他进制需要将值转换为二进制后进行按**位运算**

|  |  |
| --- | --- |
| & | 按位与 |
| | | 按位或 |
| ~ | 按位非 |
| ^ | 异或 |
| >> | 右移 |
| << | 左移 |
| >>> | 无符号右移 |

（1）&：两位都是1则为1，否则为0

int a = 5; //二进制00000000 000000000 00000000 00000101

int b = 8; //二进制00000000 000000000 00000000 00001000

按位与00000000 000000000 00000000 00000000

a&b上下按位与后都是0，所以结果是0

（2）|：只要有一个1，则为1，否则为0

int a = 5; //二进制00000000 000000000 00000000 00000101

int b = 8; //二进制00000000 000000000 00000000 00001000

按位或00000000 000000000 00000000 00001101

a|b最终输出的结果就是13

（3）~：如果是1则为0，如果是0则为1

int a = 5; //二进制00000000 000000000 00000000 00000101

按位非11111111 11111111 11111111 11111010（补码计算）

转反码11111111 11111111 11111111 11111001

转原码10000000 00000000 00000000 00000110

~a最终输出的结果 -6

（4）^：如果两位相同则为0，否则则是1

int a = 5; //二进制00000000 000000000 00000000 00000101

int b = 8; //二进制00000000 000000000 00000000 00001000

异或00000000 000000000 00000000 00001101

a^b结果是13

（5）<<：将二进制的所有位向左移指定的位数，空出来末尾补零

int a = 5; //二进制00000000 000000000 00000000 00000101

左移00000000 000000000 00000000 00010100（a<<2就是向左移两位）

a<<2结果是20

（6）>>：将二进制的所有位向右移指定的位数，高位补符号位（负补1，正补0）

int a = 5; //二进制00000000 000000000 00000000 00000101

右移00000000 000000000 00000000 00000001（a>>2就是向右移两位）

a>>2结果是1

（7）>>>：将二进制的所有位向右移指定的位数，高位补0

int a = 5; //二进制00000000 000000000 00000000 00000101

无符号右移00000000 000000000 00000000 00000001（a>>>2就是无符号向右移两位）

a>>>2结果是1

**二十二.Java里的方法**

**修饰符 返回值类型 方法名(参数){**

**方法体**

**}**

1.修饰符：是对方法进行修饰的（目前固定写public static）

2.返回值类型：方法执行完之后返回值的类型，如果没有返回值则指定为void

3.方法名：小驼峰命名（一般使用单词）

4.参数：执行方法时需要的信息，参数由参数类型和参数名组成（可以没有参数）

**方法写在类里面class不是写在main方法里**

**main方法是程序的入口，程序在运行时会自动调用**

**方法的调用在面方法里**

**package** no4;

**public** **class** demo01 {

**public** **static** **void** main(String[] args) {

*sums*(); //调用方法

}

**自定义方法(无参数和无返回值)**

**public** **static** **void** sums() {

**int** sum = 0;

**for**(**int** i=0;i<100;i++) {

sum+=i;

}

System.***out***.println(sum);

}

}

**任意两个数之间的和(有参数)**

**public** **static** **void** sumAny(**int** num1,**int** num2) {

**int** sum =0;

**if**(num1<num2) {

**for**(**int** i=num1;i<=num2;i++) {

sum+=i;

}

}**else** {

**for**(**int** i=num2;i<=num1;i++) {

sum+=i;

}

}

System.***out***.println(sum);

}

**返回值return，当有返回值时返回值的类型就不能写void**

**public** **class** demo02 {

**public** **static** **void** main(String[] args) {

**int** result =*sumAny*(1, 100);

System.***out***.println(result);

}

**public** **static** **int** sumAny(**int** num1, **int** num2) {

**int** sum = 0;

**if** (num1 < num2) {

**for** (**int** i = num1; i <= num2; i++) {

sum += i;

}

} **else** {

**for** (**int** i = num2; i <= num1; i++) {

sum += i;

}

}

**return** sum;

}

}

方法之间可以互相调用，在同一个类中可以直接使用**方法名()**调用

在不同的类中需要使用**类名.方法名()**调用

0-99之间的整数System.***out***.println((**int**)(Math.*floor*(Math.*random*()\*100)));

