**ava**

**J**

第 **3** 章

程序控制结构

程序的基本结构包括顺序结构、选择结构和循环结构。顺序结构按照语句的书写次 序顺序执行。选择结构根据条件成立与否，选择执行不同路径的程序语句。循环结构在条件满足的情况下重复执行一条或多条语句。

# 顺序结构

程序由多条语句组成，语句用于向计算机系统发出操作指令。Java 语言中的语句主要分为以下 5 类。

1. 表达式语句

视 频 讲 解

Java 语言中最常见的语句是表达式语句，在表达式后加一个分号即构成表达语句。例如：

x=33;

1. 方法调用语句

方法调用语句由方法调用加上一个分号组成。例如：

System.out.println("Java Program");

1. 复合语句

复合语句用花括号 { } 将多条语句括起来，在语法上作为一条语句使用。例如：

{

temp = a; a = b;

b = temp;

}

1. 空语句

空语句只有一个分号，不执行任何操作。设计空语句是为了语法需要。例如下面循 环语句的循环体只有一条空语句，表示执行空循环。

**Java** 程序设计

while(x>y) ;

1. 控制语句

控制语句完成一定的控制功能，包括选择语句 if 和 switch，循环语句 while、do-while 和 for，跳转语句 continue 和 break。

顺序结构是最简单的一种程序结构，程序按照语句的书写次序顺序执行。

【例 3-1】计算 Java 工程师的月薪。月薪按以下公式计算： Java 工程师月薪 = 月底薪 + 月实际绩效 + 月餐补 - 月保险

其中：月实际绩效 = 月底薪 ×25%× 月工作完成分数 ÷100，月工作完成分数最小值为 0，最大值为 150。月餐补 = 月实际工作天数 ×15。月保险 = 月底薪 ×10.5%。

//CH03\_01.java

import java.util.Scanner; public class CH03\_01 {

public static void main(String[] args){ Scanner input=new Scanner(System.in);

double engSalary; // 月薪

int baseSalary; // 底薪

int comResult; // 月工作完成分数

double workDay; // 月实际工作天数

double insurance; // 月保险System.out.println(" 请输入底薪 "); baseSalary=input.nextInt(); System.out.println(" 请输入月工作完成分数 "); comResult=input.nextInt(); System.out.println(" 请输入月实际工作天数 "); workDay=input.nextDouble(); insurace=baseSalary\*0.105;

engSalary=baseSalary+baseSalary\*0.25\*comResult/100+workDay\*15-

insurance;

}

System.out.println("Java 工程师月薪为 "+engSalary); input.close();

}

程序执行结果如下：



第 3 章 程序控制结构

# 选择结构

选择结构又称为分支结构，根据条件的成立与否决定执行哪些语句。选择语句有两种，即 if 语句和 switch 语句。

* + 1. if 语句

视 频 讲 解

if 语句是最常用的选择语句。Java 提供了三种形式的 if 语句结构，即单分支结构、双分支结构和多分支结构。

1. 单分支结构

单分支结构的 if 语句格式如下：

if( 条件表达式 ){

语句

}

如果条件表达式值为 true，则执行 {} 里的语句，否则直接越过 {} 里的语句。执行流程如图 3-1 所示。

在使用 if 语句时，要注意以下两点：

① if 后面小括号 () 内表达式的值必须是布尔类型的值，不要与 C 语言混淆。

② 花括号 {} 里如果只有一条语句，{} 也可以省略不写；但是为了增强程序的可读性和可维护性，最好不要省略 {}，这也是一种很好的编程习惯。

᲍Т㶔䓪ᐻ

false

true

䄙ऑ

图 3-1 单分支 if 结构

【例 3-2】判断下列程序语句是否存在问题？如何修改才能输出结果？

//CH03\_02.java

public class CH03\_02 {

public static void main(String[] args){ if(1){

System.out.println(" 注意 if 后条件表达式值的类型 ");

