第03天 java基础知识

今日内容介绍

* 循环结构
* 循环嵌套
* 控制循环语句
* Random随机数

# 循环结构

## for循环的格式及基本使用

### for循环语句格式：

for(初始化语句;判断条件语句;控制条件语句) {

循环体语句;

}

### 执行流程

A:执行初始化语句

B:执行判断条件语句，看其结果是true还是false

如果是false，循环结束。

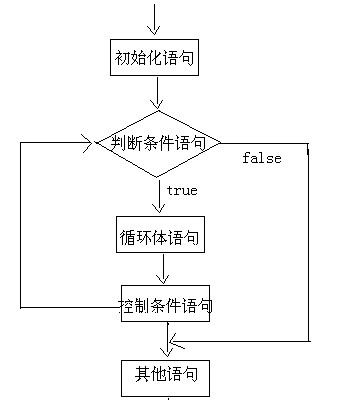
如果是true，继续执行。

C:执行循环体语句

D:执行控制条件语句

E:回到B继续

### for循环的执行流程图



### 代码案例一

**package** com.itheima;

/\*

\* for循环语句的格式：

\* for(初始化语句;判断条件语句;控制条件语句) {

\* 循环体语句;

\* }

\*

\* 执行流程：

\* A:执行初始化语句

\* B:执行判断条件语句，看其结果是true还是false

\* 如果是false，就结束循环

\* 如果是true，就继续执行

\* C:执行循环体语句

\* D:执行控制条件语句

\* E:回到B继续

\*/

**public** **class** ForDemo {

**public** **static** **void** main(String[] args) {

//需求：在控制台输出5次HelloWorld

//原始做法

System.***out***.println("HelloWorld");

System.***out***.println("HelloWorld");

System.***out***.println("HelloWorld");

System.***out***.println("HelloWorld");

System.***out***.println("HelloWorld");

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

//用for循环改进

**for**(**int** x=1; x<=5; x++) {

System.***out***.println("HelloWorld");

}

}

}

### for循环的练习

#### for循环练习之获取数据1-5和5-1

#### 代码案例二

**package** com.itheima;

/\*

\* 需求：获取数据1-5和5-1

\*/

**public** **class** ForTest {

**public** **static** **void** main(String[] args) {

//原始做法

System.***out***.println(1);

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

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

System.***out***.println(4);

System.***out***.println(5);

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

//for循环改进

**for**(**int** x=1; x<=5; x++) {

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

}

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

//获取数据5-1

**for**(**int** x=5; x>=1; x--) {

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

}

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

}

}

#### for循环练习之求1-5之间数据和

#### 代码案例三

**package** com.itheima;

/\*

\* 需求：求出1-5之间数据之和

\*

\* 分析：

\* A:定义一个求和变量，初始化值是0

\* B:获取1-5的数据，用for循环就可以实现

\* C:把每一次获取到的数据累加到求和变量

\* D:输出求和变量

\*/

**public** **class** ForTest2 {

**public** **static** **void** main(String[] args) {

//定义一个求和变量，初始化值是0

**int** sum = 0;

//获取1-5的数据，用for循环就可以实现

**for**(**int** x=1; x<=5; x++) {

//把每一次获取到的数据累加到求和变量

//sum = sum + x;

sum += x;

/\*

\* 第一次：sum = sum + x = 0 + 1 = 1

\* 第二次：sum = sum + x = 1 + 2 = 3

\* 第三次：sum = sum + x = 3 + 3 = 6

\* 第四次：sum = sum + x = 6 + 4 = 10

\* 第五次：sum = sum + x = 10 + 5 = 15

\*/

}

//输出求和变量

System.***out***.println("sum:"+sum);

}

}

#### for循环练习之求1-100之间的偶数

#### 代码案例四

**package** com.itheima;

/\*

\* 需求：求出1-100之间偶数和

\*

\* 分析：

\* A:定义求和变量，初始化值是0

\* B:获取1-100之间的偶数，用for循环实现

\* C:拿到每一个获取的数据进行判断看是否是偶数

\* 如果是偶数，就累加。

\* D:输出求和变量

\*/

**public** **class** ForTest3 {

**public** **static** **void** main(String[] args) {

//定义求和变量，初始化值是0

**int** sum = 0;

//获取1-100之间的偶数，用for循环实现

**for**(**int** x=1; x<=100; x++) {

//拿到每一个获取的数据进行判断看是否是偶数

**if**(x%2 == 0) {

//如果是偶数，就累加。

sum += x;

}

}

//输出求和变量

System.***out***.println("sum:"+sum);

}

}

#### for循环练习之打印水仙花数

#### 代码案例五

**package** com.itheima;

/\*

\* 需求：在控制台输出所有的”水仙花数”

\*

\* 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是一个水仙花数

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

\*

\* 分析：

\* A:三位数其实是告诉了我们范围，这个用for循环实现

\* B:获取每一个三位数的个位，十位，百位

\* 如何获取一个三位数的个位，十位，百位呢?

