java基础知识 第3天

**【学习目标】理解、了解、应用、记忆**

通过今天的学习，参训学员能够：

1. **【应用】能够独立使用选择流程语句完成指定操作**
   1. 【理解】阐述if语句的的第一种第二种及第三种书写格式及执行流程
   2. 【应用】独立编写代码,使用if语句完成获取两个数较大值的练习
   3. 【应用】独立编写代码,使用if语句完成判断学生等级练习
   4. 【理解】阐述switch语句的的格式及执行流程
   5. 【应用】独立编写代码，使用switch语句完成根据数字输出对应星期的练习
2. **【理解】能够独立使用循环控制语句完成指定操作**
   1. 【理解】阐述使用for循环的基本格式及执行流程
   2. 【应用】独立编写代码，使用for循环完成获取1-5之间数据之和
   3. 【应用】独立编写代码，使用for循环完成获取1-100之间的偶数和
   4. 【应用】独立编写代码，使用for循环完成控制台打印100-999之间水仙花数，并获取水仙花个数练习
   5. 【理解】阐述while循环的基本格式及执行流程
   6. 【应用】独立编写代码，使用while循环完成1-100之间数据之和
   7. 【理解】阐述do…while循环的格式及执行流程
   8. 【理解】阐述for、while、do…while三种循环的区别
3. **【应用】能够灵活使用控制循环语句**
   1. 【理解】阐述break的作用及使用场景
   2. 【理解】阐述continue的作用及使用场景

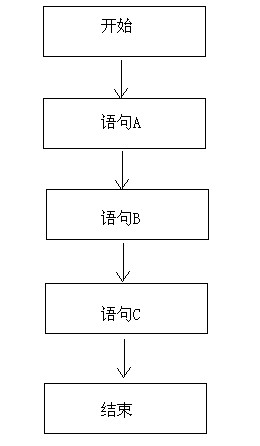
# 选择流程控制语句

## 顺序结构的基本使用

### 顺序结构概述

是程序中最简单最基本的流程控制，没有特定的语法结构，按照代码的先后顺序，依次执行，程序中大多数的代码都是这样执行的

### 顺序流程流程图



### 代码案例一

|  |
| --- |
| **package** com.igeek\_01;  /\*\*  \* **@ClassName**: OrderDemo  \* **@Description**: 顺序结构：从上往下，依次执行  \* **@date** 2017年10月12日 下午3:49:49  \* Company www.igeekhome.com  \*  \*/  **public** **class** OrderDemo {  **public** **static** **void** main(String[] args) {  System.***out***.println("开始");  System.***out***.println("语句A");  System.***out***.println("语句B");  System.***out***.println("语句C");  System.***out***.println("结束");  }  } |

## if语句的格式

### if语句格式1及执行流程

if语句第一种格式：

if(关系表达式) {

语句体

}

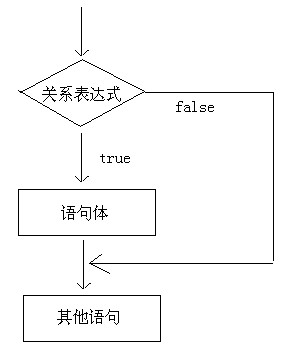
执行流程:

首先判断关系表达式看其结果是true还是false

如果是true就执行语句体

如果是false就不执行语句体

### if语句格式1执行流程图



### 代码案例二

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: IfDemo  \* **@Description**: if语句  \* **@date** 2017年10月12日 下午3:52:39  \* Company www.igeekhome.com  \*  \* if语句有三种格式。  \*  \* if语句格式1：  \* if(关系表达式) {  \* 语句体;  \* }  \*  \* 执行流程：  \* A:首先判断关系表达式看其结果是true还是false  \* B:如果是true,就执行语句体  \* C:如果是false,就不执行语句体  \*/  **public** **class** IfDemo {  **public** **static** **void** main(String[] args) {  System.***out***.println("开始");  // 定义两个变量  **int** a = 10;  **int** b = 20;  **if** (a == b) {  System.***out***.println("a等于b");  }  **int** c = 10;  **if** (a == c) {  System.***out***.println("a等于c");  }  System.***out***.println("结束");  }  } |

