第3章 顺序结构程序设计

本章主要介绍C语言的顺序结构程序设计方法，顺序、分支、循环结构是程序设计的基础。所有的程序都由这3种结构组合而成。

什么叫程序设计？对于初学者来说，往往把程序设计简单地理解为只是编写一个程序，这是不全面的。程序设计反映了利用计算机解决问题的全过程，包含多方面的内容，而编写程序只是其中的一个方面。使用计算机解决实际问题，通常是先对问题进行分析并建立数学模型，然后考虑数据的组织方式和算法，并用某一种计算机语言编写程序，最后调试程序，使之运行后能产生预期的结果，这个过程称为程序设计。在结构化程序设计中顺序结构是最简单，也是最常用的程序结构，它严格按语句出现的先后次序顺序执行。

3.1 结构化程序设计

程序是命令的有序集合，命令执行的顺序即程序的结构。一个程序的功能不仅取决于所选用的命令，还决定于命令执行的顺序。在结构化程序设计中，把所有程序的逻辑结构归纳为3种：顺序结构、选择结构(也叫分支结构)和循环结构。

3.1.1 结构化程序设计概述

Bohn和Jacopini于1966年提出了结构化程序设计的理论。结构化程序设计思想和方法的引入，使程序结构清晰，容易阅读、修改和验证，从而提高了程序设计的质量和效率。

结构化程序设计方法是用高级语言表示的结构化算法。该方法的基本思路是把一个复杂问题的求解过程分阶段进行，每个阶段处理的问题都控制在人们容易理解和处理的范围内。结构化程序设计的原则是：

(1) 自顶向下。程序设计时，应该先总体、后细节、先全局、后局部。不要一开始就过多地追求细节，应从最上层总体目标开始，逐步使问题具体化。

(2) 逐步细化。对复杂问题设计一些子目标作过渡，逐步细化。

(3) 模块化设计。设计是编码的前导。所谓模块化设计，就是按模块组装的方法编程。把一个待开发的软件分解成若干个小的简单的部分，称为模块。每个模块都独立地开发、测试，最后再组装出整个软件。这种开发方法是对待复杂事物的“分而治之”的一般原则在软件开发领域的具体体现。模块化澄清和规范了软件中各部分间的界面，便于成组的软件设计人员工作，也促进了更可靠的软件设计实践。

(4) 结构化编程。软件开发的最终目的是产生能在计算机上执行的程序。即：使用选

定的程序设计语言，把模块描述为用该语言书写的源程序。重要的是结构化编程的思想，具备了该思想，语言就只是工具。

遵循结构化程序的设计原则，按照结构化程序设计方法设计出来的程序具有两个明显的优点：其一是程序易于理解、使用和维护；其二是提高了编程工作的效率，降低了软件开发的成本。