**二十二.递归**

自己不断调用自己（一定要有退出的条件）

**public** **static** **int** cale(**int** x,**int** y) {

**if**(y==0) {

**return** 1;

}

**return** x\**cale*(x,y-1);

}

**二十三.程序调试Debug**

步骤：1.设置断点（程序暂停）

认为有问题的地方，右击行序号，点击Toggle BreakPoint

2.单步执行

右击空白Debug AS执行，点击yes后可以进入Debug区域

3. F6:单步执行

F5:单步跳入执行（会进入方法）

（界面上面也有按键）

**二十四.数组（与js[ ]不同,java写在{ }里）**

数组是**必须是相同数据类型**的一组数据

1.声明一个变量就是在内存中划出一块空间

2.声明一个数组就是在内存中划出一连串空间

3.数组的四要素：数组名、数组中的元素、元素的类型、元素的下标（数组元素的编号，索引，从0开始）

4.数组的基本用法

(1).声明数组，指定元素的类型

数据类型[ ] 数组名称; **int[ ] nums; (也可以String names[ ];)**

数据类型 数组名称;

(2).分配空间，指定数组的大小/长度

数组名称 = new 数组类型[数组长度]; **nums=new int[5];**

(3).数组赋值，向数组中添加元素

数组名称[下标]=值; **nums[0] = 1;**

(4).访问数组

数组名称[下标];

(5).简化写法(赋值时不能指定长度，长度由元素个数决定)

**String[ ] names = new String[ ]{** **"Tom","Jack","Bob"};**

**再简写：String[ ] names = {** **"Tom","Jack","Bob"};**

5.数组长度 names.length 数组的长度一旦指定，则无法被更改

**二十五.动态赋值**

循环的接收用户输入的数据，动态为数组赋值

public static void main(String[] args) {

Scanner input= new Scanner(System.in);

String[]names=new String[4];

for(int i = 0;i<names.length;i++) {

System.out.println("请输入第"+"个名字");

names[i]=input.next();

}

for(int i = 0;i<names.length;i++) {

System.out.println(names[i]);

}

}

**二十六.遍历集合操作 (foreach循环 ,但没有foreach关键字)**

**for(元素类型 循环变量:集合){**

**}**

int[] numbers= {1,2,3,4,5,6,7,9};

for(int number:numbers) {

System.out.println(number);

}

Arrays类的toString()方法，将数组变成字符串，数组元素以逗号隔开，并放在一个中括号里，需要调用Arrays类

Arrays.toString(numbers)

**二十七.默认值**

1.局部变量是没有默认值的，必须赋值才能用

布尔值，int，String都是没有默认值的

2.数组中元素的默认值

**int**[] numbers=**new** **int**[3];

System.***out***.println(numbers[2]);//0

String[] names = **new** String[3];

System.***out***.println(names[1]);//null

boolean的默认值是false

double的默认值是0.0

3.ArrayIndexOutOfBoundsException数组下标越界异常

**4.以下写法不支持**

int[] numbers=new int[3];

numbers={1,2,3};

只能通过numbers[0]=1;numbers[1]=2;numbers[2]=3;这种写法

**二十八.数据类型**

基本数据类型（8种）：byte short int long float double boolean char

引用数据类型：除了基本其他都是（除了null）如String，数组，类，接口...

数据类型 变量名=new 数据类型( )

数组特殊

int[ ] nums = new int[5];

字符串特殊

String name = “xxxxx”;  
null特殊

引用类型的默认值是null

正常的Scanner input = new Scanner();