### if语句格式2及执行流程

if语句第二种格式：

if(关系表达式) {

语句体1;

}else {

语句体2;

}

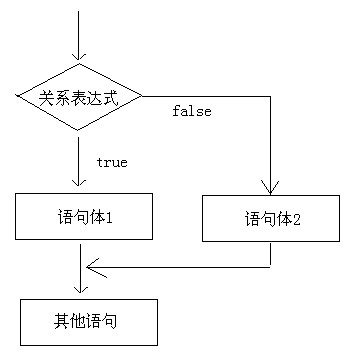
执行流程

首先判断关系表达式看其结果是true还是false

如果是true就执行语句体1

如果是false就执行语句体2

### if语句格式2执行流程图



### 代码案例三

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: IfDemo2  \* **@Description**: if-else语句格式  \* **@date** 2017年10月12日 下午3:54:57  \* Company www.igeekhome.com  \*  \* if语句格式2：  \* if(关系表达式) {  \* 语句体1;  \* }else {  \* 语句体2;  \* }  \*  \* 执行流程：  \* A:判断关系表达式的值是true还是false  \* B:如果是true，就执行语句体1  \* C:如果是false，就执行语句体2  \*/  **public** **class** IfDemo2 {  **public** **static** **void** main(String[] args) {  System.***out***.println("开始");  // 判断给定的数据是奇数还是偶数  // 定义变量  **int** a = 100;  // 给a重新赋值  a = 99;  **if** (a % 2 == 0) {  System.***out***.println("a是偶数");  } **else** {  System.***out***.println("a是奇数");  }  System.***out***.println("结束");  }  } |

### if语句格式3及执行流程

if语句第三种格式：

if(关系表达式1) {

语句体1;

}else if (关系表达式2) {

语句体2;

}

…

else {

语句体n+1;