\* 举例：153

\* 个位：153%10

\* 十位：153/10%10

\* 百位：153/10/10%10

\* ...

\* C:拿个位，十位，百位的立方和和该数本身进行比较

\* 如果相等，就说明该数是水仙花数，在控制台打印

\*/

**public** **class** ForTest4 {

**public** **static** **void** main(String[] args) {

//三位数其实是告诉了我们范围，这个用for循环实现

**for**(**int** x=100; x<1000; x++) {

//获取每一个三位数的个位，十位，百位

**int** ge = x%10;

**int** shi = x/10%10;

**int** bai = x/10/10%10;

//拿个位，十位，百位的立方和和该数本身进行比较

**if**((ge\*ge\*ge+shi\*shi\*shi+bai\*bai\*bai) == x) {

//如果相等，就说明该数是水仙花数，在控制台打印

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

}

}

}

}

#### for循环练习之统计水仙花的个数

#### 代码案例六

**package** com.itheima;

/\*

\* 需求：统计”水仙花数”共有多少个

\*

\* 分析：

\* A:定义统计变量，初始化值是0

\* B:获取三位数，用for循环实现

\* C:获取每个位上的数据

\* D:判断数据是否是水仙花数

\* 如果是，就统计变量++

\* E:输出统计变量

\*/

**public** **class** ForTest5 {

**public** **static** **void** main(String[] args) {

//定义统计变量，初始化值是0

**int** count = 0;

//获取三位数，用for循环实现

**for**(**int** x=100; x<1000; x++) {

//获取每个位上的数据

**int** ge = x%10;

**int** shi = x/10%10;

**int** bai = x/10/10%10;

//判断数据是否是水仙花数

**if**((ge\*ge\*ge+shi\*shi\*shi+bai\*bai\*bai) == x){

//如果是，就统计变量++

count++;

}

}

//输出统计变量

System.***out***.println("水仙花的个数是："+count);

}

}

## while循环格式的概述和使用

### while循环语句格式

基本格式

while(判断条件语句) {

循环体语句;

}

扩展格式

初始化语句;

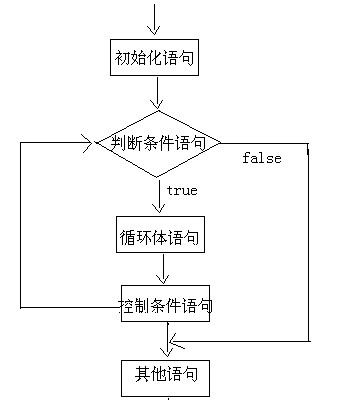
while(判断条件语句) {

循环体语句;

控制条件语句;

}

### 执行流程图



### 代码案例七

**package** com.itheima;

/\*

\* while循环的语句格式：

\* while(判断条件语句) {

\* 循环体语句;

\* }

\*

\* 完整格式：

\* 初始化语句;

\* while(判断条件语句) {

\* 循环体语句;

\* 控制条件语句;

\* }

\*

\* 回顾for循环的语句格式：

\* for(初始化语句;判断条件语句;控制条件语句) {

\* 循环体语句;

\* }

\*/

**public** **class** WhileDemo {

**public** **static** **void** main(String[] args) {

//在控制台输出5次HelloWorld

//for循环实现

/\*

for(int x=1; x<=5; x++) {

System.out.println("HelloWorld");

}

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

\*/

//while循环实现

**int** x=1;

**while**(x<=5) {

System.***out***.println("HelloWorld");

x++;

}

}

}

### while循环的练习

#### while循环练习之求1-100的数据和

#### 代码案例八

**package** com.itheima;

/\*

\* 需求：求1-100之间的数据和。

\*/

**public** **class** WhileTest {

**public** **static** **void** main(String[] args) {

//for循环实现

/\*

//定义求和变量

int sum = 0;

for(int x=1; x<=100; x++) {

//累加即可

sum += x;

}

//输出结果

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

\*/

//while循环实现

//定义求和变量

**int** sum = 0;

**int** x=1;

**while**(x<=100) {

//累加即可

sum += x;

x++;

