第四章：流程控制语句

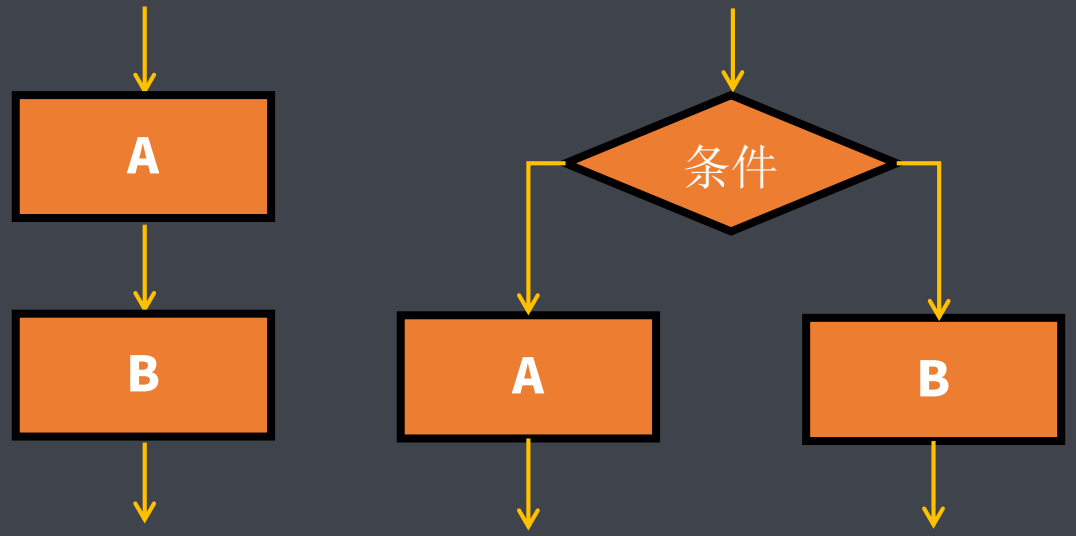
奇点
拉勾网高级Java讲师

目录

第一节：分支结构（重中之重）

分支结构的概念

- 当需要进行条件判断并做出选择时，使用分支结构。

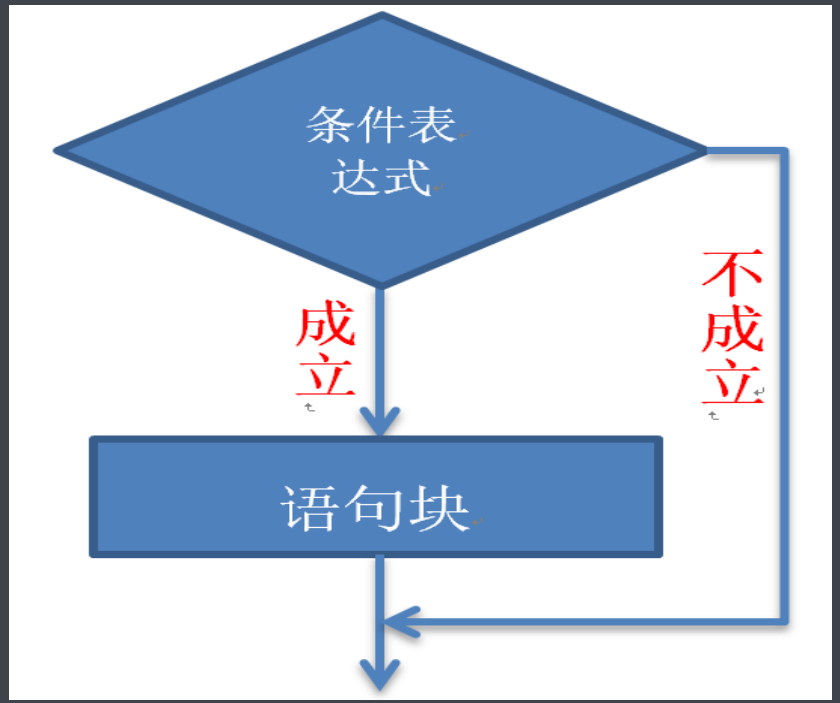


if分支结构

- if(条件表达式) {

  语句块;

