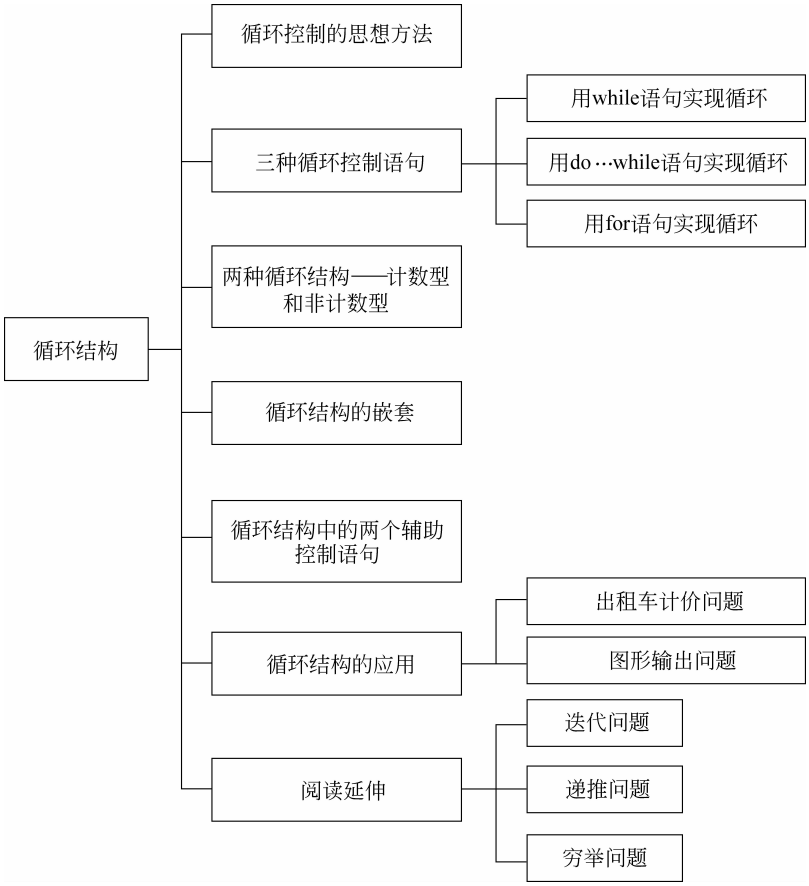
第5章 循环结构

本章知识结构图



本章学习导读

循环结构又称为重复结构，用来处理需要重复处理的问题，它是程序中一种很重要的结构。循环结构是由循环控制条件来控制循环体是否执行。

本章主要介绍三种循环语句的语法格式及用法。包括 while 语句、do…while 语句和

for 语句的使用；循环语句嵌套的运用；以及转移控制语句：break、continue。

本章以应用为中心，以算法为基础，确立循环的算法设计思想，并将此算法设计思想用流程图表示出来。在阅读延伸一节，增加了三类重要的用循环结构解决的问题——迭代、递推和穷举。

5.1 循环控制的思想方法

前面两章介绍了程序中常用到的顺序结构和选择结构，但是仅有这两种结构是不够的，还要用到循环结构。

1. 现实世界中的循环问题

生活中到处都是循环。例如，打印 50 份试卷，10000 米赛跑，旋转的车轮，包括每天的吃饭、上课、睡觉……自然界、社会上以及人生中的各种现象，似乎都在周而复始，不停地循环。

那么循环的特点是什么呢？对于循环，英文原版教材是这样描述的：repetition，重复。《辞海》上说循环是指事物周而复始的运动或变化。由这两个定义可以看出，循环重在强调重复。

现在我们可以提出一个问题：在计算机程序设计的世界里是否也有类似的这种相同操作重复出现的问题呢？我们又当如何提高程序设计的工作效率呢？在程序所处理的问题中也常常遇到需要重复处理的问题。例如：