}

//输出结果

System.***out***.println("sum:"+sum);

}

}

## dowhile循环格式的概述和使用

### dowhile循环语句格式

基本格式

do {

循环体语句;

}while((判断条件语句);

扩展格式

初始化语句;

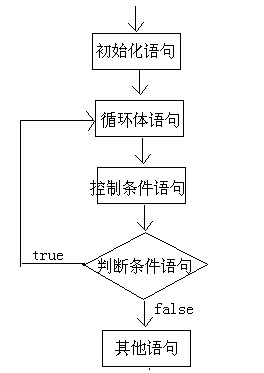
do {

循环体语句;

控制条件语句;

} while((判断条件语句);

### 执行流程图



### 代码案例九

**package** com.itheima;

/\*

\* do...while循环语句的格式：

\* 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);

\*

\* 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，看是true还是false

\* 如果是false，就结束循环

\* 如果是true，就回到B继续

\*

\* 练习：求和案例，水仙花案例

\*/

**public** **class** DoWhileDemo {

**public** **static** **void** main(String[] args) {

//在控制台输出5次HelloWorld案例

/\*

for(int x=1; x<=5; x++) {

System.out.println("HelloWorld");

}

\*/

**int** x=1;

**do** {

System.***out***.println("HelloWorld");

x++;

}**while**(x<=5);

}

}

**package** com.itheima;

/\*

\* 三种循环语句可以完成相同的事情，但是也是有小区别的：

\* do...while循环语句至少执行一次循环体。

\* 而for和while循环语句要先进行条件的判断，然后看是否执行循环体语句。

\*

\* for循环和while循环的小区别：

\* for循环结束后，初始化的变量不能被使用了。

\* 而while循环结束后，初始化的变量还可以继续被使用。

\*

\* 推荐使用的顺序：

\* for -- while -- do...while

\*/

**public** **class** DoWhileDemo2 {

**public** **static** **void** main(String[] args) {

/\*

int x = 3;

do {

System.out.println("爱生活，爱Java");

x++;

}while(x < 3);

\*/

/\*

int x = 3;

while(x < 3) {

System.out.println("爱生活，爱Java");

x++;

}

\*/

/\*

for(int x=1; x<=5; x++) {

System.out.println("爱生活，爱Java");

}

System.out.println(x);

\*/

**int** x=1;

**while**(x <= 5) {

System.***out***.println("爱生活，爱Java");

x++;

}

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

}

}

## 三种循环的区别

### 区别概述

虽然可以完成同样的功能，但是还是有小区别：

do…while循环至少会执行一次循环体。

for循环和while循环只有在条件成立的时候才会去执行循环体

for循环语句和while循环语句的小区别：

使用区别：控制条件语句所控制的那个变量，在for循环结束后，就不能再被访问到了，而while循环结束还可以继续使用，如果你想继续使用，就用while，否则推荐使用for。原因是for循环结束，该变量就从内存中消失，能够提高内存的使用效率。

### 代码案例十

**package** com.itheima;

/\*

\* 三种循环语句可以完成相同的事情，但是也是有小区别的：

\* do...while循环语句至少执行一次循环体。

\* 而for和while循环语句要先进行条件的判断，然后看是否执行循环体语句。

\*

\* for循环和while循环的小区别：

\* for循环结束后，初始化的变量不能被使用了。

\* 而while循环结束后，初始化的变量还可以继续被使用。

\*

\* 推荐使用的顺序：

\* for -- while -- do...while

\*/

**public** **class** DoWhileDemo2 {

**public** **static** **void** main(String[] args) {

/\*

int x = 3;

do {

System.out.println("爱生活，爱Java");

x++;

}while(x < 3);

\*/

/\*

int x = 3;

while(x < 3) {

System.out.println("爱生活，爱Java");

x++;

}

\*/

/\*

for(int x=1; x<=5; x++) {

System.out.println("爱生活，爱Java");

}

System.out.println(x);

\*/

**int** x=1;

**while**(x <= 5) {

System.***out***.println("爱生活，爱Java");

x++;