  }



if分支结构

- 判断条件表达式是否成立

  => 若成立，则执行语句块；

  => 若不成立，则跳过语句块；

案例题目

- 提示用户输入两个整数，使用if分支结构找到最大值并打印出来。

if else分支结构

if else分支结构

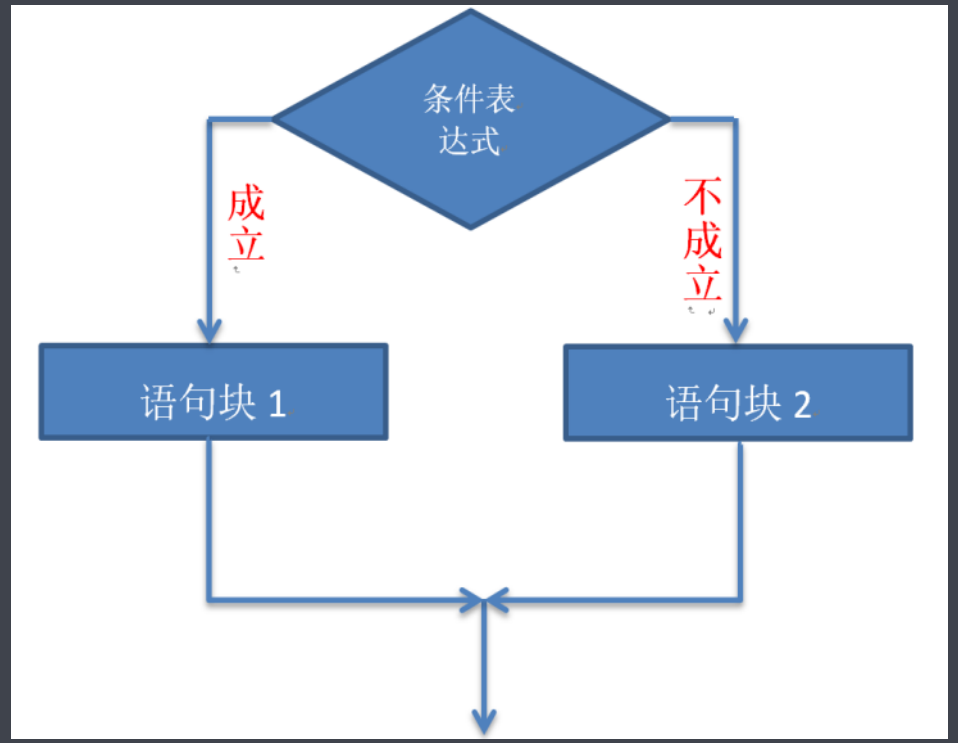
- if(条件表达式) {

  语句块1;

  } else {

  语句块2;

