**基本语法**

编写 Java 程序时，应注意以下几点：

* **大小写敏感**：Java 是大小写敏感的，这就意味着标识符 Hello 与 hello 是不同的。
* **类名**：对于所有的类来说，类名的首字母应该大写。如果类名由若干单词组成，那么每个单词的首字母应该大写，例如 **MyFirstJavaClass** 。
* **方法名**：所有的方法名都应该以小写字母开头。如果方法名含有若干单词，则后面的每个单词首字母大写。
* **源文件名**：源文件名必须和类名相同。当保存文件的时候，你应该使用类名作为文件名保存（切记 Java 是大小写敏感的），文件名的后缀为 **.java**。（如果文件名和类名不相同则会导致编译错误）。
* **主方法入口**：所有的 Java 程序由 **public static void main(String[] args)** 方法开始执行。

public class HelloWorld {

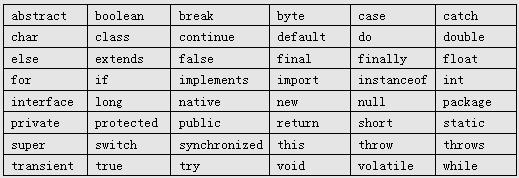
public static void main(String[] args){

System.out.println("hello imooc");

}

}

Java 中常用关键字：（区分大小写）



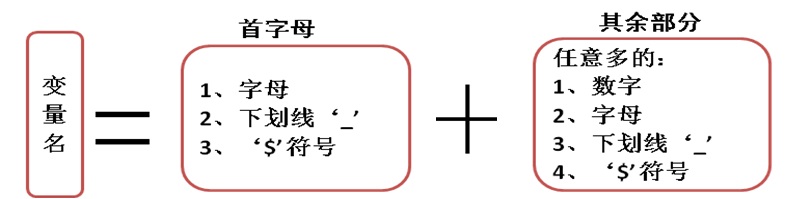
使用标识符时，需要遵守几条规则：

 1.  标识符**可以**由**字母、数字、下划线（\_）、美元符（$）**组成，但不能包含 @、%、空格等其它特殊字符**，不能以数字开头**。譬如：123name 就是不合法滴

 2.  标识符**不能是 Java 关键字和保留字**（ Java 预留的关键字，以后的升级版本中有可能作为关键字），但可以包含关键字和保留字。如：不可以使用 void 作为标识符，但是 Myvoid 可以

 3.  标识符是**严格区分大小写**的。 所以涅，一定要分清楚 imooc 和 IMooc 是两个不同的标识符哦！

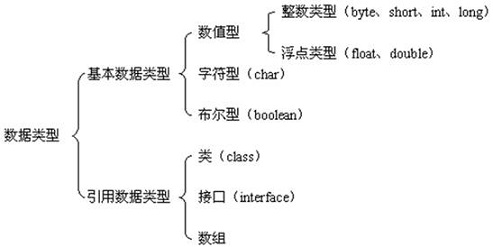
 4.  标识符的命名最好能反映出其作用，做到**见名知意**。

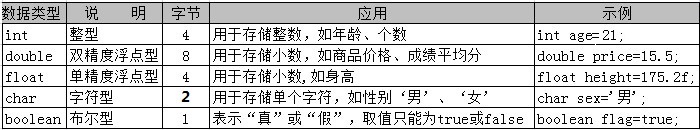


1、变量名由多单词组成时，**第一个单词的首字母小写**，**其后单词的首字母大写**，俗称骆驼式命名法（也称驼峰命名法），如 myAge

2、变量命名时，尽量简短且能清楚的表达变量的作用，做到**见名知意**。如：定义变量名 stuName 保存“学生姓名”信息

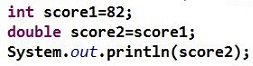
PS： Java 变量名的**长度没有限制**，但 Java 语言**是区分大小写**的，所以 price 和 Price 是两个完全不同的变量哦！





