

第5章

循环结构

计算机最大的优势就是不厌其烦地重复工作,并且不会出错。循环结构的语法并不复杂,如何灵活地使用循环结构解决问题是我们要学习的关键。本章共给出 5 个案例,将分支语句、循环语句等重要且基本的知识点巧妙地贯穿到各个案例中,帮助同学们逐步建立起程序设计的思想和思维习惯。

5.1 例 5-1: 简易计算器

5.1.1 设计说明

1. 功能说明

- (1) 简易的计算器,实现加(+)、减(-)、乘($\times$)、除($\div$)和模除($\%$)的计算。
- (2) 将整个计算器的功能放在循环体中,就能实现反复计算,设置循环结束条件是输入“1 2 3”,通过 break 语句退出循环。
- (3) 使用 switch 语句分析用户输入的“opt”的值,根据用户的输入完成对应的运算,对于“/”运算,当用户输入的第二个运算数是“0”时,提示“分母不能为零”,并利用 continue 语句返回循环入口,进行下一次循环。
- (4) 输出运算结果。

2. 变量说明

char opt: 表示用户输入的运算符。
int number1: 表示用户输入的第一个运算数。
int number2: 表示用户输入的第二个运算数。
int result: 用来存储运算结果。

3. 流程分析

计算器的执行过程如下。

- (1) 定义变量,提示用户如果输入“1 2 3”,则程序结束。
- (2) 使用 while(1)语句直接进入循环。

① 循环体中按照“%d %c %d”的格式输入运算数和运算符,即 number1,opt 和 number2 的值,如果输入的是“1 2 3”时使用 break 语句退出循环;否则使用 switch 语句进行判断,根据输入的 opt 值的不同进入不同的 case 分支进行计算。

② 如果 opt 的值是“/”时,需要判断分母是否为零,如果分母不为零,直接进行除法运算;如果分母为零,给出错误提示,并使用 continue 语句结束本次循环,直接进入下一次循环。

5.1.2 程序源代码

```
#include <stdio.h>
int main()
{
    char opt;
    int number1;
    int number2;
    int result;
    printf("输入“1 2 3”可以退出计算器。\\n");
    while(1)
    {
        printf(">>>");
        /* opt 是字符型,采用 %c 格式接受赋值 */
        /* number1 和 number2 是整型,采用 %d 格式来赋值 */
        scanf("%d %c %d", &number1, &opt, &number2);
        if((number1 == 1)&&(opt == '2')&&(number2 == 3))
        {
            break;
        }
        switch(opt)
        {
            case '+': result = number1 + number2;break;
            case '-': result = number1 - number2;break;
            case '*': result = number1 * number2;break;
            case '/':
            {
                if (number2 != 0)
                    result = number1 / number2;
                else
                {
                    printf("分母不能是零!\\n");
                    continue;
                }
                break;
            }

            case '%': result = number1 % number2;break;
            default: printf("非法运算!\\n");break;
        }
        printf("%d %c %d = %d\\n", number1, opt, number2, result);
    }
    printf("计算器已退出!\\n");
    return 0;
}
```

5.1.3 程序运行情况

运行程序后,按提示输入两个运算数和一个运算符进行计算,用户输入“1 2 3”的时候结束程序。运行情况如图 5-1 所示。

```
输入“1 2 3”可以退出计算器。  
>>>6*28  
6 * 28 = 168  
>>>8/0  
分母不能是零!  
>>>1 2 3  
计算器已退出!  
Press any key to continue.
```

图 5-1 简易计算器的运行情况

5.2 例 5-2: 计算平均分

5.2.1 设计说明

1. 功能说明

用户输入 n 名学生的成绩,计算学生的平均分并统计各个分数段的人数。

2. 变量说明

double sum: 用来存储分数总和。

double avg: 表示平均分。

int a[100]: 数组 a 用来存储输入的学生的成绩,给定初值为 100。

int num[5]: 数组 num 用来存储各个分数段的人数。

3. 流程分析

(1) 程序中第一个 for 循环用来计算学生的总分,先输入每名学生的成绩存储在数组 a 中,如果成绩不符合要求,即小于 0 或者大于 100,则使用 break 语句结束循环;如果输入的成绩是 0~100 的数,则计算累加和,存储在变量 sum 中。