  }



if else分支结构

- 判断条件表达式是否成立

  => 若成立，则执行语句块1；

  => 若不成立，则执行语句块2；

案例题目

- 提示用户输入一个整数，使用if else分支结构判断该整数是负数还是非负数并打印。
- 使用if else分支结构判断该整数是正数、负数还是零。

if else if else分支结构

if else if else分支结构

- if(条件表达式1) {

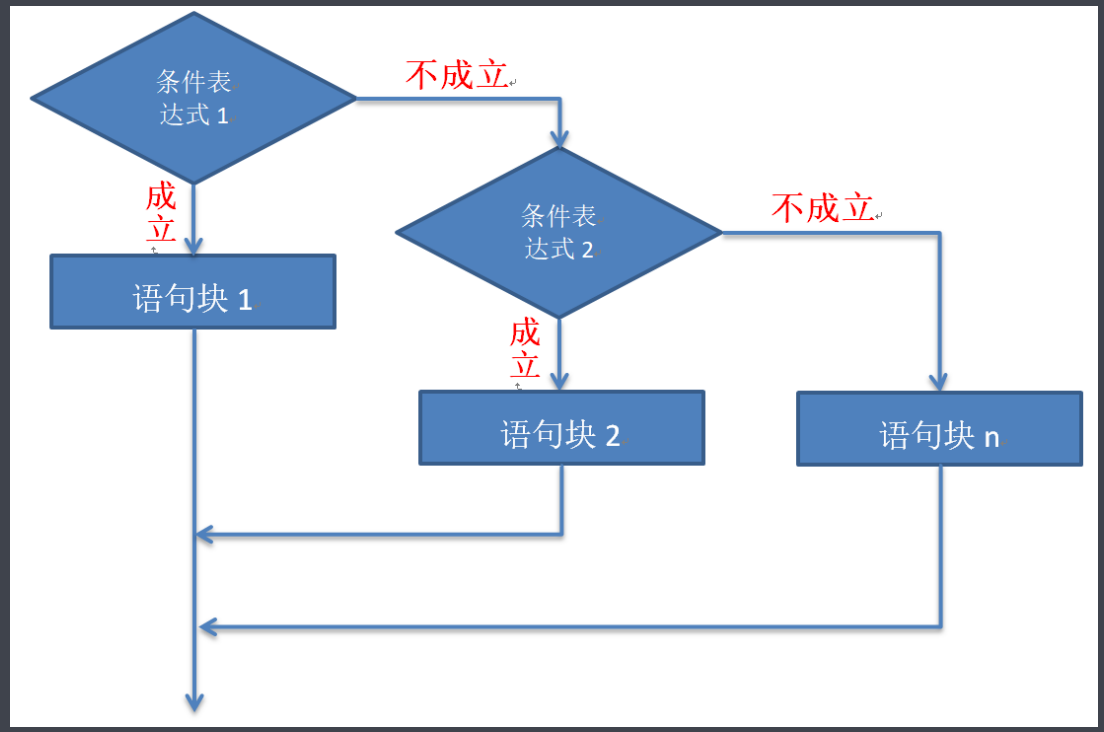
  语句块1; }

  else if(条件表达式2) {

  语句块2; }

  else {

  语句块n; }



if else if else分支结构

- 判断条件表达式1是否成立

  => 若成立，则执行语句块1；

  => 若不成立，则判断条件表达式2是否成立

  => 若成立，则执行语句块2；

  => 若不成立，则执行语句块n；

案例题目

- 根据用户输入的薪水计算个人所得税并打印出来，其中个税起征点为：5000元，具体规则如下：

全月应纳税所得额	税率	速算扣除数(元)
全月应纳税额不超过3000元(8000)	3%	0
全月应纳税额超过3000元至12000元(17000)	10%	210
全月应纳税额超过12000元至25000元(30000)	20%	1410
全月应纳税额超过25000元至35000元(40000)	25%	2660
全月应纳税额超过35000元至55000元(60000)	30%	4410
全月应纳税额超过55000元至80000元(85000)	35%	7160
全月应纳税额超过80000元	45%	15160

案例题目

- 出租车计费方式：由里程钱数和等候时间钱数相加得出。
- 里程数前3公里13元，超过3公里到15公里部分每公里2元，15公里以上部分每公里3元。
- 等候时间每2分半1元，不足部分不要钱。
- 输入公里数和等候秒数，输出车费。
- 16公里，等候290秒，车费 = 13 + (15-3) *2 + (16-15) *3 + 1 = 41

