Java语言基础组成

1.关键字

关键字的定义和特点				
定义：被Java语言赋予了特殊含义的单词 特点：关键字中所有字母都为小写				
用于定义数据类型的关键字				
class	interface	byte	short	int
long	float	double	char	boolean
void				
用于定义数据类型值的关键字				
true	false	null		
用于定义流程控制的关键字				
if	else	switch	case	default
while	do	for	break	continue
return				

用于定义访问权限修饰符的关键字				
private	protected	public		
用于定义类，函数，变量修饰符的关键字				
abstract	final	static	synchronized	
用于定义类与类之间关系的关键字				
extends	implements			
用于定义建立实例及引用实例，判断实例的关键字				
new	this	super	instanceof	
用于异常处理的关键字				
try	catch	finally	throw	throws
用于包的关键字				
package	import			
其他修饰符关键字				
native	strictfp	transient	volatile	assert

2.标识符

(1)就是给类,变量,方法起名字。

(2)组成规则：

由数字0-9,英文大小写字母,$以及_组成。

(3)注意事项：

A:只能有组成规则规定的内容组成。

B:不能以数字开头。

C:不能是Java中的关键字。

D:区分大小写。

A:包(文件夹，用于区分同名的类)

全部小写。如果多级包，用.分开。

举例：

com一级包

itheima.interview二级包

注意：www.itheima.com(域名反写)

B:类和接口

如果是一个单词,首字母大写。

举例：

Demo,Test,Student

如果是多个单词,每个单词的首字母大写。

举例：

HelloWorld,StudyJava

C:变量和方法

如果是一个单词,首字母小写。

举例：

main(),name,age.show().method()

如果是多个单词,从第二个单词开始每个单词的首字母大写。

举例：

showName(),studentName()

D:常量

全部大写，如果多个单词组成，用_连接。

举例：

PI,STUDENT_MAX_AGE

3.注释

注释：就是对程序的解释性文字。

注释的分类：

A:单行注释

a:以//开头，以回车结束。

b:单行注释是可以嵌套单行注释的。

B:多行注释

a:以/*开头，以*/结束。

b:多行注释是不可以嵌套多行注释的，但可以嵌套单行注释。

C:文档注释(了解)

将来被Javadoc工具解析，生成一个说明书。

注释的作用：

A:解释程序，提高程序的阅读性。

B:可以调试错误。

对于单行注释和多行注释，被注释的文字，不会被JVM（Java虚拟机）解释执行。

对于文档注释，是Java特有的注释，其中注释内容可以被JDK提供的工具javadoc所解析，生成一套以网页文件形式体现的该程序的

说明文档。

注释是一个程序员必须具有的良好编程习惯。

初学者编写程序可以养成习惯：先写注释再写代码。

将自己的思想通过注释先整理出来，在用代码去体现。

因为代码仅仅是思想的一种体现形式而已。

作用：

1、注解说明

2、调试程序

格式:

单行注释：//

多行注释： /* */

文档注释： /** */

建议：写代码之前写需求，思路，步骤，养成一个良好的习惯，以后一旦思想确定，后面写代码就是行云流水。

如下一个Hello 黑马程序员的小程序：

```
/*
1.需求：
    编译一个Hello 黑马程序员！！的小程序
2.思路：
    1).通过Java关键字class定义一个类
    2).通过Java特殊字main定义一个主函数
    3).定义一个输出语句，打印下Hello 黑马程序员！！
3.步骤：
    1).class Demo 定义一个类
    2).public static void main(String[] args) 定义一个主函数
```

```
        3).System.out.println("Hello 黑马程序员！！");定义一个输出语句
    */


    //创建一个类。
    class Demo{
        //主函数是程序的入口。
        public static void main(String[] args){
            //输出语句,可以打印小括号内容。
            System.out.println("Hello 黑马程序员！！");
        }
    }
```

4.常量和变量

常量：表示不能改变的数值。

(1)在程序的运行过程中，其值是不可以发生改变的量。

(2)常量的分类：

1:整数常量

12,-23

2:小数常量

12.5,-65.43

3:字符常量

'a','A','0'

4:字符串常量

"Hello"