}

}

}

**Java** 程序设计

程序执行结果如下：



将 if 后小括号内的整数 1 改为布尔值 true，程序即可正确执行并输出结果。

1. 双分支结构

双分支结构是 if 语句最通用的一种形式，语句格式如下：

if( 条件表达式 ){

语句 1

}

else{

语句 2

}

如果条件表达式值为 true，则执行 if 后 {} 里的语句 1；否则，执行 else 后 {} 里的语句

2。执行流程如图 3-2 所示。

true

᲍Т㶔䓪ᐻ

false

䄙ऑl

䄙ऑ2

图 3-2 双分支 if 结构

【例 3-3】根据给定的整数，判断该数是奇数还是偶数。

//CH03\_03.java

import java.util.Scanner; public class CH03\_03 {

public static void main(String[] args){ Scanner input=new Scanner(System.in);

System.out.println(" 请输入一个整数：");

int x=input.nextInt(); if(x%2==0){

System.out.println(x+" 是一个偶数 ");

}

else{

System.out.println(x+" 是一个奇数 ");

}

input.close();

}

}

第 3 章 程序控制结构

程序执行结果如下：



1. 多分支结构

多分支结构的 if 语句格式如下：

if( 条件表达式 1){

语句 1

}else if( 条件表达式 2){

语句 2

}

…

else if( 条件表达式 n){

语句 n

}else{

语句 n+1

}

多分支结构的 if 语句执行流程如图 3-3 所示。

 false

条件表达式1

 false

true

条件表达式2

false

false

true

条件表达式n

true

语句n

语句2

语句1

语句n+1

图 3-3 多分支 if 结构

【例 3-4】根据给定的 x 值，计算分段函数 y 的结果。

0 , x < 0

｛

y = x , 0 ≤ x < 10

-0.5x+10 , x ≥ 10

**Java** 程序设计

//CH03\_04.java

import java.util.Scanner; public class CH03\_04 {

public static void main(String[] args){ Scanner input=new Scanner(System.in);

System.out.println(" 请输入 x 的值：");

double x=input.nextDouble(); if(x<0){

System.out.println("y="+0);

}else if(x>=0 && x<10){

System.out.println("y="+x);

}else{

System.out.println("y="+(-0.5\*x+10));

}

}

input.close();

}

}

程序执行结果如下：



* + 1. switch 语句

switch 语句是 Java 的多分支语句。它提供了一种基于一个表达式的值来使程序执行不同部分的简单方法。switch 语句的格式如下：

switch( 表达式 ){

case 常量值 1: 语句 1 [break;]

case 常量值 2: 语句 2 [break;]

…

case 常量值 n: 语句 n [break;] [default: 语句 n+1]

}

switch 语句中的表达式必须为 int、short、char、byte、String 和 enum 枚举类型。case 后的值必须是与表达式类型兼容的一个常量值，而且各常量值要互不相同。

switch 语句的执行流程是：首先计算表达式的值，然后将表达式的值与每个 case 中的常量值作比较，如果找到一个与之相匹配的值，则执行该 case 后的语句，直到遇到 break 语句就跳出 switch 结构；如果没有一个 case 中的常量值与表达式值相匹配，则执行 default 后的语句。

default 为可选参数，如果没有这个参数，而且所有的常量值与表达式值都不匹配，则switch 语句不会执行任何操作。

第 3 章 程序控制结构

break 也为可选参数，主要用于跳出 switch 结构，如果没有使用 break 参数，则程序会继续向下执行下一个 case 后的语句，直到遇到 break 语句为止。

【例 3-5】根据给定的字母，判断该字母是元音字母、半元音字母，还是辅音字母。元音字母包括 a、e、i、o、u；半元音字母包括 y、w；其他字母均为辅音字母。

//CH03\_05.java

import java.util.Scanner; public class CH03\_05 {

public static void main(String[] args){ Scanner input=new Scanner(System.in);

