Day 03

2018年5月2日 8:37

课程回顾:

1.用某种编辑工具写java代码

2.通过jdk的工具,在jdk的安装目录中的bin目录 javac编译java代码

编译完后的文件是字节码文件.class文件

3.利用jdk的工具,java命令,执行.class文件

4.jdk中有jre,jre中有jvm, .java编译一次 到处运行,前提必须有对应操作系统jvm

5.带包编译,执行

javac -d

java -cp

6.

-公有类的名字必须跟文件名相同

-.java文件中公有类只能有一个,但是类可以定义多个

-main是程序的入口,可以有多个,但一个java文件中最好只有一个入口

```
public static void main(String[] args){
    //程序的主业务逻辑
}
```

7.java的标识符(就是名称)

驼峰命名:从第二个单词往后所有的单词第一个字母大写,其余的时小写

帕斯卡命名:所有单词的首字母都大写

匈牙利命名: 类型的缩写+见名知意的名称 int iage;

java没有匈牙利命名

8.注释:

```
//单行
/*
  多行注释
*/
/**
*文档注释,可以通过javadoc命令生成文档
*/
```

9.数据的表示

二进制(0b),八进制(0),十进制,十六进制(0x)

10.进制转换

十-->非十

整数部分:除权取余,倒着取余数

小数部分:乘权取整,正着取整数

非十---->十

整数部分从右往左标序号,本位乘以权位序号次方

八 --------二 ---------- 十六

三个二进制位对应一个八进制位

四个二进制对应一个十六进制位

11.java中的数据表示

原码,反码和补码

12.java中的数据表示(针对程序员)

常量:固定不变的量

变量:量值可以改变, 先定义后使用

类型 变量的标识符

13.八个基本数据类型

标记了,程序要申请多大的内存,以及在空间中能存储多大的数据

byte: 1个字节

short: 2个字节

int: 4个字节

long: 8个字节 l

float: 4个字节 f

double:8个字节 d

char:2个字节 用单引号

boolean:根据版本字节不同

引用数据类型:

数组,类,接口

数据类型之间的转换

转换的前提,一定类型类似,比如,数字/数值

byte---->short---->int----->long

char---->int---->long

float----->double

隐式转换: int i=10; long l=i;//隐式转换

int k=100;

float f=k+10.1f;

隐式转换另一种说法,向上造型

强制/显示转换:大类型的数据转小类型的数据

long l=100l;

```
int i=(int)l;
double d=123.4;
float f=(float)d;
int k=(int)d;
```

结论:轻易不要强制转换,对于数值来说,可能丢失精度,除非能接收精度不准确

有那么几个特殊的类型转换 (针对考试,不针对开发)

```
short s='a';
```

```
char c='a';
short s1=c;
```

Type mismatch: cannot convert from char to short

类型 不匹配 不能 转换 从 char 到 short

char 2个字节 0---65535

short 2个字节 --32768----32767

算数运算符:+ - * / % ++ --

byte/short/char在运算的时候会自动的提升为int类型

整数运算完成之后的结果一定是整数

当小类型和大类型一起运算的时候结果一定是大的类型

```
byte b=4+1;
```

结果是byte类型

```
int i=4;  byte b=i+1;//错的,
```

除0问题:

整数/0 算数异常 java.lang.ArithmeticException: / by zero

带有小数/0 结果:"Infinity"

非零数/0.0 结果:"Infinity"

0.0/0 结果:NaN not a number 不是一个数字

0/0.0 结果:NaN not a number 不是一个数字

0.0/0.0 结果:NaN not a number 不是一个数字

%:取余数,取模

7%3=1 2%5=2

-18%-5=-3 --- 对于负数的取余，先忽略符号按照正数取余来运算，运算完成之后看%左边数字的符号，如果左边为正，结果为正，左边为负，结果为负。

注意：小数无法精确运算，绝大部分小数在内存中无法精确存储。

java中有一个java类BigDecimal类,来处理小数的精确度

++,-- 自增,自减

i++;相当于 i=i+1; 加1操作 先用变量的值后加1

i--;相当于 i=i-1; 减1操作 先用变量的值后减1

++i;相当于 i=i+1; 加1操作 先给变量加1,在使用变量的值(加过了)

--i;相当于 i=i-1; 减1操作 先给变量减1,在使用变量的值(减过了)

int i=3;

int k=i++; 结果:k=3 i=4;

int i=3;

int k=++i; 结果: k=4 i=4;

