# DAY02 常量变量，进制，数据，数据转换，算术赋值关系

## 常量

### 02.01\_Java语言基础(常量的概述和使用)(掌握)

#### A:什么是常量

在程序执行的过程中其值不可以发生改变

#### **B:Java中常量的分类**

字面值常量

自定义常量(面向对象部分讲)

#### **C:字面值常量的分类**

字符串常量 用双引号括起来的内容

整数常量 所有整数

小数常量 所有小数

字符常量 用单引号括起来的内容,里面只能放单个数字(0~9),单个字母,或单个符号,也可以使单个的汉字,因为Java采用的是UNicode编码的,一个汉字是两个字节.也可以放得下，单也仅仅只是一个汉而已。

布尔常量 较为特殊，只有true和false两个值。

空常量 null(数组部分讲解)

注:字符串必须用双引号括起来"abc"，不能使用单引号括起来'abc'。

#### D:案例演示

用输出语句输出各种常量。null不演示

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Constant { //constant 常量

public static void main(String[] args) {

System.out.println("abc"); //字符串常量

System.out.println(123); //整数常量

System.out.println(12.3); //小数常量

//System.out.println('10'); //''中必须放的是单个字符,10代表的是1字符和0字符

//System.out.println(''); //''中什么也不放,也是不可以的,因为代表不了任何字符

System.out.println(' '); //代表空格字符，注意一个制表符tab也是一个字符

System.out.println(true); //boolean类只有两个值,true和false

System.out.println(false);

}

}

---------------------------------------------------------------------------------------------------------------------------

## 进制讲解

### 02.02\_Java语言基础(进制概述和二,八,十六进制图解)(了解)

A:什么是进制

进制：就是进位制，是人们规定的一种进位方法。 对于任何一种进制--X进制，就表示某一位置上的数运算时是逢X进一位。二进制就是逢二进一，八进制是逢八进一，十进制是逢十进一，十六进制是逢十六进一。

例如一周有七天,七进制,一年有十二个月,十二进制

B:十进制的由来

十进制的由来是因为人类有十个手指

C:二进制的由来

其实二进制来源与中国,请看史料记载

18世纪德国数理哲学大师莱布尼兹从他的传教士朋友鲍威特寄给他的拉丁文译本《易经》中，读到了八卦的组成结构，惊奇地发现其基本素数（0）（1），即《易经》的阴爻yao- -和\_\_阳爻，其进位制就是二进制，并认为这是世界上数学进制中最先进的。20世纪被称作第三次科技革命的重要标志之一的计算机的发明与应用，其运算模式正是二进制。它不但证明了莱布尼兹的原理是正确的，同时也证明了《易经》数理学是很了不起的。

D:八进制的由来

任何数据在计算机中都是以二进制的形式存在的。二进制早期由电信号开关演变而来。一个整数在内存中一样也是二进制的，但是使用一大串的1或者0组成的数值进行使用很麻烦。

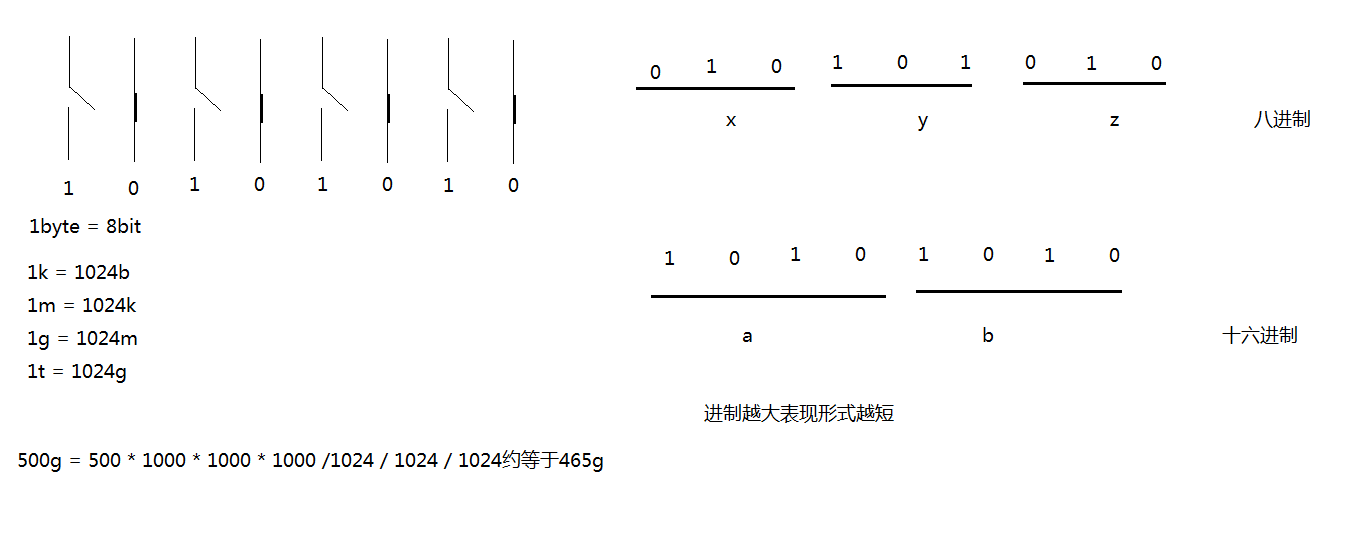
所以就想把一大串缩短点，将二进制中的三位用一位表示。这三位可以取到的最大值就是7.超过7就进位了，这就是八进制。

E:十六进制的由来

但是对于过长的二进制变成八进制还是较长，所以出现的用4个二进制位表示一位的情况，四个二进制位最大是15，这就是十六进制。

F:不同进制表现同一个数据的形式特点

**进制越大，表现形式越短**

****

### 02.03\_Java语言基础(不同进制数据的表现形式)(掌握)

A:二进制的数据表现形式

由0,1组成。以0b(b可以大写也可以小写)开头(**JDK1.7版本可以表示二进制了**)

B:八进制的数据表现形式

由0,1,…7组成。以0开头

C:十进制的数据表现形式

由0,1,…9组成。整数默认是十进制的

D:十六进制的数据表现形式

由0,1,…9,a,b,c,d,e,f(大小写均可)。以0x开头（大小写都是可以的）

E:案例演示

输出不同进制表现100的数据。

0b100

0100

100

0x100

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Scale { //scale 进制

public static void main(String[] args) {

System.out.println(0b100); //4? 二进制表示形式前面加0b(b可以大写也可以小写)

System.out.println(0100); //64? 八进制表示形式前面加0

System.out.println(100);

System.out.println(0x100); //256? 十六进制表示形式前面加0x(可以大写也可以小写)

}

}

---------------------------------------------------------------------------------------------------------------------------