(2) 计算平均分 avg, $avg = sum/n$, n 是学生数。

(3) 程序中第二个 for 循环用来统计各个分数段的人数,存储在数组 num 中,利用多分支语句来实现。其中:

将 90~100 分的人数存储在 num[0]中;

将 80~89 分的人数存储在 num[1]中;

将 70~79 分的人数存储在 num[2]中;

将 60~69 分的人数存储在 num[3]中;

将小于 60 分的人数存储在 num[4]中。

5.2.2 程序源代码

```
#include "stdio.h"
#include "stdlib.h"
int main()
{
    /* 变量定义 */
    int i,n;
    double sum = 0, avg = 0;
    int num[5] = {0};
    int a[100];
    /* 输入学生人数 */
    printf("请输入学生人数\n");
    scanf("%d",&n);
    /* for 循环中输入 n 个学生的成绩,并计算总分,存储在变量 sum 中 */
    for(i = 0;i < n;i++)
    {
        printf("请输入第 %d 个学生的成绩\n",i+1);
        scanf("%d",&a[i]);
        if(a[i]<0||a[i]>100)
            break;
        sum = sum + a[i];
    }
    /* 计算平均分 */
    avg = sum/n;
    printf("这 %d 个学生的平均成绩是: %.2f",n,avg);
    /* ---- 统计各个分数段的人数 ---- */
    for(i = 0; i < n; i++)
        if(a[i]>= 90) num[0] + = 1;
        else if(a[i]>= 80) num[1] + = 1;
        else if(a[i]>= 70) num[2] + = 1;
        else if(a[i]>= 60) num[3] + = 1;
        else num[4] + = 1;
    printf("\n===== \n");
    printf("\t成绩段\t\t人数\t\n");
    printf("\n===== \n");
    printf("\t90 分以上\t%d", num[0]);
    printf("\n\t80-90 分\t\t%d", num[1]);
    printf("\n\t70-80 分\t\t%d", num[2]);
    printf("\n\t60-70 分\t\t%d", num[3]);
    printf("\n\t不及格\t\t\t%d", num[4]);
    printf("\n===== \n");
}
```

5.2.3 程序运行情况

运行程序后,提示用户输入 5 名学生的成绩,程序会根据用户的输入,统计出每个分数段的人数和 5 名学生的平均分,运行情况如图 5-2 所示。

```
请输入学生人数
5
请输入第1个学生的成绩
67
请输入第2个学生的成绩
89
请输入第3个学生的成绩
87
请输入第4个学生的成绩
77
请输入第5个学生的成绩
49
这5个学生的平均成绩是: 73.80
=====
      成绩段      人数
=====
      90分以上      0
      80-90分      2
      70-80分      1
      60-70分      1
      不及格        1
=====
Process exited with return value 0
Press any key to continue . . .
```

图 5-2 计算学生成绩运行界面

5.3 例 5-3: 摄氏温度与华氏温度转换

5.3.1 设计说明

1. 功能说明

华氏温标与摄氏温标是两大国际主流的计量温度的标准。本程序实现的是华氏温度和摄氏温度之间的转换表。两者之间的换算公式为

$$c = (f - 32) \times 5 / 9.0$$

c 表示摄氏温度, f 表示华氏温度。

2. 预处理与变量说明

1) 预处理

#include "stdio.h": stdio.h 是标准的输入/输出头文件, 程序中使用的 printf() 和 scanf() 函数, 要对其进行预处理, 即文件包含。

#include "math.h": math.h 是数学类头文件, 本程序中用到了 abs 函数, 所以要对其进行预处理操作。abs 函数的功能是求整数的绝对值。

