



## 第二章 Java语言基础

---



# 涉及到课本中章节:

---


- **第2章 Java语言基础**



# 第2章 Java语言基础

---

- 2.1 标识符与关键字
- 2.2 数据类型
- 2.3 变量与常量
- 2.4 运算符与表达式
- 2.5 流程控制
- 2.6 数组和字符串



## 序：Java与C++(语言基础比较)

---

- 对于变量声明、参数传递、操作符、流程控制等使用和**C++**相同的传统

- 摒弃了**C**和**C++**中许多不合理的内容

- 全局变量：**Java**中没有全局变量。
- 指针：**Java**不支持指针。
- 数据类型的支持：**Java**在不同平台上数据类型都统一。
- 内存管理：**Java**自动回收无用内存。



# 第2章 Java语言基础

---

- **2.1** 关键字与标识符
- **2.2** 数据类型
- **2.3** 变量与常量
- **2.4** 运算符与表达式
- **2.5** 流程控制
- **2.6** 数组和字符串

# 关键字



由Java语言定义的具有特定含义的单词。

关键词不能被赋予别的含义



# 2.1 关键字与标识符

## 1. Java关键字

- 数据成分：
  - boolean, int, byte, short, long, char, float, double, void, null, const, true, false
- 访问控制与类、方法修饰符成分：
  - class, extends, abstract, interface, implements,
  - private, protected, public, super, this,
  - const, final, static, transient, volatile, native, synchronized,
- 运算加工成分：
  - new, =,
  - +, -, \*, /, %, ++, --, ?:, &, |, ^, ~, >>, <<, >>>,
  - ==, <, >, <=, >=, !=, instanceof,
- 控制成分：
  - for, while, do, if, else, switch, case, default, break, continue, goto, return,
  - throw, throws, try, catch, finally
- 程序结构成分：
  - import, package, (, ), [, ], {, }, ::, “, ‘, //, /\*, \*/,\*\*



## 注意:

---

- `true`、`false`和`null`为小写，而不是象在C++语言中那样为大写。
- 无`sizeof`运算符，所有类型的长度和表示是固定的。
- `goto`和`const`在Java中是关键字，但没有实际语义。





## 2.标识符

---

- 标识符是用户定义的单词
- 用来命名变量、类和方法等的名称
- 是以字母开头的字母数字序列



# 解释标识符:

---

字母:

- 可以是一个英文字母、下划线(\_)或美元符号(\$)。
- 也可以是**Unicode**字符集中的字符

另:

- 是大小写区别对待的,(大小写敏感的!)
- 不能使用关键字
- 无最大长度。



## 注意：

---

- **Java**源程序采用有效的**16-bit** 双字节字符编码标准(**UTF-16**)，而不是**8-bit ASCII**文本。
- 包含美元符号（\$）的关键字通常用的较少，因为**Java**用它来表示内部类，因而最好避免在标识符中使用它们。
- 检查哪些字符可以用作标识符可用**Character**类中的两个方法判断