**二十九.Arrays类**

（1）toString

（2）sort排序

（3）fill(nums,val)重新填充数组成同一个值，用于归零

**三十.不定长参数**

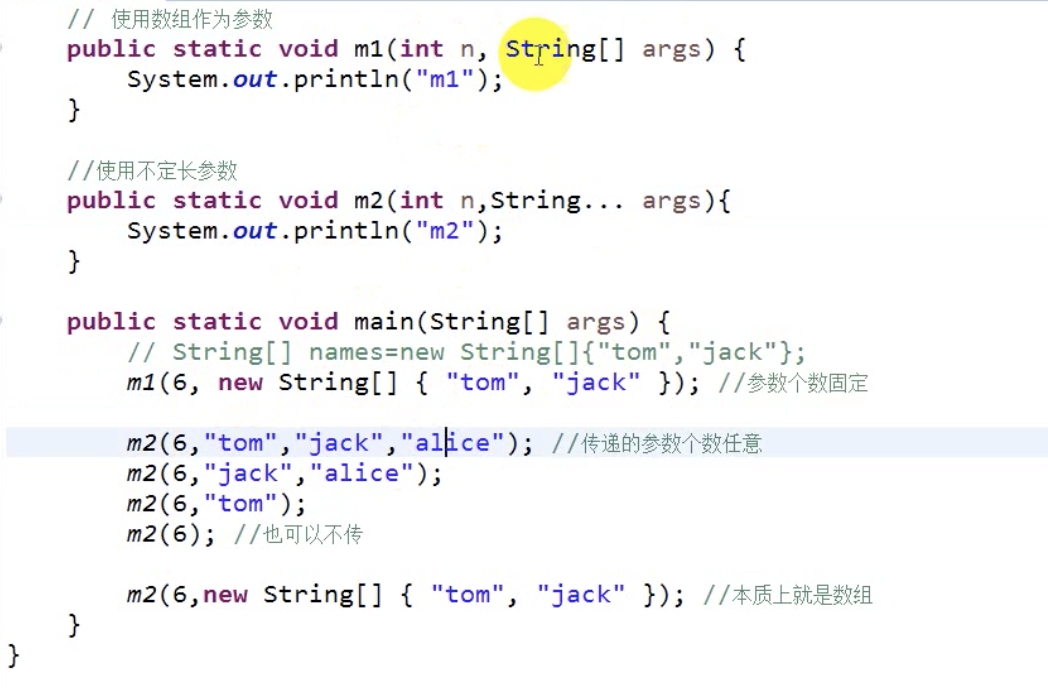
数据类型...就表示不定长参数

1.只能作为方法的最后一个参数

String...args

2.参数的个数任意

3.本质上就是特殊的数组，只能作为方法形参使用



**三十一.面向对象OOP Object Oritented Programming**

**1.创建类**

**package** oopDemo;

// 创建类

// 1.定义类，里面写属性和方法，定义类就是创建可一个数据类型，类名是大驼峰

**public** **class** Car {

// 2.添加属性

String name;

String color;

**int** speed;

**double** price;

// 3.添加方法

**public** **void** run() {

System.***out***.println(name+"车型颜色是"+color+"价格是"+price+"万");

}

}

**2.创建对象**

**package** oopDemo;

**public** **class** demo01 {

**public** **static** **void** main(String[] args) {

System.***out***.println("面向对象");

// 类是具有相同特征和行为的一组对象的抽象

// 对象就是累的具体

// 面向对象的产生，如统计路口通过的车辆品牌，价格，颜色的统计

// 创建对象

Car car1=**new** Car();

car1.color="red";

car1.name="BMW";

car1.price=123.4;

car1.speed=110;

}

}

**3.访问属性**

// 访问属性

System.***out***.println(car1.color);

**4.调用方法**

car1.run();

**5.注意**

**package** oopDemo;

// 一个文件中可以有多个类，但是public修饰的类只允许有一个，且这个类名要和文件名一样，开发中一个文件一个类

**public** **class** Car {

String name;

String color;

**int** speed;

**double** price;

**public** **void** run() {

System.***out***.println(name+"车型颜色是"+color+"价格是"+price+"万，车速"+speed+"迈");

}

}

// 定义多个类

**class** Pet{

String name;

}

**6.this关键字**

**package** oopDemo;

/\*

\* 1.全局变量和局部变量

\* 全局变量：在类中定义的变量，成为全局变量，也就是类里的属性，在整个类中都可以使用，而且有默认值

\* 局部变量：除了全局之外的都叫局部，如类中的方法里定义的变量，只能局部访问，没有默认值，必须先赋值，才能使用

\* 局部变量可以和全局变量同名，局部变量优先

\* 方法的形参也是局部变量

\* 可以用this关键字来解决全局和局部同名的问题

\* \*/

/\*

\* 2.this的用法

\* this表示当前对象

\* 只能在类的方法中使用，表示方法的调用者，即调用该方法的对象，谁调用指向谁

\* 同名时，用this.来访问全局变量

\* \*/

**public** **class** thisKeyWords {

**public** **static** **void** main(String[] args) {

Account xx = **new** Account();

xx.money = 10000.0;

xx.nameC();

}

}

**class** Account {

// 全局变量

String name = "tom";

**double** money;

**public** **void** nameC() {

// 局部变量

String name = "alice";

System.***out***.println(name); // alice 访问局部变量

System.***out***.println(**this**.name); // tom 访问全局变量 this代表xx这个对象

}

}

**7.static关键字**

**（1）静态变量(静态属性)**

**package** oopDemo;

/\*

\* static修饰的变量和方法就是静态变量和静态方法

\* 1.静态变量 也叫类变量，属于类的所有对象所共有

\* 2.非静态变量就是实例变量

\* \*/

**public** **class** staticKeyWords {

**public** **static** **void** main(String[] args) {

Student student1 = **new** Student();