【问题 5-1】 全班有 50 个学生，每个学生期末参加三门课考试，计算学生每门课的平均成绩。

【问题 5-2】 全班 50 个学生，计算每个学生三门课的平均分。

【问题 5-3】 老师检查 30 个学生的各科成绩是否及格。

【问题 5-4】 把 100 个职工的工资和打印出来。

循环就是不断重复地执行同一段程序。也即在程序设计中，从某处开始有规律地反复执行某一程序块的现象。简单说循环就是重复。

2. 循环的思想方法

人们最怕机械重复，因为重复枯燥乏味。即使用到这种思想方法也会因为手工计算过于繁琐而不愿用或不能用。而计算机则擅长重复。这种重复体现到程序中就是循环。使用循环达到便捷、效率高的目的。循环结构可以减少源程序重复书写的工作量，用来描述重复执行某段算法的问题，这是程序设计中最能发挥计算机特长的程序结构。

下面分析一下问题 5-1 的循环问题。

我们可以先输入学生 1 的三门课成绩，并计算平均值后输出：

```
scanf("%f,%f,%f", &n1, &n2, &n3);
aver=(n1+n2+n3)/3;
```

```
printf("aver=%7.2f", aver);
```

然后，输入学生 2 的三门课成绩，并计算平均值后输出：

```
scanf("%f,%f,%f", &n1, &n2, &n3);
aver=(n1+n2+n3)/3;
printf("aver=%7.2f", aver);
…
```

要对 50 个学生进行相同操作，重复 50 次。显然，这种方法是不可取的，需要引入循环的控制结构。循环结构就是设计一种能让计算机周而复始地重复地执行某些相同代码的结构。换句话说就是：相同语句只编写一次代码、并让计算机多次重复执行。这就是我们下面要介绍的三种循环控制语句。

C 语言中，有一组相关的控制语句，用以实现循环结构。

循环控制语句：for、while、do…while。

转移控制语句：break、continue、goto。

5.2 三种循环控制语句

C 语言中提供 4 种循环，即 goto 循环、while 循环、do…while 循环和 for 循环。4 种循环可以用来处理同一问题，一般情况下它们可以互相替换，在结构化程序设计方法中不提倡用 goto 循环，因为强制改变程序的顺序经常会给程序的运行带来不可预料的错误。

5.2.1 while 语句

循环结构和顺序结构、选择结构是结构化程序设计的三种基本结构，它们是各种复杂程序的基本构造单元。

1. while 语句的格式

```
while (运算式){
循环体语句;
}     //循环体为单个语句时可省略"{"和"}"
```

2. while 循环的执行过程

第一步，如果运算式成立，执行循环体语句；

重复执行第一步，直到运算式不成立。

即首先计算条件运算式的值，如果为真，则执行一次循环体；若为假，则结束循环，执行循环的后继语句。执行过程如图 5-1 所示。

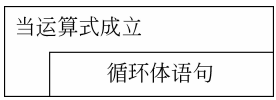


图 5-1 while 结构

3. while 循环的特点

先判断条件运算式，后执行循环体语句。

【例 5-1-1】 全班有 50 个学生，每个学生期末参加三门课考试，计算每个学生三门课的平均成绩。

(1) 问题分析

输入：每个学生的三门课成绩；三个变量 x1，x2，x3。

处理：对于每个学生计算三门课的平均成绩。

输出：每个学生的平均成绩：aver。

(2) 算法设计

算法对应的 N-S 见图 5-2。

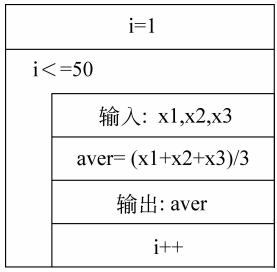


图 5-2 例 5-1-1 的 N-S 图

(3) 程序编写

```
#include<stdio.h>
int main()
{
  float x1, x2, x3, aver;
  int i;
  i=1;
  while (i<=50){
      scanf("%f%f%f", &x1 , &x2 , &x3);       //数据输入
      aver=(x1+x2+x3)/3;                      //数据处理
      printf("aver=%f\n", aver);              //数据输出
      i++;
  }  //while 循环体结束
  return 0;
}
```