**boolean isJavaIdentifierStart()**和**boolean isJavaIdentifierPart()**  
( **java.lang.Character** )

# 编码——计算机的暗号

宝塔镇河妖

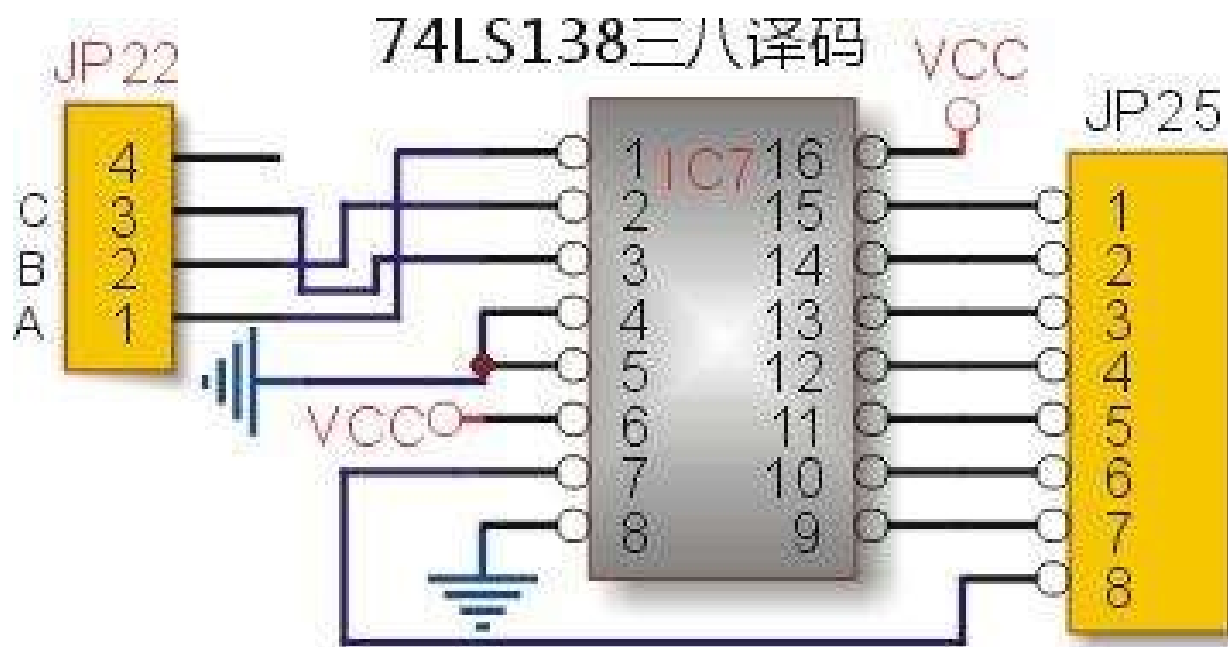
天王盖地虎



# ASCII码 对照表(b8为0)

<div> <div> 字 符 </div> <div> <math>b_7 b_6 b_5</math>  <math>b_4 b_3 b_2 b_1</math> </div> </div>	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	.	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	,	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	S	,	<	L	\	l	
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

# 编码





## 例题：(Unicode字符)

---

```
public class Example
```

```
{
```

```
■ public static void main (String args[ ])
```

```
■ {
```

```
■ char chinaWord='严',japanWord='あ';
```

```
■ int p1=20005,p2=12353;
```

```
■ System.out.println("汉字\'严\'字在Unicode表中的顺序位置:"+(int)chinaWord);
```

```
■ System.out.println("日语\'あ\'字在unicode表中的顺序位置:"+(int)japanWord);
```

```
■ System.out.println("unicode表中第20005位置上的字符是:"+(char)p1);
```

```
■ System.out.println("unicode表中第12353位置上的字符是:"+(char)p2);
```

```
■ }
```

```
■ }
```

■ 注：关于Unicode信息可以查看<http://www.unicode.org>

■ <http://unicode.org/standard/translations/s-chinese.html> (中文)



# UTF-8

---

编码	大小	支持语言
ASCII	1个字节	英文
Unicode	2个字节（生僻字4个）	所有语言
UTF-8	1-6个字节，英文字母1个字节，汉字3个字节，生僻字4-6个字节	所有语言

本着节约精神，又出现了“可变长编码”的**UTF-8**编码。**UTF-8**编码把一个**Unicode**字符根据不同的数字大小编码成**1-6**个字节，常用的英文字母被编码成**1**个字节，汉字通常是**3**个字节，只有很生僻的字符才会被编码成**4-6**个字节。





