第 5 章 程序控制结构

5.1 程序流程控制介绍

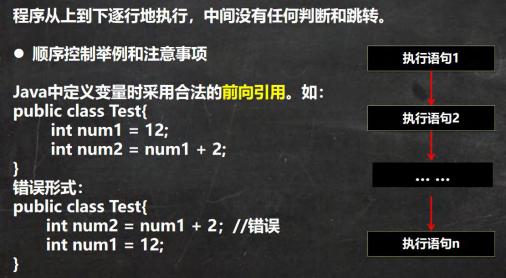
在程序中，程序运行的流程控制决定程序是如何执行的，是我们必须掌握的，主要有三大流程控制语句。

1) 顺序控制

2) 分支控制

3) 循环控制

5.2 顺序控制



5.3 分支控制 if-else

5.3. 1分支控制 if-else 介绍

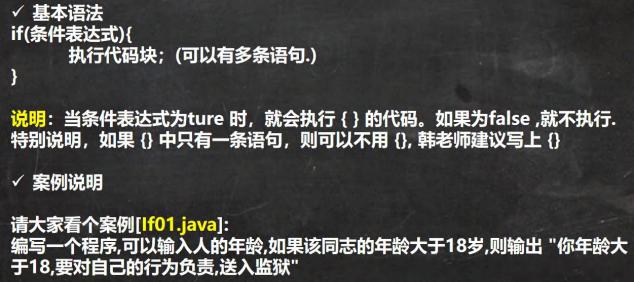
让程序有选择的的执行,分支控制有三种

1) 单分支 if

2) 双分支 if-else

3) 多分支 if-else if -....-else

5.3.2单分支

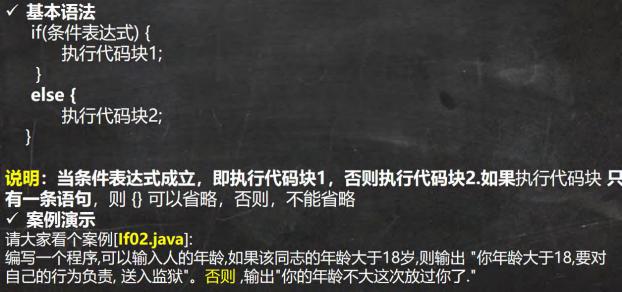


|  |
| --- |
| //if 的快速入门  import java.util.Scanner;//导入  public class If01 {  //编写一个 main 方法  public static void main(String[] args) {  //编写一个程序,可以输入人的年龄,如果该同志的年龄大于 18 岁,  //则输出 "你年龄大于 18,要对自己的行为负责,送入监狱"  //  //思路分析  //1. 接收输入的年龄, 应该定义一个 Scanner 对象  //2. 把年龄保存到一个变量 int age  //3. 使用 if 判断，输出对应信息  //应该定义一个 Scanner 对象 |

|  |
| --- |
| Scanner myScanner = new Scanner(System.in);  System.out.println("请输入年龄");  //把年龄保存到一个变量 int age  int age = myScanner.nextInt();  //使用 if 判断，输出对应信息  if(age > 18) {  System.out.println("你年龄大于 18,要对自己的行为负责,送入监狱");  }  System.out.println("程序继续...");  }  } |



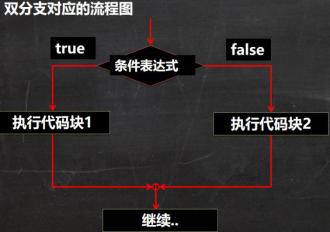
5.4 双分支



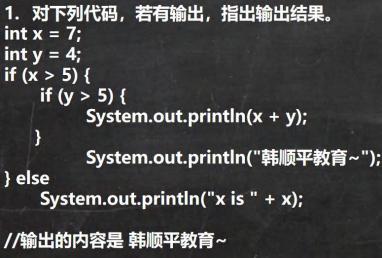
|  |
| --- |
| //if-else 的快速入门  import java.util.Scanner;//导入  public class If02 {  //编写一个 main 方法  public static void main(String[] args) {  //编写一个程序,可以输入人的年龄,如果该同志的年龄大于 18 岁,  //则输出 "你年龄大于 18,要对  //自己的行为负责, 送入监狱" 。否则 ,输出"你的年龄不大这次放过你了."  //  //思路分析  //1. 接收输入的年龄, 应该定义一个 Scanner 对象  //2. 把年龄保存到一个变量 int age  //3. 使用 if-else 判断，输出对应信息 |

|  |
| --- |
| //应该定义一个 Scanner 对象  Scanner myScanner = new Scanner(System.in);  System.out.println("请输入年龄");  //把年龄保存到一个变量 int age  int age = myScanner.nextInt();  //使用 if-else 判断，输出对应信息  if(age > 18) {  System.out.println("你年龄大于 18,要对自己的行为负责,送入监狱");  } else {//双分支  System.out.println("你的年龄不大这次放过你了");  }  System.out.println("程序继续...");  }  } |

5.4. 1双分支



5.4.2单分支和双分支练习题



//IfExercise01.java

1) 编写程序，声明 2 个 double 型变量并赋值。判断第一个数大于 10.0 ，且第 2 个数小于 20.0 ，打印两数之和。

2) 【课后自己练】定义两个变量 int ，判断二者的和，是否能被 3 又能被 5 整除，打印提示信息

3) 判断一个年份是否是闰年，闰年的条件是符合下面二者之一：(1)年份能被 4 整除，但不能被 100 整除；(2)能被 400

整除

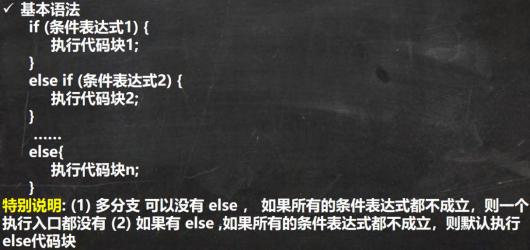
4) 代码

|  |
| --- |
| //单分支和双分支的练习  //  public class IfExercise01 {  //编写一个 main 方法  public static void main(String[] args) {  //编写程序，声明 2 个 double 型变量并赋值。  //判断第一个数大于 10.0 ，且第 2 个数小于 20.0 ，打印两数之和 |