}

执行流程

首先判断关系表达式1看其结果是true还是false

如果是true就执行语句体1

如果是false就继续判断关系表达式2看其结果是true还是false

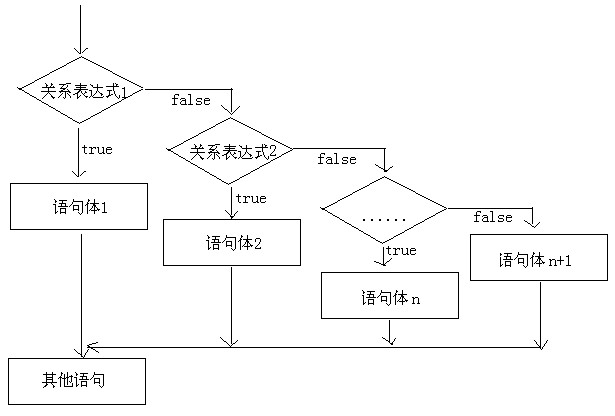
如果是true就执行语句体2

如果是false就继续判断关系表达式…看其结果是true还是false

…

如果没有任何关系表达式为true，就执行语句体n+1

### if语句格式3执行流程图



### 代码案例四

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: IfDemo3  \* **@Description**: 多重if-else if结构  \* **@date** 2017年10月12日 下午3:57:57  \* Company www.igeekhome.com  \*  \* if语句格式3：  \* if(关系表达式1) {  \* 语句体1;  \* }else if(关系表达式2) {  \* 语句体2;  \* }else if(关系表达式3) {  \* 语句体3;  \* }  \* ...  \* else {  \* 语句体n+1;  \* }  \*  \* 执行流程：  \* A:首先判断关系表达式1看其结果是true还是false  \* B:如果是true，就执行语句体1  \* 如果是false，就继续进行关系表达式2的判断看其结果是true还是false  \* C:如果是true，就执行语句体2  \* 如果是false，就继续进行关系表达式...的判断看其结果是true还是false  \* ...  \* D:如果没有一个为true的，就执行语句体n+1  \*  \* if语句的三种格式：  \* 第一种格式适合做一种情况的判断  \* 第二种格式适合做二种情况的判断  \* 第三种格式适合做多种情况的判断  \*/  **public** **class** IfDemo3 {  **public** **static** **void** main(String[] args) {  // x和y的关系满足如下：  // x>=3 y = 2x + 1;  // -1<=x<3 y = 2x;  // x<=-1 y = 2x – 1;  // 根据给定的x的值，计算出y的值并输出。  // 定义变量  **int** x = 5;    /\*  int y;  if (x >= 3) {  y = 2 \* x + 1;  } else if (x >= -1 && x < 3) {  y = 2 \* x;  } else if (x < -1) {  y = 2 \* x - 1;  }else {  y = 0;  }  \*/    **int** y = 0;  **if** (x >= 3) {  y = 2 \* x + 1;  } **else** **if** (x >= -1 && x < 3) {  y = 2 \* x;  } **else** **if** (x < -1) {  y = 2 \* x - 1;  }    System.***out***.println("y的值是："+y);  }  } |

## if语句的练习

### if语句实现获取两个数据较大值

### 代码案例五

|  |
| --- |
| **package** com.igeek\_03;  **import** java.util.Scanner;  /\*\*  \* **@ClassName**: IfTest  \* **@Description**: 键盘录入两个数据，获取这两个数据的较大值  \* **@date** 2017年10月12日 下午4:01:26  \* Company www.igeekhome.com  \*  \* 键盘录入两个数据，获取这两个数据的较大值  \*  \* 分析：  \* A:看到键盘录入，我们就应该想到键盘录入的三步骤  \* 导包，创建对象，接收数据  \* B:获取这两个数据的较大值，其实就是判断两个数据谁大，把大的输出就可以了。  \*  \* 导包：  \* A:手动导包  \* import java.util.Scanner;  \* B:鼠标点击红色叉叉，自动生成  \* C:快捷键(推荐)  \* ctrl+shift+o  \*/  **public** **class** IfTest {  **public** **static** **void** main(String[] args) {  //创建对象  Scanner sc = **new** Scanner(System.***in***);    //接收数据  System.***out***.println("请输入第一个数据：");  **int** a = sc.nextInt();    System.***out***.println("请输入第二个数据：");  **int** b = sc.nextInt();    //采用if语句格式2实现  /\*  if(a>b){  System.out.println("较大的值是："+a);  }else {  System.out.println("较大的值是："+b);  }  \*/    //拿到较大的值之后，我未必想直接输出，所以我们定义变量接收这个较大的值  **int** max;  **if**(a>b){  max = a;  }**else** {  max = b;  }  //可能做其他的操作  //max += 100;  System.***out***.println("较大的值是："+max);  }  } |