### 02.04\_Java语言基础(任意进制到十进制的转换图解)(了解)

A:任意进制到十进制的转换原理

系数：就是每一位上的数据。

基数：X进制，基数就是X。

权：在右边，从0开始编号，对应位上的编号即为该位的权。

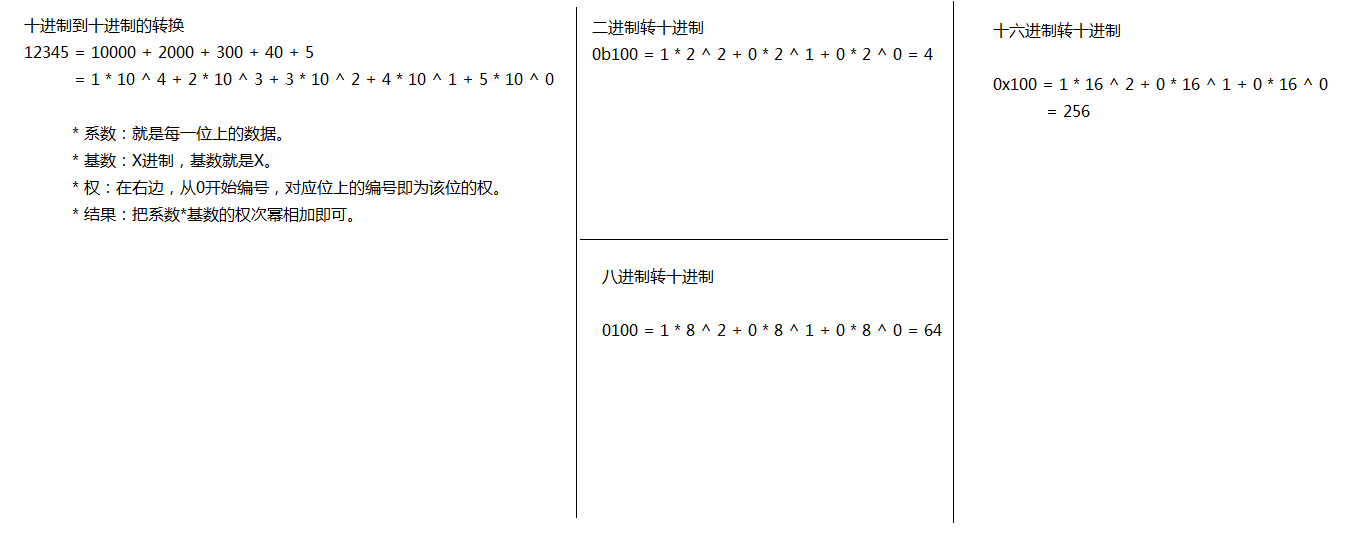
结果：把系数基数的权次幂相加即可。

B:画图练习

二进制--十进制

八进制--十进制

十六进制--十进制



### 02.05\_Java语言基础(十进制到任意进制的转换图解)(了解)

A:十进制到任意进制的转换原理

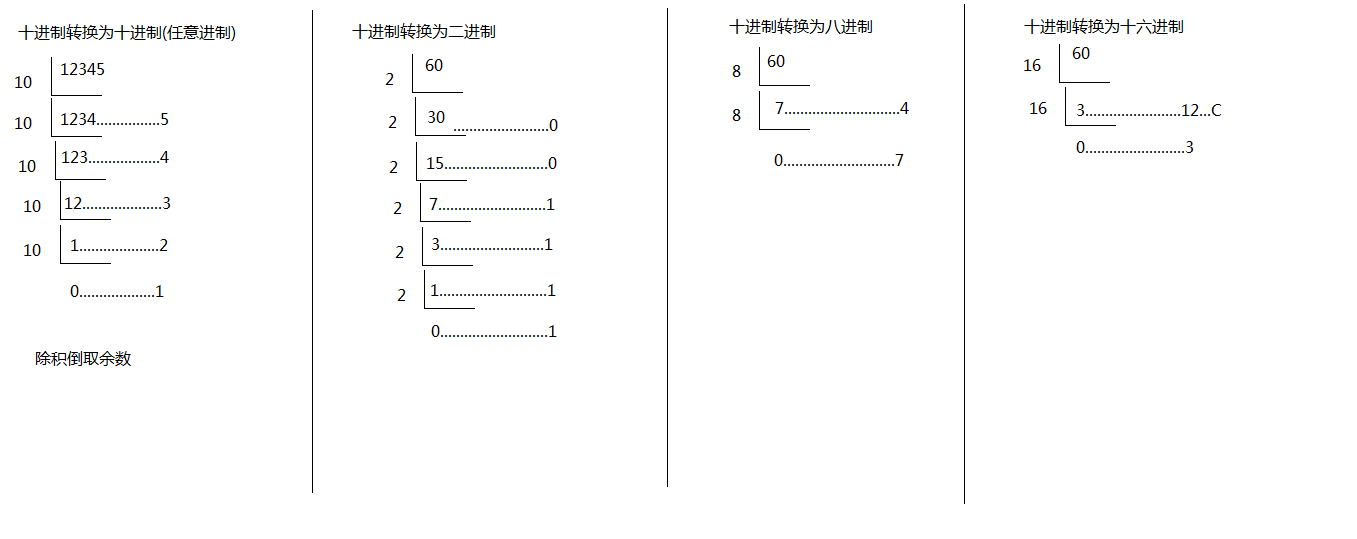
除积倒取余

B:画图练习

十进制--二进制

十进制--八进制

十进制--十六进制



### **02.06\_Java语言基础(快速的进制转换法8421码)(掌握)（仅仅只是一种快速的转换方式罢了）**

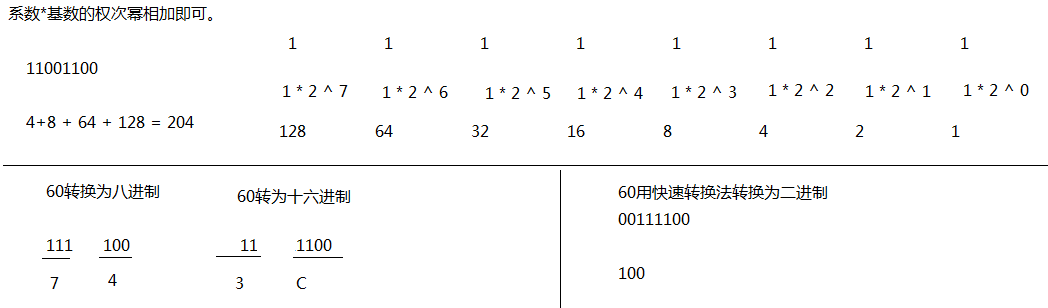
A:8421码及特点 128 64 32 16 8 4 2 1记住2^6=64

8421码是中国大陆的叫法，8421码是BCD代码中最常用的一种。在这种编码方式中每一位二值代码的1都是代表一个固定数值，把每一位的1代表的十进制数加起来，得到的结果就是它所代表的十进制数码。

B:通过8421码的方式进行二进制和十进制的相互转换

C:二进制到八进制的简易方式 从右边开始三位为一位

D:二进制到十六进制的简易方式 从右边开始四位为一位

****

### 02.07\_Java语言基础(原码反码补码)(了解)

A:为什么要学习原码反码补码?

1,强制类型转换,如果不知道有原反补会看不懂结果

2,计算机底层没有减法只有加法,减法使用的加法的补码进行计算的

-------------------

例如:

byte a = (byte)300;//byte的范围是-128~127 ,而300超过,需要进行强制转换.

300的二进制形式是:00000000  00000000  00000001  00101100

强制转换(截取)后为: 00101100    //此时为补码形式

而此时最高符号位为0,即为正数,而正数的原反补是相同的,所以最后a的值为44;

-------------------

B:有符号数据表示法的几种方式

原码

就是二进制定点表示法，即最高位为符号位，“0”表示正，“1”表示负，其余位表示数值的大小。

通过一个字节,也就是8个二进制位表示+7和-7

0(符号位) 0000111

1(符号位) 0000111 注意这个是-7的原码

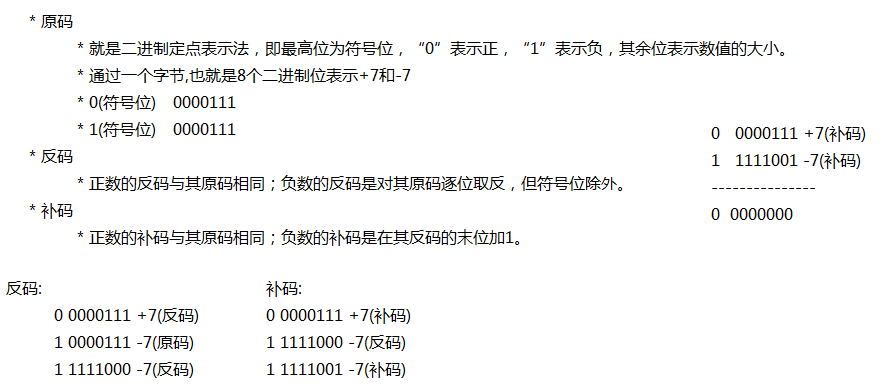
**正数的源码就是取余法,负数的源码就是其绝对值正数的源码取反加上符号位的1.**

**反码**

**正数的反码与其原码相同；负数的反码是对其原码逐位取反，但符号位除外。**

**补码**

**正数的补码与其原码相同；负数的补码是在其反码的末位加1。**



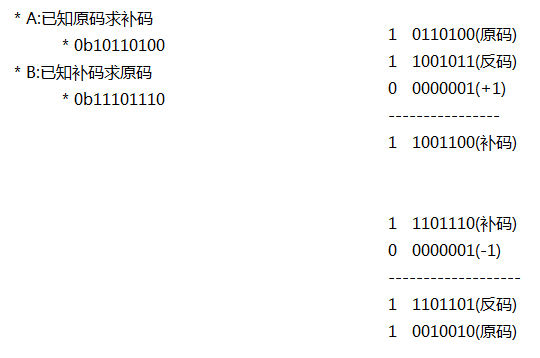
### 02.08\_Java语言基础(原码反码补码的练习)(了解)

A:已知原码求补码

0b10110100

B:已知补码求原码

0b11101110



## 变量

### 02.09\_Java语言基础(变量的概述及格式)

A:什么是变量

在程序执行的过程中，在某个范围内其值可以发生改变的量

B:变量的定义格式

**1，数据类型 变量名 = 变量值;（常用方式）**

**2，数据类型 变量名1,变量名2……**

**变量名1 = 变量值;**

**变量名2 = 变量值;**

**…….**

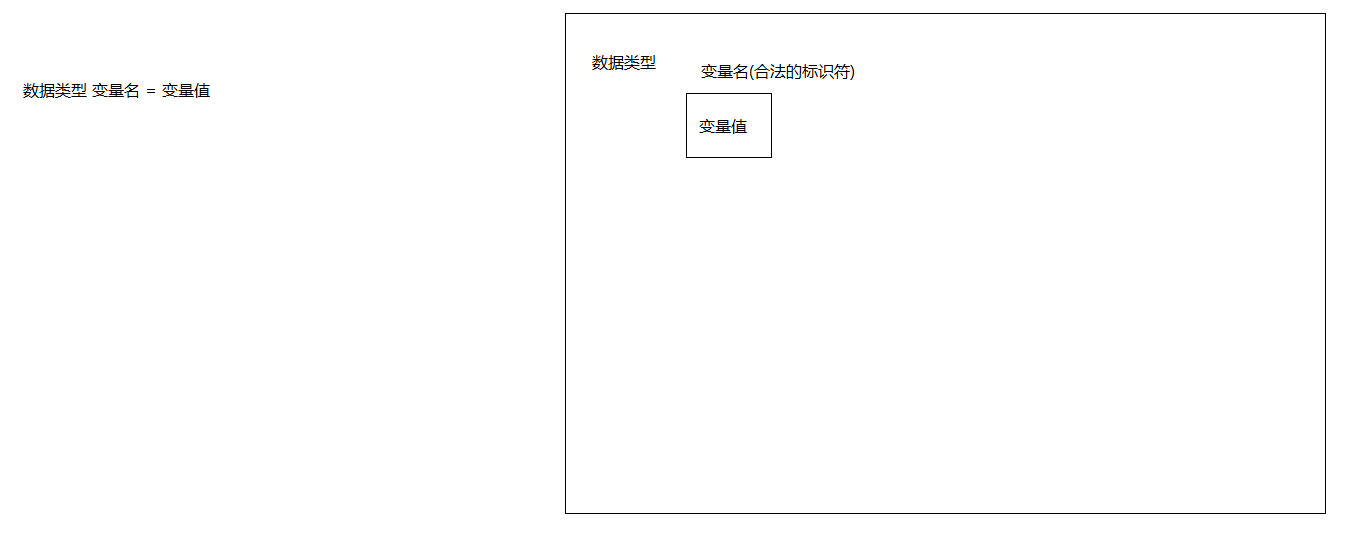
**3，数据类型 变量名1= 变量值，变量名2= 变量值;……**

**4，数据类型 变量名= （关系表达式）？ 值1：值2;9//属于是三元运算符，但是本质来说也是一种变量的声明。**

5，（等待补充）

C:为什么要定义变量

用来不断的存放同一类型的常量，并可以重复使用



### 02.12\_Java语言基础(使用变量的注意事项)

A:案例演示

a:作用域问题

同一个区域(也就是同一个大括号内的)不能使用相同的变量名

b:初始化值问题

局部变量在使用之前必须赋值,若是没有使用的话就可以不赋值

c:一条语句可以定义几个变量

int a,b,c...;

