

第3章 简单程序设计



1. 问题的提出

上一章介绍了常量、变量、运算符、表达式等，它们都是构成程序的基本成分。C语言没有专门的输入输出语句，是通过函数来实现的，所以函数也是C语言程序的基本构成单位。

前两章中涉及的简单C语言程序，都是顺序结构的程序。顺序结构程序的执行流程是按语句的书写顺序自上而下依次执行，是三大结构化程序之一。那么顺序结构程序的组成包括哪些，又如何来实现呢？

顺序结构程序只能完成一些简单的运算和操作。但是，人们经常会遇到复杂的问题，例如，要求计算机根据给定的条件，判断条件是否成立，并根据不同的判断结果让计算机执行不同的操作，这就属于选择结构。本章讲的if语句就是一种选择结构。

到本章结束，程序设计的入门篇就结束了，即将进入到基础篇，希望同学们在学习完本章之后，能够完成下面程序的设计制作，检验是否具备了设计简单软件的能力。

制作数学工具软件，如下图所示。



将上述案例进行任务分解，并依次解决，当所有任务都解决了，上述案例的程序也就完成了。

(1) 显示菜单项。

(2) 输入菜单选择项，并显示输出所选菜单项，如果输入值为5，则显示“退出软件.”。

(3) 如果输入值为1-4，则要输入半径的值，并根据输入的值计算并输出相应结果。

此部分程序的答案将在本章后面的综合应用实例中给出。

2. 简单实例

1. 从键盘上输入某学生3门课程的成绩，计算平均成绩。

执行顺序 ↓

```

#include <stdio.h>
void main()
{ float score1,score2,score3,sum,aver;
  scanf("%f%f%f",&score1,&score2,&score3);
  sum= score1+score2+score3;
  aver=sum/3;
  printf("该学生 3 门课程的平均值为: %.2fn",aver);
}

```

程序运行的结果为:

83.5 73 90.5

该学生 3 门课程的平均值为: 83.00

其中第 4 行和第 7 行是函数调用语句, scanf 函数和 printf 函数分别为数据输入函数和输出函数, “%f” 为格式控制字符, “%.2f” 表示以浮点型数输出, 保留两位小数。第 5 行和第 6 行为赋值语句。包含在 “{ }” 之间的为复合语句, 是主函数的函数体, 完成特定的功能。该程序的执行过程是按照书写的顺序自上而下一部一步一步的执行, 直到程序的结束。这种结构属于顺序结构。

2. 从键盘上输入两整数, 编程实现输出其中较大的数。

```

#include <stdio.h>
void main()
{ int a,b,max;
  scanf("%d%d",&a,&b);
  if(a>b)
    max=a;
  else
    max=b; /* 这里的 if-else 语句也可使用条件表达式 a>b?a:b; */
  printf("max=%d",max);
}

```

程序运行的结果为:

69 92

max=92

这种结构属于选择结构。

3. 本章任务

本章学习的主要任务:

- (1) 表达式语句。
- (2) 函数调用语句, 数据的输入和数据的输出。
- (3) 复合语句。
- (4) 空语句。
- (5) 简单控制语句。
- (6) 关系和逻辑表达式。
- (7) 程序流程图。
- (8) if 语句。

3.1 C 语句概述

C 语言是一种很好的结构化程序设计语言，提供比其他高级语言更丰富的流程控制语句。C 语言程序的重要组成部分就是语句，可以通过一些丰富的语句完成一些基本的操作。C 语言的语句可以分为五大类：

1. 表达式语句

在表达式的后面加上“;”就构成了表达式语句。表达式语句主要有：赋值语句、自增自减运算符构成的表达式语句和逗号表达式语句。例如：

```
x=y+z;    赋值语句;  
i++;      自增语句, i 值加 1;  
x=1,y=2;  逗号表达式语句;
```

2. 函数调用语句

函数调用语句是由“函数名”、“参数”加上“分号”(;)组成，即在调用函数末尾加上一个分号就组成了函数调用语句。其一般形式为：

函数名(参数列表);

例如：

```
printf("C Program! "); 调用库函数，输出字符串：C Program!。
```

3. 复合语句

把多个语句用括号“{ }”括起来组合在一起，称之为复合语句，又称为语句块。

例如：

```
{ int x,y,z;  
  x=10;  
  y=20;  
  z=x+y;  
  printf("%d",z);  
}
```

介于“{”和“}”之间的多条语句就构成了一条复合语句。复合语句可以从整体上看作一条语句来完成某一特定的操作。复合语句内的各条语句都必须以分号(;)结尾，但在括号“}”外不能加分号。如果在“}”后加上分号，是一个单独的空语句，而不是复合语句。

4. 空语句

只有分号“;”组成的语句称为空语句。空语句是什么也不执行的语句。

5. 控制语句

控制语句用于控制程序的流程，以实现程序的各种结构方式。C 语言有九种控制语句。

- (1) if 语句;
- (2) switch 语句;
- (3) do while 语句;
- (4) while 语句;
- (5) for 语句;
- (6) break 语句;

- (7) goto 语句;
- (8) continue 语句;
- (9) return 语句。

在后续的章节中将陆续介绍以上控制语句。

3.2 数据输出与输入

把数据（如运算的结果）显示在屏幕上、或是打印在纸上或是存储到磁盘上的操作称为“输出”。从键盘、磁盘等设备上接收的数据装入内存中称为“输入”。C 语言本身不提供输入输出语句，输入输出操作是由函数来实现的。C 程序的输入输出是通过调用标准库函数中的 printf 函数和 scanf 函数来实现的。

3.2.1 printf 函数

printf 函数称为格式输出函数，是最常用和功能最强的函数。其格式为：

```
printf(<"格式控制字符串">,<参数 1>[,<参数 2>][,……])
```

其功能是按照“格式控制字符串”的格式，把参数一一输出。

printf 函数包括“格式控制字符串”和“参数列表”两部分，两者之间用逗号隔开。格式控制字符串在“()”中，必须用双引号括起来。输出参数可以是常量、变量、表达式或可以输出的函数。输出参数之间用逗号隔开。

其中格式控制字符串用于指定输出的格式。“格式控制字符串”由“格式说明符”和“普通字符串”两部分组成。

(1) “格式说明符”由“%”和“转换字符”组成，以“%”开头的字符串，在“%”后面跟有各种格式“转换字符”（特定含义的单个字符），以说明输出数据的类型、形式、长度、小数位数等。如：“%d”将参数按十进制整数形式输出；“%f”将参数按浮点型小数形式输出；“%c”将参数按单个字符形式输出等。