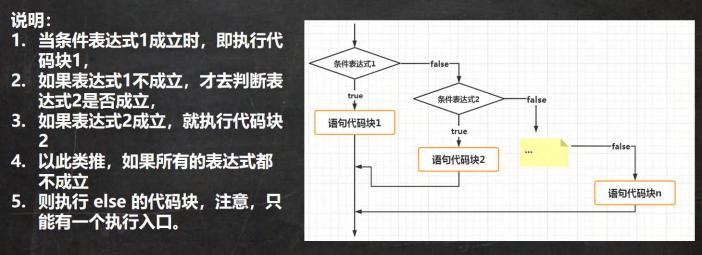
|  |
| --- |
| //思路分析  double d1 = 33.5;  double d2 = 2.6;  if(d1 > 10.0 && d2 < 20.0) {  System.out.println("两个数和=" + (d1 + d2));  }  //【课后自己练】定义两个变量 int ，判断二者的和，  //是否能被 3 又能被 5 整除，打印提示信息  //  //思路分析  //1. 定义两个变量 int num1, num2  //2. 定义一个变量 int sum = num1 + num2;  //3. sum % 3 , 5 后 等于 0 说明可以整除  //4. 使用 if - else 来提示对应信息  //走代码  int num1 = 10;  int num2 = 1;  int sum = num1 + num2;  if(sum % 3 == 0 && sum % 5 == 0) {  System.out.println("和可以被 3 又能被 5 整除");  } else {  System.out.println("和不能被 3 和 5 整除..");  } |

|  |
| --- |
| //判断一个年份是否是闰年，闰年的条件是符合下面二者之一：  //(1)年份能被 4 整除，但不能被 100 整除；(2)能被 400 整除  //  //思路分析  //1. 定义 int year 保存年  //2. 年份能被 4 整除，但不能被 100 整除,  // => year % 4 == 0 && year % 100 != 0  //3. 能被 400 整除 => year % 400 == 0  //4. 上面的 2 和 3 是 或的关系  //代码实现  int year = 2028;  if( (year % 4 == 0 && year % 100 != 0) || year % 400 == 0 ) {  System.out.println(year + " 是 闰年");  } else {  System.out.println(year + " 不是 闰年");  }  }  } |

5.4.3多分支



 多分支的流程图(重要!)



5.4.4多分支

 案例演示 1

请大家看个案例[If03.java]:

输入保国同志的芝麻信用分：

如果：

1) 信用分为 100 分时，输出 信用极好；

2) 信用分为(80 ，99]时，输出 信用优秀；

3) 信用分为[60,80]时，输出 信用一般；

4) 其它情况 ，输出 信用 不及格

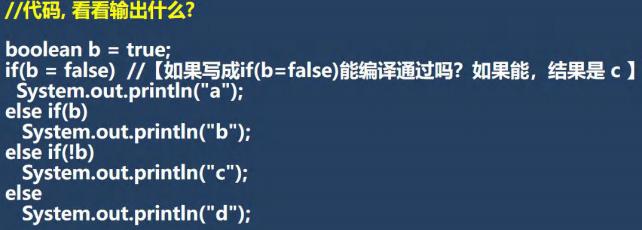
5) 请从键盘输入保国的芝麻信用分，并加以判断

代码:

|  |
| --- |
| //课堂练习  import java.util.Scanner;  public class If03 {  //编写一个 main 方法  public static void main(String[] args) {  /\*  输入保国同志的芝麻信用分：  如果：  信用分为 100 分时，输出 信用极好；  信用分为(80 ，99]时，输出 信用优秀；  信用分为[60,80]时，输出 信用一般；  其它情况 ，输出 信用 不及格  请从键盘输入保国的芝麻信用分，并加以判断  假定信用分数为 int  \*/  Scanner myScanner = new Scanner(System.in);  //接收用户输入  System.out.println("请输入信用分(1- 100):");  //请思考：如果小伙伴输入的不是整数，而是 hello..  //==>这里我们后面可以使用异常处理机制搞定-》老师点一下 |

|  |
| --- |
| int grade = myScanner.nextInt();  //先对输入的信用分，进行一个范围的有效判断 1- 100, 否则提示输入错误  if(grade >= 1 && grade <= 100) {  //因为有 4 种情况，所以使用多分支  if(grade == 100) {  System.out.println("信用极好");  } else if (grade > 80 && grade <= 99) { //信用分为(80 ，99]时，输出 信用优秀；  System.out.println("信用优秀");  } else if (grade >= 60 && grade <= 80) {//信用分为[60,80]时，输出 信用一般  System.out.println("信用一般");  } else {//其它情况 ，输出 信用 不及格  System.out.println("信用不及格");  }  } else {  System.out.println("信用分需要在 1- 100,请重新输入:)");  }  }  } |

 案例演示 2



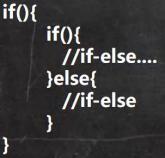
5.5 嵌套分支

5.5. 1基本介绍

**在一个分支结构中又完整的嵌套了另一个完整的分支结构**，里面的分支的结构称为内层分支外面的分支结构称为外

层分支。老师建议: 不要超过 3 层 （可读性不好）

5.5.2基本语法



5.5.3应用案例

参加歌手比赛，如果初赛成绩大于 8.0 进入决赛，否则提示淘汰。并且根据性别提示进入男子组或女子组。【可以

让学员先练习下】, 输入成绩和性别，进行判断和输出信息。[NestedIf.java]