---------------------------------------------------------------------------------------------------------------------------

class Demo2\_DataType {

public static void main(String[] args) {

//int x = 10;

int x = 20; //不能进行重复的定义所以应该注释一个定义的变量。

System.out.println(x);

int y;

y = 10;

System.out.println(y);

int a,b,c,d,e;

a = 10;

b = 20;

c = 30;

d = 40;

e = 50;

System.out.println(a);

int x = 10;

int y = 20;

int z = 30;

}

}

---------------------------------------------------------------------------------------------------------------------------

## 数据类型

### 02.10\_Java语言基础(数据类型的概述和分类)

A:为什么有数据类型

Java语言是强类型语言，对于每一种数据都定义了明确的具体数据类型，在内存中分配了不同大小的内存空间

B:Java中数据类型的分类

基本数据类型

引用数据类型

面向对象部分讲解

C:基本数据类型分类(4类8种)

整数型

byte 占一个字节 -128到127 (-2^7~2^7-1)

short 占两个字 -2^15~2^15-1 (-32768~32767)

int 占四个字节 -2^31~2^31-1

long 占八个字节 -2^63~2^63-1

浮点型

float 占四个字节 -3.403E38~3.403E38

double 占八个字节-1.798E308~1.798E308

字符型

char 占两个字节 0~65535

布尔型

boolean

boolean理论上是占八分之一个字节,因为一个开关就可以决定是true和false了,但是java中boolean类型没有明确指定他的大小

### 02.11\_Java语言基础(定义不同数据类型的变量)

A:案例演示

定义不同基本数据类型的变量，并输出

**赋值时候注意float类型,long类型(注意这两种类型应该加上后标)**

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_DataType { //DataType数据类型

public static void main(String[] args) {

//整数类型

byte b = 10; //占一个字节,-128 到 127

short s = 20; //占两个字节

int i = 30; //占四个字节 整数默认的数据类型就是int类型

long x = 8888888888L; //占八个字节 如果long类型后面加L进行标识最好加大L,因为小l太像一了

System.out.println(b);

System.out.println(s);

System.out.println(i);

System.out.println(x);

//System.out.println(12345 + 54321l);

//浮点类型

float f = 12.3F; //占四个字节

double d = 33.4; //占八个字节，小数默认的数据类型是double,double类型后面也可以用D或d标识,但是一般不加

System.out.println(f);

System.out.println(d);

//字符类型

char c = 'a'; //占两个字节

System.out.println(c);

//布尔类型

boolean b1 = true;

boolean b2 = false;

System.out.println(b1);

System.out.println(b2);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 02.16\_Java语言基础(long与float的取值范围谁大谁小)

**进行混合运算的时候,byte,short,char不会相互转换,都会自动类型提升为int类型,其他类型进行混合运算的是小的数据类型提升为大的**

**byte,short,char -- int -- long -- float -- double**

long: 8个字节

float：4个字节

IEEE754

4个字节是32个二进制位

1位是符号位

8位是指数位

00000000 11111111

0到255

1到254

-126到127

23位是尾数位

每个指数位减去127

---------------------------------------------------------------------------------------------------------------------------

class Test2\_DataTypeConversion {

public static void main(String[] args) {

float f = 12.3f;

long x = 12345;

//f = x; //隐式转换

//System.out.println(f);

x = (long)f; //强制转换

System.out.println(x);

}

}

---------------------------------------------------------------------------------------------------------------------------

A:它们底层的存储结构不同。

B:float表示的数据范围比long的范围要大

long：2^63-1

float：3.410^38 > 210^38 > 28^38 = 22^3^38 = 22^114 > 2^63-1

## 数据类型的转换

### 02.13\_Java语言基础(数据类型转换之隐式转换)(掌握)

A:案例演示

a:int + int

b:byte + int

B:Java中的默认转换规则

取值范围小的数据类型与取值范围大的数据类型进行运算,会先将小的数据类型提升为大的,再运算。只有在这个地方的转换才叫做隐性转换,其他的都叫做强制转换，强制转换的格式： 变量 = （数据类型）数值。(声明变量的时候而是强制转换byte b = 5 ),有一些隐形转换时无法满足的,之后又将要进行强制转换的例如byte b=2,b=b+3. 注意这个地方.byte b ; b = 3 + 4; 这是不进行报错的,因为这个值是不超过byte的,这就是内存优化机制,若是这个值超过了byte就会报错的,还有就是byte/short/char计算的时候自动提升为int，并不是他们之间就转化为他们之间的范围大的类型。

---------------------------------------------------------------------------------------------------------------------------

class Test1\_DataTypeConversion {

public static void main(String[] args) {

//面试题:看下面的程序是否有问题，如果有问题，请指出并说明理由。

byte b1 = 3;

byte b2 = 4;

//byte b3 = b1 + b2;

/\*

从两方面

1,byte与byte(或short,char)进行运算的时候会提升为int,两个int类型相加的结果也是int类型

2,b1和b2是两个变量,变量存储的值是变化,在编译的时候无法判断里面具体的值,相加有可能会超出byte的取值范围

\*/

//System.out.println(b3);

//byte b4 = 3 + 4; //java编译器有常量优化机制

byte b4 = 7;

System.out.println(b4);

}

}

---------------------------------------------------------------------------------------------------------------------------

C:画图解释byte+int类型的问题

---------------------------------------------------------------------------------------------------------------------------

class Demo3\_DataTypeConversion { //Conversion转换

public static void main(String[] args) {

//数据类型转换之隐式转换

/\*int x = 3;

byte b = 4;

x = x + b;

System.out.println(x);\*/

**//数据类型转换之强制转换，注意会砍掉二进制数据的前边的部分！注意原理**

/\*int x = 3;

byte b = 4;

b = (byte)(x + b);

System.out.println(b);\*/

//00000000 00000000 00000000 10000010 130的二进制

//10000010 -126补码

//00000001 -1求反码

//10000001 -126反码

//11111110 -126原码

byte b = (byte)(126 + 4); **//得出来的是补码，进行翻译成源码我们和计算机才能知道他要表示的值。**

System.out.println(b);

//00000000 00000000 00000001 00101100 300的二进制

//00101100

byte b2 = (byte)300;

System.out.println(b2);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 02.14\_Java语言基础(数据类型转换之强制转换)(掌握)

A:强制转换问题

int a = 10;

byte b = 20;

b = a + b;

B:强制转换的格式

b = (byte)(a + b);

C:强制转换的注意事项

如果超出了被赋值的数据类型的取值范围得到的结果会与你期望的结果不同

D:强制转换的几种形式

eg:byte x = 5; x++; 这里边的自增就等于是 x = (byte)(x + 1),赋值运算符亦是如此.隐含了强制转换。

例题就是上边的隐式与强制转换。

### 02.15\_Java语言基础(面试题之变量相加和常量相加的区别)

A:案例演示

**面试题:看下面的程序是否有问题，如果有问题，请指出并说明理由。**

byte b1 = 3;

byte b2 = 4;

byte b3 = b1 + b2;

从两方面去回答这个题

b1和b2是两个变量,变量里面存储的值都是变化的,所以在程序运行中JVM是无法判断里面具体的值。提示“错误: 可能损失精度”

byte类型的变量在进行运算的时候,会自动类型提升为int类型

byte b4 = 3 + 4;

3和4都是常量,java有常量优化机制,就是在编译的的时候直接把3和4的结果赋值给b4了

注意：

int i1 = 1;

int i2 = 2;

int i3 = i1 + i2;

System.out.println(i3); //编译正确，这是同一种数据类型且是在默认的Int中或者之上的。

## Char数据类型,码表

### 02.17\_Java语言基础(字符和字符串参与运算)

A:案例演示

System.out.println('a');

System.out.println('a'+1);

通过看结果知道'a'的值是多少,由此引出ASCII码表

B:ASCII码表的概述

记住三个值：

'0' 48

'A' 65

'a' 97

C:案例演示

System.out.println("hello"+'a'+1);

System.out.println('a'+1+"hello");

D:+在有字符串参与中被称为字符串连接符

System.out.println("5+5="+5+5);

System.out.println(5+5+"=5+5");

---------------------------------------------------------------------------------------------------------------------------

class Demo4\_DataTypeConversion {

public static void main(String[] args) {

//System.out.println('a' + 1); //98,因为有ASCII码表,a字符对应的是int类型的97

//System.out.println((char)('a' + 1));

System.out.println("hello"+'a'+1); //任何数据类型用+与字符串相连接都会产生新的字符串

System.out.println('a'+1+"hello");

System.out.println(" 5 + 5 = " + (5 + 5));

}

}

---------------------------------------------------------------------------------------------------------------------------

### 02.18\_Java语言基础(char数据类型)

A:char c = 97; 0到65535

B:Java语言中的字符char可以存储一个中文汉字吗?为什么呢?