总体来说，程序设计应该强调简单和清晰，做到“清晰第一，效率第一”。

3.1.2　结构化程序设计的基本结构及其特点

结构化程序设计的基本结构有 3 种，这 3 种基本结构是表示一个良好算法的基本单元。

1. 顺序结构

这是最简单的一种基本结构，依次顺序执行不同的程序块，如图 3-1(a)所示。

2. 选择结构

根据条件满足或不满足而去执行不同的程序块，如图 3-1(b)所示。如满足条件 P，则执行 A 程序块，否则执行 B 程序块。

3. 循环结构

循环结构是指重复执行某些操作，重复执行的部分称为循环体。循环结构分当型循环和直到型循环两种，如图 3-1(c)和图 3-1(d)所示。

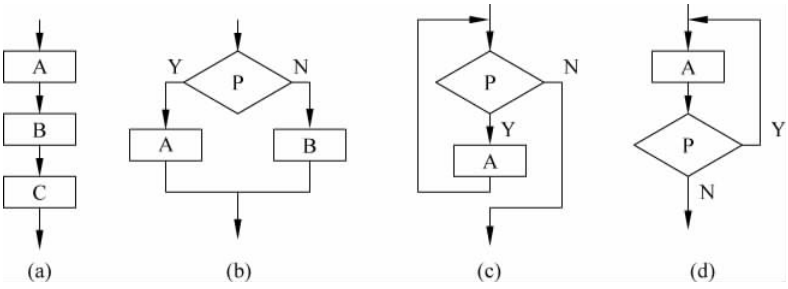

图 3-1　程序的控制结构

当型循环先判断条件是否满足，如满足条件 P 则反复执行 A 程序块，每执行一次判断一次，直到不满足条件 P 为止，跳出循环体执行它后面的基本结构。

直到型循环先执行一次，再判断条件是否满足，如满足条件 P 则反复执行 A 程序块，每执行一次判断一次，直到不满足条件 P 为止，跳出循环体执行它后面的基本结构。

3.2　算法

3.2.1　算法的基本概念

事实上，我们已经接触过很多算法，比如拨打电话、购买商品、包装一件货物、解决数学问题，等等。由于习惯，人们并没意识到做每一件事都需要事先设计，然而，事实上做每一件

事都是按一定的方法、步骤进行的。算法就是一种在有限的步骤内解决问题或完成任务的方法。

计算机程序就是告诉计算机如何去解决问题或完成任务的一组详细的、逐步执行的指令的集合。计算机编程就是用程序设计语言把算法程序化。事实上，编程是一件有趣的事，是一种创造性工作，也是一种用有形的方式表达抽象思维的方法。编程可以教会人们各种技能，如阅读思考、分析判断、综合创造以及关注细节等。

学习计算机程序设计首先应从问题描述开始，问题描述是算法的基础，而算法则是程序的基础。

数据是操作的对象，操作的目的是对数据进行加工处理，以得到期望的结果。作为程序设计人员，必须认真考虑和设计数据结构和算法。为此，1976 年瑞士计算机科学家沃思(N. Wirh)曾提出了一个著名的公式：

程序＝算法＋数据结构

实际上，在设计一个程序时，要综合运用算法、数据结构、设计方法、语言工具和环境等方面的知识。这其中，算法是程序设计的灵魂，数据结构是数据的组织形式，语言则是编程的工具。

3.2.2 算法的特性

并不是所有组合起来的操作系列都可以称为算法。算法必须符合以下 5 个基本特性：

(1) 有穷性。算法中的操作步骤必须是有限个，而且必须是可以完成的，有始有终是算法最基本的特征。

(2) 确定性。算法中每个执行的操作都必须有确切的含义，并且在任何条件下，算法都只能有一条可执行路径，无歧义性。

(3) 可行性。算法中所有操作都必须是可执行的。如果按照算法逐步去做，则一定可以找出正确答案。可行性是一个正确算法的重要特征。

(4) 有零个或多个输入。在程序运行过程中，有的数据是需要在算法执行过程中输入的，而有的算法表面上看没有输入，但实际上数据已经被嵌入其中了。没有输入的算法是缺少灵活性的。

(5) 有一个或多个输出。算法进行信息加工后应该得到至少一个结果，而这个结果应当是可见的。没有输出的算法是没有用的。

3.2.3 算法的流程图表示法

为了表示一个算法，可以用不同的方法描述。描述算法的常用方法有自然语言表示法、传统流程图表示法、N-S 图表示法、PAD 图(Problem Analysis Diagram)表示法和伪代码表示法等。使用它们的目的是把编程的思想用图形或文字表述出来。本节主要介绍用流程图描述算法的方法。

传统流程图是历史最悠久、使用最广泛的一种描述算法的方法，也是软件开发人员最熟悉的一种算法描述工具。它的主要特点是对控制流程的描绘很直观，便于初学者掌握。

流程图用一些图框表示各种类型的操作，用流程线表示这些操作的执行顺序。美国国

家标准协会 ANSI 规定了一些常用的流程图符号，如图 3-2 所示。

- 起止框：表示算法的开始或结束；
- 输入输出框：用于表示输入输出操作；
- 判断框：按条件选择操作；
- 处理框：用于表示赋值等操作；
- 流程线：表示流程及流程的方向。