# 第2章 Java语言基础

---

- 2.1 标识符与关键字
- 2.2 数据类型
- 2.3 变量与常量
- 2.4 运算符与表达式
- 2.5 流程控制
- 2.6 数组和字符串

## 2.2 数据类型

### 1. Java 数据类型

基本 数据 类型	布尔数据类型( <b>boolean</b> )	2字节( <b>16bit</b> )	
	字符类型( <b>char</b> )	2字节( <b>16bit</b> )	
	整数类型	<b>byte</b>	1字节( <b>8bit</b> )
		<b>short</b>	2字节( <b>16bit</b> )
		<b>int</b>	4字节( <b>32bit</b> )
		<b>long</b>	8字节( <b>64bit</b> )
	浮点类型	<b>float</b>	4字节( <b>32bit</b> )
		<b>double</b>	8字节( <b>64bit</b> )
引用数据 类型	类( <b>class</b> )		
	接口( <b>interface</b> )		
	数组( <b>array</b> )		

## 2. Java与C++基本数据类型比较

Java			C++	
char	char	(16bit)	char (signed char)	(8bit)(-128--127)
			unsigned char	(8bit)(0-255)
int	byte	(8bit)(-128--127)		
	int	(32bit)	int(signed int)	与机器有关
			unsigned int	
	short	(16bit)	short int(signed short int)	16bit
			unsigned short int	16bit
	long	(64bit)	long int (signed long int)	32bit
			unsigned int	32bit
浮点数	float	(32bit)	float	32bit
	double	(64bit)	double	64bit
			long double	80bit



## 比较后结论：

---

- **Java**各整数类型有固定的表示范围和字段长度，不受具体操作系统的影响，保证了**Java**程序的可移植性。
- **Java**的浮点类型具有平台独立性特性



# Java 运算类型兼容性原则

---

## ■ 基本类型转换

### --自动转换

两个不同的数据类型具有相同的性质，数据能够参加相同的运算，运算结果类型为范围大、精度高的那种类型

**byte < (short=char) < int < long < float < double**

# 生活中的类型转换



- 卖菜小钱精度高，精打细算一块两块
- 买车大钱精度低，千元以下轻松抹零





# Java 运算类型兼容性原则

---

## ■ 基本类型转换

### --强制转换

大的转成小的，或者在**short**与**char**之间进行转换，就必须强制转换，也被称作缩小转换（**narrowing conversion**），因为必须显式地使数值更小以适应目标类型。

强制转换采用转换操作符（），如**(int) abc**

# Java 运算类型兼容性原则

From \ To	byte	short	char	int	long	float	double
byte	-	(byte)	(byte)	(byte)	(byte)	(byte)	(byte)
short		-	(short)	(short)	(short)	(short)	(short)
char		(char)	-	(char)	(char)	(char)	(char)
int				-	(int)	(int)	(int)
long					-	(long)	(long)
float						-	(float)
double							-

参考地址：

<http://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html>

<http://blog.csdn.net/bingduanlbd/article/details/27790287>



### 3.基本数据类型的封装类



基本数据类型	封装类(Wrapper)
<b>boolean</b>	<b>Boolean</b>
<b>char</b>	<b>Character</b>
<b>byte</b>	<b>Byte</b>
<b>short</b>	<b>Short</b>
<b>int</b>	<b>Integer</b>
<b>long</b>	<b>Long</b>
<b>float</b>	<b>Float</b>
<b>double</b>	<b>Double</b>

