语言基础第三天：

回顾：

1. 变量：存数的

```
int a;  int b,c;
int a = 5;  int a;  a = 5;
int b = a+10; System.out.println(b);
a = a+10;
```

2. 八种基本数据类型：byte、short、int、long、float、double、boolean、char
    - int：整型，4个字节，5，25，250.....
    - long：长整型，8个字节，5L，100000000000L.....
    - double：浮点型，8个字节，5.0，3.14，25.678.....
    - boolean：布尔型，1个字节，true，false
    - char：字符型，2个字节，'你'，'y'，'6'，'*'.....
3. 类型间的转换：
    - 两种方式：
        - 自动/隐式类型转换：小到大
        - 强制类型转换：大到小
            - (要转换成为的数据类型)变量
            - 有可能溢出或丢失精度
    - 两点规则：

```
short s1 = 5;
short s2 = 6;
short s3 = (short)(s1+s2);
```

精华笔记：

1. 运算符：
    - 算术：+，-，*，/，%，++，--
    - 关系：>、<、>=、<=、==、!=
    - 逻辑：&&、||、!
    - 赋值：=、+=、-=、*=、/=、%=
    - 字符串连接：+
    - 条件：boolean? 数1：数2
2. 分支结构(上)：基于条件执行
    - if结构：1条路
    - if...else结构：2条路
    - if...else if结构：多条路

笔记：

1. 运算符：
   - 算术：+，-，*，/，%，++，--
     - %:取模/取余，余数为0即为整除

```
System.out.println(5%2); //1，商2余1
System.out.println(8%2); //0，商4余0----整除
System.out.println(2%8); //2，商0余2
```

     - ++/--:自增1/自减1，可在变量前也可在变量后
       - 单独使用时，在前在后都一样
       - 被使用时，在前在后不一样
         - a++的值为a----------(a--的值为a)
         - ++a的值为a+1--------(--a的值为a-1)

```
//演示++单独使用
int a=5,b=5;
a++; //相当于a=a+1
++b; //相当于b=b+1
System.out.println(a); //6
System.out.println(b); //6

//演示++被使用
int a=5,b=5;
int c = a++; //将a++的值5赋值给c，同时a自增1变为6
int d = ++b; //将++b的值6赋值给d，同时b自增1变为6
System.out.println(a); //6
System.out.println(b); //6
System.out.println(c); //5
System.out.println(d); //6

//演示--单独使用:
int a=5,b=5;
a--; //相当于a=a-1
--b; //相当于b=b-1
System.out.println(a); //4
System.out.println(b); //4

//演示--被使用:
int a=5,b=5;
int c = a--; //将a--的值5赋值给c，同时a自减1变为4
int d = --b; //将--b的值4赋值给d，同时b自减1变为4
System.out.println(a); //4
System.out.println(b); //4
System.out.println(c); //5
System.out.println(d); //4
```

   - 关系：>、<、>=、<=、==、!=

```
1)>(大于)、<(小于)
  >=(大于或等于)、<=(小于或等于)
  ==(等于)、!=(不等于)
2)关系运算的结果为boolean型，
  关系成立则为true，关系不成立则为false
```

```
int a=5,b=10,c=5;
boolean b1 = a>b;
System.out.println(b1);   //false
System.out.println(c<b);  //true
System.out.println(a>=c); //true
System.out.println(a<=b); //true
System.out.println(a==c); //true
System.out.println(a!=c); //false

System.out.println(a%2==0); //false
System.out.println(a+c>b);  //false
System.out.println(a++>5); //false------a自增1变为6
System.out.println(a++>5); //true-------a自增1变为7
```

- 逻辑：&&、||、！
  - 逻辑运算是建立在关系运算的基础之上的
    逻辑运算的结果也是boolean型
  - &&：短路与(并且)，两边都为真则为真，见false则false
    - 第1个条件为false时，则发生短路(后面的不执行了)

      ```
      int a=5,b=10,c=5;
      boolean b1 = b>=a && b<c;
      System.out.println(b1);          //true&&false=false
      System.out.println(b<=c && b>a); //false&&true=false
      System.out.println(a==b && c>b); //false&&false=false
      System.out.println(b!=c && a<b); //true&&true=true
      int age = 99;
      System.out.println(age>=18 && age<=50); //年龄在18到50之间
      int score = 86;
      System.out.println(score>=0 && score<=100); //成绩在0到100之间

      //演示短路:
      boolean b3 = a>b && c++>2;
      System.out.println(b3); //false
      System.out.println(c);  //5，发生短路了
      ```

  - ||：短路或(或者)，有真则为真，见true则true
    - 第1个条件为true时，则发生短路(后面的不执行了)

```
int a=5,b=10,c=5;
System.out.println(b>=a || b<c); //true||false=true
System.out.println(b<=c || b>a); //false||true=true
System.out.println(b!=c || a<b); //true||true=true
System.out.println(a==b || b<c); //false||false=false
int score = 90;
System.out.println(score<0 || score>100); //成绩不合法验证(不在0到100之间)

//演示短路:
boolean b3 = a<b || c++>2;
System.out.println(b3); //true
System.out.println(c);  //5，发生短路了
```

- !：逻辑非(取反)，非真则假，非假则真

```
int a=5,b=10,c=5;
boolean b2 = !(a<b);
System.out.println(b2);     //!true=false
System.out.println(!(a>b)); //!false=true
```