案例题目

- 提示用户输入考试的成绩，使用if-else if-else分支结构判断所在等级并打印。

  [90 ~ 100]　等级A

  [80 ~ 89]　　等级B

  [70 ~ 79]　　等级C

  [60 ~ 69]　　等级D

  [0 ~ 59]　　等级E

switch case分支结构

- switch(变量/表达式) {

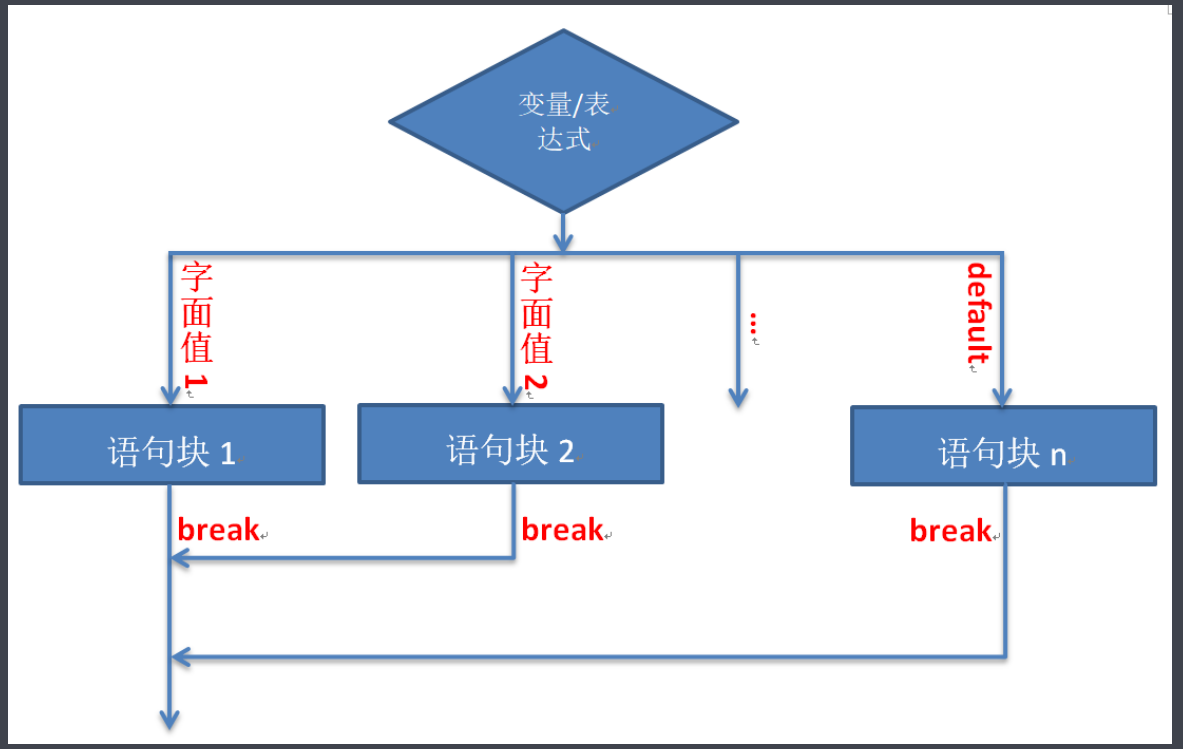
  case 字面值1: 语句块1; break;

  case 字面值2: 语句块2; break;

  ...

  default:语句块n;

  }



switch case分支结构

- 计算变量/表达式的数值 => 判断是否匹配字面值1

  => 若匹配，则执行语句块1 => 执行break跳出当前结构

  => 若不匹配，则判断是否匹配字面值2

  => 若匹配，则执行语句块2 => 执行break跳出当前结构

  => 若不匹配，则执行语句块n

switch case分支结构

- switch()中支持的数据类型有：byte、 short、 char以及int类型，从jdk1.5开始支持枚举类型，从jdk1.7开始支持String类型。