可以。因为Java语言采用的是Unicode编码。Unicode编码中的每个字符占用两个字节。中文也是占的两个字节

所以，Java中的字符可以存储一个中文汉字

---------------------------------------------------------------------------------------------------------------------------

class Demo5\_Char {

public static void main(String[] args) {

char c = 'a';

System.out.println(c);

byte b = 100;

char c1 = 97; //0 - 65535

System.out.println(c1);

char c2 = 3;

char c3 = 4;

char c4 = 5;

char c5 = 6;

System.out.println(c2);

System.out.println(c3);

System.out.println(c4);

System.out.println(c5);

//char类型是否可以存储中文

char c6 = '中';

System.out.println(c6);

}

}

---------------------------------------------------------------------------------------------------------------------------

## 算术运算符

### 02.19\_Java语言基础(算术运算符的基本用法)

A:什么是运算符

就是对常量和变量进行操作的符号。

B:运算符的分类

算术运算符，赋值运算符，比较(关系或条件)运算符，逻辑运算符，位运算符，三目(元)运算符

C:算数运算符有哪些

**+, -, \*, /, %, ++, --**

D:注意事项：

a:+号在java中有三种作用,代表正号,做加法运算,字符串的连接符

b:整数相除只能得到整数。如果想得到小数，必须把数据变化为浮点数类型

c:/获取的是除法操作的商，%获取的是除法操作的余数

%运算符

当左边的绝对值小于右边绝对值时,结果是左边

当左边的绝对值等于右边或是右边的倍数时,结果是0

当左边的绝对值大于右边绝对值时,结果是余数

%运算符结果的符号只和左边有关系,与右边无关

任何一个正整数%2结果不是0就是1可以用来当作切换条件

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Operator { //operator 运算符

public static void main(String[] args) {

//System.out.println(10 / 3); //整数相除结果只能是整数

//System.out.println(10 / 3.0); //如果想得到小数,把其中一个数变成小数,另一个数在运算的时候会自动类型提升

//System.out.println(-5 % 5);

System.out.println(13 % -5);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 02.20\_Java语言基础(算术运算符++和--的用法)

A:++,--运算符的作用

自加（++）自减（--）运算

++:自加。对原有的数据进行+1

--:自减。对原有的数据进行-1

B:案例演示

a:单独使用：

放在操作数的前面和后面效果一样。(这种用法是我们比较常见的)

b:参与运算使用：

放在操作数的前面，先自增或者自减，然后再参与运算。

放在操作数的后面，先参与运算，再自增或者自减。

---------------------------------------------------------------------------------------------------------------------------

class Demo2\_Operator {

public static void main(String[] args) {

//单独使用

/\*int a = 3;

//a++; //a = a + 1;

++a; //a = a + 1;

System.out.println(a);\*/

//参与运算使用

int a = 3;

int b;

//b = a++; //当++在变量后面的时候,会先将变量中的值取出做赋值操作,然后再自身加1

b = ++a; //当++在变量前面的时候,会先自身加1,然后在将结果赋值

System.out.println("a = " + a);

System.out.println("b = " + b);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 02.21\_Java语言基础(算术运算符++和--的练习)

A:案例演示

请分别计算出a,b,c的值?

int a = 10;

int b = 10;

int c = 10;

a = b++;

c = --a;

b = ++a;

a = c--;

B:案例演示

请分别计算出x,y的值?

int x = 4;

int y = (x++)+(++x)+(x10);

C:面试题

byte b = 10;

b++;

b = b + 1;

问哪句会报错,为什么

---------------------------------------------------------------------------------------------------------------------------

class Test1\_Operator {

public static void main(String[] args) {

//第一题

/\*int a = 10;

int b = 10;

int c = 10;

a = b++; //a = 10 ,b = 11

c = --a; //a = 9, c = 9

b = ++a; //a = 10, b = 10

a = c--; //a = 9, c = 8

System.out.println("a = " + a + ",b = " + b + ",c = " + c);\*/

//第二题

/\*int x = 4;

// 4 + 6+ 60

int y = (x++)+(++x)+(x\*10);

System.out.println("x = " + x + ",y = " + y);\*/

//第三题问哪句会报错,为什么

byte b = 10;

//b++; //b = (byte)(b + 1)

b = b + 1; //当byte与int进行混合运算的时候,会提升为int类型,两个int相加的结果还是int,赋值给byte会损失精度

System.out.println("b = " + b);

}

}

---------------------------------------------------------------------------------------------------------------------------

## 赋值运算符

### 02.22\_Java语言基础(赋值运算符的基本用法)

A:赋值运算符有哪些

a:基本的赋值运算符： =

把=右边的数据赋值给左边。

b:扩展的赋值运算符：+=, -=, \*=, /=, %=

+= 把左边和右边做加法，然后赋值给左边。

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Operator {

public static void main(String[] args) {

//赋值运算符

int a = 3; //把右边的常量值赋值给左边的变量,左边必须是变量

//a += 4; //a = a + 4;

a = a + 4;

System.out.println(a);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 02.23\_Java语言基础(赋值运算符的面试题)

A:案例演示

面试题:看下面的程序是否有问题，如果有问题，请指出并说明理由。

short s=1;s = s+1;

short s=1;s+=1;

---------------------------------------------------------------------------------------------------------------------------

class Test1\_Operator {

public static void main(String[] args) {

//short s=1;s = s+1; //当short与int进行运算的时候,会提升为int类型,两个int类型相加的结果也是int类型

short s=1;s+=1; //s = (short)(s + 1);

System.out.println(s);

}

}

---------------------------------------------------------------------------------------------------------------------------

## 关系运算符

### 02.24\_Java语言基础(关系运算符的基本用法及其注意事项)

A:关系运算符有哪些(比较运算符,条件运算符)

==, !=, >, >=, <, <=

注意事项：

无论你的操作是简单还是复杂，结果是boolean类型。

"=="不能写成"="。

02.25\_day02总结

把今天的知识点总结一遍。

# DAY03

## 逻辑运算符

### 03.01\_Java语言基础(逻辑运算符的基本用法)(掌握)

A:逻辑运算符有哪些（既然是逻辑运算符那么结果肯定是布尔类型的值，不是true就是false）

**&, |, ^, !**

**&&, || 开发中很常用的。**

B:案例演示

逻辑运算符的基本用法

注意事项：

a:逻辑运算符一般用于连接boolean类型的表达式或者值。

b:表达式：就是用运算符把常量或者变量连接起来的符合java语法的式子。

算术表达式：a + b

比较表达式：a == b(条件表达式)

C:结论：

**&逻辑与:遇false则false**。

**|逻辑或:遇true则true。**

**^逻辑异或:相同为false，不同为true。**

**!逻辑非:非false则true，非true则false。注意偶数个不改变本身。**

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Operator {

public static void main(String[] args) {

//&,|,^,!

//int x = 10;

//5 < x < 15

**//x > 5 & x < 15 注意此种此种写法。**

//逻辑与 & 并且and 遇false则false

int a = 10;

int b = 20;

int c = 30;

/\*System.out.println(a < b & b < c); //true & true = true

System.out.println(a < b & b > c); //true & false = false

System.out.println(a > b & b < c); //false & true = false

System.out.println(a > b & b > c); //false & false = false\*/

//逻辑或 或or 遇true则true

/\*System.out.println(a < b | b < c); //true | true = true

System.out.println(a < b | b > c); //true | false = true

System.out.println(a > b | b < c); //false | true = true

System.out.println(a > b | b > c); //false | flase = false\*/

//逻辑异或 ^ 两边相同为false,两边不同为true

/\*System.out.println(a < b ^ b < c); //true | true = false

System.out.println(a < b ^ b > c); //true | false = true

System.out.println(a > b ^ b < c); //false | true = true

System.out.println(a > b ^ b > c); //false | flase = false\*/

//逻辑非!

System.out.println(!true);

System.out.println(!!true);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.02\_Java语言基础(逻辑运算符&&和&的区别)

A:案例演示

**&&和&的区别?**

**a:最终结果一样**。

**b:&&具有短路效果。左边是false，右边不执行**。

&是无论左边是false还是true,右边都会执行

B:同理||和|的区别?(学生自学)。

两者的结果也是一样的，但是||只要左边正确右边就不再执行了。

C:开发中常用谁?