### if语句实现判断学生等级

### 代码案例六

|  |
| --- |
| **package** com.igeek\_03;  **import** java.util.Scanner;  /\*\*  \* **@ClassName**: IfTest2  \* **@Description**: 键盘录入学生考试成绩，请根据成绩判断该学生属于哪个级别  \* **@date** 2017年10月12日 下午4:05:01  \* Company www.igeekhome.com  \*  \* 键盘录入学生考试成绩，请根据成绩判断该学生属于哪个级别  \* 90-100 优秀  \* 80-90 好  \* 70-80 良  \* 60-70 及格  \* 60以下 不及格  \*  \* 分析：  \* A:键盘录入学生考试成绩  \* 三步骤  \* B:通过简单的分析，我们决定采用if语句格式3来实现  \*  \* 程序一定要考虑周全了。  \* 安全数据  \* 边界数据  \* 错误数据  \*/  **public** **class** IfTest2 {  **public** **static** **void** main(String[] args) {  //创建对象  Scanner sc = **new** Scanner(System.***in***);    //接收数据  System.***out***.println("请输入学生的考试成绩：");  **int** score = sc.nextInt();    //if语句格式3  /\*  if(score>=90 && score<=100){  System.out.println("你的成绩属于优秀");  }else if(score>=80 && score<90){  System.out.println("你的成绩属于好");  }else if(score>=70 && score<80){  System.out.println("你的成绩属于良");  }else if(score>=60 && score<70){  System.out.println("你的成绩属于及格");  }else {  System.out.println("你的成绩属于不及格");  }  \*/    //我们发现程序不够健壮,加入错误数据的判断  **if**(score<0 || score>100){  System.***out***.println("你的成绩是错误的");  }**else** **if**(score>=90 && score<=100){  System.***out***.println("你的成绩属于优秀");  }**else** **if**(score>=80 && score<90){  System.***out***.println("你的成绩属于好");  }**else** **if**(score>=70 && score<80){  System.***out***.println("你的成绩属于良");  }**else** **if**(score>=60 && score<70){  System.***out***.println("你的成绩属于及格");  }**else** {  System.***out***.println("你的成绩属于不及格");  }  }  } |

## switch语句的格式及执行流程

### 格式解释:

switch语句格式：

switch(表达式) {

case 值1:

语句体1;

break;

case 值2:

语句体2;

break;

...

default:

语句体n+1;

break;