【例 3.1】 输入两个实数，按数值由小到大依次输出这两个数。

分析：该问题的输入是两个实数，输出也是两个实数，但输出的两个实数是从小到大排序的。可见输入的两个实数是在比较了大小后输出的。如果输入的第一个数比第二个数小，则按输入次序输出这两个数；否则交换次序后输出即可。

经过上述分析，可以得到该题的算法。用流程图表示见图 3-3。

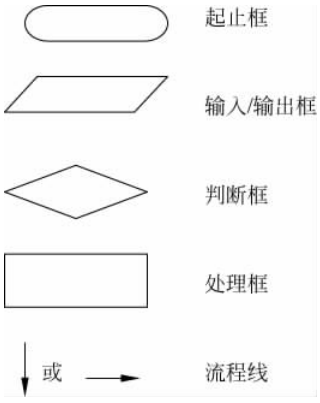

图 3-2　流程图符号

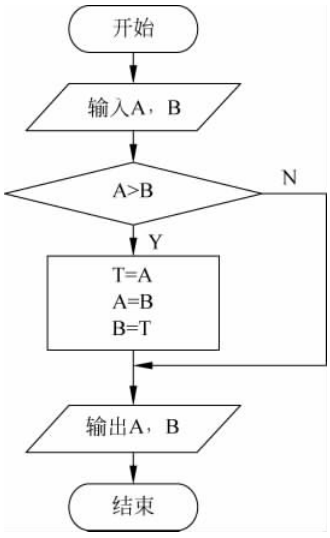

图 3-3　两个数排序的流程图

【例 3.2】 求从 1 开始的一百个自然数之和。

分析：由题意可知，这是重复百次的求和运算。即累加百次，而且每次参与累加的数就是累加的次数。据此分析，可用循环结构实现其算法。用当型循环结构表示，如图 3-4(a)所示；直到型循环如图 3-4(b)所示。

用流程图表示算法直观形象，能比较清楚地显示出各个框之间的逻辑关系。

3.2.4　基本算法

1. 累加

(1) 首先将和初始化为 0(令 sum＝0)。

(2) 循环，在每次迭代中将一个新值加到和上(如 sum＝sum＋i)。

(3) 退出循环后输出结果。

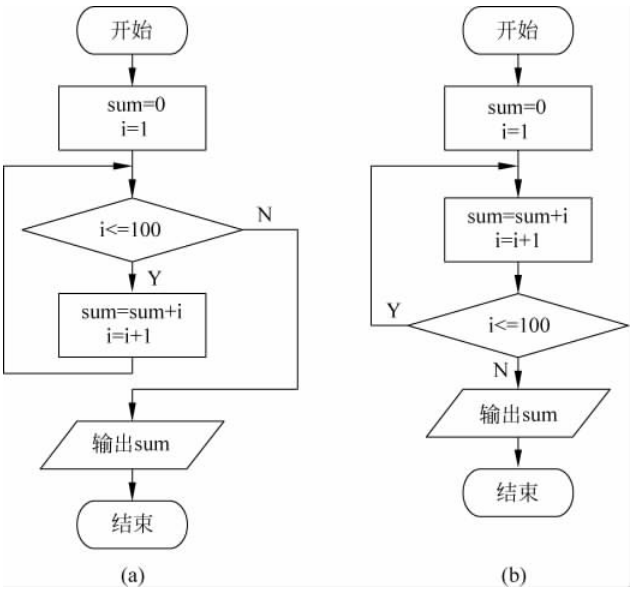

图 3-4　累加的流程图

2. 累乘

(1) 将乘积初始化为 1(令 product=1)。

(2) 循环,在每次迭代中将一个新数与乘积相乘(如 product=product * x)。

(3) 退出循环后输出结果。

3. 求最大或最小值

它的思想是通过分支结构找出两个数中的较大(小)值,然后把这个结构放在循环中,就可以得到一组数中的最大(小)值。

4. 穷举

穷举是一种重复型算法。它的基本思想是对问题的所有可能状态一一测试,直到找到解或将全部可能的状态都测试过为止。

5. 迭代(递推)