**&&,||,!最常用的三个！掌握！**

---------------------------------------------------------------------------------------------------------------------------

class Demo2\_Operator {

/\*

||与|的区别

a:最终的结果是一样

b:||具有短路效果,左边为true,右边不执行

\*/

public static void main(String[] args) {

/\*int a = 10;

int b = 20;

int c = 30;

System.out.println(a < b && b < c); //true && true = true

System.out.println(a < b && b > c); //true && false = false

System.out.println(a > b && b < c); //false && true = false

System.out.println(a > b && b > c); //false && false = false\*/

int x = 3;

int y = 4;

//System.out.println((++x == 3) & (++y == 4)); //false & false = false //注意这个地方是先自增后比较，即使是一个打印

//System.out.println("x = " + x); **//x = 4** //也是对内存有影响的，所以后边的打印是变化的！

//System.out.println("y = " + y); **//y = 5**

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

System.out.println((++x == 3) && (++y == 4)); //false & false = false

System.out.println("x = " + x); **//x = 4**

System.out.println("y = " + y); **//y = 4**

}

}

---------------------------------------------------------------------------------------------------------------------------

## 位运算(系统中做条件判断去使用)

### 03.03\_Java语言基础(位运算符的基本用法1)

A:位运算符有哪些（**所谓的位运算就是二进制位的运算**）

**&, |, ^, ~ , >>, >>>, <<**

B:案例演示

位运算符的基本用法1

&,|,^,~ 的用法

位与&:有0则0

位或|:有1则1

位异或^:相同则0，不同则1

取反~: 按位取反，是包括符号位的，就是对数字的补码进行取反的。

负数的原码变反码是除去符号位，其他位取反

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Operator {

public static void main(String[] args) {

/\*

\* &,|,^,~ 的用法

\* &:有0则0

\* |:有1则1

\* ^:相同则0，不同则1

\* ~:按位取反

\*/

System.out.println(6 & 3); //2

System.out.println(6 | 3); //7

System.out.println(6 ^ 3); //5

System.out.println(~6); //-7?

}

}

/\*

110

& 011

-----------

010

110

| 011

-----------

111

110

^ 011

-----------

101

00000000 00000000 00000000 00000110 6的原码反码补码都是本身

11111111 11111111 11111111 11111001 对6取反，其实这既是对6的补码进行的了取反之后额数值，但是不知道是多少，后边进行的计算

- 00000000 00000000 00000000 00000001 知道上边是负数，按照负数的补码就是源码取反加1之后的的数值。

---------------------------------------

11111111 11111111 11111111 11111000 反码

10000000 00000000 00000000 00000111 原码(-7)

\*/

---------------------------------------------------------------------------------------------------------------------------

### 03.04\_Java语言基础(位异或运算符的特点及面试题)

A:案例演示

位异或运算符的特点

**^的特点：一个数据对另一个数据位异或两次，该数本身不变。**总结起来有以下几种用法：1，a^b^b=a;2,a^b^a=b;3,a^a^a=a;4,a^a=不确定的。异或是可以变位置的。

B:面试题：

请自己实现两个整数变量的交换

注意：以后讲课的过程中，我没有明确指定数据的类型，默认int类型。

---------------------------------------------------------------------------------------------------------------------------

class Demo2\_Operator {

public static void main(String[] args) {

//System.out.println(5 ^ 10 ^ 10);

//System.out.println(5 ^ 10 ^ 5);

/\*

\* 请自己实现两个整数变量的交换(不需要定义第三方变量)

\* 注意：以后讲课的过程中，我没有明确指定数据的类型，默认int类型。

\*/

int x = 10;

int y = 5;

**//需要第三方变量,开发推荐用这种**

/\*int temp;

temp = x;

x = y;

y = temp;\*/

**//不需要定义第三方变量,有弊端,有可能会超出int的取值范围**

/\*x = x + y; //10 + 5 = 15

y = x - y; //15 - 5 = 10

x = x - y; //15 - 10 = 5\*/

**//不需要第三方变量,通过^来做**

x = x ^ y; // 10 ^ 5

y = x ^ y; // 10 ^ 5 ^ 5 y = 10

x = x ^ y; // 10 ^ 5 ^ 10 x = 5

System.out.println("x = " + x + ",y = " + y);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.05\_Java语言基础(位运算符的基本用法2及面试题)

A:案例演示 >>,>>>,<<的用法:

<<:左移，a左移一定的位数就是a乘以2的位数次幂，正数一定成立，但是负数就不一定了。 左边最高位丢弃，右边补齐0

>>:右移，a左移一定的位数就是a除以2的位数次幂，正负数都是可以的。 最高位是0，左边补齐0;最高为是1，左边补齐1

>>>:无符号右移 无论最高位是0还是1，左边补齐0，正数可以，负数不可以

最有效率的算出2 8的结果

---------------------------------------------------------------------------------------------------------------------------

class Demo3\_Operator {

public static void main(String[] args) {

//左移,向左移动几位就是乘以2的几次幂

//System.out.println(12 << 1); //24

//System.out.println(12 << 2); //48

/\*

00000000 00000000 00000000 00001100 12的补码

(0)0000000 00000000 00000000 000011000 24的补码

(00)000000 00000000 00000000 0000110000 48的补码

\*/

//右移,向右移动几位就是除以2的几次幂

//System.out.println(12 >> 1);

//System.out.println(12 >> 2);

/\*

00000000 00000000 00000000 00001100 12的补码

000000000 00000000 00000000 0000110(0) 6

0000000000 00000000 00000000 000011(00) 3

\*/

**//最有效率的算出2 \* 8的结果**

**System.out.println(2 << 3);**

}

}

---------------------------------------------------------------------------------------------------------------------------

## 三元运算符

### 03.06\_Java语言基础(三元运算符的基本用法)

A:三元运算符的格式

(关系表达式) ? 表达式1 : 表达式2; 或者称作是条件表达式。

B:三元运算符的执行流程

先判断关系表达式是不是成立，若是成立就将表达式1赋值给左边，若是不成了则表达式2赋值给左边，注意左边应该有一个接收赋值的的变量。

**四种情况注意方式:**

情况一:

3 > 4 ? 3 : 4;//编译报错，因为条件运算符最终会得到一个结果，我们一定要用一个变量来接收，若是直接进行打印就不会出现问题的

情况二:

int i = 3 > 4 ? 3 : 4;//编译正确

情况三:

3 > 4 ? System.out.println(3) : System.out.println(4);//编译报错

既没有进行接收，右边的输出也是不行的，要注意。

情况四:

??? a = 3 > 4 ? System.out.println(3) : System.out.println(4);//编译报错，因为条件运算符最终一定要得到一个结果，所以表达式1和表达式2里面不可以是输出语句

C:案例演示

获取两个数中的最大值

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Operator {

public static void main(String[] args) {

//(关系表达式) ? 表达式1 : 表达式2;

int x = 10;

int y = 5;

int z;

**z = (x > y) ? x : y; //注意左边有一个接收赋值的变量。**

**（int s = a > b ? a : b;这样子也是可以的！）**

System.out.println("z = " + z);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.07\_Java语言基础(三元运算符的练习)

A:案例演示

比较两个整数是否相同

B:案例演示

获取三个整数中的最大值

---------------------------------------------------------------------------------------------------------------------------

class Test1\_Operator {

public static void main(String[] args) {

//比较两个整数是否相同

/\*第一种方法

int x = 10;

int y = 10;

**//boolean b = (x == y) ? true : false;**

**boolean b = (x == y);** //左右已经是布尔运算了，右边就是他的值根本就没有什么意义的。

System.out.println("b = " + b);

第二种方法（比第一种更加的贴近开发实践！）

int a = 1;

int b = 2;

int c = 3;

int temp,max; //一般我们推荐的就是及时的定义初始值。int temp = 0；int max = 0；其实直接这么定义而下边不进行赋值的话就是错误的。当然不用不赋值就是没有问题的。

String z; //注意此处已经是String类型的值声明，多以后边是可以输出字符串的。

z=(a==b)?"两个值相同":"两个值不相同"; // 输出的两个值是不是相等，其实这就是利用三元运算符的字符串初始值的声明。

System.out.println(z);

\*/

/\*获取三个整数中的最大值

第一种方法

int a = 10;

int b = 20;

int c = 30;

**int temp = (a > b) ? a : b;//先比较任意两个数的值,找出这两个数中的最大值**

**int max = (temp > c) ? temp : c;//用前两个数的最大值与第三个数比较,获取最大**值

***//a = a > b ? a : b;***

***//a = a > c ? a : c;***

System.out.println("max =" + max);

第二种方法\*/

int max = (((a > b) ? a : b > c)) ? ((a > b) ? a : b ): c;//直接一次性的嵌套，但是要注意运算符的优先性.但是在这里三元运算符的优先性是高于其他的，也就是说temp的括号是可以不加的。

}

}

---------------------------------------------------------------------------------------------------------------------------

## Java中键盘录入

### 03.08\_Java语言基础(键盘录入的基本格式讲解)

A:为什么要使用键盘录入数据

a:为了让程序的数据更符合开发的数据

b:让程序更灵活一下

B:如何实现键盘录入呢?