案例题目

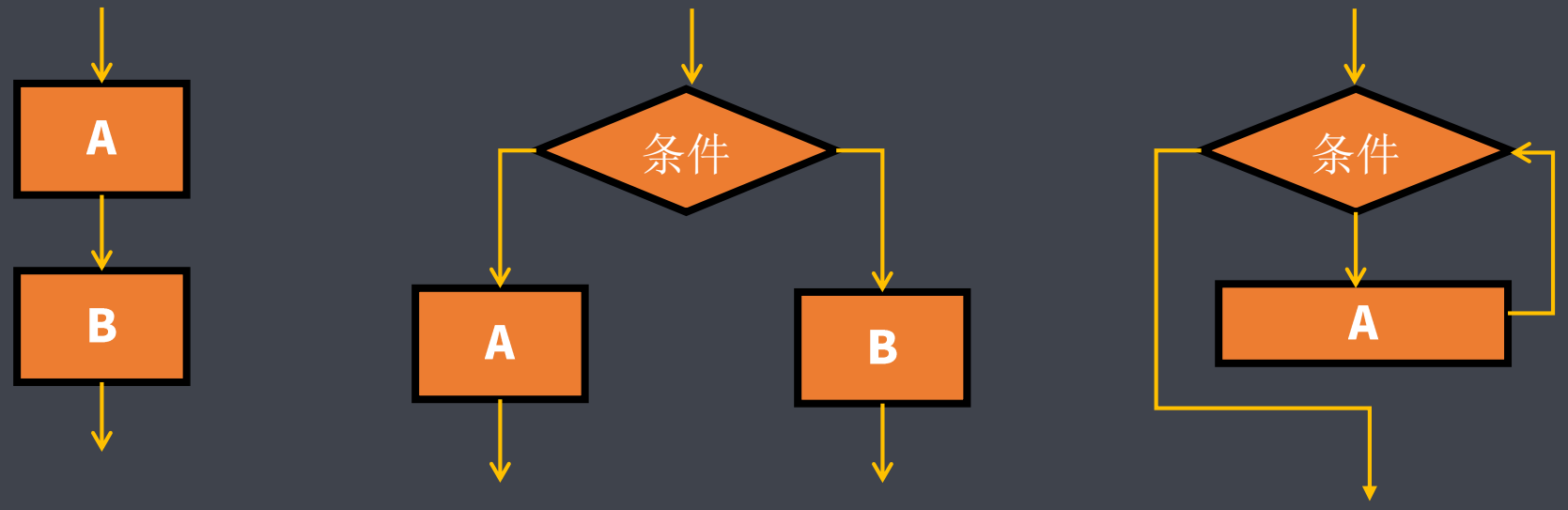
- 使用switch case分支结构模拟以下菜单效果。

```
            欢迎来到拉勾教育
------------------------------------
 [1]学员系统        [2]管理员系统
 [0]退出系统
------------------------------------
请选择要进入的系统：
```