迭代(递推)是一个不断用新值取代变量的旧值,或由旧值递推出变量的新值的过程。迭代算法只包括参数,不包括算法本身。这个算法通常包含循环。

例如,阶乘计算的迭代算法(递推公式):

$$f(n)=\begin{cases}1 & (n=0)\\ n\times(n-1)\times(n-2)\times(n-3)\times\cdots\times3\times2\times1 & (n>0)\end{cases}$$

6. 递归

递归是算法自我调用的过程。一种算法是否叫作递归,关键取决于该算法定义中是否

有它本身。递归算法不需要循环，但递归概念本身包含循环。

例如，阶乘计算的递归算法（递归函数）：

$$f(n)=\begin{cases}1 & (n=0)\\ f(n-1)\times n & (n>0)\end{cases}$$

7. 排序

排序是在待排序记录中按一定次序（递增或递减）排列数据的过程。试想在一个没有顺序的电话号码本中，查找某人的电话号码是件多么困难的事。选择排序、比较排序和插入排序是当今计算机科学中使用快速排序的基础。

8. 查找

查找是在数据列表中确定目标所在位置的过程。对于列表，有两种基本的查找方法，即顺序查找和折半查找。顺序查找可在任何列表中查找，折半查找则要求列表是有序的。

3.3　C 语句概述

程序是语句的有序集合，因此，要想学会程序设计，就必须掌握每一条语句。一条语句只能完成有限的功能，而要完成一个比较复杂的功能，则需要一组按照一定顺序排列的语句。C 语句一般可划分为表达式语句、函数调用语句、空语句、复合语句和控制语句 5 类。

1. 表达式语句

表达式语句是在各种表达式后加一个分号构成的语句，表达式语句是 C 语言的一个重要特色，其一般形式如下：

```
表达式;
```

最典型的表达式语句是由一个赋值表达式加一个分号“;”构成的赋值表达式语句。

2. 函数调用语句

函数调用语句是在函数调用的一般形式后加一个分号构成的语句，其作用是完成特定的功能。其一般形式如下：

```
函数名(实参表);
```

例如：

```
scanf("%c",&a);          /*格式输入函数调用语句*/
printf("%c",a);          /*格式输出函数调用语句*/
```

C 语言有丰富的标准函数库，可以提供各类函数供用户调用。标准库函数完成预先设定的功能，用户无须定义，也不必在程序中声明，就可以直接调用。例如，常用的数学函数 sin(x)、sqrt(x)等。调用标准库函数时，需要在程序前用编译预处理命令将有关的含有该函数原理的头文件包括到程序中来。例如，#include <stdio.h>或 #include "stdio.h"，

#include < math. h >或#include "math. h"。

3. 空语句

在C程序中只有一个分号的语句，称为空语句，其形式如下：

;

空语句在语法上占据一个语句的位置，但是它不具备任何操作功能。

4. 复合语句

复合语句是为了满足将多条语句从语法上作为一条语句使用的需要而设计的。复合语句是用一对大括号“{}”将多条语句括起来构成的语句体。

复合语句的一般形式如下：

{ 语句组 }

5. 控制语句

控制语句是C程序设计中用来构成分支结构和循环结构，完成一定控制功能的语句。C语言有9种控制语句。

(1) 条件语句：if-else

(2) 多分支选择语句：switch

(3) 当型循环语句：while

(4) 直到型循环语句：do-while

(5) 当型循环语句：for

(6) 终止本次循环语句：continue

(7) 终止整个循环或跳出switch语句：break

(8) 无条件转移语句：goto

(9) 函数返回语句：return

其中continue、break和goto虽然属于流程控制语句，但是不能单独使用，只能与其他控制语句结合使用。

3.4 输入输出函数

设计程序时，首先要由问题所给予的已知条件和所要求的输出结果来决定输入和输出界面，接着再想出一个好的算法，处理如何由输入得到输出结果的过程。

C语言不提供输入输出语句，它的输入输出操作是由输入输出函数调用语句来实现的。在C的标准函数库中提供了多种输入输出函数，使其输入输出的形式多样、方便灵活。例如，printf函数和scanf函数，在使用它们时，千万不要简单地认为它们是C语言的“输入输出语句”。printf和scanf不是C语言的关键字。由于C语言提供的函数是以库的形式存放在系统中的，它们不是C语言文本中的组成部分，因此各函数的功能和名字在各种不同的编译系统中有所不同。不过，有些通用的函数(如printf和scanf等)，各种编译系统都提