(2) “普通字符串”在输出时原样输出，起提示作用。例如，

```
printf("a=%d\n",a);
```

其中“%d”为格式说明符，“a=”为普通字符，“\n”为转义字符，代表换行。“a”是输出参数。假设 a=5；则输出结果为：

a=5

C 语言“格式控制字符串”主要有 9 种，如表 3-1 所示，一个输出参数对应一个格式说明符，有多少个输出参数，就一定有多少个格式说明符。不但数量上要一致，而且类型上也要一致。如果想输出一个“%”，则应在普通字符中，连续加入两个“%”，方可实现。

表 3-1 格式控制字符说明

格式说明符	含义
%d	以十进制整数的形式输出
%o	以八进制形式输出无符号整数
%x	以十六进制形式输出无符号整数

续表

格式说明符	含义
%u	以十进制形式输出无符号整数
%f	以小数形式输出单、双精度浮点数
%e	以指数形式输出单、双精度浮点数
%g	以%f或%e中较短的输出宽度输出单、双精度浮点数，不输出小数后面无意义的0
%c	输出单个字符
%s	输出字符串

在格式说明符中，在“%”和转换字符之间还可以插入以下几种附加格式说明符，如表 3-2 所示。

表 3-2 附加格式说明符

附加格式说明符	意义
-	输出结果左对齐；无“-”时，靠右对齐
0	数值型数据不使用的位置用0来填充
正整数 m（数据宽度）	数据的最小宽度，若宽度不够实际数据输出，则以实际宽度为准。若字符的宽度比数据宽度小，通常在其左边或者右边用空格来填充
正整数 n（精度）	若为浮点数，表示输出 n 位小数；若为字符串，表示截取字符的个数
长度修饰符 l（字母）	是否按长整数输出。可加在格式字符 d、o、x、u 前面
。（英文句号）	将数据宽度和后面表示精度的正整数分开。通常看到的是：m.n

根据以上格式说明，可以归纳如图 3-1 所示，其中-、0、m、n 和 l 都可以省略。

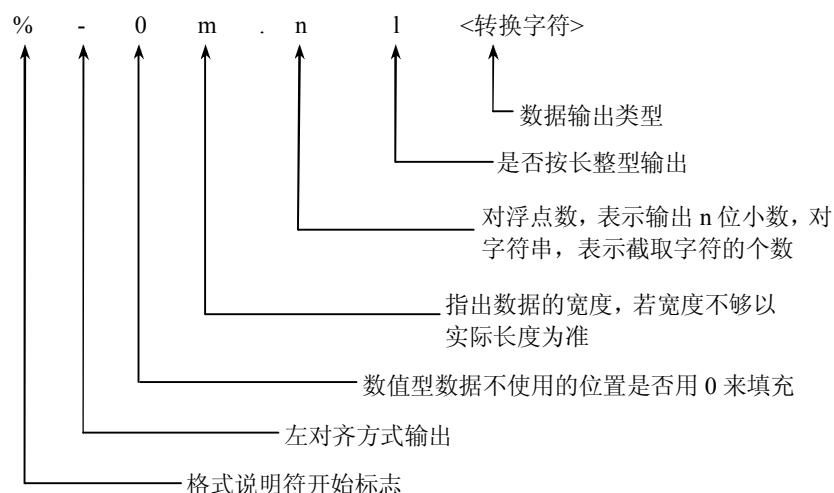


图 3-1 格式说明及含义

下面举例说明，见表 3-3。

表 3-3 printf 函数举例及其说明 (□表示空格)

输出语句	输出结果	说明
printf("%d",315);	315	以十进制数的形式输出
printf("%5d",315);	□□315	指定宽度为 5，左边补空格
printf("%-6d",315);	315□□□	指定宽度为 6，右边补空格
printf("%06d",315);	000315	指定宽度为 6，左边补 0
printf("%f", 123.456);	123.456000	以小数形式输出浮点数
printf("%12f", 123.456);	□□123.456000	指定宽度为 12，左边补空格
printf("%e",123.456);	1.23456e+02	以指数形式输出浮点数
printf("%12e",123.400);	□1.23400e+02	指定宽度为 12，左边补空格
printf("%g",123.400);	123.4	%f 和 %e 宽度较短的那种形式
printf("%8.3f",123.8);	□123.800	宽度为 8，精度为 3 (3 位小数)
printf("%-8.0f",123.8);	124□□□□□	宽度为 8，精度为 0，右补空格
printf("%.4f",123.45678);	123.4568	精度为 4
printf("%.5s","abcdef");	abcde	截取 5 个字符
printf("%3s","abcdefg");	abcdefg	超过实际宽度以实际宽度输出
printf("%3c",'a');	□□a	宽度为 3，左边补空格
printf("%.2f%%",1.0/4*100);	25.00%	输出 % 形式，小数为 2 位

注意：输入数据的实际精度并不取决于格式控制中的修饰字符“m.n”，而是取决于数据所占存储单元的大小。通常系统对 float 类型提供 6 位有效数字，若以指数形式输出时，缺省精度为 5；对 double 类型提供 16 位有效数字。

【例 3.1】写出程序的运行结果。

```
#include<stdio.h>
void main()
{   int a=65,b=97;
    printf("%d %d\n",a,b);
    printf("%d,%d,5d,-5d\n",a,b,a,b);
    printf("%3c, %-3c,%3d,%-3d\n",a,b,a,b);
    printf("a=%d,b=%d",a,b);
}
```

程序运行的结果为：

```
65□97
65,97,□□□65,97□□□
□□A,a□□, □65,97□
a=65,b=97
```

本例中多次输出了 a, b 的值，但由于格式说明符的不同，输出的结果也不相同。第四行的输出语句格式控制串中，两个格式字符“%d”之间加了一个空格，所以输出的 a, b 值之间有一个空格。第五行的 printf 语句格式控制串中加入的是普通字符逗号，因此输出的 a, b 值之间加了一个逗号。第六行的格式字符分别按照字符和十进制整数的形式输出 a, b 值。第七

行中为了提示输出结果又增加了普通字符（“a=”、“b=”和“,”）。其中“\n”是转义字符，代表回车换行。其他附加修饰字符的含义见表 3-2。

3.2.2 scanf 函数

scanf 函数称为格式输入函数，是最常用和功能最强的输入函数。scanf 函数与 printf 函数对应，是从标准的输入设备中读取信息，按照格式描述把读入的信息转换为指定数据类型的数据，并把这些数据赋给指定的变量。scanf 函数的格式与 printf 函数的格式类似，其格式如下：

scanf(<“格式控制字符串”>, <&变量 1>[, <&变量 2>][, ...])