System.out.println(" 请输入一个 97~122 的整数：");

char c=(char)input.nextInt(); switch (c){

case 'a':

case 'e':

case 'i':

case 'o':

case 'u': System.out.println(c+" 是元音字母 "); break; case 'y':

case 'w': System.out.println(c+" 是半元音字母 "); break;

default: System.out.println(c+" 是辅音字母 ");

}

input.close();

}

}

程序执行结果如下：



# 循环结构

如果同样的语句需要被执行多次，则需要使用循环结构。循环结构的特点是重复执行一段代码直到满足一定的条件为止。Java 中主要有三种循环结构， 即 for 循环、while 循环和 do-while 循环。

* + 1. for 循环

视 频 讲 解

for 循环是程序设计时常用的一种循环形式。在事先能够确定循环次数的情况下，应首选 for 循环。for 循环语句的格式如下：

for( 表达式 1; 表达式 2; 表达式 3){

循环体

**Java** 程序设计

表达式 1 通常是为循环控制变量赋初值的表达式，在整个循环过程中表达式 1 仅被

执行一次；表达式 2 是用于判断是否继续执行循环的条件表达式；表达式 3 一般是用于更改循环控制变量值的表达式，通常使用 ++ 或 - - 运算符，用于改变循环条件。

for 循环执行流程：第 1 步，执行表达式 1，为循环控制变量赋初值；第 2 步，计算表达式 2，如果表达式 2 的值为真，则执行循环体，如果为假，则结束 for 循环；第 3 步， 计算表达式 3，更改循环控制变量的值；第 4 步，重复执行第 2 步和第 3 步，直到表达式 2 的值为假，结束 for 循环。for 循环执行过程的流程图如图 3-4 所示。

㶔䓪ᐻ1

㶔䓪ᐻ2

true

false

ᓖ⣛ѿ

㶔䓪ᐻ3

图 3-4 for 循环执行过程流程图

【例 3-6】使用 for 循环在控制台输出 1~10。

//CH03\_06.java

public class CH03\_06 {

public static void main(String[] args){ for(int i=1;i<=10;i++){

System.out.print(i+" ");

}

}

}

程序执行结果如下：



for 循环语句中的三个表达式可以部分省略或者全部省略，但是两个分号不能省略。例 3-6 的 for 循环语句也可以改为如下的表示形式。

①省略表达式 1 int i=1; for(;i<=10;i++){

System.out.print(i+" ");

第 3 章 程序控制结构

②省略表达式 1 和表达式 3

int i=1; for(;i<=10;){

System.out.print(i+" "); i++;

}

③三个表达式全部省略

int i=1; for(;;){

if(i>10){

break;

}

System.out.print(i+" "); i++;

}

* + 1. while 循环

当不知道循环次数而只知道循环条件时，可以使用 while 循环。while 循环语句的格式如下：

while( 条件表达式 ){

循环体

}

只要条件表达式的值为真，循环体就被执行。只有当条件表达式的值为假时，才结 束 while 循环。在 while 的循环体中，一般需要给出控制条件表达式值的语句，通过使条件表达式的值为假来结束循环。while 循环执行过程的流程图如图 3-5 所示。

᲍Т㶔䓪ᐻ

false

true

ᓖ⣛ѿ

图 3-5 while 循环执行过程的流程图

【例 3-7】使用 while 循环在控制台输出 1~10。

//CH03\_07.java

public class CH03\_07 {

public static void main(String[] args){ int i=1;

**Java** 程序设计

while(i<=10){

System.out.print(i+" ");

i++; // 用于控制条件表达式的值

}

}

}

【注意】 while(i<=10) 后面一定不要加分号，如果加了分号，则变成“while (i< =10) ; ”，这条语句等价于：

while(i<=10)