}

switch表示这是switch语句

表达式的取值：byte,short,char,int

JDK5以后可以是枚举

JDK7以后可以是String

case后面跟的是要和表达式进行比较的值

语句体部分可以是一条或多条语句

break表示中断，结束的意思，可以结束switch语句

default语句表示所有情况都不匹配的时候，就执行该处的内容，和if语句的else相似。

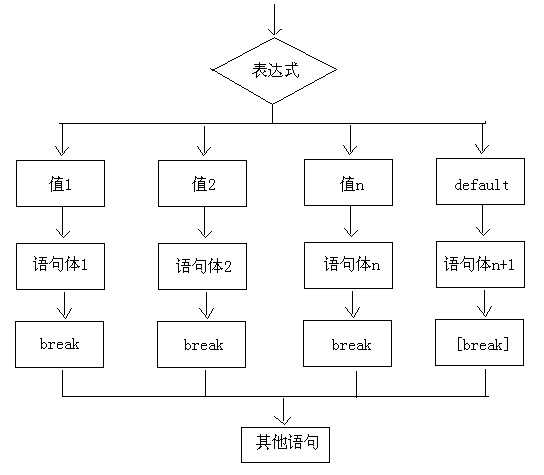
### 执行流程

首先计算出表达式的值

其次，和case依次比较，一旦有对应的值，就会执行相应的语句，在执行的过程中，遇到break就会结束。

最后，如果所有的case都和表达式的值不匹配，就会执行default语句体部分，然后程序结束掉。

### switch语句执行流程图



## switch语句的练习

### switch语句实现根据数字输出对应星期

### 代码案例七

|  |
| --- |
| **package** com.igeek\_05;  **import** java.util.Scanner;  /\*\*  \* **@ClassName**: SwitchDemo  \* **@Description**: switch语句格式  \* **@date** 2017年10月12日 下午4:10:18  \* Company www.igeekhome.com  \*  \* switch语句格式：  \* switch(表达式) {  \* case 值1:  \* 语句体1;  \* break;  \* case 值2:  \* 语句体2;  \* break;  \* ...  \* default:  \* 语句体n+1;  \* break;  \* }  \* 格式解释：  \* 表达式:byte,short,char,int  \* JDK5以后可以是枚举  \* JDK7以后可以是字符串  \* case:就是要和表达式进行比较的值  \* break:表示中断,结束的意思。  \* default:表示所有的情况都不匹配的时候，就执行语句体n+1。和if语句的else相似。  \* 执行流程：  \* A:计算出表达式的值  \* B:拿计算出来的值和case后面的值依次比较，一旦有对应的值，就执行该处的语句，在执行过程中，遇到 break，就结束。  \* C:如果所有的case都不匹配，就会执行default控制的语句，然后结束。  \* 需求：  \* 根据键盘录入的数值1，2，3，…7输出对应的星期一，星期二，星期三…星期日。  \*/  **public** **class** SwitchDemo {  **public** **static** **void** main(String[] args) {  //创建键盘录入对象  Scanner sc = **new** Scanner(System.***in***);    //接收数据  System.***out***.println("请输入一个数字(1-7)：");  **int** weekday = sc.nextInt();    //switch语句实现选择  **switch**(weekday) {  **case** 1:  System.***out***.println("星期一");  **break**;  **case** 2:  System.***out***.println("星期二");  **break**;  **case** 3:  System.***out***.println("星期三");  **break**;  **case** 4:  System.***out***.println("星期四");  **break**;  **case** 5:  System.***out***.println("星期五");  **break**;  **case** 6:  System.***out***.println("星期六");  **break**;  **case** 7:  System.***out***.println("星期日");  **break**;  **default**:  System.***out***.println("你输入的数字有误");  **break**;  }  }  } |

# 循环流程控制语句

## 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。

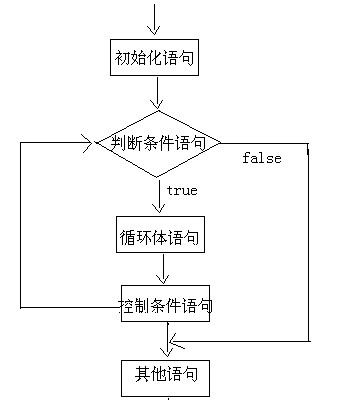
如果是true，继续执行。

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



### 代码案例八

|  |
| --- |
| **package** com.igeek\_01;  /\*\*  \* **@ClassName**: ForDemo  \* **@Description**: for循环语句格式  \* **@date** 2017年10月12日 下午4:17:21  \* Company www.igeekhome.com  \*  \* 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，看结果是true还是false  \* 如果是true，就继续执行  \* 如果是false，就结束循环  \* C:执行循环体语句  \* D:执行控制条件语句  \* E:回到B继续  \*  \* 需求：  \* 在控制台输出10次”HelloWorld”的案例。  \*/  **public** **class** ForDemo {  **public** **static** **void** main(String[] args) {  //原始写法  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("HelloWorld");  System.***out***.println("-------------------------");    //用循环改进  **for**(**int** x=1; x<=10; x++) {  System.***out***.println("HelloWorld");  }  }  } |

## for循环的练习

### for循环实现获取指定范围数据

### 代码案例九

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: ForTest  \* **@Description**: 获取数据1-5和5-1  \* **@date** 2017年10月12日 下午4:20:32  \* Company www.igeekhome.com  \*  \* 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("-------------");    //用循环改进  **for**(**int** x=1; x<=5; x++) {  System.***out***.println(x);  }  System.***out***.println("-------------");    //1-5的数据我们获取到了，如何获取5-1呢?  **for**(**int** x=5; x>=1; x--){  System.***out***.println(x);  }  }  } |