其中，“格式控制字符串”的作用与 printf 函数类似，同样包括“格式说明符”（以%开头，后跟 d、f、c、s 等）和“普通字符串”。除格式字符外，其他每一个变量的前面加取址运算符“&”，代表各变量的地址。它指出转换后的每一个输入数据的存储位置。例如，&a、&b 分别表示变量 a 和变量 b 的地址。对于 a=123 赋值表达式，a 是变量名，123 是 a 的值，&a 是变量 a 的地址。

与 printf 函数相比，需要注意的是，格式说明符从字符“%”开始到转换字符结束，中间可以含有指明数据的宽度的正整数、指明数据类型为长整型的小写字母 l（如%ld）或指明数据类型为短整型的字母 h（如%hd），但是不能含有指定输入数据的精度表示，如%3.2f，是不合法的。当输入的数据是 long 型时，应该使用%ld 形式，当输入的数据为 double 型时，则使用%lf 或%le，否则得不到正确的输入数据。其中，转换字符 f 和 e 作用相同，用来输入浮点数，输入是小数形式或指数形式都可以。

如果两个格式说明之间没有其他字符，则在输入时，两个数据之间必须以空格或 Tab 键或回车键来分隔输入的数据。但如果在格式说明符中提供了数据宽度，就不需要有任何分隔符，系统能自动识别。如果在两个格式说明之间还有其他字符（如“,”），则在输入数据时，用同样的字符分隔数据。需要注意的是，scanf 函数中的参数必须为地址参数，前面的“&”绝对不能省略，而且输入的数据类型必须与定义类型相符合。举例见表 3-4。

表 3-4 scanf 函数举例及其说明

输入语句	输入数据格式（□代表空格）	说明
char ch1,ch2; scanf("%c%c",&ch1,&ch2);	Ax	输入两个字符，即将'A'赋值于变量 ch1，'x'赋值于变量 ch2
int m,n; scanf("%d%d",&m,&n);	8□12 或 8（Tab 键）12 或回车键隔开	输入两个整数
float f1,f2; scanf("%f:%e",&f1,&f2);	-9.87:1.23e5	包含普通字符“:”，输入两个浮点数以：隔开
short h; long l; scanf("%hd,%ld",&h,&l);	168,2269998	包含逗号，输入短整数和长整数，以“,”隔开
unsigned d; scanf("%u",&d);	65535	输入无符号整数，也可以用“%d”格式说明符
int d1,d2; scanf("%2d%2d",&d1,&d2);	1234	说明两个整数的宽度都为 2，故不需要分隔符，将 12 赋值于 d1，34 赋值于 d2

【例 3.2】输入三个小写字母，输出其 ASCII 码和对应的大写字母。

```
#include<stdio.h>
void main()
{   char a,b,c;
    int a1,b1,c1;
    printf("input character a,b,c: ");
    scanf("%c,%c,%c",&a,&b,&c);
    a1=a-('a'-'A');
    b1=b-('a'-'A');
    c1=c-('a'-'A');
    printf("%d,%d,%d\n%c,%c,%c\n",a,b,c,a1,b1,c1);
}
```

程序运行的结果为：

```
input character a,b,c: a,b,c
97,98,99
A,B,C
```

在 ASCII 码表中，大小写字母是按照顺序排列的，大写字母在前，小写字母紧跟其后，大小写字母之间相差 ASCII 值恒为 32，也就是'a'-'A'的值。

3.3 字符输入输出函数

1. putchar 函数

putchar 函数是 C 语言标准库函数中提供的字符输出函数，用于完成单个字符的输出。一般形式为：

```
putchar(c);
```

其功能是用来输出字符变量 c 的值。例如：

```
putchar('A');          /*输出大写字母 A*/
putchar(x);            /*输出字符变量 x 的值*/
putchar('\n');         /*换行。“\n”控制字符执行控制功能，不在屏幕上显示*/
```

使用本函数前必须要用文件包含命令：`#include<stdio.h>`或`#include "stdio.h"`。并且写在文件的开头位置，表示将“标准 I/O 库函数”包含在本文件中。

【例 3.3】输出单个字符。

```
#include<stdio.h>
void main()
{   char a='c',b='o',c='m';
    putchar(a);putchar(b);putchar(c);
    putchar('\n');
}
```

程序运行结果为：

```
com
```

2. getchar 函数

getchar 函数也是 C 语言标准库函数中提供的输入函数，其功能是从键盘上输入一个字符。一般形式为：

```
getchar();
```

通常把输入的字符赋予一个字符变量，例如：

```
char c;
```

```
c=getchar();
```

从键盘上输入一个字符，按回车键结束，表示将该字符保存到变量 `c` 所指的内存单元中。

同样，使用本函数前必须使用文件包含命令：`#include<stdio.h>`或`#include "stdio.h"`。

3.4 关系表达式和逻辑表达式

3.4.1 关系运算符及其表达式

关系运算符用于确定两个数据之间是否存在某种关系。经常利用关系运算符写出关系表达式比较两个数据，判断其比较的结果是否符合某种条件。

C 语言的关系运算符共有 6 个，如表 3-5 所示。

表 3-5 关系运算符

运算符	含义
< <= > >=	小于 小于等于 大于 大于等于
== !=	等于 不等于

关系运算符的优先级低于算术运算符，高于赋值运算符。在六个关系运算符中，`<`、`<=`、`>`、`>=`的优先级相同，高于`==`和`!=`，其中`==`和`!=`的优先级相同。C 语言规定在关系运算符构成的表达式中，若关系成立，则表达式值为非零整数（系统实现时一般为 1），若不成立，则表达式的值为零。可以看出，在 C 语言中用整型数值替代逻辑值。需要注意的是，C 语言中的等于关系用关系运算符“`==`”表示，而不是赋值运算符“`=`”。那么，关系表达式的一般形式为：

<表达式> <关系运算符> <表达式>

例如，由关系运算符构成的几个关系表达式实例：

```
a+b>c-d
```

```
i==j
```

```
ch <='a'
```

```
t!=k+1
```

关系表达式的结果只有非零整数和零两种。假设 `a=5;b=8;`，则 `a>b` 的结果为零，`a!=b` 的结果为非零整数（系统实现时一般为 1）。

3.4.2 逻辑运算符及其表达式

关系表达式所描述的结果只有两种。即关系成立，或者关系不成立。所以说，关系表达式描述的是一种逻辑判断。当一个关系成立时，人们常说这个关系表达式所表达的关系为“真”，或者说它具有逻辑值“真”；而在其关系表达式不成立时就说该关系为“假”，或者说表达式具有逻辑值“假”。C 语言规定值为零，表示逻辑值“假”；非零数值，表示逻辑值“真”。C 语言没有专用的逻辑类型，但 C 语言中提供了逻辑运算符，如表 3-6 所示。