提示: double score; char gender;

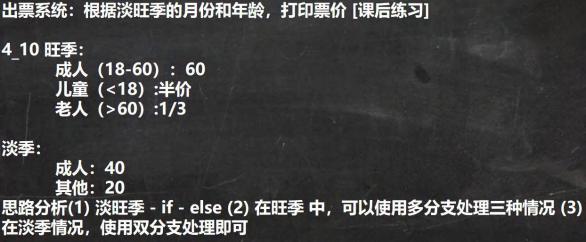
接收字符: char gender = scanner.next().charAt(0)

|  |
| --- |
| import java.util.Scanner; |

|  |
| --- |
| public class NestedIf {  //编写一个 main 方法  public static void main(String[] args) {  /\*  参加歌手比赛，如果初赛成绩大于 8.0 进入决赛，  否则提示淘汰。并且根据性别提示进入男子组或女子组。  【可以让学员先练习下】, 输入成绩和性别，进行判断和输出信息。  [NestedIf.java]  提示: double score; char gender;  接收字符: char gender = scanner.next().charAt(0)  \*/  //思路分析  //1. 创建 Scanner 对象，接收用户输入  //2. 接收 成绩保存到 double score  //3. 使用 if-else 判断 如果初赛成绩大于 8.0 进入决赛，否则提示淘汰  //4. 如果进入到 决赛，再接收 char gender, 使用 if-else 输出信息  //代码实现 => 思路 --> java 代码  Scanner myScanner = new Scanner(System.in);  System.out.println("请输入该歌手的成绩");  double score = myScanner.nextDouble();  if( score > 8.0 ) {  System.out.println("请输入性别"); |

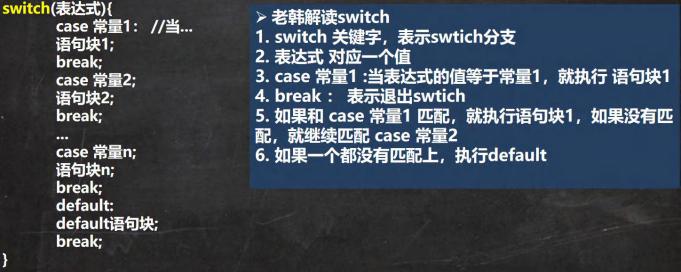
|  |
| --- |
| char gender = myScanner.next().charAt(0);  if( gender == '男' ) {  System.out.println("进入男子组");  } else if(gender == '女') {  System.out.println("进入女子组");  } else {  System.out.println("你的性别有误，不能参加决赛~");  }  } else {  System.out.println("sorry ,你被淘汰了~");  }  }  } |

5.5.4应用案例 2, 一定自己做.



5.6 switch 分支结构

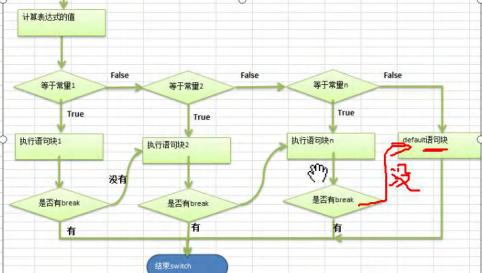
5.6. 1基本语法



5.6.2流程图

1) 画出 swtich 出流程

2) 案例说明流程图



5.6.3快速入门

案例：Switch01.java

请编写一个程序，该程序可以接收一个字符，比如:a,b,c,d,e,f,g

a 表示星期一，b 表示星期二 …

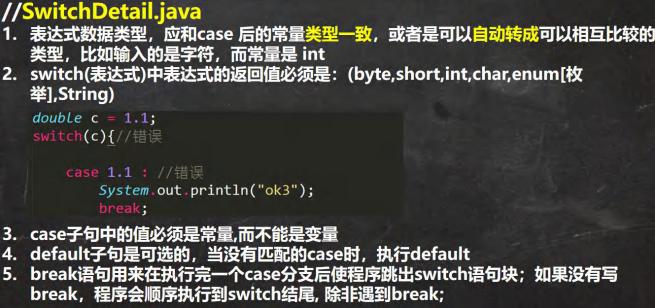
根据用户的输入显示相应的信息.要求使用 switch 语句完成

代码

|  |
| --- |
| import java.util.Scanner;  public class Switch01 {  //编写一个 main 方法  public static void main(String[] args) {  /\*  案例：Switch01.java  请编写一个程序，该程序可以接收一个字符，比如:a,b,c,d,e,f,g  a 表示星期一，b 表示星期二 …  根据用户的输入显示相应的信息.要求使用 switch 语句完成  思路分析  1. 接收一个字符 , 创建 Scanner 对象  2. 使用 switch 来完成匹配,并输出对应信息  代码  \*/  Scanner myScanner = new Scanner(System.in);  System.out.println("请输入一个字符(a-g)");  char c1 = myScanner.next().charAt(0);//  //在java 中，只要是有值返回，就是一个表达式  switch(c1) {  ' '  case a : |

|  |
| --- |
| System.out.println("今天星期一,猴子穿新衣");  break;  case 'b' :  System.out.println("今天星期二,猴子当小二");  break;  ' '  case c :  System.out.println("今天星期三,猴子爬雪山..");  break;  //.....  default:  System.out.println("你输入的字符不正确，没有匹配的");  }  System.out.println("退出了 switch ,继续执行程序");  }  } |

5.6.4switch 注意事项和细节讨论



5.6.5switch 课堂练习(学员先做)