### for循环实现1-5之间数据求和

### 代码案例十

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: ForTest2  \* **@Description**: 求出1-5之间数据之和  \* **@date** 2017年10月12日 下午4:21:56  \* Company www.igeekhome.com  \*  \* 需求：求出1-5之间数据之和  \*  \* 分析：  \* A:定义求和变量，初始化值是0  \* B:获取1-5之间的数据，用for循环实现  \* C:把每一次获取到的数据，累加起来就可以了  \* D:输出求和变量即可  \*/  **public** **class** ForTest2 {  **public** **static** **void** main(String[] args) {  //定义求和变量，初始化值是0  **int** sum = 0;    //获取1-5之间的数据，用for循环实现  **for**(**int** x=1; x<=5; x++) {  //把每一次获取到的数据，累加起来就可以了  //sum = sum + x;  /\*  \* 第一次：sum = 0 + 1 = 1  \* 第二次：sum = 1 + 2 = 3  \* 第三次：sum = 3 + 3 = 6  \* 第四次：sum = 6 + 4 = 10  \* 第五次：sum = 10 + 5 = 15  \*/  sum += x;  }    //输出求和结果  System.***out***.println("sum:"+sum);  }  } |

### for循环实现1-100之间偶数和

### 代码案例十一

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: ForTest3  \* **@Description**: 求出1-100之间偶数和  \* **@date** 2017年10月12日 下午4:24:07  \* Company www.igeekhome.com  \*  \* 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之间的数据，用for循环实现  \* C:把获取到的数据进行判断，看是否是偶数  \* 如果是，就累加  \* D:输出求和结果  \*/  **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之间的数据，用for循环实现  **for**(**int** x=1; x<=100; x++) {  //把获取到的数据进行判断，看是否是偶数  **if**(x%2 ==0) {  sum += x;  }  }    //输出求和结果  System.***out***.println("sum:"+sum);  }  } |

### for循环实现在控制台打印水仙花数

### 代码案例十二

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: ForTest4  \* **@Description**: 在控制台输出所有的”水仙花数”  \* **@date** 2017年10月12日 下午4:28:43  \* Company www.igeekhome.com  \*  \* 需求：在控制台输出所有的”水仙花数”  \*  \* 分析：  \* 什么是水仙花数呢?  \* 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就是一个水仙花数。  \* 153 = 1\*1\*1 + 5\*5\*5 + 3\*3\*3  \*  \* A:三位数其实就告诉了我们水仙花数的范围  \* 100-999  \* B:如何获取一个数据的每一个位上的数呢?  \* 举例：我有一个数据153，请问如何获取到个位，十位，百位  \* 个位：153%10 = 3;  \* 十位：153/10%10 = 5;  \* 百位：153/10/10%10 = 1;  \* 千位：...  \* 万位：...  \* C:让每个位上的立方和相加，并和该数据进行比较，如果相等，就说明该数据是水仙花数，在控制台输出  \*/  **public** **class** ForTest4 {  **public** **static** **void** main(String[] args) {  //通过循环获取到每一个三位数  **for**(**int** x=100; x<1000; x++) {  //获取个位，十位，百位  **int** ge = x%10;  **int** shi = x/10%10;  **int** bai = x/10/10%10;    //让每个位上的立方和相加，并和该数据进行比较，如果相等，就说明该数据是水仙花数，在控制台输出  **if**((ge\*ge\*ge+shi\*shi\*shi+bai\*bai\*bai) == x) {  System.***out***.println(x);  }  }  }  } |

### for循环实现统计水仙花的个数

### 代码案例十三

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: ForTest5  \* **@Description**: 统计”水仙花数”共有多少个  \* **@date** 2017年10月12日 下午4:30:42  \* Company www.igeekhome.com  \*  \* 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:获取三位数的个位，十位，百位  \* D:判断这个三位数是否是水仙花数，如果是，统计变量++  \* E:输出统计结果就可以了  \*/  **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;    //判断这个三位数是否是水仙花数，如果是，统计变量++  **if**((ge\*ge\*ge+shi\*shi\*shi+bai\*bai\*bai) == x) {  count++;  }  }    //输出统计结果就可以了  System.***out***.println("水仙花数共有："+count+"个");  }  } |