表 3-6 关系运算符

操作符	含义
!	逻辑非
&&	逻辑与
	逻辑或

下面列出由逻辑运算符构成的几个表达式实例：

```
a>b && c>d
!b||x<=2
!(sex=="男"&& age<=18)
```

对于“!”运算符，它是将非零数值（真）转化为零（假），把零转化为非零数值（系统实现时一般为 1）。对于“&&”运算符，只有左右两个操作数都为非零（真），返回值才为非零（真），否则返回值为零（假）。而对于“||”运算符，只要左右两个操作数中有一个为非零（真），返回值为非零（真），否则返回值为零（假）。逻辑值的运算规则可以用真值表来说明，如表 3-7 所示。

表 3-7 真值表

操作数 a	操作数 b	!a	a&& b	a b
非 0	非 0	0	1	1
非 0	0	0	0	1
0	非 0	1	0	1
0	0	1	0	0

不同的运算符，其优先级不同，下面列出常用运算符的优先级，如图 3-2 所示。

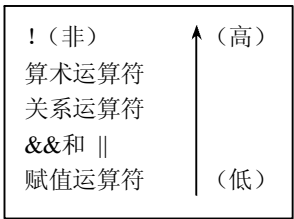


图 3-2 运算符的优先级

按照运算符的优先顺序可以得出：

```
(1) 1>3 && c>d && x>y      等价于    (1>3)&&(c>d)&&(x>y)
(2) 5>3||d<a||x<y          等价于    (5>3)||((d<a)||(x<y))
```

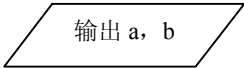
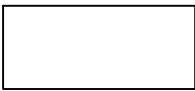
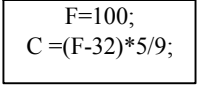
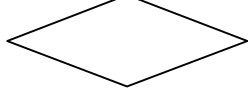
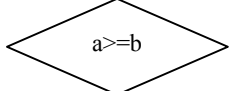
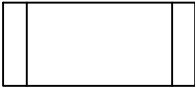
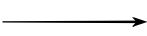
因为“&&”和“||”运算符的优先级低于关系运算符，上述示例的圆括号可以省略，这也反映了 C 语言简洁的特点。对于(1)表达式，1>3 不成立，即为零（假），则不再做后面的“&&”的运算，则整个表达式的值为零（假）。对于(2)表达式，5>3 成立，即为非零（真），则不再进行后面的“||”运算，则整个表达式的值为非零（真）。这就说明，在由一个或多个“&&”

相连的表达式中，只要遇到第一个操作数为零，就不再做后面的“&&”运算，则整个表达式的值为零。而在由一个或多个“||”相连的表达式中，只要遇到第一个操作数为非零，就不再做后面的“||”运算，则整个表达式的值为非零。所以，在 C 语言中，把“&&”和“||”称为短路运算符。

3.5 流程图简介

流程图是使用一些规定图框表示各种操作。使用规定的图形符号描述算法，具有直观、形象、易于理解等优点。流程图符号及其含义如表 3-8 所示。

表 3-8 流程图符号及其功能

图形符号	名称	含义	举例
	起止框	流程的起点和终点	
	输入输出框	数据的输入和输出	
	处理框	各种形式数据的处理	
	判断框	判断选择，根据条件满足与否选择合适的路径	
	特定过程	一个定义过程，如函数	
	流程线	连接各个图框，表示执行的顺序	
	连接点	表示与流程图其他部分相连	

一般情况下，在编写一个复杂的程序之前，先画出流程图，它是计算方案的形象描述。流程图的每一个框表示一段程序（包括一条或多条语句）的功能，各框内写明要做的事情，说明要简洁，不能含糊不清。

为了解决实际问题，对于较复杂的问题，可以分为 4 个步骤来考虑：

- （1）分析问题，确定计算的方案。
- （2）根据确定的方案，画出流程图。
- （3）检查流程图无误后，进行程序的编写。
- （4）运行和调试程序。

例如：输入三角形的三个边长，判断是否能构成三角形，并求出三角形的周长和面积。首先分析构成三角形的条件，三个边长都必须大于 0 而且满足两边之和大于第三边。如果能构成三角形，则计算三角形的面积。否则，就不能构成三角形。用流程图表示如图 3-3 所示，两