赋值运算符: = += -= *= /= %= &= |= ^= <<= >>= >>>=

= 是把等号右边运算完的结果赋值等号左边的变量中

i+=2 相当于i=i+2

```
int i=5;
i += i -= i*= i++;  -> i=-15
i = 5 + ( 5 - (5 * 5));
```

关系运算符:== != >= <= > <

结果一定布尔类型 true和false

3>5 false

3<5 true

instanceof 运算符 instanceof左边的对象是否是右边的类类型的数据

"abc" instaceof String true

123 instanceof int //错的

123 instanceof Integer //错的

new Integer(123) instanceof Integer true

逻辑运算符:&与---And ,|或---Or ,!非---Not, ^异或---Xor, &&--短路与, ||--短路或

true&true=true,

true&false=false ,

false&true=false ,

false&false=false

（同为true，结果才为true,只要有一个false结果就为false）

true|true=true，

true|false=true ，

false|true=true ，

false|false=false

（有一个为true，结果就为true）

!true=false ，!false=true（ ！非，取反）

true^true=false，

true^false=true ，

false^true=true ，

false^false==false

（两个不同为true,相同的为false）

&&：如果&&左边的结果为false，那么&&右边的表达式就不再运算---短路

||：如果||左边的结果为true，那么||右边的表达式就不再运算

注意：||可以把&&短路掉，但是&&不能把||短路掉。

&& 优先级级大于||

位运算: &与， |或， ^异或 ，<<左移， >>右移 ，>>>无符号右移，

~取反

位运算规则:

6 0110 &9 1001 0 0000	6 0110 |9 1001 15 1111
&:将数据转化为其补码形式，然后低位次对齐，将1看作true，将0看作false，进行按位与操作，最后将结果转化为十进制显式 任何一个数&偶数=偶数 判断一个数字的奇偶性	|:将数据转化为其补码形式，然后低位次对齐，将1看作true，将0看作false，进行按位或操作，最后将结果转化为十进制显式 任何一个数|奇数=奇数

6 0110 ^9 1001 15 1111
^:将数据转化为其补码形式，然后低位次对齐，将1看作true，将0看作false，进行按位异或操作，最后将结果转化为十进制显式 a^b^b=a

<< 左移位,把数据转换成二进制的补码,然后再左移对应的位数

高位移出的部分舍弃,低位空出部分用0补位

>>右移位,把数据转换成二进制的补码,然后再右移对应的位数

低位移出的部分舍弃,高位空出部分,如果正数补0

如果是负数补1

3 0000 0011 3

3<<2 0000 1100 12

左移位一次乘2

12 0000 1100 12

12>>2 0000 0011 3

右移位一次除2

>>> 无符号右移位, 高位空出一律补0,低位部分舍弃

~ 取反,把数据转换成二进制补码,按位取反,就是二进制(补码)

三元运算符: ? :

```
int i=10;
int j=5;

int result= (i>j)? i : j;
```

结论:三元运算符的结果一定是":"两边的数据之一

条件为真值就取":"左侧的值,

条件为假值就取":"右侧的值

必须有一个变量来接收三元运算符的结果

":"两边的数据类型要一致

java程序执行流程:

1.保证程序是顺序执行, 从上执行到下,一直执行到末尾,

中间不能返回执行,程序重新执行是可以的(铁律)

2.在顺序执行的基础上,会有分支结构

3.在顺序执行的基础上,会有循环结构

所有的程序都遵守这三种基本结构,顺序,选择,循环

选择分支结构: if结构 if:如果

语法:

```
if(条件){
    //代码块(很多条java语句),完成某个功能
}
```

说明:

条件:返回结果是boolean类型数据(关系运算符,逻辑运算符)

条件返回结果为真值true就执行代码块

条件返回结果为假值false就执行if(){}后面的代码

代码块:可以是多条java代码,代表某个功能

代码块也可以是一条语句,如果是一条语句,大括号可以省略

如果没有大括号,if体的语句,离if最近的那条语句

语法:

```
if(条件){
    //代码块1(很多条java语句),完成某个功能
}else{
    //代码块2(很多条java语句),完成某个功能
}
```

说明:

条件:返回结果是boolean类型数据(关系运算符,逻辑运算符)

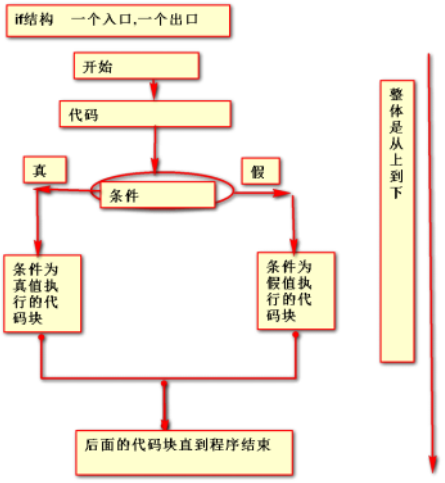
条件返回结果为真值true就执行代码块1,然后执行if(){}else{}后面的代码