先照格式来。

a:导包

格式：

import java.util.Scanner;

位置：

在class上面。

b:创建键盘录入对象

格式：

Scanner sc = new Scanner(System.in); //注意只创建一次。下边可以录入很多次，但是要注意变量是不相同的就可以了。

c:通过对象获取数据

格式：

int x = sc.nextInt();

简而言之就是一下的方式进行的：

分三步:

步骤一:

导包: import java.util.Scanner;(class 类名的上面)

步骤二:

创建对象: Scanner sc = new Scanner(System.in);

步骤三:

调用方法: int a = sc.nextInt();

C:案例演示

键盘录入1个整数，并输出到控制台。

键盘录入2个整数，并输出到控制台。

### 03.09\_Java语言基础(键盘录入的练习1)

A:案例演示

键盘录入练习：键盘录入两个数据，并对这两个数据求和，输出其结果

B:案例演示

键盘录入练习：键盘录入两个数据，获取这两个数据中的最大值

---------------------------------------------------------------------------------------------------------------------------

import java.util.Scanner; //导入包中的类Scanner

class Test2\_Scanner {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); //创建键盘录入对象

//键盘录入练习：键盘录入两个数据，并对这两个数据求和，输出其结果

/\*System.out.println("请输入第一个整数:");

int x = sc.nextInt(); //将键盘录入的数据存储在x中

System.out.println("请输入第二个整数:");

int y = sc.nextInt(); //将键盘录入的数据存储在y中

int sum = x + y;

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

//键盘录入练习：键盘录入两个数据，获取这两个数据中的最大值

System.out.println("请输入第一个整数:");

int x = sc.nextInt(); //将键盘录入的数据存储在x中

System.out.println("请输入第二个整数:");

int y = sc.nextInt(); //将键盘录入的数据存储在y中

int max = (x > y) ? x : y; //获取x和y中的最大值

System.out.println("max = " + max);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.10\_Java语言基础(键盘录入的练习2)

A:案例演示

键盘录入练习：键盘录入两个数据，比较这两个数据是否相等

B:案例演示

键盘录入练习：键盘录入三个数据，获取这三个数据中的最大值

---------------------------------------------------------------------------------------------------------------------------

import java.util.Scanner; //导包

class Test3\_Scanner {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); //创建键盘录入对象

//键盘录入练习：键盘录入两个数据，比较这两个数据是否相等

/\*System.out.println("请输入第一个整数:");

int x = sc.nextInt(); //将键盘录入的数据存储在x中

System.out.println("请输入第二个整数:");

int y = sc.nextInt(); //将键盘录入的数据存储在y中

//boolean b = (x == y)? true : false;

boolean b = (x == y);

System.out.println(b); 注意这样子是很不人性化的，记得用字符串类型的输出\*/

//键盘录入练习：键盘录入三个数据，获取这三个数据中的最大值

System.out.println("请输入第一个整数:");

int x = sc.nextInt(); //将键盘录入的数据存储在x中

System.out.println("请输入第二个整数:");

int y = sc.nextInt(); //将键盘录入的数据存储在y中

System.out.println("请输入第三个整数:");

int z = sc.nextInt(); //将键盘录入的数据存储在z中

//定义临时变量记录住比较出前两个变量中的最大值

int temp = (x > y) ? x : y;

//将比较后的结果与第三个变量中的值比较,比较出三个数中的最大值

int max = (temp > z) ? temp : z;

System.out.println(max);

}

}

---------------------------------------------------------------------------------------------------------------------------

## 结构控制语句

### 03.11\_Java语言基础(顺序结构语句)

A:什么是流程控制语句

流程控制语句：可以控制程序的执行流程。

B:流程控制语句的分类

顺序结构 就是正常的顺序输出

选择结构 if 和 switch

循环结构 while 和 for

C:执行流程：

从上往下，依次执行。

D:案例演示

输出几句话看效果即可

### 03.12\_Java语言基础(选择结构if语句格式1及其使用)

A:选择结构的分类

if语句

switch语句

B:if语句有几种格式

格式1

格式2

格式3

C:if语句的格式1

if(比较表达式) {

语句体;

}

比较或者关系表达式，只有一个正确的表达式输出结果，若是错误就什么都没有了，很不人性化。若是只想选择一个就没什么问题了。

D:执行流程：

**先计算比较表达式的值，看其返回值是true还是false。比较运算符和逻辑运算符的结果都是Boolean类型的，返回值也就说直接是true或者false也是可以的！**

如果是true，就执行语句体；

如果是false，就不执行语句体；

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_If {

public static void main(String[] args) {

int age = 17;

if (age >= 18) { //注意if后边是不加分号的，一条语句的时候可以省略，但是不建议，因为可能会发生未知的错误。

System.out.println("可以浏览本网站");

}

System.out.println("完了");

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.13\_Java语言基础(选择结构if语句注意事项)

**A:案例演示**

***a:比较表达式无论简单还是复杂，结果必须是boolean类型,或者值就是false和true在里边进行赋值运算也是可以的！***

b:if语句控制的语句体如果是一条语句，大括号可以省略；

如果是多条语句，就不能省略。建议永远不要省略。

c:一般来说：有左大括号就没有分号，有分号就没有左大括号

d：表达式后边是没有分号的，然后加上了分号那么就等于是if语句已经结束，且后边括号中的内容是没有任何的限制直接执行的。

e：这只能是一种情况的语句，不能增加的。

---------------------------------------------------------------------------------------------------------------------------

class Demo2\_If {

public static void main(String[] args) {

int age = 17;

if (age >= 18 && age <= 60) { //比较运算符和逻辑运算符的结果都是Boolean类型的。

System.out.println("可以浏览本网站");

**//int x = 10; 是两句话,int x声明是一句,x = 10 赋值是一句**

}

System.out.println("完了");

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.14\_Java语言基础(选择结构if语句格式2及其使用)

A:if语句的格式2

if(比较表达式) {

语句体1;

}else {

语句体2;

}

非正即错的表达方式，还是选择太少，不符合日常生活.但是在复杂的逻辑中注意其是一个利器.

B:执行流程：

首先计算比较表达式的值，看其返回值是true还是false。

如果是true，就执行语句体1；

如果是false，就执行语句体2；

C:案例演示

a:获取两个数据中较大的值

b:判断一个数据是奇数还是偶数,并输出是奇数还是偶数

注意事项：else后面是没有比较表达式的，只有if后面有。

---------------------------------------------------------------------------------------------------------------------------

class Demo3\_If {

public static void main(String[] args) {

/\*int x = 0;

if (x == 1) {

System.out.println("男厕所欢迎您");

}else {

System.out.println("女厕所欢迎您");

}\*/

//a:获取两个数据中较大的值

/\*int x = 10;

int y = 20;

int z;

if (x > y) {

z = x;

}else {

z = y;

}

System.out.println(z);\*/

//b:判断一个数据是奇数还是偶数,并输出是奇数还是偶数

int num = 11;

if (num % 2 == 0) { //也可以写成：**if (num >> 1 == 0),位运算是更快的。**

System.out.println(num + "是一个偶数");

}else {

System.out.println(num + "是一个奇数");

}

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.15\_Java语言基础(if语句的格式2和三元的相互转换问题)

A:案例演示

if语句和三元运算符完成同一个效果

B:案例演示

if语句和三元运算符的区别

三元运算符实现的，都可以采用if语句实现。反之不成立。

**什么时候if语句实现不能用三元改进呢?**

**当if语句控制的操作是一个输出语句的时候就不能**。

为什么呢?因为三元运算符是一个运算符（也就是说他只是一个数字，而不是一种输出语句，若是想达到同一种效果，是要改变三元运算符的某些内容和声明的，是这样吗，等待求证？），运算符操作完毕就应该有一个结果，而不是一个输出。

---------------------------------------------------------------------------------------------------------------------------

class Demo4\_If {

public static void main(String[] args) {

/\*

int x = 10;

int y = 20;

if (x > y) {

System.out.println(x + "是最大值");

}else {

System.out.println(y + "是最大值");

}

\*/

/\*

int x = 10;

int y = 20;

String z;

int z = (x > y)?System.out.println(x + "是最大值"):System.out.println(y + "是最大值");

System.out.println(z);

\*/ //这个到底是怎么写的呢？目前还是乖乖的听老师的吧

/\*

三元运算符

int a = 20;

int b = 30;

int c = (a > b)? a : b;

System.out.println(c);

\*/

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.16\_Java语言基础(选择结构if语句格式3及其使用)

A:if语句的格式3：

if(比较表达式1) { 如果

语句体1;

}else if(比较表达式2) { 否则如果

语句体2;

}else if(比较表达式3) { 否则如果

语句体3;

}

...

else { 什么都不是那就是

语句体n+1;

} 很不错，这种贴合实际，但是注意最后的else是可以省略的，一般不建议省略，因为这里可以写上备胎，也就是当用户输入错误的时候，我们来一个温馨提示，保证了软件的健壮性。

B:执行流程：

首先计算比较表达式1看其返回值是true还是false，

如果是true，就执行语句体1，if语句结束。

如果是false，接着计算比较表达式2看其返回值是true还是false，

如果是true，就执行语句体2，if语句结束。

如果是false，接着计算比较表达式3看其返回值是true还是false，

如果都是false，就执行语句体n+1。

**C:注意事项:最后一个else可以省略,但是建议不要省略,可以对范围外的错误值提示**

---------------------------------------------------------------------------------------------------------------------------

class Demo5\_If {

public static void main(String[] args) {

int x = 2;

if (x == 1) {

System.out.println("男厕所欢迎您");

}else if (x == 0) {

System.out.println("女厕所欢迎您");

}else {

System.out.println("无法识别您的性别");

}

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.17\_Java语言基础(选择结构if语句格式3练习)

A:练习1

需求：键盘录入一个成绩，判断并输出成绩的等级。

90-100 优

80-89 良

70-79 中

60-69 及

0-59 差

B:练习2

需求：

键盘录入x的值，计算出y的并输出。

x>=3 y = 2 x + 1;

-1<x<3 y = 2 x;

x<=-1 y = 2 x - 1;

---------------------------------------------------------------------------------------------------------------------------

import java.util.Scanner;

class Test1\_If {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

/\*需求：键盘录入一个成绩，判断并输出成绩的等级。

System.out.println("请输入学生成绩范围在1到100之间");

int x = sc.nextInt();

if (x >= 90 && x <= 100) { //注意此处的短路与 && 的运用，如果x < 90直接排除，若是用 & ，就会判断后边，虽然结果是一样的但是增加了计算的时间，软件没有达到最优。

System.out.println("优");