; // 空语句

空语句即为循环体。因为循环体中并没有使 i 值发生变化的语句，而 i 的初值是 1， 所以循环条件一直为真，导致了死循环。

* + 1. do-while 循环

在程序设计时，需要执行一次循环体后，再判断循环条件，则需要使用 do-while 循环。do-while 循环语句的格式如下：

do{

循环体

}while( 条件表达式 ); // 注意此处的分号必须要有

do-while 循环总是先执行循环体，然后计算条件表达式，如果条件表达式的值为真， 则继续循环；否则，循环结束。do-while 循环的循环体至少被执行一次。do-while 循环执行过程的流程图如图 3-6 所示。

 false

ᓖ⣛ѿ

᲍Т㶔䓪ᐻ

true

图 3-6 do-while 循环执行过程的流程图

【例 3-8】使用 do-while 循环在控制台输出 1~10。

//CH03\_08java

public class CH03\_08 {

public static void main(String[] args){ int i=1;

do{

System.out.print(i+" "); i++;

第 3 章 程序控制结构

}while(i<=10);

}

}

* + 1. 嵌套循环

嵌套循环指一个循环结构的循环体内包含了另一个循环结构，又称为多重循环。

while、do-while、for 这三种循环语句均可以互相嵌套。for 循环语句的嵌套经常被使用， 尤其是在处理二维数组时。for 循环语句的嵌套格式如下：

for( 表达式 1; 表达式 2; 表达式 3){ // 外层循环

…

for( 表达式 1; 表达式 2; 表达式 3){ // 内层循环

…

}

}

嵌套循环的特点是外层循环每执行一次，内层循环就会完整地执行一遍。

【例 3-9】编写程序打印用 \* 组成的直角三角形。

//CH03\_09.java

public class CH03\_09 {

public static void main(String[] args){ for(int i=1;i<=5;i++){

for(int j=1;j<=i;j++){ System.out.print("\*");

}

System.out.println();

}

}

}

程序执行结果如下：



* + 1. 跳转语句 break 和 continue

break 和 continue 语句是和循环紧密相关的两条语句。break 的中文含义是中断， continue 的中文含义是继续。通过这两条语句可以控制循环的执行流程。

1. break 语句

break 语句可用于 switch 结构或 for、while、do-while 循环结构，程序执行到 break 语

**Java** 程序设计

句时，将强制退出 switch 结构或循环结构。break 语句是用 break 加分号构成的语句。

【例 3-10】编写程序输出 2 〜 20 间的素数。

素数是一个大于 1 的自然数，除了 1 和它本身外，它不能被其他自然数整除。判断整数 i 是否为素数的算法是：如果在 2 〜 i-1 中存在某个数 j，使 i 能被 j 整除，则 i 不是素数；否则 i 是素数。

//CH03\_10.java

public class CH03\_10 {

public static void main(String[] args){ int i,j;

for(i=2;i<=19;i++){ for(j=2;j<=i-1;j++){

if(i%j==0){

break; // 如果 i 被 j 整除了则退出循环 ,i 不是素数

}

}

// 退出内层循环有两种情况，break 退出和循环条件 j<=i-1 为假退出

if(j==i){ //j 等于 i, 说明 i 值一直未被 j 整除 ,i 是素数

System.out.print(i+" ");

}

}

}

}

程序执行结果如下：



1. continue 语句

continue 语句可用于 for、while、do-while 循环结构。程序执行到 continue 语句，跳过循环体中当前循环还未执行的其余语句，回到循环条件处，判断是否执行下一次循环。

【例 3-11】编写程序输出 1~20 内不是 3 或 3 的倍数的数据。

//CH03\_11.java

public class CH03\_11 {

public static void main(String[] args){ for(int i=1;i<=20;i++){

if(i%3==0){

continue;

}

System.out.print(i+" ");