注：所有的封装类都提供了静态的**valueOf(String s)**方法，把给定的**String**类型转换成对应的基本数据类型





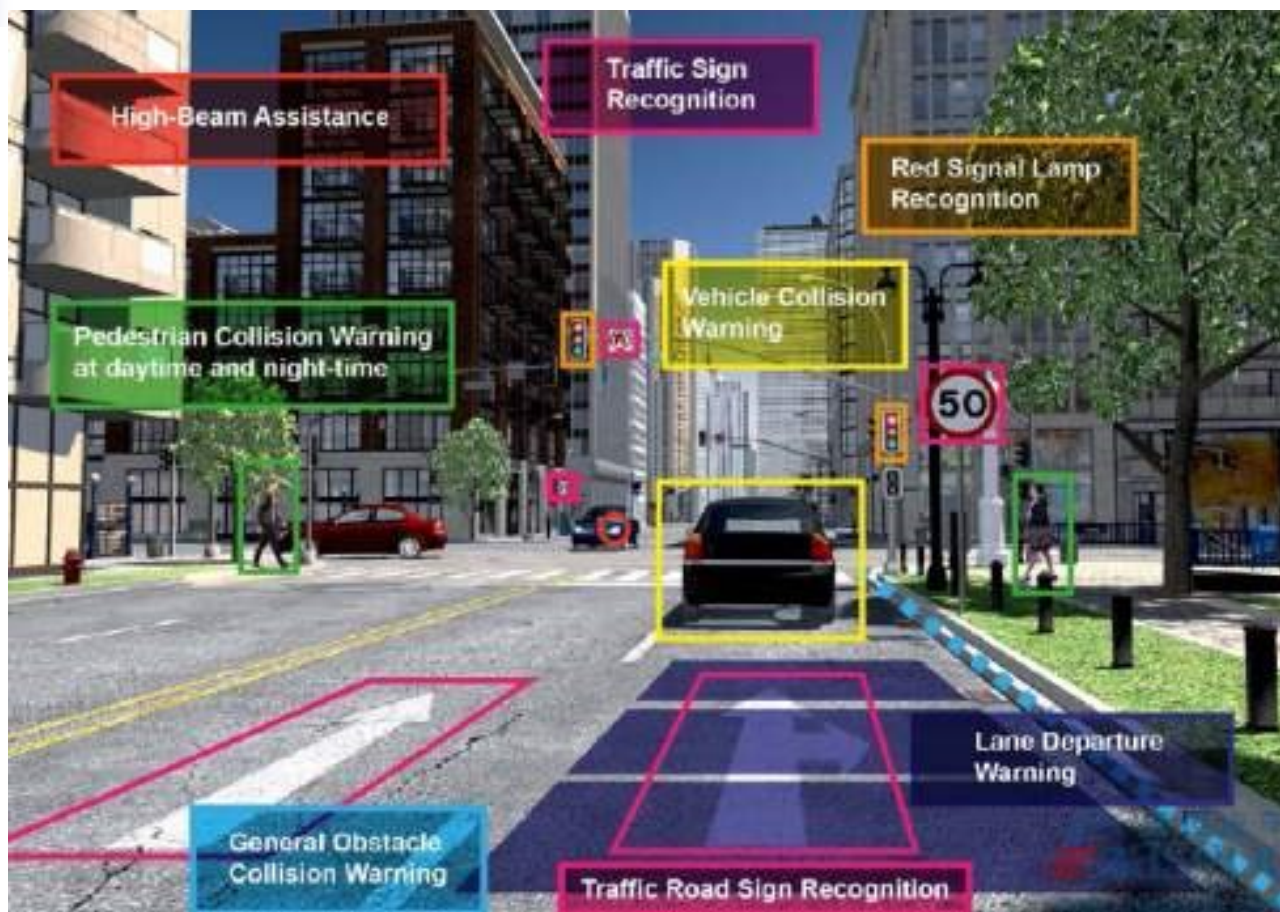
# 第2章 Java语言基础

---

- 2.1 标识符与关键字
- 2.2 数据类型
- **2.3 变量与常量**
- 2.4 运算符与表达式
- 2.5 流程控制
- 2.6 数组和字符串

# 设想一下自动驾驶程序

交通变量



交通目标 → Java变量



# 设想一下自动驾驶程序

交通语法

编程语言要素	Java语言	交通活动
关键字	class、String、int	地名、路名
标识符	自定义变量、类	回趟家、远离重型车辆
数据类型	字符、整数、浮点数	交通要素类型（红绿灯、车辆、行人）
变量、常量	i=1; PI=3.14	车速、油价
运算符(操作符)	+ >= >> 优先级	操控（加速、红灯秒、变道）人行道
表达式	i+1	速度+10，速度+1
流程控制	if{} else{} else{}	路径规划、路径选择
数组	j[]={1,2,3,4,5}	车队、斑马线行人
字符串	k="Hello Java"	"左转车辆进入待行区"