}else if (x >= 80 && x <= 89 ) {

System.out.println("良");

}else if (x >= 70 && x <= 79 ) {

System.out.println("中");

}else if (x >= 60 && x <= 69 ) {

System.out.println("及");

}else if (x >= 0 && x <= 59 ) {

System.out.println("差");

}else {

System.out.println("成绩录入错误");

}\*/ // 也是可以运用switch的，只要你不嫌麻烦。但是有一些情况你未必能写全哦！例如下边的边界问题。

//需求： 键盘录入x的值，计算出y的并输出

System.out.println("请输入一个整数:");

int x = sc.nextInt();

int y = 0; //注意这个地方y不可能包含了全部的情况，只要你写条件肯定有没有写到的！若是这个地方实在是不想写初始值那么就在后边写上else，那么不管怎么样都会执行一个的，备胎的存在还是有意义的。所以说为了避免出错所有的变量都要在开始的时候有一个初始值。**若是没有进行初始化值那么就是编译报错的，因为语法是错误的，下边条件中的方式是不会进行判断的！**

if (x >= 3) {

y = 2 \* x + 1;

}else if (x > -1 && x < 3) {

y = 2 \* x;

}else if (x <= -1) {

y = 2 \* x - 1;

}

System.out.println(y);

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.18\_Java语言基础(选择结构if语句的嵌套使用)

A:案例演示

需求：获取三个数据中的最大值

if语句的嵌套使用。每一次消灭一种情况然后再在后边进行排序，就是一种大换套小环的结构。注意若是获取四个值中的最大值，那么情况比较多可能会用到较多的if选择了，主要就是看你怎么给他分情况了。

---------------------------------------------------------------------------------------------------------------------------

class Demo6\_IfIf {

public static void main(String[] args) {

int a = 40;

int b = 50;

int c = 30;

if (a > b) {

if (a > c) {

System.out.println(a + "是最大值");

}else {

System.out.println(c + "是最大值");

}

}else { //b >= a

if (b > c) {

System.out.println(b + "是最大值");

}else {

System.out.println(c + "是最大值");

}

}

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.19\_Java语言基础(选择结构switch语句的格式及其解释)

A:switch语句的格式

switch(表达式) {

case 值1：

语句体1;

break;

case 值2：

语句体2;

break;

…

default：

语句体n+1;

break;

}

B:switch语句的格式解释

C:面试题：表达式可以是 byte，short，char，int，1.5之后的枚举，1.7之后的String字符串

byte可以作为switch的表达式吗?

long可以作为switch的表达式吗?

String可以作为switch的表达式吗?

C:执行流程

先计算表达式的值

然后和case后面的匹配，如果有就执行对应的语句，否则执行default控制的语句

---------------------------------------------------------------------------------------------------------------------------

class Demo1\_Switch {

public static void main(String[] args) {

String name = "rose" //多个人就要用到变量，若是一个人直接可以打印的时候是字符串。

String gender = "妖";

**switch (gender) { //这就是字符串**

case "男士":

System.out.println(name + "是一位" + gender + "喜欢吃饭睡觉打dota");

break;

case "女士":

System.out.println(name + "是一位" + gender + "喜欢逛街购物美容");

break;

default:

System.out.println(name + "是一位" + gender + "打雌性激素维持美貌容颜");

break;

}

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.20\_Java语言基础(选择结构switch语句的练习)

A:整数(给定一个值,输出对应星期几)

---------------------------------------------------------------------------------------------------------------------------

class Test1\_Switch {

public static void main(String[] args) {

int week = 1;

switch (week) {

case 1:

System.out.println("星期一");

break;

case 2:

System.out.println("星期二");

break;

case 3:

System.out.println("星期三");

break;

case 4:

System.out.println("星期四");

break;

case 5:

System.out.println("星期五");

break;

case 6:

System.out.println("星期六");

break;

case 7:

System.out.println("星期日");

break;

default:

System.out.println("对不起没有对应的星期");

break;

}

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.21\_Java语言基础(选择结构switch语句的注意事项)

A:案例演示

a:case后面只能是常量，不能是变量，而且，多个case后面的值不能出现相同的

b:default可以省略吗?

可以省略，但是不建议，因为它的作用是对不正确的情况给出提示。也TM是一个备胎。

特殊情况：

case就可以把值固定。

A,B,C,D

c:break可以省略吗?

最后一个可以省略,其他最好不要省略

中间的省略会出现一个现象：case穿透。只会穿透到下一个，但是前提是下一个的语句之后有break（下一个的条件是不进行判断直接执行的！）。

最终我们建议不要省略

d:default一定要在最后吗?

不是，可以在任意位置。但是建议在最后。我认为倒是可以放在最前边，因为不会忘记，有了备胎就不会出现意外的情况。但是前提是一定要有break语句，小心打乱了后边的程序。

e:switch语句的结束条件

a:遇到break就结束了

b:执行到switch的右大括号就结束了

### 03.22\_Java语言基础(选择结构switch语句练习)

A:看程序写结果：

int x = 2;

int y = 3;

switch(x){

default:

y++;

break;

case 3:

y++;

case 4:

y++;

}

System.out.println("y="+y); //4

B:看程序写结果：

int x = 2;

int y = 3;

switch(x){

default:

y++;

case 3:

y++;

case 4:

y++;

}

System.out.println("y="+y); //6

### 03.23\_Java语言基础(选择结构if语句和switch语句的区别)

A:总结switch语句和if语句的各自使用场景

switch建议判断固定值的时候用

if建议判断区间或范围的时候用

B:案例演示

分别用switch语句和if语句实现下列需求：

键盘录入月份，输出对应的季节

---------------------------------------------------------------------------------------------------------------------------

import java.util.Scanner;

class Test3\_SwitchIf {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in); //创建键盘录入对象

System.out.println("请输入月份");

int month = sc.nextInt(); //将键盘录入的结果存储在month

/\*switch (month) {

case 3:

case 4:

case 5:

System.out.println(month + "月是春季");

break;

case 6:

case 7:

case 8:

System.out.println(month + "月是夏季");

break;

case 9:

case 10:

case 11:

System.out.println(month + "月是秋季");

break;

case 12:

case 1:

case 2:

System.out.println(month + "月是冬季");

break;

default:

System.out.println("对不起没有对应的季节");

break;

}\*/

//用if语句来完成月份对应季节

**if (month > 12 || month < 1) {**

System.out.println("对不起没有对应的季节"); //特殊情况先判断，也可以放在后边，只是因为这里的12,1,2不是一个范围罢了！

}else if (month >= 3 && month <= 5) {

System.out.println(month + "月是春季");

}else if (month >= 6 && month <= 8) {

System.out.println(month + "月是夏季");

}else if (month >= 9 && month <= 11) {

System.out.println(month + "月是秋季");

}else {

System.out.println(month + "月是冬季");

}

}

}

---------------------------------------------------------------------------------------------------------------------------

### 03.24\_day03总结

把今天的知识点总结一遍。

今天常见的代码梳理：

DAY04 for,(do)while,break.continue,方法，重载

### 04.01\_Java语言基础(循环结构概述和for语句的格式及其使用)

A:循环结构的分类

for,while,do...while

B:循环结构for语句的格式：（for循环一般是用在已经知道的循环次数的情况中）

for(初始化表达式;条件表达式;循环后的操作表达式) {

循环体;

}

for (int i = 1;i <= 10 ;i++ ) {

System.out.println("helloworld");

}

C执行流程：

a:执行初始化语句

b:执行判断条件语句,看其返回值是true还是false（就是执行初始化语句是不是符合条件表达式，这里的是返回值是不是真假并不是条件表达式是不是真假的意思）//注意这里的条件表达式只要是为Boolean类型的true就行了，和前边的初始化表达式可以不产生关系，若是产生了关系就限定了循环的次数，条件表达式/（循环后的操作表达式-初始化表达式）=循环次数。条件表达式默认为true。

如果是false，就结束循环

c:执行循环体语句;

d:执行循环后的操作表达式(这一句放在循环体的最后也是可以的)

e:回到b继续。

f:初始化表达式中的变量最后是被释放的，不影响后边的重复定义，同时后边若是想用到也是不可以的。

g：初始化表达式可以放在外边。

h：最简单的无限循环。

for(;;){

语句体1；

}

D:案例演示

在控制台输出10次"helloworld"

for (int i = 1;i <= 10 ;i++ ) {

System.out.println("helloworld");

for (int i = 1;; ) {

System.out.println("helloworld");

}

### 04.02\_Java语言基础(循环结构for语句的练习之获取数据)

**A:案例演示**

需求：请在控制台输出数据1-10

需求：请在控制台输出数据10-1

找出下边程序的bug：

class Noname1 {

public static void main(String[] args) {

for (int x = 0;x <= 10 ;x++ ) {

System.out.println("x = "+ x);

int sum = 0;  **//for循环中的变量是出不去循环体的，所以说的这**

**里的sum根本是无法打印出来的。**

sum += x;

}

System.out.println("sum = "+sum);//其实前边出不来，这里就无法打

印，根本就没有定义变量。所以会编译错误。

for (int x = 10;x >= 0 ;x-- ) {

System.out.println("x = "+ x);

}

}

}

B:注意事项

a:判断条件语句无论简单还是复杂结果是boolean类型。

b:循环体语句如果是一条语句，大括号可以省略；如果是多条语句，大括号不能省略。建议永远不要省略。这点和if是一样一样的.