SwitchExercise.java

1) 使用 switch 把小写类型的 char 型转为大写(键盘输入) 。只转换 a, b, c, d, e. 其它的输出 "other"。

2) 对学生成绩大于 60 分的，输出"合格"。低于 60 分的，输出"不合格"。(注：输入的成绩不能大于 100), 提示 成绩/60

3) 根据用于指定月份，打印该月份所属的季节。3,4,5 春季 6,7,8 夏季 9, 10, 11 秋季 12, 1, 2 冬季 [课堂练习, 提示 使

用穿透 ]

**代码**:

|  |
| --- |
| import java.util.Scanner;  public class SwitchExercise {  //编写一个 main 方法  public static void main(String[] args) {  //使用 switch 把小写类型的 |

|  |
| --- |
| //char 型转为大写(键盘输入) 。只转换 a->A, b->B, c, d, e.  //其它的输出 "other"。  //创建 Scanner 对象  // Scanner myScanner = new Scanner(System.in);  // System.out.println("请输入 a-e");  // char c1 = myScanner.next().charAt(0);  // switch(c1) {  // case 'a' :  // System.out.println("A");  // break;  // case 'b' :  // System.out.println("B");  // break;  // case 'c' :  // System.out.println("C");  // break;  // case 'd' :  // System.out.println("D");  // break;  // case 'e' :  // System.out.println("E");  // break;  // default :  // System.out.println("你的输入有误~"); |

|  |
| --- |
| // }  //对学生成绩大于 60 分的，输出"合格" 。低于 60 分的，  //输出"不合格" 。(注：输入的成绩不能大于 100), 提示 成绩/60  //思路分析  //1. 这道题，可以使用 分支来完成， 但是要求使用 switch  //2. 这里我们需要进行一个转换, 编程思路 :  // 如果成绩在 [60, 100] , (int)(成绩/60) = 1  // 如果成绩在 [0,60) , (int)(成绩/60) = 0  //代码实现  double score = 1. 1;  //使用 if-else 保证输入的成绩有有效的 0- 100  //看了当老师的分析和代码演示后， 自己一定要独立完成（不看老韩代码，也能写）  if( score >= 0 && score <= 100) {  switch ((int)(score / 60)) {  case 0 :  System.out.println("不合格");  break;  case 1 :  System.out.println("合格");  break;  // default :  // System.out.println("输入有误"); |

|  |
| --- |
| }  } else {  System.out.println("输入的成绩在 0- 100");  }  //根据用于指定月份，  //打印该月份所属的季节。  //3,4,5 春季 6,7,8 夏季 9, 10, 11 秋季 12, 1, 2 冬季  //[课堂练习, 提示 使用穿透 ]  //  //思路分析  //1. 创建 Scanner 对象， 接收用户输入  //2. 使用 int month 接收  //3. 使用 switch 来匹配 ,使用穿透来完成，比较简洁  Scanner myScanner = new Scanner(System.in);  System.out.println("输入月份");  int month = myScanner.nextInt();  switch(month) {  case 3:  case 4:  case 5:  System.out.println("这是春季");  break;  case 6:  case 7: |

|  |
| --- |
| case 8:  System.out.println("这是夏季");  break;  case 9:  case 10:  case 11:  System.out.println("这是秋季");  break;  case 1:  case 2:  case 12:  System.out.println("这是冬季");  break;  default :  System.out.println("你输入的月份不对(1- 12)");  }  }  } |

5.6.6switch 和 if 的比较

1) 如果**判断的具体数值不多**，而且符合 byte 、 short 、int 、 char, enum[枚举], String 这 6 种类型。虽然两个语句都可

以使用，建议使用 swtich 语句。

2) 其他情况：对区间判断，对结果为 boolean 类型判断，使用 if，if 的使用范围更广

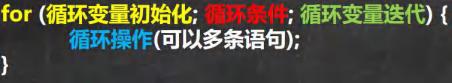
5.7 for 循环控制(!!!)

基本介绍:听其名而知其意,就是让你的代码可以循环的执行.

5.7. 1看一个实际需求

请大家看个案例[For01.java]:编写一个程序, 可以打印 10 句 "你好，韩顺平教育!" 。请大家想想怎么做?

5.7.2基本语法



> 老韩说明

1. for 关键字，表示循环控制

2. for 有四要素: (1)循环变量初始化(2)循环条件(3)循环操作(4)循环变量迭代

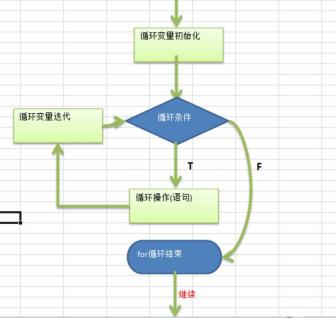
3. 循环操作 , 这里可以有多条语句，也就是我们要循环执行的代码

4. 如果 循环操作(语句) 只有一条语句，可以省略 {}, 建议不要省略

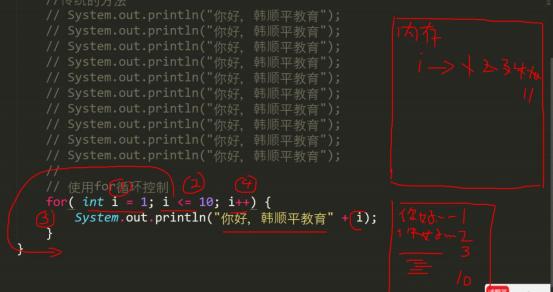
5.7.3for 循环执行流程分析

1) 使用 for循环完成前面的题

2) 画出 for 流程图



3) 代码执行, **内存分析法(初学者)**图



5.7.4注意事项和细节说明

ForDetail.java

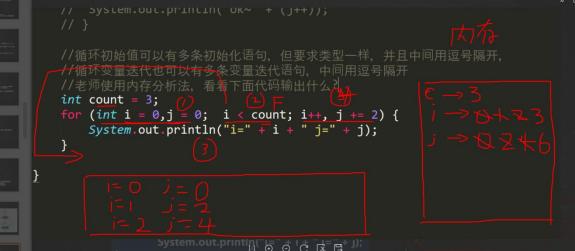
1) 循环条件是返回一个布尔值的表达式

2) for(;循环判断条件;) 中的初始化和变量迭代可以写到其它地方，但是两边的分号不能省略。

3) 循环初始值可以有多条初始化语句，但要求类型一样，并且中间用逗号隔开，循环变量迭代也可以有多条变量迭代

语句，中间用逗号隔开。

4) 使用内存分析法，老师分析输出下面代码输出什么?