## while循环的格式及基本使用

### 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;

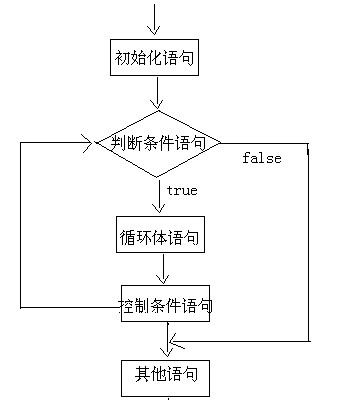
while(判断条件语句) {

循环体语句;

控制条件语句;

}

### 执行流程图



### 代码案例十四

|  |
| --- |
| **package** com.igeek\_03;  /\*\*  \* **@ClassName**: WhileDemo  \* **@Description**: while循环语句的基本格式  \* **@date** 2017年10月12日 下午4:34:27  \* Company www.igeekhome.com  \*  \* while循环语句的基本格式：  \* 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;  \* }  \*  \* 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) {  //输出10次HelloWorld  /\*  for(int x=1; x<=10; x++) {  System.out.println("HellloWorld");  }  \*/    //while循环实现  **int** x=1;  **while**(x<=10) {  System.***out***.println("HellloWorld");  x++;  }  }  } |

## while循环的练习

### while循环实现1-100之间数据求和

### 代码案例十五

|  |
| --- |
| **package** com.igeek\_04;  /\*\*  \* **@ClassName**: WhileTest  \* **@Description**: 求1-100之和  \* **@date** 2017年10月12日 下午4:36:45  \* Company www.igeekhome.com  \*  \* 求1-100之和。  \*/  **public** **class** WhileTest {  **public** **static** **void** main(String[] args) {  //回顾for循环实现    /\*  //定义求和变量  int sum = 0;  //获取1-100之间的数据  for(int x=1; x<=100; x++) {  //累加  sum += x;  }  System.out.println("1-100的和是："+sum);  \*/    //while循环实现  //定义求和变量  **int** sum = 0;  **int** x = 1;  **while**(x<=100) {  sum += x;  x++;  }  System.***out***.println("1-100的和是："+sum);  }  } |

## do…while循环的格式及基本使用

### do…while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;

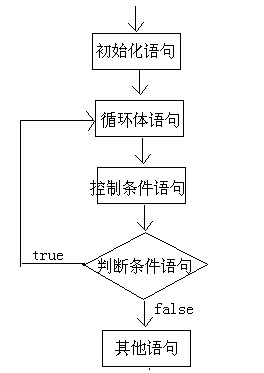
do {

循环体语句;

控制条件语句;

} while(判断条件语句);

### 执行流程图



### 代码案例十六

|  |
| --- |
| **package** com.igeek\_05;  /\*\*  \* **@ClassName**: DoWhileDemo  \* **@Description**: do...while循环的基本格式  \* **@date** 2017年10月12日 下午4:39:59  \* Company www.igeekhome.com  \*  \* do...while循环的基本格式：  \* 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：  \* 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true，回到B继续  \* 如果是false，就结束  \*  \* 练习：  \* 求和案例  \* 统计水仙花个数  \*/  **public** **class** DoWhileDemo {  **public** **static** **void** main(String[] args) {  //输出10次 HelloWorld  /\*  for(int x=1; x<=10; x++) {  System.out.println("HelloWorld");  }  \*/    //do...while改写  **int** x=1;  **do** {  System.***out***.println("HelloWorld");  x++;  }**while**(x<=10);  }  } |