}

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

}

}

# 循环嵌套

### 请输出一个4行5列的星星(\*)图案

#### 案例代码十一

**package** com.itheima;

/\*

\* 需求：请输出一个4行5列的星星(\*)图案。

\* 结果：

\* \*\*\*\*\*

\* \*\*\*\*\*

\* \*\*\*\*\*

\* \*\*\*\*\*

\*

\* 循环嵌套：就是循环体语句本身是一个循环语句。

\*

\* 结论：

\* 外循环控制的是行，内循环控制的是列

\*/

**public** **class** ForForDemo {

**public** **static** **void** main(String[] args) {

//原始的做法

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

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

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

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

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

//虽然我们完成了题目的要求，但是不好

//假如我们有很多这样的要求，而且行和列是变化的

//所以我们要改进目前的代码的写法

//我要在一行上输出一颗\*

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

//我要在一行上输出5颗\*

/\*

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

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

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

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

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

\*/

//之所以出现这样的结果，是因为System.out.println()每次把内容输出后加一个换行

//有解决方案呢?有

//用System.out.print()就可以解决

/\*

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

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

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

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

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

\*/

/\*

//用循环改进代码

for(int x=1; x<=5; x++) {

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

}

//采用下面的内容就可以实现换行

System.out.println();

//第二行的5颗\*

for(int x=1; x<=5; x++) {

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

}

System.out.println();

//第三行的5颗\*

for(int x=1; x<=5; x++) {

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

}

System.out.println();

//第四行的5颗\*

for(int x=1; x<=5; x++) {

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

}

System.out.println();

\*/

//重复的代码执行多次，用循环改进

**for**(**int** y=1; y<=4; y++) {

**for**(**int** x=1; x<=5; x++) {

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

}

System.***out***.println();

}

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

**for**(**int** y=1; y<=7; y++) {

**for**(**int** x=1; x<=8; x++) {

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

}

System.***out***.println();

}

}

}

### 循环嵌套练习之打印正三角形

#### 案例代码十二

**package** com.itheima;

/\*

\* 需求：请输出如下图形

\* \*

\* \*\*

\* \*\*\*

\* \*\*\*\*

\* \*\*\*\*\*

\*/

**public** **class** ForForTest {

**public** **static** **void** main(String[] args) {

//通过简单的观察，我们发现这是一个5行的，列数变化的形状

//首先来实现一个5行5列的形状

**for**(**int** x=1; x<=5; x++) {

**for**(**int** y=1; y<=5; y++) {

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

}

System.***out***.println();

}

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

//虽然我们实现了一个5行5列的形状

//但是这不是我们想要的

//我们要的形状是列数变化的

//怎么办呢?

//第一行：1 y=1; y<=1

//第二行：2 y=1; y<=2

//第三行：3 y=1; y<=3

//第四行：4 y=1; y<=4

//第五行：5 y=1; y<=5

//我们需要一个变量的变量是从1开始，到5结束

**int** z = 1;

**for**(**int** x=1; x<=5; x++) {

**for**(**int** y=1; y<=z; y++) {

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

}

System.***out***.println();

z++;

}

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

//我们现在已经实现了我们的需求，但是我们继续观察会发现

//第一次：x的值是1

//第二次：x的值是2

//...

//x和z是一样的变化过程，这样的话，我们就可以省略z，直接用x替代

**for**(**int** x=1; x<=5; x++) {

**for**(**int** y=1; y<=x; y++) {

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

}

System.***out***.println();

}

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

}

}

### 循环嵌套练习之打印九九乘法表

#### 案例代码十三

**package** com.itheima;

/\*

\* 需求：在控制台打印九九乘法表

\*

\* \t:转移字符，表示一个tab键的位置

\*/

**public** **class** ForForTest2 {

**public** **static** **void** main(String[] args) {

//先打印一个9行9列的星形(列是变化的)

**for**(**int** x=1; x<=9; x++) {

**for**(**int** y=1; y<=x; y++) {

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

}

System.***out***.println();

}

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

/\*

1\*1=1

1\*2=2 2\*2=4

1\*3=3 2\*3=6 3\*3=9

...

\*/

**for**(**int** x=1; x<=9; x++) {

**for**(**int** y=1; y<=x; y++) {

System.***out***.print(y+"\*"+x+"="+y\*x+"\t");

}

System.***out***.println();

}

}

}

# 控制循环语句

## 跳转控制语句break的概述和使用

### break的使用场景

在选择结构switch语句中

在循环语句中

离开使用场景的存在是没有意义的

break的作用：

跳出单层循环

跳出多层循环

* + 带标签的跳出
  + 格式：
    1. 标签名:循环语句
    2. 标签名要符合Java的命名规范

### 代码案例十四

**package** com.itheima;

/\*

\* break:中断的意思

\*

\* 使用场景：

\* A:switch语句中，用于结束switch语句

\* B:循环语句中，用于结束循环

\* 如何使用：

\* A:跳出单层循环

\* B:跳出多层循环

\* 用带标签的语句格式。

\*/

**public** **class** BreakDemo {

**public** **static** **void** main(String[] args) {

//break可以用于结束当前的循环。

**for**(**int** x=1; x<=5; x++) {

**if**(x == 3) {

**break**;

}

System.***out***.println("HelloWorld");

}

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

//如果是多层循环，break到底结束的是哪个循环呢?