// 静态变量的访问

// 对象名.属性名或者类名.属性名（推荐)

student1.name = "jack";

// student1.hobby = "sleep";

Student.*hobby* = "sleep";

}

}

**class** Student {

String name;// 实例属性

**static** String *hobby*;// 静态变量属于整个类

}

**（2）静态方法**

**package** oopDemo;

/\*

\* 1.静态方法 也叫类方法，属于类的所有对象所共有

\* 2.非静态方法就是实例方法

\* \*/

**public** **class** staticKeyWords {

**public** **static** **void** main(String[] args) {

Student student1 = **new** Student();

student1.show();

// 静态方法的访问

// 对象名.方法名或者类名.方法名（推荐)

Student.*print*();

}

}

**class** Student {

// 实例方法

**public** **void** show() {

System.***out***.println("实例方法");

}

// 静态方法

**public** **static** **void** print() {

System.***out***.println("静态方法");

}

}

// 注意：1.静态方法中只能访问静态变量和静态方法，2.实例方法中可以访问静态变量和静态方法

// 注意：写在方法里的是变量，写在类里的是属性

**8.方法的重载**

**package** oopDemo;

/\*

\* 方法的重载

\* 一个类中包含两个或两个以上同名的方法，但其参数不同

\* 特征：1.在同一个类中

\* 2.方法名一样

\* 3.参数列表要有不同（参数个数，参数类型，参数顺序）

\* 4.与修饰符，返回值类型，形参名无关

\* 优点：调用时会很方便，会根据参数的不同，会调用相应的重载方法

\* 作用：同一种功能的多种实现方式

\* 例如：输出时的print()方法，里面可以传不同类型的参数，但方法名都是print

\* \*/

**public** **class** overloadOfMethods {

**public** **static** **void** main(String[] args) {

System.***out***.println("方法的重载");

Calculator c =**new** Calculator();

System.***out***.println(c.sum(3,4));

System.***out***.println(c.sum(3,4,5));

}

}

**class** Calculator{

// 两个数的和

**public** **int** sum(**int** num1,**int** num2) {

**return** num1+num2;

}

// 三个数的和

**public** **int** sum(**int** num1,**int** num2,**int** num3) {

**return** num1+num2+num3;

}

}

**9.对象数组**

**package** oopDemo;

**import** java.util.Scanner;

//对象数组

**public** **class** arrayObjectText {

**public** **static** **void** main(String[] args) {

/\*

\*Car[] cars = new Car[3]; // 定义一个装Car类型的数组cars

\*Car car1 = new Car();

\*car1.color = "red";

\*car1.name = "BMB";

\*car1.price = 120.1;

\*car1.speed = 100;

\*cars[0] = car1;

\*cars[1] = new Car(); // !必须要实例化

\*cars[1].color="blue";

\*/

Car[] cars = **new** Car[3];

*inputInfo*(cars);

*displayInfo*(cars);

}

// 定义方法录入车辆信息

**public** **static** **void** inputInfo(Car[] cars) {

Scanner input = **new** Scanner(System.***in***);

**for** (**int** i = 0; i < cars.length; i++) {

cars[i] = **new** Car();// 第一步：实例化Car对象

System.***out***.println("=========请输入第" + (i + 1) + "辆车的信息=========");

System.***out***.println("请输入车的名字");

cars[i].name = input.next();

System.***out***.println("请输入车的颜色");

cars[i].color = input.next();

System.***out***.println("请输入车的价格");

cars[i].price = input.nextDouble();

System.***out***.println("请输入车的时速");

cars[i].speed = input.nextInt();

}

}

// 展示循环输出

**public** **static** **void** displayInfo(Car[] cars) {

System.***out***.println("名字\t颜色\t价格\t时速");

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

**for** (Car car : cars) {

System.***out***.println(car.name + "\t" + car.color + "\t" + car.price + "\t" + car.speed);

}

}

}

**10.构造函数**

**package** oopDemo;

//构造函数/构造方法/构造器

//用于创建和初始化对象

//特点：方法名要和类名相同,不允许有返回值，也不能void，当创建类的构造对象，会被自动调用，一个类可以有多个构造方法（方法的重载）

//如果没有写构造方法，系统会自动加一个无参的构造方法

//但是如果写了有参的构造方法，没写无参的，那么调用无参的就会报错（建议加上一个无参）

**public** **class** constructionMethod {

**public** **static** **void** main(String[] args) {

// Dog dog1 = new Dog();//1

Dog dog = **new** Dog("tom", 20);// 2 赋值后就可以调用构造函数

}

}

//

**class** Dog {

String name;