种流程图的表示方法，都能表示程序计算的方案和程序的执行顺序，但是显然图 3-3 (b) 简洁明了，建议读者采用此流程图表示。

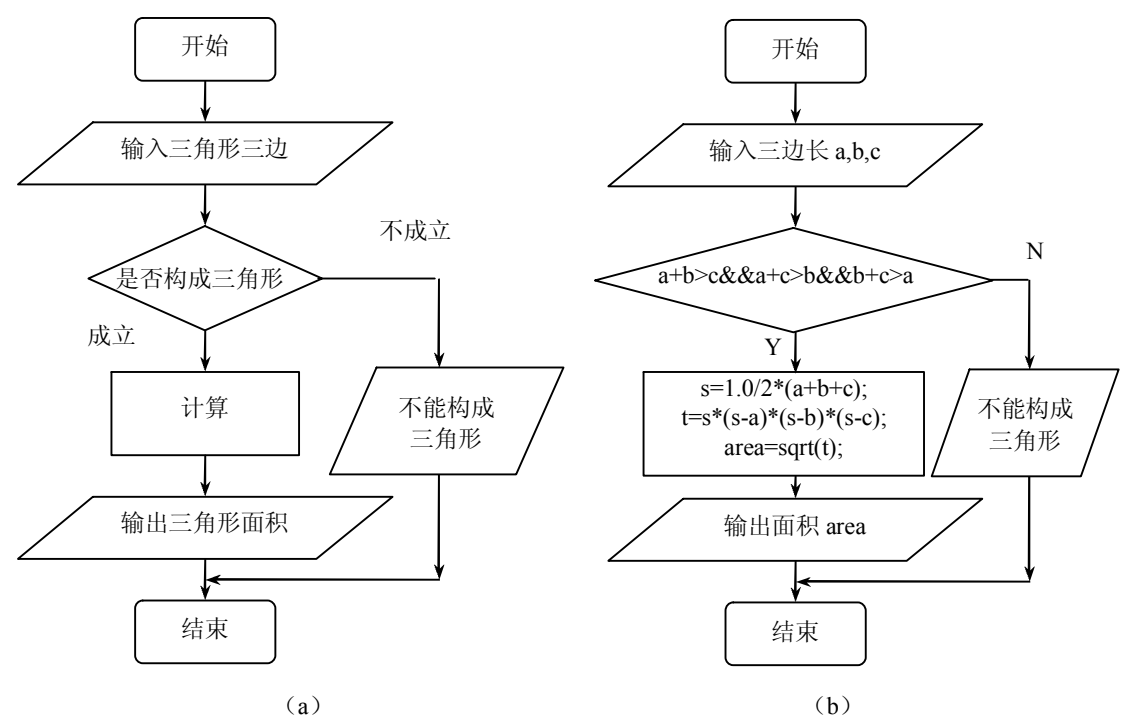


图 3-3 两种流程图的比较

类似的还有，比较两个数的大小，输出较大数，分段函数等；对于诸如这样的问题，画出流程图时，发现使用顺序程序无法满足要求，必须让计算机按照给定的条件进行分析，比较和判断，并对判断后的不同情况进行不同的处理。这种问题属于选择结构，选择语句就是为了解决这种问题而设计的。C 语言提供了 `if` 和 `switch` 来完成这种功能。本章将详细介绍选择控制语句。

对于初学者来说，画出流程图，有助于编写出完整正确的程序。

3.6 简单的 if 语句

3.6.1 单分支 if 语句

单分支 if 语句的格式为：

`if(<表达式>)[语句;]`

其含义是判断“表达式”是否成立，如果“表达式”的值为非零（真）时，执行“语句”，否则什么也不执行，直接转到 if 语句后面的第一条语句。其执行过程如图 3-4 所示。

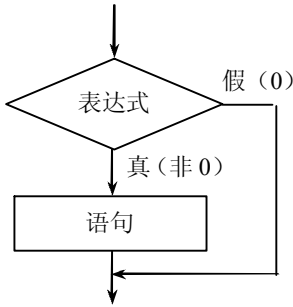


图 3-4 单分支 if 语句的执行流程

说明：

(1) if 是 C 语言的关键字，紧跟 if 后面的表达式两侧的圆括号必不可少，括号的后面一般不加分号，而在“语句”的后面加分号。

(2) “表达式”可以是任何符合 C 语言规则的表达式，如逻辑表达式、关系表达式、算术表达式、赋值表达式等，也可以是任意类型的数据，如整型、浮点型、字符型等。

(3) “语句”构成了 if 语句的一个分支，可以包含一条或多条语句。当包含两条以上的语句时，则用“{ }”把多条语句括起来组成复合语句，从整体上看，仍然满足“一条语句”的功能。

只有一条语句的 if 结构形式，如：

```
if(a<=b)printf("%d",a);
```

有多条语句的 if 结构形式，如：

```
if(a>b)
{   t=a;
    a=b;
    b=t;
}
```

3.6.2 双分支 if...else 语句

双分支 if...else 语句格式：

```
if(<表达式>)
    [语句 1;]
else
    [语句 2;]
```

双分支 if 语句是使用比较频繁的一种条件语句。其功能如下：如果表达式的值为真，则执行“语句 1”，否则执行“语句 2”。“语句 1”和“语句 2”是 if 语句的双分支语句。“语句 1”和“语句 2”都可以包含一条或多条语句，如果是多条语句，同样使用“{ }”括起来，构成复合语句。其执行过程如图 3-5 所示。

【例 3.4】输入三角形的三个边长，判断是否能构成三角形，并求出三角形的面积。流程图如图 3-3 所示。

```
#include <stdio.h>
#include <math.h>
void main()
{   float a,b,c,s,area;
    scanf("%f,%f,%f",&a,&b,&c);
    if(a+b>c&& a+c>b&& b+c>a)
    {   s=1.0/2*(a+b+c);
        area=sqrt(s*(s-a)*(s-b)*(s-c));
        printf("三角形的面积为%.2f\n",area);
    }
    else
        printf("不能构成三角形\n");
}
```

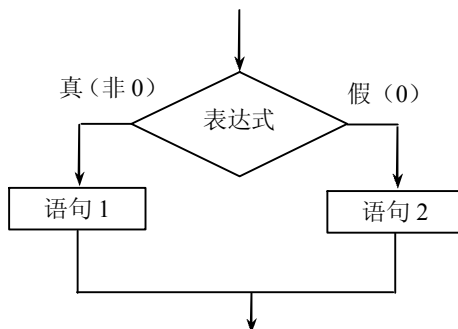


图 3-5 双分支 if 语句的执行流程

运行的结果为：

```
3.0 4.0 5.0
三角形的面积为 6.00
5.8 9.8 2.3
不能构成三角形
```

【例 3.5】输入一个正整数，判断输入的正整数是偶数，还是奇数。

分析：如果能被 2 整除，则说明是偶数。用 C 语言表达式表示： $x \% 2 == 0$ 。否则，输入的整数为奇数。

```
#include <stdio.h>
void main()
{
    int x;
    printf("请输入一个正整数: ");
    scanf("%d",&x);
    if(x%2==0)
        printf("您输入的正整数%d 是偶数。\\n",x);
    else
        printf("您输入的正整数%d 是奇数。\\n",x);
}
```

运行结果为：

```
请输入一个正整数: 208
您输入的正整数 208 为偶数。
请输入一个正整数: 119
您输入的正整数 119 为奇数。
```

注意：在程序中关键字 `else` 必须与关键字 `if` 配对，共同组成一条 `if...else` 语句。`else` 只是 `if` 语句的一部分，不能单独使用。对于选择结构程序只能执行“语句 1”或者“语句 2”，绝对不可能同时执行两个分支。

【例 3.6】根据定期存款期限和相应利率，计算本息合计。

存款期限	年利率(%)	本金	本息合计
三个月	1.71		
六个月	1.98		
一年	2.25		
二年	2.79		
三年	3.33		
五年	3.60		

```
#include <stdio.h>
void main()
{
    float c,r; /*存款数额和利率*/
    int m; /*存款期限*/
    scanf("%f,%d",&c,&m);
    if(m==3)
        r=1.71;
```

```

if(m==6)
    r=1.98;
if(m==12)
    r=2.25;
if(m==24)
    r=2.79;
if(m==36)
    r=3.33;
if(m==60)
    r=3.60;
s=c*(1+r/100);
printf("s=%f\n",s);
}

```

3.7 C 语言中几个特殊知识点

3.7.1 类型转换

当某个运算符的运算对象具有不同数据类型时，就出现了混合类型计算。例如表达式：

$3.14 + 315$

这里的一个运算对象 3.14 为 double 型，而另一个运算对象 315 是 int 型。而 C 语言规定，只有不同类型的操作数转换为同一类型时，才能计算。转换的方法有两种，一种是自动类型转换，另一种是强制类型转换。