//break结束的是离他最近的那个循环

//如果我要跳出外层循环，可不可以呢?

//可以。如何实现呢?

//带标签的语句：

//格式：标签名:语句

wc:**for**(**int** x=1; x<=3; x++) {

nc:**for**(**int** y=1; y<=4; y++) {

**if**(y == 2) {

**break** wc;

}

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

}

System.***out***.println();

}

}

}

## 跳转控制语句continue的概述和使用

### continue的使用场景

在循环语句中

离开使用场景的存在是没有意义的

continue的作用：

单层循环对比break，然后总结两个的区别

break 退出当前循环

continue 退出本次循环

### 代码案例十五

**package** com.itheima;

/\*

\* continue:继续的意思

\*

\* 使用场景：

\* 循环中。离开使用场景是没有意义的。

\*

\* break和continue的区别：

\* break:跳出整个循环

\* continue:跳出这一次的操作，进入下一次的执行

\*/

**public** **class** ContinueDemo {

**public** **static** **void** main(String[] args) {

**for**(**int** x=1; x<=5; x++) {

**if**(x == 3) {

**continue**;

}

System.***out***.println("HelloWorld"+x);

}

}

}

# Random随机数

## Random的概述和使用

### Random的使用步骤

* 作用：
  1. 用于产生一个随机数
* 使用步骤(和Scanner类似)
  1. 导包
     1. import java.util.Random;
  2. 创建对象
     1. Random r = new Random();
  3. 获取随机数
     1. int number = r.nextInt(10);
     2. 产生的数据在0到10之间，包括0，不包括10。
     3. 括号里面的10是可以变化的，如果是100，就是0-100之间的数据

### 案例代码十六

**package** com.itheima;

**import** java.util.Random;

/\*

\* Random:用于产生随机数的类。用法和Scanner类似。

\*

\* 使用步骤：

\* A:导包

\* import java.util.Random;

\* B:创建对象

\* Random r = new Random();

\* C:获取随机数

\* int number = r.nextInt(10);

\* 获取的范围：[0,10) 包括0，不包括10

\*/

**public** **class** RandomDemo {

**public** **static** **void** main(String[] args) {

//创建对象

Random r = **new** Random();

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

//获取随机数

**int** number = r.nextInt(10);

System.***out***.println("number:"+number);

}

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

//如何获取一个1-100之间的随机数呢?

**int** i = r.nextInt(100)+1;

System.***out***.println(i);

}

}

## 猜数字小游戏案例:

### 案例代码十七

package com.itheima;

import java.util.Random;

import java.util.Scanner;

/\*

\* 需求：猜数字小游戏。

\* 系统产生一个1-100之间的随机数，请猜出这个数据是多少?

\*

\* 分析：

\* A:系统产生一个1-100之间的随机数

\* Random r = new Random();

\* int number = r.nextInt(100)+1;

\* B:键盘录入我们要猜的数据

\* C:比较这两个数据，看我们猜的是否正确

\* 如果大了，提示：你猜的数据大了

\* 如果小了，提示：你猜的数据小了

\* 如果相等，提示：恭喜你，猜中了

\* D:为了实现多次猜数据，我们就要加入循环，而我们又不知道猜多少次能中。怎么办呢?

\* 死循环：while(true) {...}

\* for(;;) {...}

\*/

public class RandomTest {

public static void main(String[] args) {

//系统产生一个1-100之间的随机数

Random r = new Random();

//获取随机数

int number = r.nextInt(100)+1;

//多次猜数据

while(true) {

//创建键盘录入对象

Scanner sc = new Scanner(System.in);

//给出提示

System.out.println("请输入你要猜的整数(1-100)：");

int guessNumber = sc.nextInt();

//比较这两个数据，看我们猜的是否正确

if(guessNumber > number) {

System.out.println("你猜的数据"+guessNumber+"大了");

}else if(guessNumber < number) {

System.out.println("你猜的数据"+guessNumber+"小了");

}else {

System.out.println("恭喜你，猜中了");

break; //跳出循环

}

}

}

}