2) 变量说明

double c: 用来表示摄氏温度。

double f: 用来表示华氏温度。

int begin: 表示用户输入的温度起始值。

int end: 表示用户输入的温度终止值。

int step: 步长。

3. 流程说明

(1) 提示用户输入华氏温度的起始值、终止值和步长值,起始值、终止值分别存储在变量 begin 和 end 中,步长值存储在变量 step 中。

(2) 如果 $\text{begin} > \text{end}$, 则步长取其相反数, 即 $\text{step} = -\text{step}$ 。

(3) 进入循环, 循环体中按格式“\t%.2f\t\t%.2f\n”逐个打印华氏温度和摄氏温度, “\t”和“\n”是转义字符, 控制格式“.2f”表示输出小数点后保留两位的实数(double 型), 当 $(f - \text{end}) < \text{step}$ 时, 使用 break 语句结束循环。abs() 是求绝对值函数。

5.3.2 程序源代码

```
#include <stdio.h>
#include <math.h>
int main()
{
    double c;                /* 摄氏温度 */
    double f;                /* 华氏温度 */
    int begin;                /* 温度起始值 */
    int end;                  /* 温度终止值 */
    int step;                 /* 步长 */
    printf("欢迎使用温度表小程序\n");
    printf("请输入【华氏温度】的起始值和结束值:\n");
    scanf("%d %d", &begin, &end);
    printf("请输入步长(1~10 的整数):\n");
    scanf("%d", &step);
    if(begin < end)
        step = step;
    else
        step = -step;
    f = begin;
    printf("\t 华氏温度 -----> 摄氏温度\n");
    while(1)
    {
        c = (f - 32) * 5 / 9.0;
        printf("\t %.2f\t\t %.2f\n", f, c);
        if(abs(f - end) < abs(step))
            break;
        f += step;
    }
    return 0;
}
```

5.3.3 程序运行情况

运行程序后,提示用户输入华氏温度的起始值、结束值以及步长,程序根据用户输入的数据进行计算,打印出对应的华氏温度与摄氏温度的换算表,如图 5-3 所示。

```
欢迎使用温度表小程序
请输入【华氏温度】的起始值和结束值:
20
150
请输入步长(1~10的整数):
9
    华氏温度  ——> 摄氏温度
    20.00      -6.67
    29.00      -1.67
    38.00       3.33
    47.00       8.33
    56.00      13.33
    65.00      18.33
    74.00      23.33
    83.00      28.33
    92.00      33.33
   101.00      38.33
   110.00      43.33
   119.00      48.33
   128.00      53.33
   137.00      58.33
   146.00      63.33
Press any key to continue
```

图 5-3 摄氏温度与华氏温度转换程序运行图

5.4 例 5-4: 猜数字游戏 1

5.4.1 设计说明

1. 功能说明

猜数字小游戏,首先由系统随机产生 1~100 的随机数,然后提示用户输入一个数字,将用户输入的数字与系统随机产生的数进行比较,并给出“偏大了!”或“偏小了!”的提示,如果两数相等,表示猜对了,此时打印出游戏进行的次数。游戏共进行 10 次,如果 10 次都没有猜对,给出猜错的提示信息。

2. 预处理与常量、变量说明

1) 预处理

#include "stdio.h": stdio.h 是标准的输入/输出头文件,程序中使用 printf() 和 scanf() 函数,必须对其进行预处理,即文件包含。

#include "time.h": time.h 是 C/C++ 中的日期和时间头文件。time(NULL) 函数是 C 标准库函数,其功能是获取当前的系统时间,返回的结果是当前时间的秒数, NULL 是系统常量,表示 0,这里指空指针。

#include "stdlib.h": stdlib.h 是标准库头文件(即 standard library),stdlib 头文件里包含 C、C++ 语言最常用的系统函数,如 srand() 函数和 time() 函数。

srand() 函数是在调用 rand() 函数之前使用的,rand() 函数的功能是一个产生随机数,而 srand() 函数的功能是设置随机数的种子。这两个函数一起使用才能完成产生随机数的功能。