## Java中的自动类型转换

在 Java 程序中，不同的基本数据类型的数据之间经常需要进行相互转换。例如：

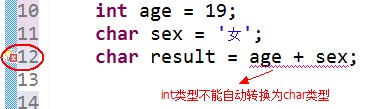
[](https://img.mukewang.com/5359d0c70001ffe202530066.jpg)，

代码中 int 型变量 score1 可以直接为 double 型变量 score2 完成赋值操作，运行结果为：**82.0**

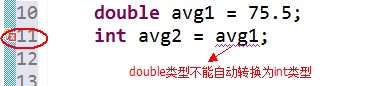
这种转换称为**自动类型转换**。

当然自动类型转换是需要**满足特定的条件**的：

1.  目标类型能与源类型兼容，如 double 型兼容 int 型，但是 char 型不能兼容 int 型

[](https://img.mukewang.com/53688a3200014fc803800109.jpg)

2.  目标类型**大于**源类型，如 double 类型长度为 8 字节， int 类型为 4 字节，因此 double 类型的变量里直接可以存放 int 类型的数据，但反过来就不可以了

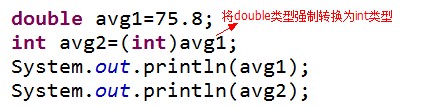
[](https://img.mukewang.com/53688a460001539403740086.jpg)

## Java中的强制类型转换

当程序中需要将 double 型变量的值赋给一个 int 型变量

显然，这种转换是不会自动进行的！因为 int 型的存储范围比 double 型的小。此时就需要通过**强制类型转换**来实现了。

语法：**( 数据类型 ) 数值**

[](https://img.mukewang.com/53688a9a0001f73404330109.jpg)

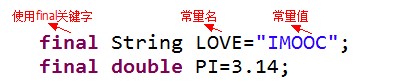
运行结果：

[](https://img.mukewang.com/5359d73a00016cde00500033.jpg)

## Java常量的应用

所谓常量，我们可以理解为是一种特殊的变量，它的值被设定后，在**程序运行过程中不允许改变**。

语法：**final 常量名 = 值;**

[](https://img.mukewang.com/53689a880001620703990081.jpg)

程序中使用常量可以提高代码的**可维护性**。例如，在项目开发时，我们需要指定用户的性别，此时可以定义一个常量 SEX，赋值为 "男"，在需要指定用户性别的地方直接调用此常量即可，避免了由于用户的不规范赋值导致程序出错的情况。

注意啦：常量名一般使用**大写字符**

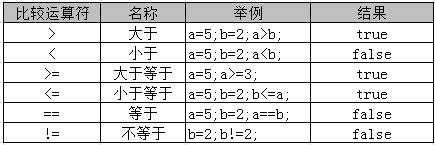
Java 中注释有三种类型：**单行注释、多行注释、文档注释**

[](https://img.mukewang.com/535ce105000118b504440337.jpg)

## Java中的比较运算符

比较运算符用于判断两个数据的大小，例如：大于、等于、不等于。比较的结果是一个布尔值（ true 或 false ）。

Java 中常用的比较运算符如下表所示：

[](https://img.mukewang.com/536063f00001b7b904350145.jpg)

注意哦：

1、  > 、 < 、 >= 、 <= 只支持左右两边操作数是数值类型

2、  == 、 != 两边的操作数既可以是数值类型，也可以是引用类型

## Java中的逻辑运算符

逻辑运算符主要用于进行逻辑运算。Java 中常用的逻辑运算符如下表所示：

[](https://img.mukewang.com/5360ef9700017b9405870095.jpg)