条件返回结果为假值false就执行代码块2,然后执行if(){}else{}后面的代码

代码块:可以是多条java代码,代表某个功能

代码块也可以是一条语句,如果是一条语句,大括号可以省略

如果没有大括号,if体的语句,离if最近的那条语句



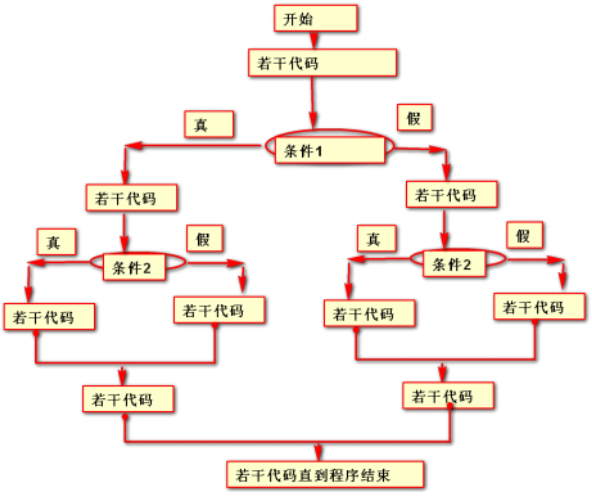
if嵌套

语法

```
if(条件1){
    if(条件2){

    }else{
    }
}else{
    if(条件3){
```

```
        }else{
        }
    }
```



if嵌套层数过多,不是你不懂if结构,你忘了最顶端的条件是啥了

```
If(女){
    if(年龄大){
        if(钱多){
        }else{
            if(漂亮女儿){

            }else{
                不要
            }
        }
    }else{
        if(没钱){
            if(特别漂亮){
            }else{
                if(会持家){
                }else{
                    不要
                }
            }
        }else{
            if(有钱){
                if(难看){
                }
            }
        }
}
```

```
    }
}else{
    If(爹有钱){
        If(小鲜肉){
        }else{

        }
    }else{
        If(本人有钱){
            If(很丑){

            }
        }
    }
}
```

从键盘录入数据:

```
Scanner input=new Scanner(System.in);
int java=input.nextInt();
```

说明:

Scanner是java系统的一个类

Scanner input=new Scanner();创建一个Scanner的对象

System.in,代表是从键盘获取数据

input是scanner的对象名,可以随意起名,但符合标识符的命名规则

input.nextInt();从键盘获取一个整数　next下一个　Int整数

int java=input.nextInt();键盘数据的整数存储给java变量

补充内容:

点击project----clean, 把bin目录中的所有的class文件删除

重新编译,生成class文件

```
Scanner input =new Scanner(System.in);
int i=input.nextInt();接收整型数据
```

double d=input.nextDouble();接收double数据

String s=Input.next();从键盘接收的字符串

运算符的优先级:从高到低

() ++ -- ~ ！ 算数运算符 << >> >>> 关系运算符 && || ^ 三元 赋值

eclipse的快捷键：

ctrl+c 复制

ctrl+v 粘贴

ctrl+a 全选

ctrl+z 撤销

ctrl+y 取消撤销

ctrl+x 剪切

ctrl+d 删除光标所在的行，删除所选择的内容

ctrl+/ 注释和取消注释

ctrl+shift+/ 多行注释

ctrl+f 打开查找和替换的窗口

ctrl+h 打开复杂查找窗口

ctrl+w 关闭当前文档

ctrl+shift+w 关闭所有已经打开的文档窗口

ctrl+s 保存当前文档

ctrl+shift+s 全部保存

ctrl+-> 光标右移一个单词，遇到空格就移动一个空格

shift+-> 选择一个字符

ctrl+shift+-> 右移一个单词，且选中

shift+home 从当前光标位置一直选择到行头

shift+end 从当前光标位置一直选择到行尾

home 定位光标到行头

end 定位光标到行尾

ctrl+shift+home 从当前光标位置一直选择到文件头

ctrl+shift+end 从当前光标位置一直选择到文件尾

ctrl+shift+o　导入类所需要包

ctrl+shift+f　格式化源代码

alt+shift+z　给选择代码添加模板

f2　选择后按f2是更名