c:一般来说：有左大括号就没有分号，有分号就没有左大括号

### 04.03\_Java语言基础(循环结构for语句的练习之求和思想)

**A:案例演示**

需求：求出1-10之间数据之和

B:学生练习

需求：求出1-100之间偶数和

需求：求出1-100之间奇数和

写出上边的的三个需要代码

int x = 1;

int sum = 0;

for( x = 1;x <= 10;x++ ){

sum += x;

}

System.out.println("sum = "+sum);

----------

int x = 1;

int sum = 0;

for( x = 0;x <= 100;x++ ){ //也是可以在i++上边做手脚的!

if ( x % 2 == 0) {

sum += x;

}

}

System.out.println("sum = "+sum);

----------

int x = 1;

int sum = 0;

for( x = 0;x <= 100;x++ ){

if ( x % 2 != 0) {

sum += x;

}

}

System.out.println("sum = "+sum);

### 04.04\_Java语言基础(循环结构for语句的练习之水仙花)

**A:案例演示**

需求：在控制台输出所有的”水仙花数”

所谓的水仙花数是指一个三位数，其各位数字的立方和等于该数本身。

举例：153就是一个水仙花数。

153 = 1\*1\*1 + 5\*5\*5 + 3\*3\*3 = 1 + 125 + 27 = 153

for (int i = 100;i <= 999 ;i++ ) {

int ge = i % 10; //123 % 10

int shi = i / 10 % 10; //12 % 10;

int bai = i / 10 / 10 % 10; //1 % 10

if (ge ge ge + shi shi shi + bai bai bai == i) {

System.out.println(i);

}

}

### 04.05\_Java语言基础(循环结构for语句的练习之统计思想)

**A:案例演示**

需求：统计”水仙花数”共有多少个

class Test4\_FlowerCount {

public static void main(String[] args) {

***int count = 0; //不能放在里边是因为每一次都进行了返回赋值***

for (int i = 100;i <= 999 ;i++ ) {

int ge = i % 10;

int shi = i / 10 % 10;

int bai = i / 10 / 10;

if (i == ge ge ge + shi shi shi + bai bai bai) {

count ++; //满足条件就自增,计数器思想

}

}

System.out.println(count);

}

}

### 04.06\_Java语言基础(循环结构while语句的格式和基本使用)

A:循环结构while语句的格式：

while循环的基本格式：

while(判断条件语句) {

循环体语句;

}

完整格式：和if一样判断语句至少执行一次也比其他的多一次

**初始化语句;**

**while(判断条件语句) {**

**循环体语句;**

**控制条件语句;**

**}**

B:执行流程：

a:执行初始化语句

b:执行判断条件语句,看其返回值是true还是false

如果是true，就继续执行

如果是false，就结束循环

c:执行循环体语句;

d:执行控制条件语句

e:回到B继续。

C:案例演示

需求：请在控制台输出数据1-10

class Demo1\_While {

public static void main(String[] args) {

int x = 1;

while (x <= 10) {

**System.out.println("x = " + x); //在这里写x++也可以的**

x++;

}

}

}

### 04.07\_Java语言基础(循环结构while语句的练习)

A:求和思想

求1-100之和

int sum = 0;

int i = 1;

while (i <= 100) {

sum += i; //sum = sum + i;

i++; //让变量i自增

}

System.out.println("sum = " + sum);

B:统计思想

统计”水仙花数”共有多少个

int count = 0; //计数器

int i = 100;

while (i <= 999) {

int ge = i % 10;

int shi = i / 10 % 10;

int bai = i / 100;

if (i == ge ge ge + shi shi shi + bai bai bai) {

count ++;

}

i++;

}

System.out.println("count =" + count);

附加:

int j = 1;

while (j <= 10000) { **//若是将左大括号前边加上分号,将会无限循环,执行不下去的,憋死了.**

System.out.println("I Love You!!!");

j++;

}

### 04.08\_Java语言基础(循环结构do...while语句的格式和基本使用)

A:循环结构do...while语句的格式：

do {

循环体语句;

}while(判断条件语句);

完整格式；

初始化语句;

do {

循环体语句;

控制条件语句;

}while(判断条件语句);

B:执行流程：

a:执行初始化语句

b:执行循环体语句;

c:执行控制条件语句

d:执行判断条件语句,看其返回值是true还是false

如果是true，就继续执行

如果是false，就结束循环

e:回到b继续。

C:案例演示

需求：请在控制台输出数据1-10

int i = 1;

do {

System.out.println("i = " + i); **//i最后是11了，注意这一点。**

i++;

}

while (i <= 10);

### 04.09\_Java语言基础(循环结构三种循环语句的区别)

A:案例演示

**三种循环语句的区别:**

do...while循环至少执行一次循环体。

而for,while循环必须先判断条件是否成立，然后决定是否执行循环体语句。

B:案例演示

**重点:**for循环和while循环的区别：

A:如果你想在循环结束后，继续使用**控制条件的那个变量**，用while循环(do..while循环也是一样的也是没有释放变量的)，否则用for循环。不知道用谁就用for循环。因为变量及早的从内存中消失，可以提高内存的使用效率。

int j = 11;

while (j <= 10) { //先判断不满足就无法输出

System.out.println("j = " + j);

j++;

}

for (int i = 1;i <= 10 ;i++ ) { **//注意最后的i是11，但是释放了，后边无法使用。**

System.out.println("i = " + i);

### 04.10\_Java语言基础(循环结构注意事项之死循环)

A:一定要注意控制条件语句控制的那个变量的问题，不要弄丢了，否则就容易死循环。

B:两种最简单的死循环格式

while(true){...} //while里边默认的是false.

for(;;){...}

---------------------------------------------------------------------------------------------------------------------------

//while语句的无限循环

/\*while (true) {

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

}\*/

**//System.out.println("hello world"); //永远无法执行到,程序编译报错!用于对象的无限录入,再利用选择,当遇到quiet时候讲退出.**

//for语句的无限循环

for (; ; ) {

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

}

---------------------------------------------------------------------------------------------------------------------------

### 04.11\_Java语言基础(循环结构循环嵌套输出4行5列的星星)

A:案例演示

需求：请输出一个4行5列的星星(\*)图案。

如图：

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

注意：System.out.println("\*");和System.out.print("\*");的区别

for (int i = 1;i <= 4 ;i++ ) { //外循环决定的是行数

for (int j = 1;j <= 5 ;j++ ) { //内循环决定的是列数

System.out.print("\*");

}

System.out.println();

}

B:结论：

外循环控制行数，内循环控制列数

### 04.12\_Java语言基础(循环结构循环嵌套输出正三角形)

A:案例演示

需求：请输出下列的形状

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

for (int i = 1;i <= 5 ; i++) { //外循环决定行数

for (int j = 1;j <= i ;j++ ) { //内循环决定列数

System.out.print("\*");

}

System.out.println(); //将光标换到下一行的行首

### 04.13\_Java语言基础(循环结构九九乘法表)

A:案例演示

需求：在控制台输出九九乘法表。

for (int i = 1;i <= 9 ;i++ ) { //行数

for (int j = 1;j <= i ;j++ ) { //列数

System.out.print(j + "\*" + i + "=" + (i j) + "\t" );

}

System.out.println();

}

B:代码优化

注意：

'\x' x表示任意，\是转义符号,这种做法叫转移字符。

'\t' tab键的位置

'\r' 回车

'\n' 换行

'\"'

'\''

### 04.14\_Java语言基础(控制跳转语句break语句)

A:break的使用场景

**只能在switch和for循环中,不能在其他环境或者使用。break下边不能写任何语句,因为不能访问到,就像无限循环那个一样.**

for (int x = 1;x <= 10 ;x++ ) {

if (x == 4) {

break; //跳出当前循环，if只是选

}

System.out.println("x = " + x); //输出1,2,3，当x=4的时候就跳出。

### 04.15\_Java语言基础(控制跳转语句continue语句)

A:continue的使用场景

**只能在循环中,下边也是不能写任何的语句.**

for (int x = 1;x <= 10 ;x++ ) {

if (x == 4) {

continue; //终止本次循环,继续下次循环

}

System.out.println("x = " + x); //输出1,2,3,5,6....

### 04.16\_Java语言基础(控制跳转语句标号)

标号:标记某个循环对其控制

标号组成规则:其实就是合法的标识符

**outer:** for (int i = 1;i <= 10 ;i++ ) { **//a就是标号,只要是合法的标识符即可**

System.out.println("i = " + i);

**inner:** for (int j = 1;j <= 10 ;j++ ) {

System.out.println("j = " + j);

**break outer; //本来只是跳出当前的，加了标记就是标记处。**

}

}

System.out.println("大家好");

http://www.heima.com //后边的双斜线是注释，前边的http：只是一种标记！

System.out.println("才是真的好");

会输出：大家好

才是真的是好

### 04.17\_Java语言基础(控制调整语句练习)

A:练习题

for(int x=1; x<=10; x++) {

if(x%3==0) {

//break; //return结束方法与也是可以的 //输出2次:“Java基础班“

//continue; //输出7次:“Java基础班“

System.out.println("Java基础班");//输出13次:“Java基础班“

}

System.out.println(“Java基础班”);

}

### 04.18\_Java语言基础(控制跳转语句return语句)

A:return的作用

返回

它的作用不是结束循环的，而是结束方法的。

B:案例演示

return和break以及continue的区别?

return是结束方法

break是跳出循环

continue是终止本次循环继续下次循环

for (int i = 1;i <= 10 ;i++ ) {

if (i == 4) {

//break; //停止循环，输出“循环结束了”

return; //返回的意思,用来返回方法，整个函数都停止了！没有后边的输出了。

}

}

System.out.println("循环结束了");