(4) 测试运行

为简化，我们给出 5 个学生的运行结果如下：

```
67 89 56
aver=70.666667
78 75 80
aver=77.666667
90 86 65
aver=80.333333
89 78 76
aver=81.000000
55 89 98
aver=80.666667
```

本例 while 是一个循环结构，其内部包括数据输入、数据处理和数据输出三个部分。其算法思想类似于手工计算。这是重复 50 次的问题，应该用循环来解决。

我们在手工计算的过程中，需要一个计数的过程，是用大脑记忆着我们数到哪个学生了，在编程中，我们用变量 i 来记忆。我们称其为计数器变量。简称为计数器。其中第一个数为 i=1，下一个数为 i=i+1。

注意：循环体语句超过一条语句必须用大括号括起来，以复合语句形式出现。

while()的流程是条件→循环体→条件→循环体→…→条件假→循环后面继续。

利用循环的好处是节省了编程的书写时间，减少了程序源代码的存储空间，提高了程序的质量。这就是程序设计过程中循环的本质。循环结构的三要素如下：循环变量赋初值、循环条件和循环变量增量。

(1) 循环变量赋初值

在上述程序中，被用作循环计数器 i 的数值变量称为循环变量。循环变量有初值。

(2) 循环条件

循环条件和循环变量相关，是关于循环变量的关系或逻辑运算式。如上例中的 i<=50。

(3) 循环变量增量

循环变量增量即循环变量的步长，可以是正数或负数，如上例中的 i++。

除此之外，在循环体中，包括要重复的部分。

5.2.2 do…while 语句

1. do…while 语句的一般形式

do…while 语句的一般形式如下：

```
do{
循环体语句
}
while(运算式);
```

2. do…while 循环的执行过程

do…while 循环是先执行循环体语句，后判断条件运算式是否为真。如果为真，则重复执行下一次循环体；若为假，则结束循环，执行循环的后继语句。执行过程如图 5-3 所示。

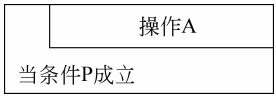


图 5-3 do…while 循环结构

3. do…while 循环的特点

这个循环与 while 循环的不同在于：它先执行循环中的语句，然后再判断条件是否为真，如果为真则继续循环；如果为假，则终止循环。因此，do…while 循环至少要执行一次循环语句。同样当有许多语句参加循环时，要用“{”和“}”把它们括起来。

【例 5-1-2】 用 do…while 循环实现例 5-1-1。

算法对应的 N-S 图如图 5-4 所示。

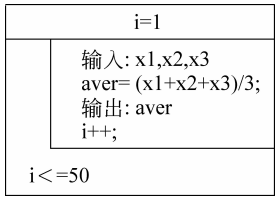


图 5-4 例 5-1-2 的 N-S 图

```
#include<stdio.h>
int main()
{
  float x1, x2, x3, aver;
  int i;
  i=1;
  do {
      scanf("%f%f%f", &x1 , &x2, &x3);      //数据输入
      aver= (x1+x2+x3)/3;                   //数据处理
      printf("aver=%f\n", aver);            //数据输出
      i++;
  }while (i<=50);                           //do…while 循环体结束
  return 0;
}
```

本例 do…while 是一个循环结构,其内部包括数据输入、数据处理和数据输出三个部分。

do…while 循环的流程是循环体→条件→循环体→条件→循环体→…→条件假→循环后面继续。

5.2.3 for 语句

1. for 循环的语句格式

```
for (运算式 1; 运算式 2; 运算式 3){
循环体语句;
}   //循环体为单个语句时可省略"{"和"}"
```