Java语法、交通语法类比表

# 设想一下自动驾驶程序

路径选择



交通路径选择 → Java流程控制



## 2.3 常量与变量

---

- **Java**中数据都存储在常量和变量中
- **Java**是强类型语言,遵循”先声明后使用”的原则



# 常量

直接常量、符号常量

## ■ 直接常量:

例如:**123,-6.87, 'v','中' ,”中国” ,true**等

↑-数值型-↑      ↑ ---- 非数值型 ---- ↑

**random(100)**

**System.out.print(“Hello Java”)**

## ■ 符号常量:

例: **final int MAX=10; final float PI=3.14f;**

--修饰符为关键字**final**，程序中不能改变

--全大写，提高可读性





# 变量:

---

- 保存在程序中可被改变的数据
- 有四个基本要素:名字,类型,值和作用域  
要素齐全,把一件事说明白。

注: 名字(称) ---由标识符组成

类型 ---可以是基本数据类型也可以是引用数据类型

值 ---可以在程序中改变

作用域 ---声明的位置决定了作用域

例: 监考员在**356**教室的黑板书写**2**类考试说明 (考试科目和时间)

例: 红绿灯的提示语**3** “**左转车辆进入待行区**” 是提醒当前路口左转车辆的



# 变量（先声明后赋值再使用）

---

## ■ 1.变量的声明:

[<修饰符>]<类型><变量名>[=<初值>][,<变量名>[=<初值>].....]

**int i;**

**public int i;**

**public int i=0;**

**public int i=1,j=2;**

**注：**变量分局部变量和成员变量



# 变量

---

## 2.变量的初始化（赋值）

例如：

```
int i=0;
```

```
boolean truth=true;
```

```
char c='A';
```

```
float x=3.14f;
```



### 3.变量的分类:

---

- 按照所属的数据类型分:

基本数据类型变量(**primitive type**)

引用数据类型变量(**reference type**)

- 按照声明的位置: 一决定了变量的作用域

成员变量—方法体外, 类体内声明的变量

局部变量—在方法内部 (或者形参) 声明的变量



# 变量默认值

---

类型	默认值
■ boolean	false
■ byte	0
■ short	0
■ int	0
■ long	0L
■ char	\u0000
■ float	0.0f
■ double	0.0d
■ 对象引用	null



# 第2章 Java语言基础

---

- 2.1 标识符与关键字
- 2.2 变量与常量
- 2.3 数据类型
- **2.4 运算符与表达式**
- 2.5 流程控制
- 2.6 数组和字符串



# 算术运算符、数学算式

---

■  $+$   $-$   $\times$   $\div$   $x^2$   $\sqrt[2]{2}$   $\int_0^1 y dx$

■  $1+1$

■  $1+1=2$

■  $1+1 \times 2$        $(1+1) \times 2$



# 1.常用运算符

---

- 算术运算符:  $+$ ,  $-$ ,  $*$ ,  $/$ ,  $\%$
- 递增递减运算符:  $++$ ,  $--$      **!!建议不与其他表达式混用**
- 关系运算符:  $==$ ,  $>$ ,  $<$ ,  $>=$ ,  $<=$ ,  $!=$
- 条件运算符:  $a? b:c$  (选择运算, 三元)
- 逻辑运算符:  $\&$ ,  $|$ ,  $!$ ,  $^$ ,  $\&\&$ ,  $||$
- 位运算符:  $\&$ ,  $|$ ,  $\sim$ ,  $^$ ,  $>>$ ,  $<<$ ,  $>>>$      **二进制**

## 数学函数和常量:

- **Math**类中提供`pow`,`sqrt`, `random`, `sin`, `cos`, `tan`, `atan`, `atan2`, `exp`, `log`, **PI**, **E**。(java.lang)
- 若要保证平台无关性, 应使用**StrictMath**类。





# 运算符

---

- 赋值运算符: `=` !!不用于判断语句
- 字符串连接运算符: `+`
- 括号运算符: `() []`
- 强制类型转换符: `()`
- 点运算符: `.`
- 对象运算符: `instanceof`
- `new`运算符: `new`