5.7.5for 循环练习题(学员先做)

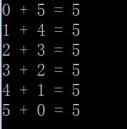
ForExercise.java

1) 打印 1~ 100 之间所有是 9 的倍数的整数，统计个数 及 总和.[**化繁为简,先死后活**]

|  |
| --- |
| public class ForExercise {  //编写一个 main 方法  public static void main(String[] args) {  //打印 1~ 100 之间所有是 9 的倍数的整数，统计个数 及 总和.[化繁为简,先死后活]  //老韩的两个编程思想(技巧)  //1. 化繁为简 : 即将复杂的需求，**拆解成简单的需求**，逐步完成 编程 = **思想** **--练习-> 代码**  //2. 先死后活 : 先考虑固定的值，然后转成可以灵活变化的值  //  //思路分析  //打印 1~ 100 之间所有是 9 的倍数的整数，统计个数 及 总和  //化繁为简 |

|  |
| --- |
| //(1) 完成 输出 1- 100 的值  //(2) 在输出的过程中，进行过滤，只输出 9 的倍数 i % 9 ==0  //(3) 统计个数 定义一个变量 int count = 0; 当 条件满足时 count++;  //(4) 总和 , 定义一个变量 int sum = 0; 当条件满足时累积 sum += i;  //先死后活  //(1) 为了适应更好的需求，把范围的开始的值和结束的值，做出变量  //(2) 还可以更进一步 9 倍数也做成变量 int t = 9;  int count = 0; //统计 9 的倍数个数 变量  int sum = 0; //总和  int start = 10;  int end = 200;  int t = 5; // 倍数  for(int i = start; i <= end; i++) {  if( i % t == 0) {  System.out.println("i=" + i);  count++;  sum += i;//累积  }  }  System.out.println("count=" + count);  System.out.println("sum=" + sum);  }  } |

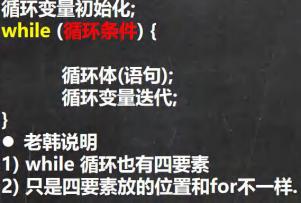
2) 完成下面的表达式输出 [老师评讲 ForExercise02.java]



|  |
| --- |
| public class ForExercise02 {  //编写一个 main 方法  public static void main(String[] args) {  //化繁为简  //先死后活  int n = 9;  for( int i = 0; i <= n; i++) {  System.out.println(i + "+" + (n-i) + "=" + n);  }  }  } |

5.8 while 循环控制

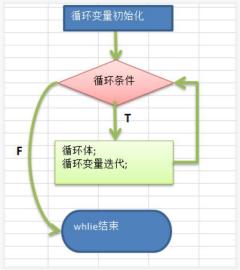
5.8. 1基本语法



5.8.2while 循环执行流程分析

While01.java

1) 画出流程图

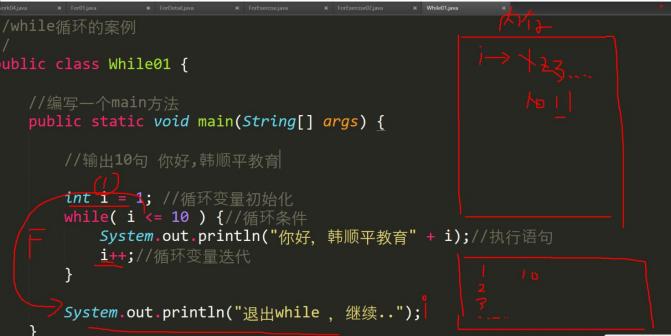


2) 使用 while 循环完成前面的题

|  |
| --- |
| //while 循环的案例  //  public class While01 {  //编写一个 main 方法  public static void main(String[] args) { |

|  |
| --- |
| //输出 10 句 你好,韩顺平教育  int i = 1; //循环变量初始化  while( i <= 10 ) {//循环条件  System.out.println("你好，韩顺平教育" + i);//执行语句  i++;//循环变量迭代  }  System.out.println("退出 while ，继续.." + i); // 11  }  } |

3) 代码执行内存分析图



5.8.3注意事项和细节说明

1) 循环条件是返回一个布尔值的表达式

2) while 循环是先判断再执行语句

5.8.4课堂练习题[学员先做]

WhileExercise.java

1) 打印 1— 100 之间所有能被 3 整除的数 [使用 while, 老师评讲 ]

2) 打印 40—200 之间所有的偶数 [使用 while, 课后练习]

代码

|  |
| --- |
| public class WhileExercise {  //编写一个 main 方法  public static void main(String[] args) {  // 打印 1— 100 之间所有能被 3 整除的数 [使用 while, 老师评讲 ]  // 化繁为简, 先死后活  int i = 1;  int endNum = 100;  while( i <= endNum) {  if( i % 3 == 0) {  System.out.println("i=" + i);  }  i++;//变量自增 |

|  |
| --- |
| }  // 打印 40—200 之间所有的偶数 [使用 while, 课后练习]  // 化繁为简, 先死后活(利于思考)  //  System.out.println("========");  intj = 40; //变量初始化  while ( j <= 200) {  //判断  if( j % 2 == 0) {  System.out.println("j=" + j);  }  j++;//循环变量的迭代  }  }  } |