**int** age;

**public** **void** run() {

System.***out***.println("奔跑");

}

// 构造方法

**public** Dog() {

System.***out***.println("无参构造方法");// 1自动调用

}

**public** Dog(String name, **int** age) {// 2自动调用

**this**();// 必须写在第一行，除了注释

**this**.age = age;

**this**.name = name;

System.***out***.println("有参构造方法");

}

}

**//this(参数)在构造方法里调用另外的构造方法，必须写在第一行**

**11.包**

**package** oopDemo;//声明包

//import java.util.Scanner;//导包

**import** java.util.\*;//不建议

//包的作用

//1.区分同名的类，不同包下可以有相同类

//2.类多时方便管理

//3.便于访问控制

//声明包

//用来指定当前类所在的包

//声明写在第一行

//语法：package 包名

//包的命名

//域名的反向网址

//如百度翻译:fanyi.baidu.com

//包名:com.baidu.fanyi

//包名统一使用小写字母

//不能以点号开头结尾

//导入包

//import 包名.类名;

//import 包名.\*;导入全部类

//写在package下面

//常用的包

//java.lang 比如System、String、Math等该宝会自动被导入

//java.util 包含实用的工具类，比如Scanner、Arrays等

//java.io 包含输入和输出的类

//java.sql 包含数据库

//java.net 网络编程

//java.awt 图形窗口

//注意定义类必须制定包名，严禁将类放在src下

**public** **class** packageStudent {

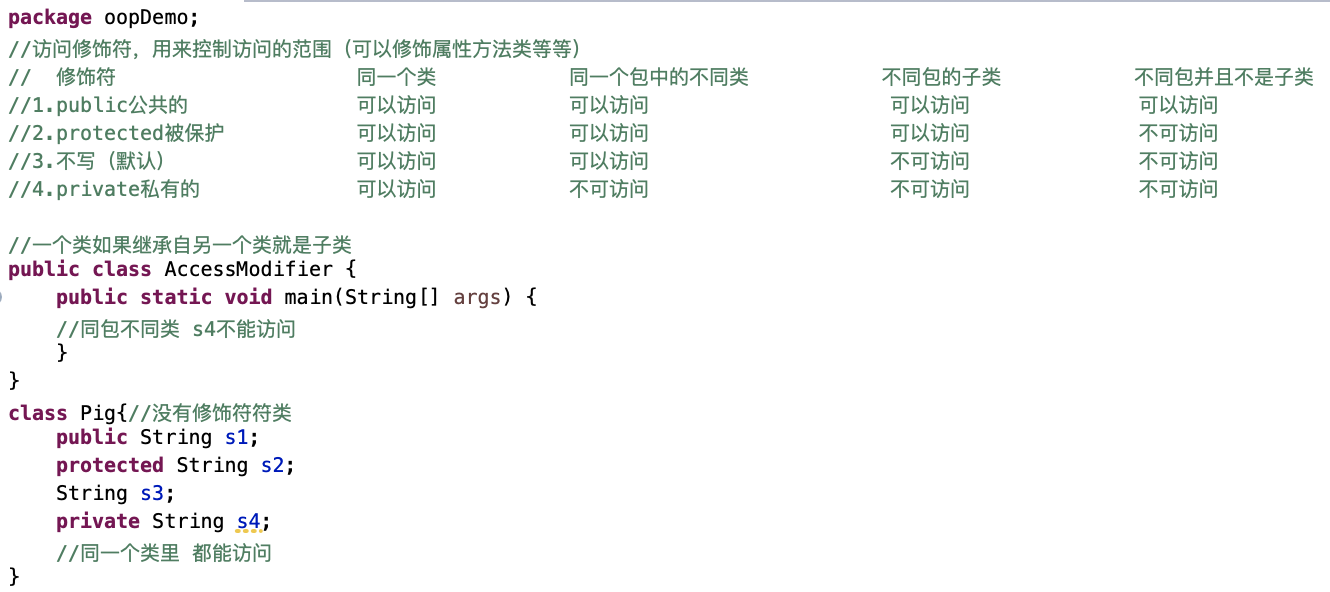
Car car= **new** Car();

Scanner input=**new** Scanner(System.***in***);

**int**[] arrayInt = **new** **int**[3];

}

**12.修饰符**



**13.对象的初始化顺序**