2. for 语句的执行过程

第一步,首先执行运算式 1,运算式 1 只执行一次。

第二步,执行运算式 2,如果运算式 2 成立,就执行循环体语句,然后执行运算式 3。

重复执行第二步,直到运算式 2 不成立为止。执行过程如图 5-5 所示。

【例 5-1-3】 用 for 循环实现例 5-1-1。

算法对应的 N-S 图如图 5-6 所示。

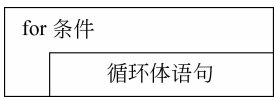


图 5-5 for 循环结构

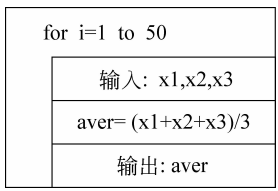


图 5-6 例 5-1-3 的 N-S 图

下面用 for 循环实现例 5-1-1。

```
#include<stdio.h>
int main()
{
  float x1, x2, x3, aver;
  int i;
  for (i=1; i<=50; i++){
      scanf("%f %f %f", &x1, &x2, &x3);     //数据输入
      aver=(x1+x2+x3)/3;                    //数据处理
      printf("aver=%f\n", aver);            //数据输出
  } //for 循环体结束
  return 0;
}
```

本例 for 是一个循环结构，其内部包括数据输入、数据处理和数据输出三个部分。

从该程序可以看出，for 循环的流程是条件→循环体→增值→条件→循环体→增值→…→条件假→循环后面继续。从上面三个程序看出，使用 for，while 和 do…while 求解同样的问题，基本思路类似。

【例 5-2】 计算 1＋2 ＋…＋100 的值。

(1) 问题分析

输入：无

处理：从 1 至 100 重复 100 次加法运算，结果存入 sum。

输出：和 sum。

(2) 算法设计

算法对应的 N-S 见图 5-7。

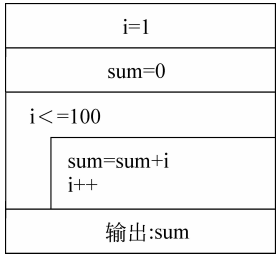


图 5-7　例 5-2 的 N-S 图

(3) 程序编写

```
#include<stdio.h>
int main()
{
  int i, sum;
  sum=0;                        //数据处理部件：变量 sum 的初值为 0
  i=1;                          //变量 i 的初值为 1
  while (i<=100){
      sum=sum+i;
      i++;
  }                             //while 循环体结束
  printf("sum=%d\n", sum);      //数据输出：sum
  return 0;
}
```

(4) 测试运行

```
sum=5050
```

本例 while 是一个循环结构，循环结构内部是数据处理：累加和计数。

其算法思想类似于计算器的累加过程。

首先，在日常生活中，用计算器进行 100 个数的加法，应该怎么做呢？读第一个数 1，按一下计算器上的“＋”，保存；再读下一个数 2，再按一下计算器上的“＋”，再保存……这个过程一直继续，直到加完第 100 个数为止。上面我们读的每个数，我们是用大脑记着这个数；在程序中，可以用一个变量 i 表示，i 称为计数器。

其次，上述计算器计算过程中的每次结果是存到计算器中了，我们在程序中可以用一个内存变量 s 表示。我们把用于存放累加结果的变量 s 称为累加器。

第三，这里重复的是什么？是“＋”，从而我们可以把日常生活中的累加写成公式：

```
s=s+i
```

这个公式的含义是将当前读的数 i 与 s 进行累加，保存在 s 中，从 1 至 100 重复这个过程。要重复 100 次加法运算，可用循环实现。

想一想本程序用到的循环的三要素都是什么？

for 语句完全可以代替 while 语句，例如：

```
i=1;
while (i<=100){
    sum=sum+i;
    i++;
}
```

等价于

```
for (i=1;i<=100;i++)
      sum=sum+i;
```

5.3 两种循环结构——计数型和非计数型

1. 计数型循环

此类问题的循环程序基本部件分别为：

```
i=1;                          //循环变量初值
while (i<=n){                 //循环条件
    scanf...;                 //循环体语句
    aver=...;                 //循环体语句
    printf...;                //循环体语句
    i++;                      //循环变量增量
```

```
}                              //while 循环体结束
for (i=1; i<=n; i++){          //循环变量初值,条件,增量
    scanf...;                  //循环体语句
    aver=...;                  //循环体语句
    printf...;                 //循环体语句
}                              //for 循环体结束
```