供，成为各种编译系统的标准函数(标准输入输出库的一部分)。在使用标准输入输出库函数时，要用预编译命令 include 将 stdio.h 头文件包含到用户的源程序中。即：

```
#include "stdio.h"
```

stdio.h 是 standard input & output 的缩写，它包含了与标准输入输出库中有关的变量定义和宏定义。在需要使用标准输入输出库中的函数时，应在程序的前面使用上述预编译命令，程序在编译连接时，用户程序与标准文件相连。

常用的标准输入输出函数有两种：用于格式输入输出的函数(scanf/printf)，用于字符输入输出的函数(getchar/putchar)。

3.4.1　格式化输出函数 printf

1. printf 函数

printf 函数的调用形式：

```
printf("格式字符串",输出项表);
```

功能：按格式字符串中的格式依次输出输出项表中的各输出项。

说明：字符串是用双引号括起来的一串字符，如"study"。格式字符串用来说明输出项表中各输出项的输出格式。输出项表列出要输出的项(常量、变量或表达式)，各输出项之间用逗号分开。若没有输出项表，且格式字符串中不含格式信息，则输出的是格式字符串本身。因此实际调用时有两种格式。

格式 1：

```
printf("字符串");
```

功能：按原样输出字符串。

格式 2：

```
printf("格式字符串",输出项表);
```

功能：按格式字符串中的格式依次输出输出项表中的各输出项。

例如：

```
printf("Happy new year to you!\n");
```

输出：

```
Happy new year to you!              /*\n 表示换行.*/
```

又如：

```
printf("r = %d,s = %f/n",3,3*3*3.14);
```

格式说明　　输出表列

输出：r=2,s=28.260000。用格式%d 输出整数 3，用%f 输出 3*3*3.14 的值 28.26，%f 格式默认小数后 6 位，因此在 28.26 后补充了 4 个 0。"r="和"s="不是格式符，原样输出。

2. 格式字符串

格式字符串中有两类字符：

(1) 非格式字符。

非格式字符(或称普通字符)一律按原样输出。如上例的“r=”“s=”等。

(2) 格式字符。

格式字符的形式：

```
%[附加格式说明符]格式符
```

如%d、%.2f 等，其中%d 格式符表示用十进制整型格式输出，而%f 表示用实型格式输出，附加格式说明符“.2”表示输出 2 位小数。常用的格式符见表 3-1，常用的附加格式说明符见表 3-2。

表 3-1 格式符

格式字符	输出形式	举例	输出结果
d(或 i)	十进制整数	int a = 255; printf("%d",a);	255
x(或 X)	十六进制整数	int a = 255; printf("%x",a);	ff
o	八进制整数	int a = 65; printf("%o",a);	101
u	不带符号的十进制整数	int a = 65; printf("%u",a);	65
c	单个字符	int a = 65; printf("%c",a);	A
s	字符串	char st[ ] = "奥运 2008"; printf("%s", st);	奥运 2008
e(或 E)	指数形式的浮点数	float a = 123.5f; printf("%e", a);	1.235000e+002
f	小数形式的浮点数	float a = 123.5f; printf("%f", a);	123.500000
g(或 G)	e 和 f 较短的一种，不输出无效值	float a = 123.5f; printf("%g", a);	123.5
%	百分号本身	printf("%%");	%

表 3-2 附加格式说明符

附加格式说明符	功能
-	数据左对齐输出，无-时默认右对齐输出
m(m 为正整数)	数据输出宽度为 m，如数据宽度超过 m，按实际输出
.n(n 为正整数)	对实数，n 是输出的小数位数，对字符串，n 表示输出前 n 个字符
l	ld 输出 long 型数据，lf，le 输出 double 型数据
h	用于格式符 d、o、u、x 或 X，表示对应的输出项是短整型