1. 自动类型转换

自动转换发生在不同数据类型的数据进行混合运算时，由编译系统自动完成。自动转换的基本原则是把表示范围小的数值转换为表示范围大的类型数值。转换规则如图 3-6 所示。

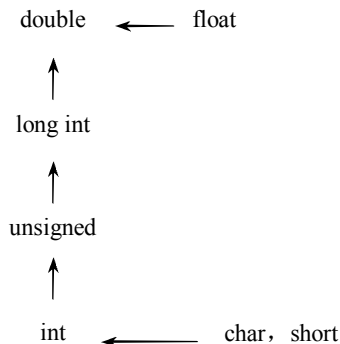


图 3-6 类型转换规则

水平箭头表示必定转换，即在运算时，char 型和 short 型必定转换为 int 型，float 型必定转换为 double 型。参与运算的两个操作数都为 float 型，也要转换为 double 型，以提高运算精度。垂直箭头表示运算的操作数类型不同时的转换方向，按照 int、unsigned、long、double 的顺序由低向高直接转换。需要指出的是，不能认为 int 型先转换为 unsigned 型，然后再转换为 long

型，而是直接由一种类型转换为另一种类型。例如有如下定义：

```
int m;  
float n;  
double d;  
long int c;
```

则表达式：

```
('a'+'d')*10+m*n-d/c
```

上述表达式是这样转换的：计算('a'+'d')时，先将'a'和'd'转换为其 ASCII 码值对应的整型数据 97 和 100，运算的结果为 197；计算 m*n 时，先将 m 和 n 都转换为双精度型；计算 d/c 时，先将 c 转换为双精度型；然后运算('a'+'d')*10 的结果为 1970，再将 1970 转换为双精度型，然后与 m*n 的结果相加，最后再减去运算结果为双精度型 d/c 的结果。表达式计算完毕，结果为双精度型。

如下表达式：

```
3.1415926*2*1.50+'a'-2345678
```

其中 3.1415926 是双精度浮点数，2 是整数，1.50 是单精度浮点数，'a'为字符型，2345678 为长整型，按照规则，均转换为双精度型，然后计算。

2. 强制类型转换

强制类型转换是把一种类型的数据强制地转换为另一种类型的数据。其一般形式为：

```
(<类型说明符>)<表达式>
```

其功能是把表达式的运算结果强制转换成类型说明符所表示的类型。

例如：

```
(float) a      把 a 转换为单精度浮点型  
(int)(x+y)     把 x+y 的结果转换为整型
```

在使用强制转换时应注意以下问题：

(1) 类型说明符和表达式都必须加括号（单个变量可以不加括号），如把(int)(x+y)写成(int)x+y 则成了把 x 转换成 int 型之后再与 y 相加了。

(2) 无论是强制转换还是自动转换，都只是为了本次运算的需要而对变量的数据类型进行的临时性转换，而不改变数据说明时该变量定义的类型。假如 a 是 int 型，则表达式(float)a 的类型为 float，但 a 仍保留原先的 int 型，a+(float)a 的结果就是浮点型。

3.7.2 逗号运算符和逗号表达式

在 C 语言中“,”（逗号）也是一种运算符，称为逗号运算符。其作用就是把两个表达式连接起来组成一个表达式，称为逗号表达式。其一般形式为：

```
<表达式 1>,<表达式 2>
```

其功能就是先计算“表达式 1”的值，再计算“表达式 2”的值，整个逗号表达式的值就是“表达式 2”的值。例如逗号表达式：

```
a=3*2,a-2
```

则先计算 a=3*2，得 a 的值为 6，然后计算 a-2，得 4，故整个逗号表达式的值为 4。

【例 3.7】写出程序运行结果。

```
#include <stdio.h>  
void main()
```

```

{   int a=3,b=5,y;
    y=(a=3*4,b+a);    /*整体为赋值语句，其中 a=3*4,b+a 为逗号表达式*/
    printf("%d,%d",y,a);
}

```

运行的结果为：

17,12

本例中，y 等于整个逗号表达式的值，也就是表达式 2 的值，a 是表达式 1 的值。需要注意的是，对于表达式 `y=(a=3*4,b+a)` 最外面的括号不能省略，只有这样才是将逗号表达式 `a=3*4,b+a` 的值赋给变量 y。因为逗号运算符是所有运算符中优先级最低的运算符。所以不要将逗号表达式写成 `(a=3*4),(b+a)`。

3.7.3 条件表达式

构造条件表达式需要使用条件运算符，这是 C 语言中唯一的一个具有三个操作数的运算符，称为三目运算符。条件表达式的形式为：

`<表达式 1>?<表达式 2>:<表达式 3>`

条件表达式的计算方式比较特殊，其含义：首先计算“表达式 1”的值，如果为非零（真），则返回“表达式 2”的值，即“表达式 2”的值就是该表达式的值。否则，该表达式的值就为“表达式 3”的值。这里的“?”和“:”总是成对出现。例如：

`min=(x<y)?x:y`

该表达式就是将 x, y 两数的最小值赋值于变量 min。即就是将“=”右边的条件表达式 `(x<y)?x:y` 的值赋值于 min，所以说整体上这是一个赋值表达式。

这个表达式和下面程序执行的结果是相同的：

```

if(x<y)
    min=x;
else
    min=y;

```

C 语言有丰富的运算符，这是一个优点，但它的缺点在于运算符的优先级和结合性较为繁杂，不容易记忆。所谓的结合性是指运算符先与左边的操作数或者参与运算的分量进行计算，还是运算符先与右边的操作数或参与运算的分量进行计算。前者称为左结合，后者称为右结合。有关 C 语言运算符的优先级和结合性规则，请参见“附录”。

3.8 综合应用实例

3.8.1 程序分析

1. 输入下面程序，分析并验证输出结果。

【提示】该程序主要练习字符变量的定义与初始化；字符输入函数 `getchar()`、字符输出函数 `putchar()` 和 `printf()` 函数的使用方法；使用 `printf()` 函数按照“%c”格式输出单个字符。

```

#include "stdio.h"
void main()
{   char a,ch;

```

```

a='C';
putchar(a);
putchar('\n');
printf("请输入一个字符: ");
ch=getchar();
printf("%c\n",ch);
}

```

程序运行后:

输出结果为:

```

C
请输入一个字符: p<回车>

```

屏幕显示: p

【想一想】

(1) 将语句“ch=getchar();”改成scanf()语句如何实现?

(2) 程序中 putchar(a);语句的功能是什么?