}

}

}

第 3 章 程序控制结构

程序执行结果如下：



# 小结

本章首先通过顺序结构介绍了程序中包含的 5 类语句，其次详细地介绍了解决选择分支问题的单分支 if 结构、双分支 if-else 结构、多分支 if-else 结构和 switch 结构， 最后介绍了解决重复问题的 for、while、do-while 三种循环结构及与循环有关的 break 和continue 语句。通过本章的学习，读者应掌握各种分支结构和循环结构的语法，并能灵活使用各种结构编写程序。

习题三

一．选择题

1. 下列选项中，（ ）不属于 Java 语言流程控制语句。

A. 分支语句 B. 赋值语句 C. 循环语句 D. 跳转语句

1. 有语句“int a=1; ”，则下列选项中（ ）是合法的条件语句。

A. if(a){} B. if(a=2){} C. if(a<<3){} D. if(true){}

1. 假设 a 和 b 为 long 类型变量，x 和 y 为 float 类型变量，ch 为 char 类型变量，且均已被赋初值，下列语句中正确的是（ ）。

A. switch(x+y){} B. switch ch{} C. switch(ch+1){} D. switch(a+b){}

1. 下面程序段中，while 循环的循环次数是（ ）。

int k=5; while(k>1){

k--;

}

A. 5 B. 0 C. 4 D. 3

1. 下面程序段中，do-while 循环的循环次数是（ ）。

int i=1; do{

System.out.print(i);

}while(i<5);

A. 4 B. 5 C. 0 D. 死循环

1. 关于下面程序段，说法正确的是（ ）。

byte b=1; while(++b>0)

;

System.out.println("Loop?");

**Java** 程序设计

A．进入死循环，什么也不输出 B．输出一次“Loop?” C．输出多次“Loop?” D．程序中含有编译错误

1. 下面程序段输出的结果是（ ）。

int a=0; while(a<5) {

switch (a) {

case 0:

case 3: a = a + 2; case 1:

case 2: a = a + 3; default: a = a + 5;

}

}

System.out.print(a);

A. 0 B. 5 C. 10 D. 其他

1. 下面程序段输出的结果是（ ）。

int n=6;

for(int i=1;i<=10;i++){ if((i+n)>10){

break;

}

System.out.print(i+" ");

}

A. 1 2 3 4 5 6 B. 7 8 9 10 C. 1 2 3 4 D. 5 6 7 8

1. 下面程序段，while 循环的循环次数是（ ）。

int i=0; while(i<10){

if(i<1){

continue;

}

if(i==5){

break;

} i++;

}

A. 1 B. 10 C. 6 D. 死循环

1. 下面程序段输出的结果是（ ）。

int m=37,n=13; while(m!=n){

while(m>n){

第 3 章 程序控制结构

m-=n;

}

while(n>m){

n-=m;

}

}

System.out.print(m);

A. 13 B. 11 C. 1 D. 2

二．判断题

1. break 语句可以用在循环结构和 switch 结构中。
2. continue 语句用在循环结构中表示继续执行下一次循环。
3. while 循环至少执行一遍。
4. 嵌套循环的特点是外层循环执行 1 遍，内层循环执行 1 遍。
5. 嵌套循环的次数为外层循环的次数加上内层循环的执行次数。

三．编程题

1. 编写程序，输出 1000 以内所有的“水仙花数”。水仙花数是一个 3 位数，其各位数字的立方和等于该数本身。例如，153 是一个水仙花数，因为 153=13+53+33。
2. 编写程序，输出 2 〜 100 内的所有完数。完数指一个整数等于该数所有因子之和。

例如，6 的因子是 1、2、3，而 6=1+2+3，所以 6 是完数。

1. 编写程序，计算 2020 〜 2100 年有多少个闰年，并输出相应年份。闰年是能被 400

整除的年份，或者不能被 100 整除但能被 4 整除的年份。

1. 编写程序，打印如下图形。



5. 编写程序，求满足 1!+2!+3!+…+n! ≤ 2566 的最大整数 n。四．Java 程序员面试题

简述在 Java 中如何跳出循环。