5:布尔常量

true,false

6:空常量(后面讲)

null

自定义常量(后面讲)

(3)常量可以直接被输出。

(4)进制

(1)是一种进位的方式。X进制，表示逢x进1。

(2)Java中整数常量的表示

A:二进制 由0,1组成。以0b开头。JDK 1.7以后的新特性。

B:八进制 由0-7组成。以0开头。

C:十进制 由0-9组成。默认就是十进制。

D:十六进制 由0-9，A-F(不区分大小写)组成，以0x开头。

(3)进制转换：

A:其他进制转十进制

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:十进制到其他进制

除基取余，直到商为0，余数反转。

变量：就是讲不确定的数据进行存储，也就是需要在内存中开辟一个空间。

(1)程序的运行过程中，在指定范围内发生改变的量。

(2)格式：

格式1:

数据类型 变量名 = 初始化值;

格式2:

数据类型 变量名;

变量名 = 初始化值;

举例：

方式1：

```
byte b = 10;
```

方式2：

```
byte b;

b = 10;
```

理解：变量就如同数学中的未知数。

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

数据分类

1)基本类型：4类8种。

整型：占用字节

byte(1)short(2)int(4)long(8)

布尔型：

boolean(1) 不明确。可以认为是1个字节。

只有俩字结果，一个是true，另一个是false。

字符型：

char(2)

浮点型：

float(4)double(8)

2)引用类型：类，接口，数组。(先了解)

注意：

整数默认是int类型。long类型需要加L或者l后缀。

浮点数默认是double类型。float类型需要加F或者f后缀。

类型转换

1、boolean类型不参与转换。（因为boolean值是常量，默认只有俩个值，要么true，要么false）

2、隐式转换(从小到大)

自动类型提升，从小到大的提升。

(byte,short,char) --> int --> long --> float --> double

3、强制转换(从大到小)

强制类型转换，从大到小的转换。

intb;b=(byte)(b+2)

格式：

(数据类型)数据;

表达式的数据类型自动提升

所有byte型、short型和char的值将被提升到int型。

如果一个操作数是long型，计算结果就是long型。

如果一个操作数是float型，计算结果就是float型。

如果一个操作数是double型，计算结果就是double型。

分析：

System.out.println('a')与System.out.println('a'+1)的区别。

System.out.println('a')打印的是a。

System.out.println('a'+1)打印的是98，自动类型提升，把a转换成ASCII编码值(97)与1相加。

变量的什么时候定义？

当数据不确定时，需要对数据进行存储时，就定义一个变量来完成存储动作。

转义字符

\n :换行

\b:退格

\r:按回车

\t:制表符

面试题：

```
byte b = 4;b = 3+7;
System.out.println(b);//打印出来是10
```

```
byte b = 4;byte b1 = 3;byte b2 = 7;
b = b1 + b2;
System.out.println(b);
```

提示可能损失精度

byte b=4; 编译器在编译的时候，右边默认是整型，但是他会判断.这数字在不在byte字节这个范围之内(-128~127)，如果在的话，把这个4做了一个默认的强转，把最后一个字节赋到b中，这时b是在byte范围内所以编译不会报错。

b1和b2是变量，是不确定的值，不能检查。

```
int x;
int x1 =Integer.MAX_VALUE;//2147483647Z最大数
int x2 = 2;
x = x1+x2;
System.out.println(x);
```

不出错，打印出来是负数。

int是默认类型，一旦超出范围，底层有强制转换过程，只保留自己原有位，高位全舍弃。

5.运算符

(1)就是把常量和变量连接的符号，一般参与运算使用。

(2)分类：

算术运算符 +,-,*,/,%,++,--

+)：正号，加法，字符串连接符。

```
System.out.println("5+5="+5+5);//5+5=55
System.out.println(5+5+"=5+5");//10=5+5
```

%)：取得余数

左边如果大于右边，结果是余数。

左边如果小于右边，结果是左边。

左边如果等于右边，结果是0。

正负号跟左边一致。

++与--：

++ 其实相当于把数据+1

- - 其实相当于把数据-1

单独使用：在数据的前后，结果一致。

参与操作使用：

如果在数据的后边，数据先操作，在++/--

如果在数据的前边，数据先++/--，在操作。

赋值运算符 =,+=,-=,*=,/=,%=

int a = 10;