同学们可自行输入任何单个字符进行验证。

2. 输入下面程序, 分析并验证输出结果。

【提示】该程序主要练习整型变量、单精度类型变量、字符型变量的定义、初始化及“%d”、“%md”、“%f”、“%m.nf”、“%c”几种格式的输出形式。

```

#include "stdio.h"
void main()
{
    int a=17,b=5;
    float x=123.45,y=-98.765;
    char c='a';
    printf("%d,%d\n",a,b);
    printf("%-3d,%3d\n",a,b);
    printf("%d\n",a/b);
    printf("%f,%8.2f\n",x,y);
    printf("%c,%d\n",c,c);
}

```

程序运行后, 输出结果为:

```

17,5
17□,□□5
3
123.449997,□□-98.76
a,97

```

说明: 符号“□”表示空格。

【想一想】

(1) 使用“%f”输出数据, 小数点后保留几位? “%8.2f”表示的含义是什么?

(2) 输出格式“%-3d”表示的含义是什么?

3. 已知三角形的三边长分别为 a, b, c, 编写程序, 计算三角形的面积。

面积公式为 $area = \sqrt{s(s-a)(s-b)(s-c)}$, 其中 $s = \frac{a+b+c}{2}$ 。

【提示】注意构成三角形的条件是“任意两边之和大于第三边”，只有满足条件时，才可计算三角形的面积。任意两边之和大于第三边需要用关系运算符及逻辑运算符一起构成一个逻辑表达式： $(a+b)>c \ \&\& \ (b+c)>a \ \&\& \ (c+a)>b$ ，将该表达式作为 if 语句的条件。

参考程序代码：

```
#include "stdio.h"
#include "math.h"
void main()
{
    float a,b,c,area,s;
    scanf("%f%f%f",&a,&b,&c);
    if((a+b)>c && (b+c)>a && (c+a)>b)
    {
        s=(a+b+c)/2;
        area=sqrt(s*(s-a)*(s-b)*(s-c));
        printf("area=%0.2f",area);
    }
}
```

程序运行后：

输入数据为：3.0 4.0 5.0

输出结果为：area=6.00

再次运行程序后：

输入数据为：3.0 9.0 5.0

无输出信息。（因程序中只对构成三角形的条件进行了判断，是单分支结构。）

【想一想】

(1) 程序中将“&&”改为“||”可以吗？

(2) 在上述的 if 语句中的“{...}”表示什么含义？

4. 根据每个人的身高和体重之间的关系，得到一个体重指数，即：体重指数=体重(kg)/(身高)²。编写程序，利用体重指数进行判断：若体重指数<25，属于正常；否则属于过胖。

【提示】设变量 weight 表示体重、height 表示身高、index 表示体重指数，若体重指数<25，属于正常；否则属于过胖。利用 if 语句实现双分支结构。

参考程序代码：

```
#include "stdio.h"
void main()
{
    float weight,height,index;
    printf("\n 请输入你的体重(千克):");
    scanf("%f",&weight);
    printf("\n 请输入你的身高(米):");
    scanf("%f",&height);
    index=weight/(height*height);
    printf("\n 你的体重指数是: %f",index);
    if(index<25)
        printf("\n 正常!\n");
    else
        printf("\n 过胖!\n");
}
```

程序运行后:

输入数据为: 请输入你的体重(千克): 52<回车>

请输入你的身高(米): 1.60<回车>

输出结果为: 你的体重指数是: 20.3125

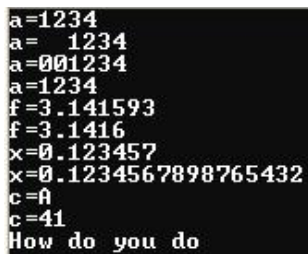
正常!

3.8.2 应用实例

1. 由本章所学的 `printf` 函数, 并结合上一章学习的数据类型, 编写如下的程序, 以加深对 C 语言数据类型的理解。

```
#include<stdio.h>
void main()
{   char c;
    int a=1234;
    float f=3.141592653589;
    double x=0.12345678987654321;
    c='\x41';
    printf("a=%d\n", a);          /*结果输出十进制整数 a=1234*/
    printf("a=%6d\n", a);         /*结果输出 6 位十进制数 a= 1234*/
    printf("a=%06d\n", a);        /*结果输出 6 位十进制数 a=001234*/
    printf("a=%2d\n", a);         /*a 超过 2 位, 按实际值输出 a=1234*/
    printf("f=%f\n", f);          /*输出浮点数 f=3.141593*/
    printf("f=%6.4f\n", f);       /*输出共 6 位其中小数点后 4 位的浮点数
                                   f=3.1416*/
    printf("x=%lf\n", x);         /*输出长浮点数 x=0.123457*/
    printf("x=%18.16lf\n", x);    /*输出共 18 位其中小数点后 16 位的长浮点数
                                   x=0.1234567898765432*/
    printf("c=%c\n", c);         /*输出字符 c=A*/
    printf("c=%x\n", c);         /*输出字符的 ASCII 码值 c=41*/
    printf("%s\n", "How do you do"); /*输出字符串 "How do you do" */
}
```

程序运行的结果如图 3-7 所示。



```
a=1234
a= 1234
a=001234
a=1234
f=3.141593
f=3.1416
x=0.123457
x=0.1234567898765432
c=A
c=41
How do you do
```

图 3-7 格式说明及含义

2. 在银行存款, 计算本息和的公式为:

$$p1=p(1+r)^n$$

其中 p 为存款金额, n 为年数, r 为年利率 (假设年利率为 2.72%), $p1$ 为 n 年后本金和利

息之和。输入存款金额 p ，计算 3 年后的本息和，要求结果保留两位小数。

```
#include<stdio.h>
void main()
{
    float p,p1,r;
    r=0.0272;
    printf("请输入存款金额: ");
    scanf("%f",&p);
    p1=p*(1+r)*(1+r)*(1+r);
    printf("3 年后您的本息和为: %.2f 元\n",p1);
}
```

运行结果为:

```
请输入存款金额: 1000
3 年后您的本息和为 1083.84 元
```

3. 计算等差数列前 n 项和。

已知等差数列的第一项为 a_1 ，公差为 d ，前 n 项和的计算公式为： $s=a_1*n+n(n-1)*d/2$ 。请编写程序实现，从键盘上输入等差数列的第一项 a_1 ，公差 d 和 n ，计算前 n 项和。

```
#include<stdio.h>
void main()
{
    int a1,n,d,s;
    printf("请输入等差数列的 a1,d 和 n: ");
    scanf("%d,%d,%d",&a1,&d,&n);
    s=a1*n+n*(n-1)*d/2;
    printf("a1=%d,d=%d,n=%d,s=%d\n",a1,d,n,s);
}
```