(1) d 格式符。将输出数据视作整型数据，并以十进制形式输出。例如：

```
printf("%d,%+5d,%05d,%-5d,%5ld",123,123,123,123,123);
```

将输出：

```
123,␣+123,00123,123␣␣,␣␣123
```

说明：

␣——表示空格。

＋——正数要输出“＋”号，负数输出“－”号。

0——在“%05d”中，如果宽度不足5位，则在数前补0。

(2) o格式符。将输出数据视作无符号整型数据，并以八进制形式输出。由于将内存单元中的各位值(0或1)按八进制形式输出，输出的数值不带符号，符号位也一起作为八进制数的一部分输出。例如：

```
printf("%#o,%4o,%d,%4lo",045,045,045,-1L);
```

将输出：

```
045,␣␣45,37,37777777777
```

说明："#"表示输出八进制(或十六进制)常量的前缀。

(3) x(或X)格式符。将输出数据视作无符号整型数据，并以十六进制形式输出。与o格式符一样，符号位作为十六进制数的一部分输出。对于x和X分别用字符a、b、c、d、e、f和A、B、C、D、E、F表示9之后的6个十六进制数字。例如：

```
printf("%x,%4x,%6lX",045,045,-1L);
```

将输出：

```
0x25,␣␣25,FFFFFFFF
```

(4) u格式符。将输出数据视作无符号整型数据，以十进制形式输出，一个整型数据可以用d格式、o格式和x格式输出，也可以用u格式输出。反之亦然，值的类型转换按相互赋值规则处理。例如：

```
printf("%d,%u,%lu",-1,-1,-1L);
```

将输出：

```
-1,65535,4294967295
```

(5) c格式符。视输出数据为字符的ASCII码，输出一个字符。一个整数，只要它的值在0～255范围内，就可以用字符形式输出，输出以该整数为ASCII代码的字符。反之一个字符数据也可以用整数形式输出，输出该字符的ASCII代码值。例如：

```
printf("%d,%c,%d,%c",65,65,'A','A');
```

将输出：

```
65,A,65,A
```

(6) s格式符。用于输出一个字符串。例如：

```
printf("%4s,%5.3s,%-6.3s,%.4s","ABCDEF","ABCDEF","ABCDEF","ABCDEF");
```

将输出：

```
ABCDEF,␣␣ABC,ABC␣␣␣,ABCD
```

(7) f格式符。用于输出实型数据，并以“整数部分、小数部分”形式输出。小数点后的

数字个数为 n 个，n 的默认值为 6。若 n 为 0，不显示小数。格式转换时有四舍五入处理。例如：

```
printf("%f,%8.3f,%6.0f,%.1f",123456.78,123456.78,123456.78,123456.78);
```

将输出：

```
123456.780000,123456.780,123457,123456.8
```

(8) e(或 E)格式符。用于输出实型数据，并以指数形式输出。尾数 1 位整数，默认 6 位小数，指数至多 3 位。例如：

```
printf("%e,%8.3e,%6.0e,%.1e",123456.78,123456.78,123456.78,123456.78);
```

将输出：

```
1.234568e+05,1.235e+05,␣1e+05,1.2e+05
```

(9) g(或 G)格式符。用于输出实型数据。g 格式能根据表示数据所需字符的多少自动选择如 f 格式的形式或 e 格式的形式输出实数。选择是以输出时所需字符少为标准。选择这种输出形式时，附加格式说明符“#”的有无也对输出形式有影响。如“#”省略，则输出时小数部分无意义的 0 及小数点不输出；如有“#”，则无意义的 0 及小数点照常输出。如：

```
printf("%g,%#g,%g,,%#g",123456.78,123456.78,120000000.883,120000000.883);
```