我们可以从“投票选举”的角度理解逻辑运算符：

1、 **与**：要求所有人都投票同意，才能通过某议题

2、 **或**：只要求一个人投票同意就可以通过某议题

3、 **非**：某人原本投票同意，通过非运算符，可以使其投票无效

4、 **异或**：有且只能有一个人投票同意，才可以通过某议题

## Java中的条件运算符

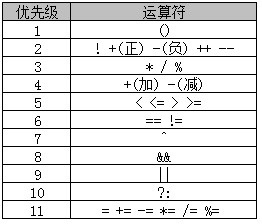
条件运算符（ ? : ）也称为 “三元运算符”。

语法形式：**布尔表达式 ？ 表达式1 ：表达式2**

运算过程：如果布尔表达式的值为 **true**，则返回 **表达式1** 的值，否则返回**表达式2**的值

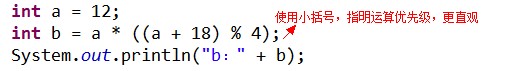
## Java中运算符的优先级

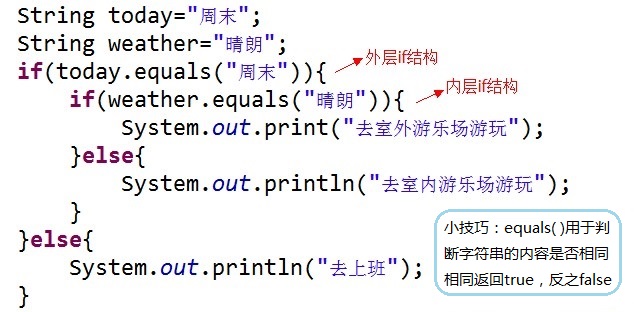
所谓优先级，就是在表达式中的运算顺序。Java 中常用的运算符的优先级如下表所示：

[](https://img.mukewang.com/5360ffb90001b4f002620224.jpg)

**级别为 1 的优先级最高，级别 11 的优先级最低**。譬如，x = 7 + 3 \* 2  得到的结果是 13 “先乘后加”嘛!

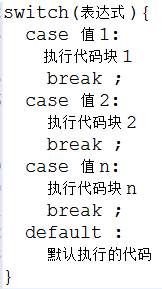
PS：大家没必要去死记运算符的优先级顺序，实际开发中，一般会使用**小括号**辅助进行优先级管理。例如：

[](https://img.mukewang.com/5361000a000129a005060071.jpg)



## Java条件语句之 switch

语法：

**[](https://img.mukewang.com/5364f4c800010b1f01620289.jpg)**

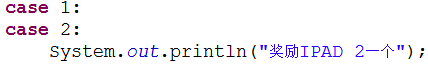
不得不说的几点小秘密：

1、 switch 后面小括号中表达式的值必须是**整型或字符型**

2、 case 后面的值**可以是常量数值**，如 1、2；也**可以是一个常量表达式**，如 2+2 ；但不能是变量或带有变量的表达式，如 a \* 2

3、 case 匹配后，执行匹配块里的程序代码，如果没有遇见 **break**会继续执行下一个的 case 块的内容，直到遇到 break 语句或者 switch 语句块结束 如

4、 可以把功能相同的 case 语句合并起来，如

[](https://img.mukewang.com/5364f6950001a57604280064.jpg)

5、 default 块可以出现在**任意位置**，也**可以省略**

## Java循环跳转语句之 continue

**continue** 的作用是跳过循环体中剩余的语句执行下一次循环。

// 每打印完一行后进行换行

System.out.println();

只打印System.out.print(" ");

输入一个数num

Int num = new Scanner(System.in).nextInt();

## 如何使用 Java 中的数组

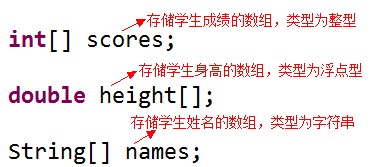
Java 中操作数组只需要四个步骤：

**1、 声明数组**

语法：  **数据类型[ ] 数组名；**

或者   **数据类型 数组名[ ]；**

其中，数组名可以是任意合法的变量名，如：

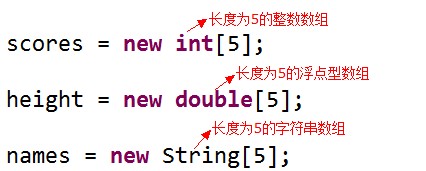
[](https://img.mukewang.com/5371e91d000147e903790167.jpg)