5.9 do..while 循环控制

5.9. 1基本语法

**循环变量初始化;**

**do{**

**循环体(语句);**

**循环变量迭代;**

**}while(循环条件);**

5.9.2老韩说明:

1. do while 是关键字

1. 也有循环四要素, 只是位置不一样

2. 先执行，再判断，也就是说，一定会至少执行一次

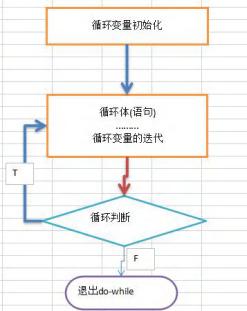
3. 最后 有一个 分号 ;

4. while 和 do..while 区别举例: 要账

5.9.3do...while 循环执行流程分析

DoWhile01.java

1) 画出流程图



2) 使用 do...while 循环完成前面的题

3) 代码执行内存分析图



5.9.4注意事项和细节说明

1) 循环条件是返回一个布尔值的表达式

2) do..while 循环是先执行，再判断， 因此它至少执行一次

5.9.5课堂练习题【学员先做】

1) 打印 1— 100 [学生做]

2) 计算 1— 100 的和 [学生做]

3) 统计 1---200 之间能被 5 整除但不能被 3 整除的个数 (DoWhileExercise01.java)

|  |
| --- |
| public class DoWhileExercise01 {  //编写一个 main 方法  public static void main(String[] args) {  //统计 1---200 之间能被 5 整除但不能被 3 整除的 个数 |

|  |
| --- |
| //化繁为简  //(1) 使用 do-while 输出 1-200  //(2) 过滤 能被 5 整除但不能被 3 整除的数 %  //(3) 统计满足条件的个数 int count = 0;  //先死后活  //(1) 范围的值 1-200 你可以做出变量  //(2) 能被 5 整除但不能被 3 整除的 , 5 和 3 可以改成变量  int i = 1;  int count = 0; //统计满足条件的个数  do {  if( i % 5 == 0 && i % 3 != 0 ) {  System.out.println("i=" + i);  count++;  }  i++;  }while(i <= 200);  System.out.println("count=" + count);  }  } |

4) 如果李三不还钱，则老韩将一直使出五连鞭，直到李三说还钱为止

[System.out.println("老韩问：还钱吗？y/n")] do...while ..

DoWhileExercise02.java

|  |
| --- |
| import java.util.Scanner;  public class DoWhileExercise02 { |

|  |
| --- |
| //编写一个 main 方法  public static void main(String[] args) {  //如果李三不还钱，则老韩将一直使出五连鞭，直到李三说还钱为  //[System.out.println("老韩问：还钱吗？y/n")] do...while ..  //  //化繁为简  //(1) 不停的问还钱吗?  //(2) 使用 char answer 接收回答, 定义一个 Scanner 对象  //(3) 在 do-while 的 while 判断如果是 y 就不在循环  //一定自己动脑筋..  Scanner myScanner = new Scanner(System.in);  char answer = ' ';  do {  System.out.println("老韩使出五连鞭~");  System.out.println("老韩问：还钱吗？y/n");  answer = myScanner.next().charAt(0);  System.out.println("他的回答是" + answer);  }while(answer != 'y');//判断条件很关键  System.out.println("李三还钱了");  }  } |

5. 10 多重循环控制(难点! 重点!)

5.10.1 介绍

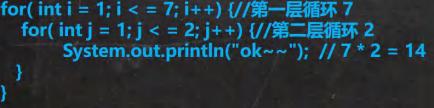
1) 将一个循环放在另一个循环体内，就形成了嵌套循环。其中，for ,while ,do …while 均可以作为外层循环和内层循环。

【建议一般使用两层，最多不要超过 3 层, 否则，代码的可读性很差】

2) 实质上，嵌套循环就是把内层循环当成外层循环的循环体。当只有内层循环的循环条件为 false 时，才会完全跳出内

层循环，才可结束外层的当次循环，开始下一次的循环[听不懂，走案例]。

3) 设外层循环次数为 m 次，内层为 n 次，则内层循环体实际上需要执行 m\*n 次。



5.10.2 多重循环执行步骤分析：

请分析 下面的多重循环执行步骤, 并写出输出 => 韩老师的内存分析法

//双层 for MulFor.java

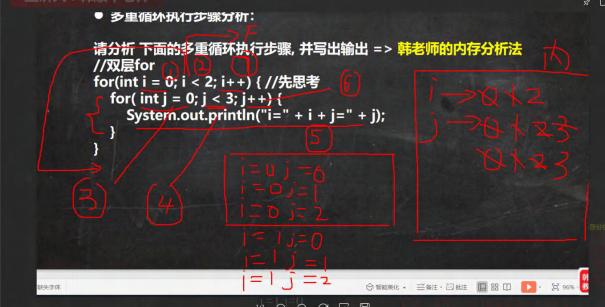
for(int i = 0; i < 2; i++) { //先思考

for( intj = 0; j < 3; j++) {

System.out.println("i=" + i + j=" + j);