C 语言中,srand(time(NULL)) 的功能是使用当前时间作为随机数发生器的初始值。

2) 常量说明

#define MAX 10: 这里 MAX 设置允许用户猜数字的次数。

3) 变量说明

int number: 用来存储随机产生的数值。

int answer: 存储用户输入的数值。

int i: 循环变量。

3. 程序流程分析

(1) 使用下面两条语句随机生成 1~100 的随机数,存储在变量 number 中。

```
srand(time(NULL));  
number = rand() % 100 + 1;
```

(2) 进入循环,循环次数从 1 到 MAX,这里设置 MAX 值为 10,循环体中提示用户输入一个值,存储在变量 answer 中,使用多分支语句将 answer 值与 number 值进行比较,如果 (answer > number),提示“偏大了”,否则提示“偏小了”,如果两者相等则提示“猜对了”,并给出当前循环变量 i 的值,即猜的次数;如果 i > MAX,表示用户在 10 次内没有猜对,给出提示,程序运行结束。

5.4.2 程序源代码

```
#include "stdio.h"  
#include "time.h"  
#include "stdlib.h"  
#define MAX 10  
int main()  
{  
    int number;  
    int answer;  
    int i;  
    srand(time(NULL));  
    /* rand() % 100 的功能是产生 0~99 的随机整数 */  
    number = rand() % 100 + 1;  
    printf("==== 欢迎使用猜数小游戏 ===== \n");  
    printf("==== 目标是 1~100 的整数 ===== \n");  
    for( i = 1; i <= MAX; i++)  
    {  
        printf("请输入一个整数:");  
        scanf("%d", &answer);  
        printf("你猜的整数是: %d, ", answer);  
        if( (answer < 1) || (answer > 100) )  
        {  
            printf("超出区间[1, 100]!\n");  
        }  
        else if(answer > number)  
        {  
            printf("偏大了!\n");  
        }  
        else if(answer < number){  
            printf("偏小了!\n");  
        }  
        else  
        {  

```



```
        printf( "猜对了!一共猜测 %d 次.\n", i );
        break;
    }
}
if( i > MAX )
{
    printf( "目标整数是 %d.\n", number);
    printf( "抱歉,只允许猜测 %d 次,再见!\n", MAX);
}
return 0;
}
```

5.4.3 程序运行情况

程序运行后,提示用户输入一个 1~100 的数,将用户输入的数与系统随机产生的数进行比较,给出偏大或偏小的提示,程序运行情况如图 5-4 所示。

```
=====欢迎使用猜数小游戏=====
=====目标是1~100的整数=====
请输入一个整数: 80
你猜的整数是: 80, 偏小了!
请输入一个整数: 90
你猜的整数是: 90, 偏大了!
请输入一个整数: 85
你猜的整数是: 85, 偏大了!
请输入一个整数: 83
你猜的整数是: 83, 偏小了!
请输入一个整数: 82
你猜的整数是: 82, 偏小了!
请输入一个整数: 81
你猜的整数是: 81, 偏小了!
请输入一个整数: 84
你猜的整数是: 84, 猜对了! 一共猜测7次。
Press any key to continue
```

图 5-4 猜数字游戏 1 的运行情况

5.5 例 5-5: 猜数字游戏 2

5.5.1 设计说明

1. 设计说明

猜数字小游戏,游戏规则如下: 计算机随机生成一个 1~100 的随机数 `number`, 由用户来猜。将用户输入的数字 `answer` 与计算机产生的随机数 `number` 进行比较, 如果 `answer > number`, 系统给出“大了”的提示; 如果 `answer < number`, 系统给出“小了”的提示; 当 `answer = number` 时, 系统提示“恭喜你! 猜对了”。

每次游戏可以猜 5 次。次数由变量 `no` 来统计, 当 `no` 值超过 5 次(即 `no` 大于或等于 5) 时, 游戏结束。

提示: 本例题与 5.4 节属于同类型, 区别在于系统随机数生成的时间点不一致, 5.4 节是在循环之前随机数就生成了, 而本例题是在循环体中生成随机数。同学们可以对比着进行练习。