**2、 分配空间**

简单地说，就是指定数组中最多可存储多少个元素

语法：  **数组名 = new  数据类型 [ 数组长度 ];**

其中，数组长度就是数组中能存放元素的个数，如：

[](https://img.mukewang.com/5371ecc10001c97804260171.jpg)

话说，我们也可以将上面的两个步骤合并，在声明数组的同时为它分配空间，如：[](https://img.mukewang.com/5371ed4200012bee03590033.jpg)

**3、 赋值**

分配空间后就可以向数组中放数据了，数组中元素都是通过下标来访问的，例如向 scores 数组中存放学生成绩

[](https://img.mukewang.com/5371ed9f0001accb02100053.jpg)

**4、 处理数组中数据**

我们可以对赋值后的数组进行操作和处理，如获取并输出数组中元素的值

[](https://img.mukewang.com/5371eec50001332707460043.jpg)

在 Java 中还提供了另外一种直接创建数组的方式，它将声明数组、分配空间和赋值合并完成，如

[](https://img.mukewang.com/5371ee8b000101aa04540056.jpg)

它等价于：

[](https://img.mukewang.com/5371ef0f0001528c04830059.jpg)

**输入输出流：**

Java.io.File类用于表示文件（目录）;

File类只用于文件（目录）的信息（大小名称等），不能用于文件内容的访问。

先导入包import java.io.File;

创建File类对象：File file=new File(“目录”);

判断文件是否存在：

System.out.println(file.exists());

存在结果为true 否则为false。

file.mkdir();//当file文件不存在创建file，且上级只有一级目录，当有多级目录时file.mkdirs();

创建文件file.creatNewFile();

//是否是一个目录

System.out.println(file.isDirectory());

//是否是一个文件

System.out.println(file.isFile());

file.delete();//删除文件

System.out.print(file);//file路径;=file.getAbsolutePath();

System.out.print(file.getName());//最后一级的目录名。

file.getParent();//获得父级目录。

**RandomAccessFile:java提供对文件内容的访问，既可以读文件，也可以写文件。**

**支持随机访问文件，可以访问文件的任意位置**

**(1)Java文件模型**

**在硬盘上的文件是byte byte byte存储的，是数据的集合**

**(2)打开文件**

**有两种模式 “rw”（读写）“r”（只读）。**

**例： RandomAccessFile raf=new RandomAccessFile(file,”rw”);**

文件指针：打开文件时指针在开头pointer=0;

(3)写方法

raf.writer(int)-->只写一个字节（后八位），同时指针指向下一个位置。准备再次写入。

(4)读方法。

int b=raf.read();-🡪读一个字节

(5)文件读写完成后一定要关闭（官方说明）。

**IO流（输入流，输出流）**

**字节流，字符流**

**1,字节流**

1. InputStream OutputStream

InputStream抽象了应用程序读取数据的方式

OutputStream抽象了应用程序写出数据的方式

2）EOF=End 读到-1就读到结尾

3)输入流基本方法

int b = in.read(); 读取一个字节无符号填充到int低八位。-1是EOF。

in.read(byte[] buf) //读取数据填充到字节数组buf

in.read(byte[] buf,int start,int size)// 读取数据到字节数组buf,从buf的start位置开始存放size长度的数据。

4）输出流基本方法

out.writer(int b) 写出一个byte到流，b的第8位

out.writer(byte[] buf)将buf字节数组都写入到流

out.writer(byte[] buf,int start,int size)

5)FileInputStream-🡪具体实现了在文件上读取数据

6）FileOutputStream实现了向文件中写出byte数据的方法

7)DataOutputStream/ DataInputStream

对“流”功能的扩展，可以更加方面的读取int,long,字符等类型数据

DataOutputStream

writerInt()/writerDouble()/writerUTF()

**Java语法2**

Java创建类的语法

public class Dog{

//类的内容

}

\*类名要有直白的意义，最好是英文

\*首字母大写

\*不要数字开头，不要包含特殊字符

创建类的实例对象

类名 对象名 = new 类名（）;

Dog lucky = new Dog();

