2021.4.24

赋值运算符

=，+=，-=，*=，/=，%=等.

特点：

1) 顺序从右向左

2) 等号左边为变量，右侧可以是常量、表达式、变量

3) 复合赋值运算符会进行类型转换

```
byte b = 2;
b += 2; // 等价 b = (byte)(b+2); 底层的强制类型转换
```

三元运算符

基本语法：条件表达式？表达式1：表达式2；

运算规则：条件为true输出1，为false输出2.

细节：① 表达式1和表达式2必须与接收变量的类型相符(或可以自动转换)

② 底层实现为if-else语句.

运算符优先级：

单目运算符(++,--)，赋值运算符从右向左运算.

优先级	运算符	结合性
1	()、[]、{}	从左向右
2	!、+、-、~、++、--	从右向左
3	*、/、%	从左向右
4	+、-	从左向右
5	«、»、>>>	从左向右
6	<、<=、>、>=、instanceof	从左向右
7	==、!=	从左向右
8	&	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

P82 - P85 Java_标识符和关键字保留字

标识符的命名规则和规范

定义：1> Java 对各种变量、方法和类等命名时使用的字符序列
2> 凡是可以自己起名字的地方

标识符命名规则：（必须遵守）

1> 由26个英文大小写、0-9、_或$组成
2> 数字不可开头
3> 不可以使用关键字和保留字，但能包含关键字和保留字.
4> 严格区分大小写，长度无限制
5> 标识符不能包含空格.

标识符命名规范：（更加专业）

1> 包名：多单词组成时所有字母都小写，如 com.hsp.crm
2> 类名、接口名：多单词组成时，所有单词首字母都大写，如 TankShotGame（大驼峰）
3> 变量名、方法名：多单词组成时，第一个单词首字母小写，第二个单词开始首字母大写，
如 tankShotGame（小驼峰，简称驼峰法）
4> 常量名：所有字母都大写，多个单词时用下划线连接，如 TAX_RATE

关键字

定义：被Java语言赋予了特殊含义，用做专门用途的字符串（单词）
特点：字母都为小写

保留字

定义：现有Java版本尚未使用，但以后版本可能作为关键字使用
例：byValue，cast，future，generic，inner，operator，outer，rest，var，goto，const

P86 Java_键盘输入语句

步骤：1>导入该类所在的包 java.util.*
2>创建该类对象
3>调用功能（方法）

P87 - P Java_进制(4种)

进制的转换:

其他进制转十进制

abcde(n进制) → $a\times n^{5-1}+b\times n^{4-1}+c\times n^{3-1}+d\times n^{2-1}+e\times n^{1-1}$

0b1010 → $1\times 2^{4-1}+0\times 2^{3-1}+1\times 2^{2-1}+0\times 2^{1-1}=8+2=10$

0234 → $2\times 8^{3-1}+3\times 8^{2-1}+4\times 8^{1-1}=128+24+4=156$

0x23A → $2\times 16^{3-1}+3\times 16^{2-1}+10\times 16^{1-1}=512+48+10=570$

10 11 12 13 14
a b c d e

十进制转其他进制

34 → 0B00100010

2|34	0
2|17	1
2|8	0
2|4	0
2|2	0
1	1

131 → 0203

8|131	3
8|16	0
2	2

237 → 0xED

16|237	13→d
14	e

二进制转八、十六进制

二转八: 从低位开始将三位看成一组,转成对应的八进制数.

二转十六: 从低位开始将四位看成一组,转成对应的十六进制数

0b 110 101 01 → 0325 (3 2 5)

0b 1101 0101 → 0xD5 (13 5)

八、十六转二进制

八转二: 将八进制数每一位,转成对应的一个3位二进制数

十六转二: 将十六进制数每一位,转成对应的一个4位二进制数

0237 八转二→ 0b 010 011 111

0x23B 十六转二→ 0b 0010 0011 1011

原码、反码、补码 ! 重点、难点

对于有符号的而言:

1> 二进制的最高位是符号位: 0表示正, 1表示负 (0正1负)

2> 正数的原码,反码,补码都一样(三码合一)

3> 负数的反码 = 原码符号位不变,其他位取反(0→1, 1→0)

4> 负数的补码 = 反码+1, 负数的反码 = 补码-1

5> 0的反码、补码都是0

6> java没有无符号数, java中所有数都有符号

7> 在计算机运算的时候,都是以补码的方式来运算的 -> 原因: 补码把正、负联系起来

8> 当我们看运算结果时,要看原码.

位运算符

共7个位运算符（&、|、^、~、>>、<< 和 >>>）

按位与 &：两位全为1，结果为1，否则为0

按位或 |：两位有一个1，结果为1，否则为0

按位异或 ^：两位一个为0，一个为1，结果为1，否则为0

按位取反 ~：0->1，1->0

详解见：0424/BitOperator.java

算术右移 >>：低位溢出，符号位不变，并用符号位补溢出的高位

算术左移 <<：符号位不变，低位补0

逻辑右移 >>>（无符号右移）：低位溢出，高位补0

无 <<< 符号及运算规则

本章作业

1. 计算下列表达式的结果

10/3 = ? ; 10/5 = ? ; 10%2 =? ; -10.5%3 = ?;

3　　2　　0

a%b = a - a/b × b

= a - (int)a / b × b

= -10.5 - (-10) / 3 × 3

= -10.5 + 9 = -1.5

a%b 公式.

小数参与运算结果为近似值.

2. 试说出下面代码的结果

```
int i=66;
System.out.println(++i+i);
```

(++i) + i　　i = 66->67

67 + 67 -> 134

3. 在Java中，以下赋值语句正确的是()。

A) int num1=(int)"18"; × Integer.parseInt("18")

B) int num2=18.0; × double -> int

C) double num3=3d; ✓

D) double num4=8; ✓

E) int i=48; char ch = i+1; × (i+1)类型为int

F) byte b = 19; short s = b+2; × 为 int -> shor