}

}



5.10.3 应用实例：

//MulForExercise01.java

1) 统计 3 个班成绩情况，每个班有 5 名同学，求出各个班的平均分和所有班级的平均分[学生的成绩从键盘输入]。

2) 统计三个班及格人数，每个班有 5 名同学。

|  |
| --- |
| import java.util.Scanner;  public class MulForExercise01 {  //编写一个 main 方法  public static void main(String[] args) {  //统计 3 个班成绩情况，每个班有 5 名同学，  //求出各个班的平均分和所有班级的平均分[学生的成绩从键盘输入]。  //统计三个班及格人数，每个班有 5 名同学。  //  //**思路分析:** |

|  |
| --- |
| //化繁为简  //(1) 先计算一个班 , 5 个学生的成绩和平均分 , 使用 for  //1. 1 创建 Scanner 对象然后，接收用户输入  //1.2 得到该班级的平均分 , 定义一个 doubel sum 把该班级 5 个学生的成绩累积  //(2) 统计 3 个班(每个班 5 个学生) 平均分  //(3) 所有班级的平均分  //3. 1 定义一个变量，double totalScore 累积所有学生的成绩  //3.2 当多重循环结束后，totalScore / (3 \* 5)  //(4) 统计三个班及格人数  //4. 1 定义变量 int passNum = 0; 当有一个学生成绩>=60, passNum++  //4.2 如果 >= 60 passNum++  //(5) 可以优化[效率，可读性, 结构]  //创建 Scanner 对象  Scanner myScanner = new Scanner(System.in);  double totalScore = 0; //累积所有学生的成绩  int passNum = 0;//累积 及格人数  int classNum = 3; //班级个数  int stuNum = 5;//学生个数  for( int i = 1; i <= classNum; i++) {//i 表示班级  double sum = 0; //一个班级的总分  for( intj = 1; j <= stuNum; j++) {//j 表示学生  System.out.println("请数第"+i+"个班的第"+j+"个学生的成绩");  double score = myScanner.nextDouble(); |

|  |
| --- |
| //当有一个学生成绩>=60, passNum++  if(score >= 60) {  passNum++;  }  sum += score; //累积  System.out.println("成绩为" + score);  }  //因为 sum 是 5 个学生的总成绩  System.out.println("sum=" + sum + " 平均分=" + (sum / stuNum));  //把 sum 累积到 totalScore  totalScore += sum;  }  System.out.println("三个班总分="+ totalScore  + " 平均分=" + totalScore / (classNum\*stuNum));  System.out.println("及格人数=" + passNum);  }  } |

3) 打印出九九乘法表[课后题]



5.10.4 经典的打印金字塔

使用 for 循环完成下面的案例

请编写一个程序，可以接收一个整数,表示层数（totalLevel），打印出金字塔。(Stars.java) [化繁为简, 先死后活]



|  |
| --- |
| public class Stars {  //编写一个 main 方法  public static void main(String[] args) {  /\*  \*  \* \*  \* \*  \*\*\*\*\*\*\*\*  思路分析  化繁为简  1. 先打印一个矩形  \*\*\*\*\*  \*\*\*\*\* |

|  |  |  |
| --- | --- | --- |
| \*\*\*\*\*  \*\*\*\*\*  \*\*\*\*\*  2. 打印半个金字塔  \* //第 1 层 有 1 个\* \*\* //第 2 层 有 2 个\* \*\*\* //第 3 层 有 3 个\*  \*\*\*\* //第 4 层 有 4 个\*  \*\*\*\*\* //第 5 层 有 5 个\* | | |
| 3. 打印整个金字塔  \* //第 1 层 有 1 个\* \*\*\* //第 2 层 有 3 个\*  \*\*\*\*\* //第 3 层 有 5 个\*  \*\*\*\*\*\*\* //第 4 层 有 7 个\*  \*\*\*\*\*\*\*\*\* //第 5 层 有 9 个\* | 2 \* 1 - 1 2 \* 2 - 1 2 \* 3 - 1 2 \* 4 - 1  2 \* 5 - 1 | 有 4=(总层数- 1)个空格  有 3=(总层数-2)个空格  有 2=(总层数-3)个空格  有 1=(总层数-4)个空格  有 0=(总层数-5)个空格 |
| 4. 打印空心的金字塔 [最难的]   |  |  |  | | --- | --- | --- | | \* | //第 1 层 有 1 个\* | 当前行的第一个位置是\*,最后一个位置也是\* | | \* \* | //第 2 层 有 2 个\* | 当前行的第一个位置是\*,最后一个位置也是\* | | \* \* | //第 3 层 有 2 个\* | 当前行的第一个位置是\*,最后一个位置也是\* | | \* \* | //第 4 层 有 2 个\* | 当前行的第一个位置是\*,最后一个位置也是\* | | \*\*\*\*\*\*\*\*\* | //第 5 层 有 9 个\* | 全部输出\* |   先死后活 | | |

|  |
| --- |
| 5 层数做成变量 int totalLevel = 5;  //小伙伴 技术到位，就可以很快的把代码写出  \*/  int totalLevel = 20; //层数  for(int i = 1; i <= totalLevel; i++) { //i 表示层数  //在输出\*之前，还有输出 对应空格 = 总层数- 当前层  for(int k = 1; k <= totalLevel - i; k++ ) {  System.out.print(" ");  }  //控制打印每层的\*个数  for(intj = 1;j <= 2 \* i - 1;j++) {  //当前行的第一个位置是\*,最后一个位置也是\*, 最后一层全部 \*  if(j == 1 || j == 2 \* i - 1 || i == totalLevel) {  System.out.print("\*");  } else { //其他情况输出空格  System.out.print(" ");  }  }  //每打印完一层的\*后，就换行 println 本身会换行  System.out.println("");  }  }  } |

5. 11 跳转控制语句-break

5.11.1 看下面一个需求

随机生成 1- 100 的一个数，直到生成了 97 这个数，看看你一共用了几次?