程序运行的结果为:

```
请输入等差数列的 a1,d 和 n: 2, 3, 20
a1=2, d=3, n=20, s=610
```

4. 交换两个变量的值。

```
#include <stdio.h>
void main()
{
    int a,b,t;
    scanf("%d,%d",&a,&b);
    printf("a=%d,b=%d\n",a,b);
    t=a;
    a=b;
    b=t;
    printf("a=%d,b=%d\n",a,b);
}
```

运行时输入:

```
88, 45
```

程序运行的结果为:

```
a=88, b=45
a=45, b=88
```

在 C 语言程序中，要进行两个变量值之间的交换，必须借助于第三个变量，才能实现两

个数的交换。对于上述例题，t 就是中间变量，假如 a=88，b=45；首先 t=a，就是将 88 赋值给变量 t，t 的值就是 88；然后 a=b，就是将 45 赋值给变量 a，这时变量 a 的值变为 45；最后 b=t，就是将 88 赋值于变量 b，从而实现两个变量值的交换。所以绝对不能写成如下程序段：

```
a=b;
b=a;
```

这样，最终变量 a、b 的值都为 45，请读者仔细想想，为什么？

5. 输入三个整数 x，y，z，请按照由小到大的顺序输出。

分析：先假设 x 为最小数，然后将 x 与 y 进行比较，如果 x>y 则将 x 与 y 的值进行交换；然后再用 x 与 z 进行比较，如果 x>z 则将 x 与 z 的值进行交换，最终使得 x 为最小数；以此类推，y 次之，z 为最大数。

```
#include <stdio.h>
void main()
{
    int x,y,z,t;
    printf("请输入 3 个数 x, y, z: ");
    scanf("%d,%d,%d",&x,&y,&z);
    if(x>y)
        {t=x;x=y;y=t;}          /*交换 x,y 的值*/
    if(x>z)
        {t=x;x=z;z=t;}          /*交换 x,z 的值*/
    if(y>z)
        {t=y;y=z;z=t;}          /*交换 z,y 的值*/
    printf("从小到大的顺序为: %d %d %d\n",x,y,z);
}
```

程序运行的结果为：

请输入 3 个数 x，y，z：100，8，45

从小到大的顺序为：8 45 100

6. 输入整数 a 和 b，如果 $a^2+b^2>100$ ，则输出 a^2+b^2 之和的百位以上的数字，否则直接输出 a^2+b^2 的和。

```
#include <stdio.h>
void main()
{
    int a,b,x,y;
    printf("请输入整数 a, b: ");
    scanf("%d,%d",&a,&b);
    x=a*a+b*b;
    if(x>100)
    {
        y=x/100;          /*截取百位以上的数字*/
        printf("百位以上的数字为: %d",y);
    }
    else
        printf("a,b 平方之和为: %d\n",x);
}
```

程序运行结果为：

请输入整数 a，b：5，6

a,b 平方之和为：61

请输入整数 a, b: 25, 10

百位以上的数字为: 7

7. 本章导学提出问题的答案。

//数学工具小软件

//功能: 计算圆的周长、圆的面积、球的表面积、球的体积

//公式 1: 圆的周长 $C=2\pi r$

//公式 2: 圆的面积 $S=\pi r^2$

//公式 3: 球的表面积 $GS=4\pi r^2$

//公式 4: 球的体积 $GV=(4/3)\pi r^3$ (r 为半径)

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <conio.h>
```

```
#include <ctype.h>
```

```
#define PI 3.14
```

```
void main()
```

```
{
```

```
int n;
```

```
float r,c,s,gs,gv;
```

```
printf("欢迎来到软件园.....\n\n");
```

```
printf("-----选项-----\n\n");
```

```
printf(" 1.计算圆的周长.\n");
```

```
printf(" 2.计算圆的面积.\n");
```

```
printf(" 3.计算球的表面积.\n");
```

```
printf(" 4.计算球的体积.\n");
```

```
printf("-----\n\n");
```

```
printf("请输入选项:(回车键确认)");
```

```
scanf("%d",&n);
```

```
printf("\n 请输入半径的值:(回车键确认)");
```

```
scanf("%f",&r);
```

```
if (n==1)
```

```
{
```

```
    c=2*PI*r;
```

```
    printf("\n 圆的周长为: %f\n",c);
```

```
}
```

```
if (n==2)
```

```
{
```

```
    s=PI*r*r;
```

```
    printf("\n 圆的面积为: %f\n",s);
```

```
}
```

```
if (n==3)
```

```
{
```

```
    gs=4*PI*r*r;
```

```
    printf("\n 球的表面积为: %f\n",gs);
```

```
}
```

```
if (n==4)
```

```
{
```

```

        gv=(4/3.0)*PI*r*r*r;
        printf("\n 球的体积为: %.2lf\n",gv);
    }
}

```



本章小结

C 语言程序的基本组成单位是函数，而函数组成语句。C 语言中，语句可分为流程控制语句（有 if 等 9 种）、表达式语句、复合语句、函数调用语句和空语句五类。流程控制语句又分选择控制语句、循环控制语句和控制转移语句。表达式后跟一个分号构成表达式语句。用大括号括起的一条或多条语句称为复合语句，它在语法上被看作一条语句。空语句由一个分号构成，常用在那些语法上需要一条语句，而实际上并不需要任何操作的场合。

C 语言中没有专门的输入输出语句，而是通过函数来实现的。在 C 语言库函数中提供了一组输入输出函数。主要介绍了标准的格式输入输出的函数：scanf 函数和 printf 函数，及其格式说明等。

本章介绍的语句和函数可以进行顺序结构和简单的选择结构程序设计。



习题 3

- 简述 C 语言程序的语句组成。
- 写出下列 printf 语句的输出结果。
 - printf("%6d\n",1234);
 - printf("%-8d\n",1234);
 - printf("%8.4f\n",1234.5678);
 - printf("%3c\n",65);
 - printf("%3d\n",'a');
 - printf("%3c\n",'a');
 - printf("%10.5s\n","abcdefg");
- 请写出能得到以下输出格式和结果的程序。已知变量 a=4, b=8, c=10, x=2.5, y=3.6, z=-4.8, ch1='a', ch2='b'。要求有关变量的定义，并给变量赋值和输出语句等组成一个完整的 C 简单程序（注意空格的输出）。
 - a=□4 b=□8 c=10
 - x=2.500000, y=3.600000, z=-4.800000
 - x+y=6.10, y+z=-1.20
 - ch1=a OR ch1=97
 - ch2=□□b OR ch2=98
- 下述程序的输出结果。


```

#include <stdio.h>
void main()
{   int a =011, b=101;

```