## 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。

### 代码案例十七

|  |
| --- |
| **package** com.igeek\_06;  /\*\*  \* **@ClassName**: DoWhileDemo  \* **@Description**: 三种循环的区别  \* **@date** 2017年10月12日 下午4:43:18  \* Company www.igeekhome.com  \*  \* 三种循环的区别：  \* A:do...while至少执行一次循环体  \* B:for,while循环先判断条件是否成立，然后决定是否执行循环体  \*  \* for和while的小区别：  \* for循环的初始化变量，在循环结束后，不可以被访问。而while循环的初始化变量，是可以被继续使用的。  \* 如果初始化变量，后面还要继续访问，就使用while，否则，推荐使用for。  \*  \* 循环的使用推荐：  \* for -- while -- do...while  \*/  **public** **class** DoWhileDemo {  **public** **static** **void** main(String[] args) {  /\*  int x = 3;  while(x<3) {  System.out.println("我爱极客营");  x++;  }  System.out.println("--------------");  int y = 3;  do {  System.out.println("我爱极客营");  y++;  }while(y<3);  \*/      **for**(**int** x=1; x<=10; x++){  System.***out***.println("爱生活，爱Java");  }  //这里的x无法继续访问  //System.out.println(x);  System.***out***.println("-----------------");    **int** y = 1;  **while**(y<=10) {  System.***out***.println("爱生活，爱Java");  y++;  }  System.***out***.println(y);  }  } |

# 控制循环语句

## 控制跳转语句break

### Break的使用场景和作用

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，不能单独使用

break的作用：

跳出单层循环

### 代码案例十八

|  |
| --- |
| **package** com.igeek\_01;  /\*\*  \* **@ClassName**: BreakDemo  \* **@Description**: break:中断的意思  \* **@date** 2017年10月12日 下午4:51:45  \* Company www.igeekhome.com  \*  \* break:中断的意思  \* 使用场景：  \* A:switch语句中  \* B:循环中  \* 注意：  \* 离开使用场景是没有意义的。  \* 作用：  \* 跳出循环，让循环提前结束  \*/  **public** **class** BreakDemo {  **public** **static** **void** main(String[] args) {  //break;    **for**(**int** x=1; x<=10; x++) {  **if**(x == 3) {  **break**;  }  System.***out***.println("HelloWorld");  }  }  } |

## 控制跳转语句continue

### 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 结束本次循环，进入循环的下一次

### 代码案例十九

|  |
| --- |
| **package** com.igeek\_02;  /\*\*  \* **@ClassName**: ContinueDemo  \* **@Description**: continue:继续的意思  \* **@date** 2017年10月12日 下午4:54:00  \* Company www.igeekhome.com  \*  \* continue:继续的意思  \* 使用场景：  \* 循环中  \* 注意：  \* 离开使用场景是没有意义的  \* 作用：  \* 结束一次循环，继续下一次的循环  \* 区别：  \* break:退出循环  \* continue:结束一次循环，继续下一次的循环  \*/  **public** **class** ContinueDemo {  **public** **static** **void** main(String[] args) {  //continue;    **for**(**int** x=1; x<=10; x++) {  **if**(x == 3) {  //break;  **continue**;  }  System.***out***.println("HelloWorld");  }  }  } |

## 控制跳转语句练习

### 练习需求:

按要求分析结果，并验证

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

if(x%3==0) {

**//分别写break，continue，说说输出几次**

}

System.out.println(“我爱极客营”);

}

### 代码案例二十

|  |
| --- |
| **package** com.igeek\_03;  /\*\*  \* **@ClassName**: BreakAndContinueDemo  \* **@Description**: break、continue的区别  \* **@date** 2017年10月12日 下午4:59:39  \* Company www.igeekhome.com  \*  \* 按要求分析结果，并验证  \*  \* break：输出2次  \* continue:输出7次  \*/  **public** **class** BreakAndContinueDemo {  **public** **static** **void** main(String[] args) {  **for** (**int** x = 1; x <= 10; x++) {  **if** (x % 3 == 0) {  // 分别写break，continue，说说输出几次  //break;  **continue**;  }  System.***out***.println("我爱Java");  }  }  } |

重点和总结

1、选择流程控制语句：if、switch

2、循环流程控制语句：for、while、do-while

3、控制循环语句：break、continue