第二节：循环结构（重中之重）

循环结构的概念

- 在Java程序中若希望重复执行一段代码时，就需要使用循环结构。
- 任何复杂的程序逻辑都可以通过顺序、分支、循环三种程序结构实现。

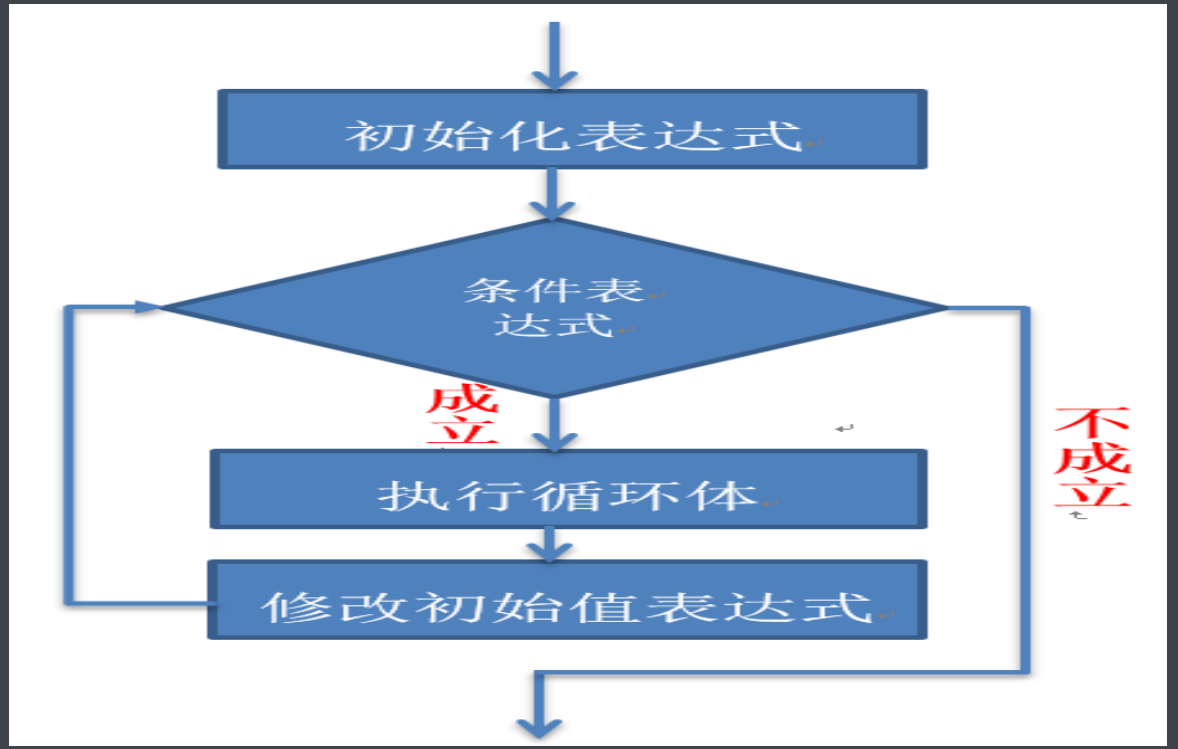


for循环

- for(初始化表达式; 条件表达式; 修改初始值表达式) {

  循环体;

  }



for循环

- 执行初始化表达式 => 判断条件表达式是否成立

  => 成立则执行循环体 => 修改初始值表达式 => 判断条件是否成立

  => 若不成立，则循环结束

案例题目

- 使用for循环打印1-100的所有奇数，使用三种方式。

案例题目

- 使用for循环实现累加：1+2+...+10000=？最后打印出来。

案例题目

- 使用for循环打印三位数中所有水仙花数。
- 所谓“水仙花数”即一个整数满足其值等于各个数位的立方和。
- 如：153是一个水仙花数，因为153=1^3+5^3+3^3。

continue关键字

- continue语句用在循环体中，用于结束本次循环而开始下一次循环。

案例题目

- 使用for循环打印1 ~ 20之间的所有整数，若遇到5的倍数则跳过不打印。

break关键字

- break用于退出当前语句块，break用在循环体中用于退出循环。
- for(;;) - 这种没有循环条件的循环叫做 无限循环，俗称“死循环”。

案例题目

- 不断地提示用户输入聊天内容并输出，直到用户输入”bye”结束聊天。

案例题目

- 猜数字游戏
- 随机生成数字n(1-100), 等待用户输入猜测数据, 根据用户的输入比较输出猜大了，猜小了，猜对了, 如果用户猜对了就结束游戏 。

双重for循环的格式

- for(初始化表达式1; 条件表达式2; 修改初始值表达式3) {

  for(初始化表达式4; 条件表达式5; 修改初始值表达式6) {

  循环体;