把10赋值给int类型的变量a。

a += 20;

把左边和右边的和赋值给左边。

注意事项：

a = a + 20;

a += 20;

结果是等价的，理解不是等价的。

因为+=这种运算符，内含了强制类型转换功能。

比如：

short s = 2;

s+=3;

等价于 s = (short)(s+3);

关系运算符 ==,!=,>,>=,<,<=

特点：关系运算符的结果都是boolean类型。

请千万注意：== 不要写成 =

逻辑运算符

&与运算 |或运算 ^异或运算 !非运算 &&短路与运算 ||短路或运

&:符号的运算特点：

true & true = true;

true & false = false;

false & true = false;

false & false = false;

&:运算规律：

&运算的两边只有有一个是false，结果肯定是false。

只有两边都为true，结果才是true。

|:符号的运算特点：

true & true = true;

true & false = true;

false & true = true;

false & false = false;

|:运算规律：

|运算的两边只要有一个是true，结果肯定是true。

只有两边都为false。结果是false。

^:异或 ：和或有点不一样。

^:运算特点。

true ^ true = false;

true ^ false = true;

false ^ true = true;

false ^ false = false;

^异或的运算规律：

^符号的两边结果如果相同，结果是 false。

两边的结果不同，结果是true。

!:非运算，判断事物的另一面。

!true =false

!false=true;

!!true=true;

面试题：

&&和&运算的结果是一样的。但是运算过程有点小区别。

&：无论左边的运算结果是什么，右边都参与运算。

&&：当左边为false时，右边不参与运

||和|运算的结果是一样的。但是运算过程有点小区别。

|：无论左边的运算结果是什么，右边都参与运算。

||：当左边为true时，右边不参与运算的。

位运算是直接对二进制进行运算

<<左移运算 >>右移运算 >>>无符号右移运算 &与运算 |或运算 ^异或运算 ~反码运算

位运算符的细节	
<<	空位补0，被移除的高位丢弃，空缺位补0。
>>	被移位的二进制最高位是0，右移后，空缺位补0； 最高位是1，空缺位补1。
>>>	被移位二进制最高位无论是0或者是1，空缺位都用0补。
&	二进制位进行&运算，只有1&1时结果是1，否则是0;
|	二进制位进行 | 运算，只有0 | 0时结果是0，否则是1;
^	相同二进制位进行 ^ 运算，结果是0；1^1=0，0^0=0 不相同二进制位 ^ 运算结果是1。1^0=1，0^1=1

<<左移几位其实就是该数据乘以2的几次方。左移：可以完成2的次幂运算

>>右移几位其实就是该数据除以2的几次幂。右移：对于高位出现的空位，原来高位是什么就用什么补这个空位。

>>>：无符号右移：数据进行右移时，高位出现的空位，无论原高位是什么，空位都用0补。

^一个数异或同一个数两次，结果还是这个数。

练习：

对两个整数变量的值进行互换（不需要第三方变量）

开发时，使用第三方变量的形式，因为阅读性强。

```
int a = 3,b = 5, c;

c = a ;a = b ;b = c ;

Sytem.out.println("a="+a+",b="+b);
```

这种方式不要用，如果两个整数的数值过大，会超出int范围，会强制转换。数据会变化。

```
a= a + b;//a = 3 + 5;a = 8;

b= a - b;//3+5-5 = 3;b = 3;

a= a - b;//3+5-3 = 5;a = 5;
```

面试的时候用。

```
a= a ^ b;//a = 3 ^ 5;

b= a ^ b;//b = (3^5)^5; b = 3;

a= a ^ b;//a = (3^5)^3; a = 5;
```

三元运算符

格式：（条件表达式）?表达式1:表达式2 ;

如果条件为true，运算后的结果是表达式1，

如果条件为flase，运算后的结果是表达式2

示例：获取两个数中大数。

```
int x = 3, y = 4, z;

z = (x>y) ? x:y;  //z变量存储的就是两个数的大数。
```