2. 变量说明

int number=0: 用来保存系统生成的随机数。

int no=0: 保存猜的第几次。

int answer: 用来保存答案。

char input='y': 保存用户输入(是否继续)。

3. 流程分析

(1) 生成 1~100 的随机数,方法同 5.4 节。

(2) 直接进入循环,使用 no 来存储猜的次数,也是循环次数,循环体内是一个双分支结构。

① 当两者相等时,即 `answer == number`,提示猜对了,并提示是否进行下一次猜数字游戏,如果用户输入“y”或者“Y”,则重新生成随机数,并使用 `continue` 语句结束本次循环,进行下一次循环。

② 当 `(answer == number)` 值为假时:

- 利用双分支语句判断是“大了”还是“小了”。
- 如果 no 值大于 5,循环结束。
- 提示用户是否进行下一次游戏,如果用户输入“y”或者“Y”,则重新生成随机数,并使用 `continue` 语句结束本次循环,进行下一次循环。

5.5.2 程序源代码

```
#include "stdio.h"
#include "time.h"
#include "stdlib.h"
main()
{
    int number = 0;           /* 用来保存系统生成的随机数 */
    int no = 0;               /* 保存猜的第几次 */
    int answer = 0;          /* 用来保存答案 */
    char input = 'y';         /* 保存用户输入是否继续 */
    srand((unsigned)time(NULL)); /* 为随机数设置种子 */
    /* rand() % 100 功能是生成 0~99 的随机数 */
    number = rand() % 100 + 1;
    printf("***** 猜 1~100 数字小游戏 ***** \n");
    printf("***** 您有 5 次机会 ***** \n");
    while (1)                 /* 游戏主循环 */
    {
        no++;                 /* no 用来存储猜的次数,进入游戏就增 1 */

        printf("第 %d 次猜, 请输入 1~100 的整数:", no);
        scanf("%d", &answer);

        if(answer == number)
        {
```

```
printf("恭喜你! 用了 %d 次猜对了!\n", no);
printf("再来一次? (y/n):");
getchar();
input = getchar();
if (input == 'y' || input == 'Y')
{
    no = 0;
    srand((unsigned)time(NULL));    /* 为随机数设置种子 */
    number = rand() % 100 + 1;
    continue;
}
else
{
    printf("Bye bye...\n");
    break;
}
}
else
{
    if( answer > number )
    {
        printf("大了...\n");
    }
    else
    {
        printf("小了...\n");
    }

    if (no > 5)
    {
        printf("猜的次数超过 5 次, 游戏结束!");
        break;
    }
    printf("继续猜? (y/n):");
    getchar();
    input = getchar();
    if (input == 'y' || input == 'Y')
    {
        continue;
    }
    else
    {
        printf("Bye bye...\n");
        break;
    }
}

}
return 0;
}
```

5.5.3 程序运行情况

程序运行后,提示用户输入 1~100 的整数,共有 5 次机会。如果用户输入的数大于系统本次运行时产生的随机数,给出“大了”的提示;如果用户输入的数小于随机数,给出“小了”的提示;如果相等,给出“猜对”的提示,同时打印出猜对的次数,运行情况如图 5-5 所示。

```
***** 猜 1~100 数字小游戏 *****  
***** 您有5次机会*****  
第1次猜, 请输入1~100的整数:70  
大了...  
继续猜? (y/n):y  
第2次猜, 请输入1~100的整数:50  
小了...  
继续猜? (y/n):y  
第3次猜, 请输入1~100的整数:60  
小了...  
继续猜? (y/n):y  
第4次猜, 请输入1~100的整数:65  
小了...  
继续猜? (y/n):y  
第5次猜, 请输入1~100的整数:68  
恭喜你! 用了5次猜对了!  
再来一次? (y/n):■
```

图 5-5 猜数字游戏 2 的运行情况