  }

}

双重for循环的执行流程

- 执行表达式1 => 判断表达式2是否成立

  => 若成立，则执行表达式4 => 判断表达式5是否成立

  => 若成立，则执行循环体 => 执行表达式6 => 表达式5是否成立

  => 若不成立，则内层循环结束 => 表达式3 => 表达式2是否成立

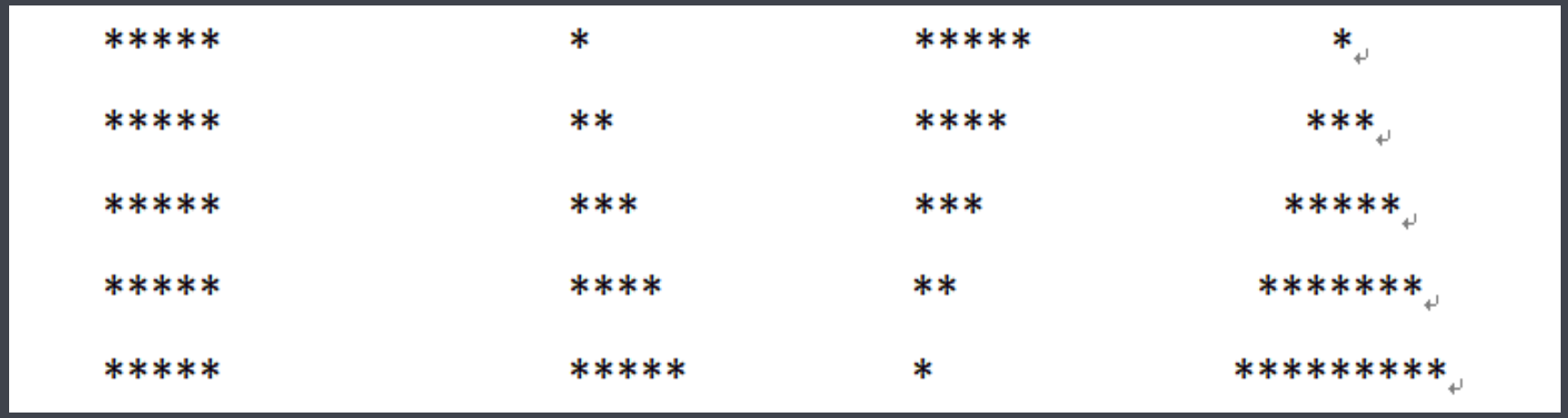
  => 若不成立，则外层循环结束

双重for循环的特点

- 外层循环用于控制打印的行数，内层循环用于控制打印的列数，外层循环改一下，内层循环从头到尾跑一圈。
- 在以后的开发中若需要打印多行多列时，需要使用双重循环。
- 多重循环不宜嵌套太多层，否则效率很低。一般到三重循环即可，最常见的就是双重循环。

案例题目

- 使用双重for循环分别打印以下图案



案例题目

- 使用双重for循环打印九九乘法表。

```
1*1=1
1*2=2 2*2=4
1*3=3 2*3=6 3*3=9
1*4=4 2*4=8 3*4=12 4*4=16
1*5=5 2*5=10 3*5=15 4*5=20 5*5=25
1*6=6 2*6=12 3*6=18 4*6=24 5*6=30 6*6=36
1*7=7 2*7=14 3*7=21 4*7=28 5*7=35 6*7=42 7*7=49
1*8=8 2*8=16 3*8=24 4*8=32 5*8=40 6*8=48 7*8=56 8*8=64
1*9=9 2*9=18 3*9=27 4*9=36 5*9=45 6*9=54 7*9=63 8*9=72 9*9=81
```

break关键字跳出多层循环

- 在嵌套的循环结构中，break用于退出所在循环体。
- 如果要退出外层循环体，需要使用标号的方式。

```
for (...) {
  for(...) {
    break;
  }
}
```

```
outer: for (...) {
  for(...) {
    break outer;
  }
}
```

案例题目

- 使用双重for循环打印2~100之间的所有素数。
- 当一个数只能被1和它本身整除时，这个数就叫做素数或质数。

while循环

- while(条件表达式) {

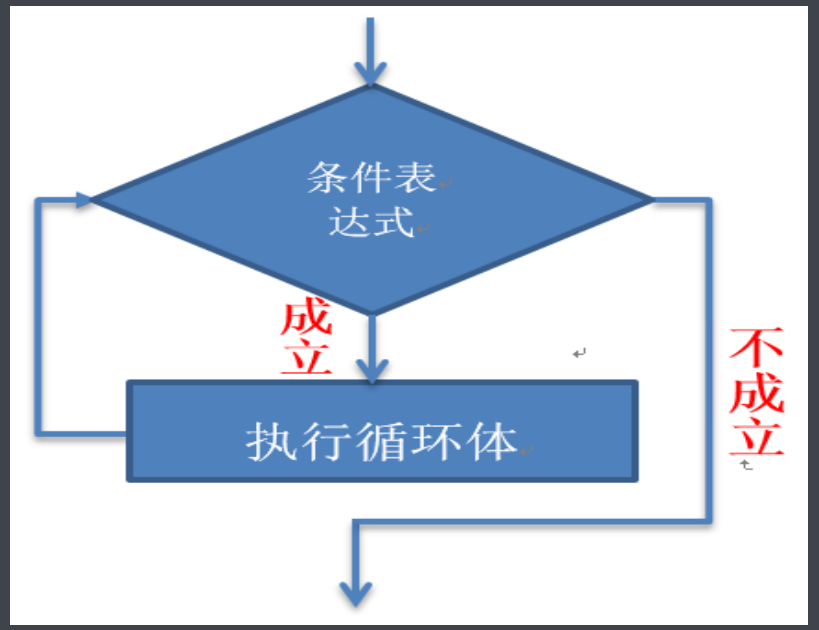
  循环体;