提示使用 (int)(Math.random() \* 100) + 1

思路分析：

循环，但是循环的次数不知道. -> break , 当某个条件满足时，终止循环

通过该需求可以说明其它流程控制的必要性，比如 break

5.11.2 基本介绍：

**break 语句用于终止某个语句块的执行**，一般使用在 switch 或者循环[for , while , do-while]中

5.11.3 基本语法：

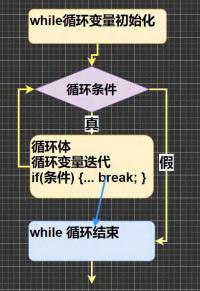
{ … …

break;

……

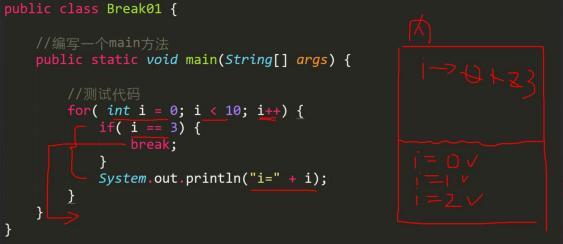
}

5.11.4 以 while 使用 break 为例,画出示意图

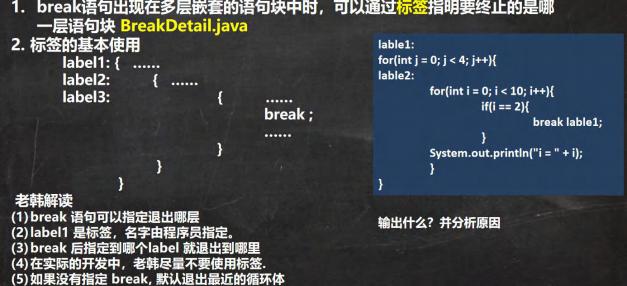


5.11.5 快速入门:

Break01.java



5.11.6 注意事项和细节说明：



5.11.7 课堂练习题：

BreakExercise.java

1) 1- 100 以内的数求和，求出 当和 第一次大于 20 的当前数 【for + break】

|  |
| --- |
| public class BreakExercise {  //编写一个 main 方法  public static void main(String[] args) {  //1- 100 以内的数求和，求出 当和 第一次大于 20 的当前数 【for + break】  //思路分析  //1. 循环 1- 100, 求和 sum  //2. 当 sum > 20 时，记录下当前数，然后 break  //3. 在 for 循环外部，定义变量 n , 把当前 i 赋给 n  int sum = 0; //累积和  //注意 i 的作用范围在 for{}  int n = 0;  for(int i = 1; i <= 100; i++) {  sum += i;//累积  if(sum > 20) {  System.out.println("和>20 时候 当前数 i=" + i);  n = i;  break;  }  }  System.out.println(" 当前数=" + n);  } |

|  |
| --- |
| } |

2) 实现登录验证，有 3 次机会，如果用户名为"丁真" ,密码"666"提示登录成功，否则提示还有几次机会，请使用 for+break

完成 BreakExercise02.java

|  |
| --- |
| import java.util.Scanner;  public class BreakExercise02 {  //编写一个 main 方法  public static void main(String[] args) {  //实现登录验证，有 3 次机会，如果用户名为"丁真" ,密码"666"提示登录成功，  //否则提示还有几次机会，请使用 for+break 完成  //  // 思路分析  // 1. 创建 Scanner 对象接收用户输入  // 2. 定义 String name ; String passwd; 保存用户名和密码  // 3. 最多循环 3 次[登录 3 次] ，如果 满足条件就提前退出  // 4. 定义一般变量 int chance 记录还有几次登录机会  //  // 代码实现  Scanner myScanner = new Scanner(System.in);  String name = "";  String passwd = "";  int chance = 3; //登录一次 ，就减少一次  for( int i = 1; i <= 3; i++) {//3 次登录机会 |

|  |
| --- |
| System.out.println("请输入名字");  name = myScanner.next();  System.out.println("请输入密码");  passwd = myScanner.next();  //比较输入的名字和密码是否正确  //补充说明字符串 的内容 比较 使用的 方法 equals  if("丁真".equals(name) && "666".equals(passwd)) {  System.out.println("恭喜你，登录成功~");  break;  }  //登录的机会就减少一次  chance--;  System.out.println("你还有" + chance + "次登录机会");  }  }  } |

5. 12 跳转控制语句-continue

5.12.1 基本介绍：

1) **continue 语句用于结束本次循环**，**继续执行下一次循环**。

2) continue 语句出现在多层嵌套的循环语句体中时，可以通过标签指明要跳过的是哪一层循环 , 这个和前面的标签的

使用的规则一样.

5.12.2 基本语法：

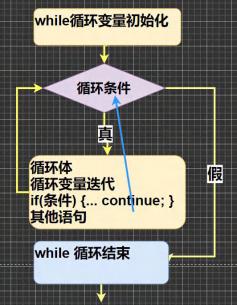
{ … …

continue;

……

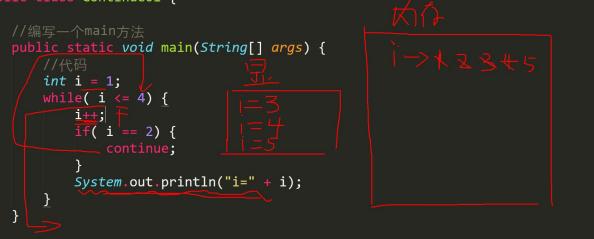
}

5.12.3 以 while 使用 continue 为例,画出示意图



5.12.4 快速入门案例

Continue01.java



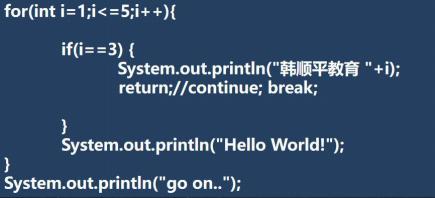
5.12.5 细节案例分析和说明： ContinueDetail.java

|  |
| --- |
| label1:  for(intj = 0; j < 4; j++){  label2:  for(int i = 0; i < 10; i++){  if(i == 2){  //看看分别输出什么值，并分析  //continue ;  //continue label2;  continue label1;  }  System.out.println("i = " + i);  }  } |

5. 13 跳转控制语句-return

return 使用在方法，表示跳出所在的方法，在讲解方法的时候，会详细的介绍，这里我们简单的提一下。注意：如

果 return 写在 main 方法，退出程序..Return01.java



5. 14 本章作业