2. 非计数类循环

【例 5-3】 从键盘输入任意多个年号,判断是否为闰年,直到输入的年号为 0 时停止。

(1) 问题分析

输入:年号 year。

处理:设 year 为被检测的年份,若 year 能被 4 整除,但不能被 100 整除,则 year 是闰年;若 year 能被 100 整除,又能被 400 整除,则 year 是闰年。

输出:是否是闰年。

(2) 算法设计

算法对应的 N-S 见图 5-8。

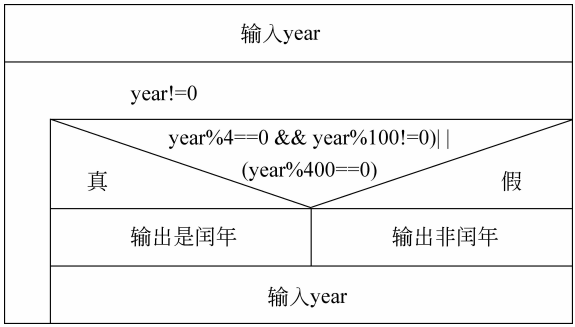


图 5-8 例 5-3 的 N-S 图

(3) 程序编写

```
#include<stdio.h>
int main()
{
  int year;
  scanf("%d", &year);                    //数据输入:第一个数据
  while (year !=0){
  if ((year%4==0 && year%100 !=0) || (year%400==0))
    printf("%d is ", year);
  else
    printf("%d is not ", year);          //数据计算:if()
  printf("a leap year.\n");              //数据输出
  scanf("%d", &year);                    //数据输入:下一个数据
```

```
}    //while 循环体结束
  return 0;
}
```

(4) 测试运行

```
2014
2014 is not a leap year.
2016
2016 is a leap year.
0
```

本例 while 是一个循环结构，循环结构包括数据输入、数据处理和数据输出，注意数据输入语句的位置，对比和计数型循环有什么不同。

5.4 循环结构的嵌套

循环结构的嵌套又称多重循环，即循环结构体中还包含有循环结构。

【例 5-4】 有 N 个歌手参加比赛，8 个评委为每个歌手打分。去掉一个最高分和一个最低分，求每个歌手的得分。

(1) 问题分析

输入：输入每一个歌手的 8 个评委的评分。

处理：去掉一个最高分和一个最低分，计算每个歌手的平均分。

输出：每个歌手的平均分。

(2) 算法设计

算法对应的 N-S 见图 5-9。

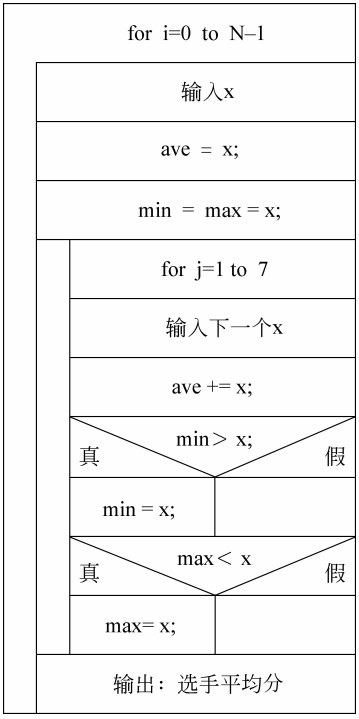


图 5-9 例 5-4 的 N-S 图

(3) 程序编写

```
#include<stdio.h>
#define  N  5
int main()
{
  int i,j,x,max,min;
  float ave=0.0;
  for(i=0;i<N;i++){
    printf("输入一个歌手的 8 个评委的评分：");
    scanf("%d", &x);
    ave+=x;
    min=max=x;
    for (j=1;j<=7; j++){
```