- 赋值：=、+=、-=、*=、/=、%=
  - 简单赋值运算符：=
  - 扩展赋值运算符：+=,-=,*=,/=,%=
    - 注意:扩展赋值运算符自带强转功能

```
int a = 5;
a += 10; //相当于a=(int)(a+10)
System.out.println(a); //15
a *= 2; //相当于a=(int)(a*2)
System.out.println(a); //30
a /= 6; //相当于a=(int)(a/6)
System.out.println(a); //5

//小面试题:
short s = 5;
//s = s+10; //编译错误，需强转，改为:s=(short)(s+10);
s += 10; //相当于s=(short)(s+10);
```

- 字符串连接：+
  - 若两边为数字，则做加法运算
  - 若两边(任意边)出现了字符串，则做字符串连接

```
int age = 39;
System.out.println("age="); //age=
System.out.println(age);    //39
System.out.println("age="+age); //age=39
System.out.println("我今年"+age+"岁了"); //我今年39岁了

String name = "WKJ";
System.out.println("大家好，我叫"+name); //大家好，我叫WKJ
System.out.println("大家好，我叫"+name+"，今年"+age+"岁了"); //大家好，我叫WKJ，今年39岁了
```

- 任何类型的数据与字符串连接，结果都会变为字符串型

```
System.out.println(10+20+""+30); //3030---------String
System.out.println(""+10+20+30); //102030-------String
System.out.println(10+20+30+""); //60-----------String
```

- 条件：boolean? 数1：数2
  - 语法：
    - boolean?数1:数2
  - 执行过程：
    - 整个条件运算是有值的，它的值要么是?号后的数1，要么是:号后的数2
    - 计算boolean的值：
      - 若为true，则整个表达式的值为?号后的数1
      - 若为false，则整个表达式的值为:号后的数2

```
int num = 5;
int flag = num>0?1:-1;
System.out.println(flag); //1

int a=8,b=5;
int max = a>b?a:b;
System.out.println("max="+max); //max=8
```

2. 分支结构(上)：基于条件执行
   - if结构：1条路

```
1)语法：
   if(boolean){
      语句块--------------基于条件执行的语句
   }
2)执行过程：
    判断boolean的值：
       若为true，则执行语句块(整个结束)
       若为false，则直接结束
```

```
//1)满500打8折:
double price = 300.0; //消费金额  带数(600.0,300.0)
if(price>=500){   //满500
    price *= 0.8; //打8折
}
System.out.println("最终消费金额为:"+price);

//2)判断成绩是否合法
int score = 555; //成绩  带数(95,-5,555)
if(score>=0 && score<=100){
    System.out.println("成绩合法");
}
System.out.println("继续执行...");
```

- if...else结构：2条路

```
1)语法:
   if(boolean){
      语句块1
   }else{
      语句块2
   }
2)执行过程:
    判断boolean的值:
      若为true，则执行语句块1(整个结束)
      若为false，则执行语句块2(整个结束)
3)说明:
    语句块1和语句块2，必走其中之一-------------2选1
```

```
//1)满500打8折，不满500打9折:
double price = 300.0;   //带数(600.0,300.0)
if(price>=500){ //满
    price*=0.8;
}else{ //不满
    price*=0.9;
}
System.out.println("最终消费金额为:"+price);

//2)判断成绩合法还是不合法
int score = 95; //带数(95,-5,555)
if(score>=0 && score<=100){
    System.out.println(score+"是合法成绩");
}else{
    System.out.println(score+"是不合法成绩");
}
```

- if...else if结构：多条路

```
1)语法:
  if(boolean-1){
    语句块1
  }else if(boolean-2){
    语句块2
```

```
  }else if(boolean-3){
    语句块3
  }else{
    语句块4
  }
2)执行过程:
    判断boolean-1，若为true则执行语句块1(结束)，若为false则
    再判断boolean-2，若为true则执行语句块2(结束)，若为false则
    再判断boolean-3，若为true则执行语句块3(结束)，若为false则执行语句块4(结束)
3)说明:
    语句块1/2/3/4，必走其中之一---------多选1
```

```
//1)满2000打5折，满1000不满2000打7折，满500不满1000打8折，不满500打9折:
double price = 6000.0; //带数(2000.0,1000.0,600.0,300.0)
if(price>=2000){
    price*=0.5;
}else if(price>=1000){
    price*=0.7;
}else if(price>=500){
    price*=0.8;
}else{
    price*=0.9;
}
System.out.println("最终消费金额为:"+price);
```

补充:

1. 任何复杂的程序逻辑都可以通过三种结构来实现:
    - 顺序结构：从上往下逐行执行，每句必走
    - 分支结构：有条件的执行某语句，并非每句必走
    - 循环结构：明天讲
2. 明日单词:

```
1)Scanner/scan：扫描仪
2)import:引入、导入
3)System:系统
4)in:进入
5)new:新的
6)nextInt:下一个整数
7)nextDouble:下一个浮点数
8)switch:开关
9)case:案例
10)break:中断、退出
11)command:命令
12)by:通过
13)times:次数
14)while:当...的时候，循环的一种
15)do:做、干
16)math:数字
17)random:随机
18)guess/guessing:猜
```

19)game:游戏
20)count:数量