# 数学中的逻辑运算

## 新授

### 2、逻辑函数

我们学过的  $F = A \bullet B$

$F = \overline{A+B}$ ,  $F = \overline{A \bullet B}$  都是

其中  $A$ ,  $B$  叫逻辑变量

特点:

(1) 逻辑函数的逻辑只能取0和1;

(2) 逻辑因变量和逻辑关系是由“与”、“或”、算决定的。

## 探究

例: 列出逻辑函数  $F = A + (B \bullet C)$  的真值表

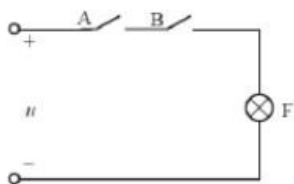
$A$	$B$	$C$	$B \bullet C$	$A + (B \bullet C)$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

# 电路中的逻辑运算

## 1. 与运算

只有当决定一件事情的条件都具备时，这件事情才会发生，这种因果关系称为与逻辑，其逻辑关系、真值表及逻辑符号如图 6.8 所示。

若用逻辑表达式来描述，则



(a) 电路

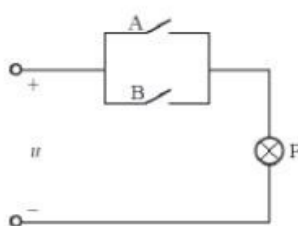
下图 6.8 为实现与运算的二极管与门电路。当输入 A 和 B 中只要有一个为低电平，则输出 F 为低电平；只有当输入 A 和 B 同时为高电平时，输出 F 才为高电平。

图

## 2. 或运算

当决定一件事情的几个条件中任何一个条件满足时，这件事情就会发生，这种因果关系称为或逻辑，其逻辑关系、真值表及逻辑符号如图 6.9 所示。

若用逻辑表达式来描述，则



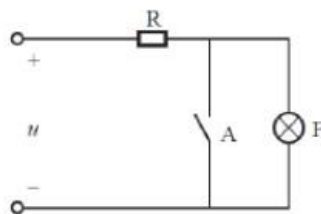
(a) 电路

下图 6.10 为实现或运算的二极管或门电路。当输入 A 和 B 中只要有一个为高电平，则输出 F 为高电平；只有当输入 A 和 B 同时为低电平时，输出 F 才为低电平。

图

## 3. 非运算

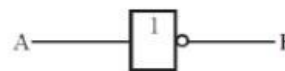
某事情发生与否，仅取决于一个条件，而且是对该条件的否定，即条件具备时事情不发生；条件不具备时事情才发生，其逻辑关系、真值表及逻辑符号如图 6.11 所示。



(a) 电路

A	F
0	1
1	0

(b) 真值表



(c) 逻辑符号

图 6.11 或运算

若用逻辑表达式来描述，则可写为： $Y = \bar{A}$

下图 6.12 为晶体管非门电路。当输入为高电平，晶体管饱和，输出为低电平；当输入为低电平，晶体管截止，输出为高电平，实现了非门功能。

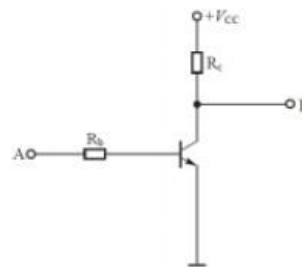


图 6.12 非运算的二极管与门电路



# Java中的逻辑运算

逻辑与 “&&”	逻辑或 “  ”	逻辑非 “！”	逻辑异或 “^”	逻辑与 “&”	逻辑或 “ ”		
逻辑运算符的真值表							
A	B	A && B	A    B	!A	A ^ B	A & B	A   B
T	T	T	T	F	F	T	T
T	F	F	T	F	T	F	T
F	T	F	T	T	T	F	T
F	F	F	F	T	F	F	F



# 引申问题:

## ■ 问: & 与 && 的区别?

提示:

**&& ||** 具有短路功能  
**&** 按位与以及逻辑与  
**&&** 条件与只能是布尔

### 1. 逻辑&的运算

```
boolean a = true;
boolean b = false;
int i = 10;
if(b&(i++)>0)
    System.out.print(i);    //输出11, 即&的右边有进行运算
else
    System.out.print(i);
```

### 2. 短路&&的运算

```
boolean a = true;
boolean b = false;
int i = 10;
if(b&&(i++)>0)    后面不用算也知道最终结果
    System.out.print(i);    //输出10, 即&&的右边没有运算
else
    System.out.print(i);
```



## 举例：(运算符)

```
class Example2_3
```

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

```
{char a1='十',a2='点',a3='进',a4='攻';
```

//好的命名习惯是一行只定义一个变量

```
char secret='8';
```

```
a1=(char)(a1^secret); a2=(char)(a2^secret);
```

```
a3=(char)(a3^secret); a4=(char)(a4^secret);
```

```
System.out.println("密文:"+a1+a2+a3+a4);
```

```
a1=(char)(a1^secret); a2=(char)(a2^secret);
```

```
a3=(char)(a3^secret); a4=(char)(a4^secret);
```

```
System.out.println("原文:"+a1+a2+a3+a4);
```

```
}
```

```
}
```

# 异或加密解密

```
1 public class P30{
2     public static void main(String args[]){
3         char a1 = '+', a2 = '点', a3 = '进', a4 = '攻';
4         char secret = 'A';
5         a1 = (char)(a1^secret);
6         a2 = (char)(a2^secret);
7         a3 = (char)(a3^secret);
8         a4 = (char)(a4^secret);
9         System.out.println("密文:"+a1+a2+a3+a4);
10        a1 = (char)(a1^secret);
11        a2 = (char)(a2^secret);
12        a3 = (char)(a3^secret);
13        a4 = (char)(a4^secret);
14        System.out.println("原文:"+a1+a2+a3+a4);
15    }
16 }
17 }
```

C:\Program Files (x86)\JCreator Pro\GE2001.exe

密文:匀悔犇馐

原文:十点进攻

Press any key to continue...

## 2. 常用运算符的优先级

### ■ 优先级由高到低为:

[ ] . ( ) ; ,

~ ++ -- +(正号) -(负号) (类型) new

\* / %

+ -

<< >> >>>

> < >= <= instanceof

== !=

&

^

|

&&

||

?:

= += -= \*= /= %= <<= >>= >>>= &= ^= |=



# 优先级及结合性

优先级	运算符	结合性
1	() [] .	从左向右
2	! + (正) - (负) ~ ++ --	从右向左
3	* / %	从左向右
4	+ (加) - (减)	从左向右
5	<< >> >>>	从左向右
6	< <= > >= instanceof	从左向右
7	== !=	从左向右
8	&(按位与)	从左向右
9	^	从左向右
10		从左向右
11	&&	从左向右
12		从左向右
13	?:	从右向左
14	= += -= *= /= %= &=  = ^= ~= <<= >>= >>>=	从右向左

# 运算符优先级示例

```
2 public class Language {  
3     public static void main(String[] args){ //main  
4  
5         int i = 5;  
6         i = i ++;  
7         System.out.println(i);  
8  
9         int y;  
10        int x = 5;  
11        y = (x++) + (++x) + (x*10);  
12        System.out.println(y);  
13    }  
14 }
```

```
15        int a = 5;  
16        int b = 4;  
17        int c = a++ - --b * ++a / b-- >> 2 % a--;
```



# 表达式:

---

- 用运算符将操作数连接起来的符合语法规则的运算式

- 例:

**a=123+b**

**x++**

**(w+1)%7**

**x%100==0**

注意：表达式和语句的区别。



# 表达式书写注意事项:

---

- 乘法运算符的\*不能省略
- 数学运算的 $\neq$ ,  $\leq$ ,  $\geq$ 必须写成 $!=$ ,  $>=$ ,  $<=$
- 赋值运算符= 关系运算符==
- 数学中的分式必须写成除式.
- 只有()能改变运算顺序,不能使用[]或者{}改变运算顺序.