对象命名规则

1. 对象名要有直白的意义，最好是英文
2. 首字母小写
3. 不要数字开头，不要包含特殊字符

用法

1. *public class* dog {  
    String name;  
    *int* age;  
    *float* weight;  
    *public static void* main(String[] args) {  
    dog duoduo = *new* dog();  
    duoduo*.*name = "Duoduo";  
    duoduo*.*age = 20;  
    System*.*out*.*println(duoduo*.*name);  
    dog lucky = *new* dog();  
    }  
   }

**方法（c中的函数）**

*public class* meathod {  
 *public* String isOddNumber(*int* number) {  
 *if*(number%2 == 0){  
 *return* "偶数";  
 }*else*{  
 *return* "奇数";  
 }  
 }  
 *public static void* main(String[] args) {  
 *int* a=10;  
 meathod meathod = *new* meathod();//创建meathod对象才可以调用这个方法。  
 String r1 =meathod*.*isOddNumber(a);  
 System*.*out*.*println(r1);  
 }

**包-package**

命名规范：采用“逆域名法”进行命名，用“.”分割，单词全部小写

标准格式：域名后缀.组织机构名.项目名[.模块名].包的职能。

//要使用其他包下的类必须进行事先import导入。

**访问修饰符**

作用：用于控制类，成员变量，方法的访问范围

访问等级比较：public > protected > default > private

* **public**:不管包外包内，所有类（子类+非子类）均可使用
* **protected**
  + 包内：所有类可使用
  + 包外：子类可使用，非子类不可使用
* **Default（默认）**
  + 包内：所有类可使用
  + 包外：所有类不可使用
* **private**：仅本类可使用

**列：**

* *public class* xiushifu {  
   *//public代表其他任何类都能访问的成员变量  
   public* String name;  
   *//private代表只能在当前类中使用的变量  
   private void* sayhello(){  
   System*.*out*.*println("hello world");  
   }  
   *//不写修饰符代表默认修饰符，表示只能被包中其他类访问  
   void* saygoodbye() {  
   System*.*out*.*println("good bye");  
   }  
   *public static void* main(String[] args) {  
   xiushifu xiushifu = *new* xiushifu();  
   xiushifu*.*sayhello();  
   xiushifu*.*saygoodbye();  
   }  
    
  }

**面向对象之封装**

概述：将对象的属性和实现细节隐藏起来，不让外部程序直接进行访问，将属性私有化，仅对外公开接口，让外部程序通过类提供的方法来对隐藏信息进行访问和操作。好处是外部程序只能通过类规定的方法对数据进行访问，避免外界程序对类内部属性进行破坏。

### 继承

继承可以对代码进行复用以提高编程的效率，继承就是子类获取父类的成员变量及成员方法。已经存在的类称为父类（也叫基类，超类），新构建的类称为子类（派生类）。Java中通过extend关键字声明一个类从另一个类继承而来。

注意：

1.父类可以有多个子类，但子类只能继承一个父类（单继承，Java中不支持多继承）。一个子类不能继承多个父类，但可以实现多个接口。

2.子类不能继承父类中private修饰的属性和方法。

3.子类可以重写父类的方法。

多态

在Java中，多态指调用同一个方法，不同的对象可能会有不同的行为。

多态实现的三个前提：

1.存在继承关系

2.必须要有方法的重写

3.父类的引用指向子类对象



Mysql关键字不区分大小写

连接数据库

命令行连接：

Mysql -uroot -p123456 --连接数据库

‘localhost’ –修改用户密码

创建数据库 :  create database [if not exists] 数据库名;

删除数据库 : drop database [if exists] 数据库名;

查看数据库 : show databases;

使用数据库 : use 数据库名;