  }

- 判断条件表达式是否成立

  => 若成立，则执行循环体 => 判断条件表达式是否成立

  => 若不成立，则循环结束



案例题目

- 使用while循环计算调和数列的和并打印，即： 1/1 + 1/2 + ... + 1/n。

while循环和for循环比较

- while循环和for循环完全可以互换，当然推荐使用for循环。
- while循环更适合于明确循环条件但不明确循环次数的场合中。
- for循环更适合于明确循环次数或范围的场合中。
- while(true) 等价于 for(;;) 都表示无限循环。

案例题目

- 提示用户输入一个任意位数的正整数然后反向输出。

do while循环（熟悉）

- do {

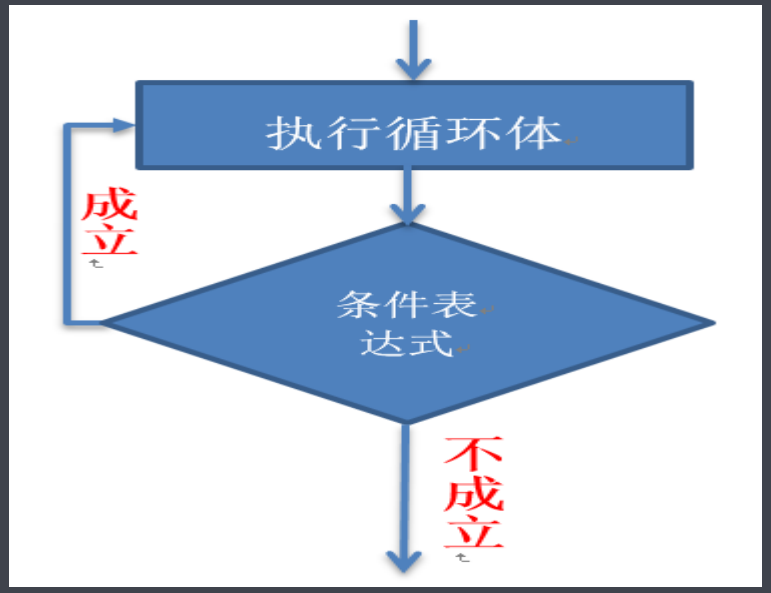
  循环体;

  } while(条件表达式);

- 执行循环体 => 判断条件表达式是否成立

  => 若成立，则执行循环体 => 判断条件表达式是否成立

  => 若不成立，则循环结束



do while循环

- do-while循环主要用于至少执行一次循环体的场合中。

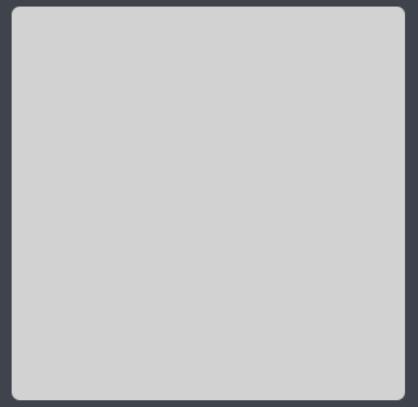
案例题目

- 使用do while循环来模拟学习任务是否合格的检查， 如果合格则停止，否则就重新完成学习任务。

总结和答疑

拉勾教育

—互联网人实战大学—



下载「拉勾教育App」
获取更多内容

L / A / G / O / U