2.1操作数据库

操作数据库>操作数据库中的表>操作数据库中的表的数据

1. 操作数据库
2. 创建数据库
3. 删除数据库
4. 使用数据库

如果表名或者字段名是一个特殊字符，就需要带‘ ‘

2.2数据库的数据类型



日期和时间型数值类型



NULL值

* 理解为 "没有值" 或 "未知值"
* 不要用NULL进行算术运算 , 结果仍为NULL

### DML语言

**数据库意义** ： 数据存储、数据管理

**管理数据库数据方法：**

* 通过SQLyog等管理工具管理数据库数据
* 通过**DML语句**管理数据库数据

**DML语言**   ：数据操作语言

* 用于操作数据库对象中所包含的数据
* 包括 :
  + INSERT (添加数据语句)
  + UPDATE (更新数据语句)
  + DELETE (删除数据语句)

### 添加数据

INSERT命令

**语法：**

INSERT INTO 表名[(字段1,字段2,字段3,...)] VALUES('值1','值2','值3')

**注意 :**

* 字段或值之间用英文逗号隔开 .
* ' 字段1,字段2...' 该部分可省略 , 但添加的值务必与表结构,数据列,顺序相对应,且数量一致 .
* 可同时插入多条数据 , values 后用英文逗号隔开 .

-- 使用语句如何增加语句?  
-- 语法 : INSERT INTO 表名[(字段1,字段2,字段3,...)] VALUES('值1','值2','值3')  
INSERT INTO grade(gradename) VALUES ('大一');  
  
-- 主键自增,那能否省略呢?  
INSERT INTO grade VALUES ('大二');  
  
-- 查询:INSERT INTO grade VALUE ('大二')错误代码：1136  
Column count doesn`t match value count at row 1  
  
-- 结论:'字段1,字段2...'该部分可省略 , 但添加的值务必与表结构,数据列,顺序相对应,且数量一致.  
  
-- 一次插入多条数据  
INSERT INTO grade(gradename) VALUES ('大三'),('大四');

### 删除数据

DELETE命令

语法：

DELETE FROM 表名 [WHERE condition];

注意：condition为筛选条件 , 如不指定则删除该表的所有列数据

TRUNCATE命令

作用：用于完全清空表数据 , 但表结构 , 索引 , 约束等不变 ;

语法：

TRUNCATE [TABLE] table\_name;  
  
-- 清空年级表  
TRUNCATE grade

**注意：区别于DELETE命令**

* 相同 : 都能删除数据 , 不删除表结构 , 但TRUNCATE速度更快
* 不同 :
  + 使用TRUNCATE TABLE 重新设置AUTO\_INCREMENT计数器
  + 使用TRUNCATE TABLE不会对事务有影响

### DQL语言

**DQL( Data Query Language 数据查询语言 )**

* 查询数据库数据 , 如**SELECT**语句
* 简单的单表查询或多表的复杂查询和嵌套查询
* 是数据库语言中最核心,最重要的语句
* 使用频率最高的语句

SELECT语法

SELECT [ALL | DISTINCT]  
{\* | table.\* | [table.field1[as alias1][,table.field2[as alias2]][,...]]}  
FROM table\_name [as table\_alias]  
  [left | right | inner join table\_name2]  -- 联合查询  
  [WHERE ...]  -- 指定结果需满足的条件  
  [GROUP BY ...]  -- 指定结果按照哪几个字段来分组  
  [HAVING]  -- 过滤分组的记录必须满足的次要条件  
  [ORDER BY ...]  -- 指定查询记录按一个或多个条件排序  
  [LIMIT {[offset,]row\_count | row\_countOFFSET offset}];  
   -- 指定查询的记录从哪条至哪条

**注意 : [ ] 括号代表可选的 , { }括号代表必选得**

数据库操作

create table `huangzhiwei`(  
 `name` varchar(20) not null comment &apos;用户&apos;,  
`price` int(20) not null comment &apos;价格&apos;,  
    `num` int(11) default null comment &apos;件数&apos;,  
    `state` varchar(50) not null comment &apos;商品状态&apos;,  
    `time` DATETIME not null comment &apos;操作时间&apos;,  
    primary key (`name`)  
)engine=myisam default charset=utf8;  
INSERT INTO `huangzhiwei` (`name`, `price`, `num`, `state`, `time`) VALUES (),(),();