将输出：

```
123457,123457.,1.2e+08,1.20000e+08
```

3.4.2 格式化输入函数 scanf

1. scanf 函数

与格式化输出函数 printf 相对应的是格式化输入函数 scanf。

scanf 函数的调用形式：

```
scanf("格式字符串",输入项地址表);
```

功能：按格式字符串中规定的格式，通过键盘输入各输入项的数据，并依次赋给各输入项。

说明：格式字符串与 printf 函数基本相同，需要特别注意的是，输入项以其地址的形式出现，而不是输入项的名称。如

```
scanf("%d,%f",&a,&b);
```

&a、&b 分别表示变量 a、b 的地址，其中 & 是取地址运算符(优先级及结合性与++相同)。若在键盘上输入“5,5.5”，则 5 赋给 a，5.5 赋给 b。

2. 格式字符串

scanf 函数中格式字符串的构成与 printf 函数基本相同，但使用时有不同之处。

（1）附加格式说明符 m 可以指定数据宽度，但不允许用附加格式说明符.n。例如，

```
scanf("%6.1f,%6f",&a,&b);          //其中%6.1f是错误的
```

（2）输入 long 型数据必须用%ld，l 输入 double 数据必须用%lf 或%le。而在 printf 函数中输出 double 型数据可以用%f 或%e。

（3）附加格式说明符“*”允许对应的输入数据不赋给相应变量。如

```
double x;int y;float z;
scanf("%f,%3d,%*d,%3f",&x,&y,&z);
```

在键盘上输入：

```
6.2,52,4562,1234.5↙(↙表示回车符)
```

输入后，x 的值为 0，y 的值为 52，z 的值为 123。x 的值不正确，原因是格式符用错了。x 是 double 型，所以输入 x 用%lf 或%le，用%f 是错误的；%*d 对应的数据是 4562，因此 4562 实际未赋给 z 变量，把 1234.5 按%3f 格式截取 123 赋给 z。

3. 关于输入方法

（1）普通字符按原样输入。

```
scanf("x=%d,y=%d",&x,&y);
```

若输入序列为：

```
1,2↙
```

则输出结果 x 的值为 0，y 的值不确定。

若输入序列为：

```
x=1,y=2↙
```

则输出结果 x 的值为 1，y 的值为 2。

（2）按格式截取输入数据。

```
scanf("%d,%3d",&a,&b);
```

若输入序列为：

```
12,1234.5↙
```

则 a=12，b=123，虽然输入的是 1234.5，但%3d 宽度为 3 位，截取前 3 位，即 123。

（3）输入数据的结束。

输入数据时，表示一个数据结束有下列 3 种情况：

- 第一个非空字符开始，遇空格、跳格(TAB 键)或回车符结束；
- 遇宽度结束；
- 遇非法输入结束。

3.4.3 字符输出函数 putchar

putchar 函数的作用是向标准输出设备输出一个字符。例如：

```
putchar(c);
```

这条语句的作用是向标准输出设备输出字符变量 c 的值。c 可以是字符型变量或整型变量。字符输出函数还可以输出控制字符，如 putchar('\n')，输出一个换行符。

【例 3.3】 理解下列程序，分析其运行结果。

```
#include <stdio.h>
void main( )
{
    char a,b,c;
    a = 'O'; b = 'K'; c = '\101';
    putchar(a); putchar(b);
    putchar('\n'); putchar(c);
   }
```

程序运行结果：

```
OK
A
```

注意：调用 putchar 函数时，必须用 #include "stdio.h"或 #include <stdio.h>编译预处理命令，将 stdio.h 文件包含到用户源文件中去。

3.4.4 字符输入函数 getchar

此函数的作用是从标准输入设备输入一个字符，该函数没有参数，其函数值就是从输入设备得到的字符。函数调用的一般形式为：

```
getchar();
```

【例 3.4】 输入并回显一个字符。

```
#include <stdio.h>
void main( )
{
    char c;
    c = getchar( );
    putchar(c);
   }
```

程序运行结果：

```
A↙
A
```

注意：getchar()只能接收一个字符。getchar 函数得到的字符可以赋给一个字符变量或整型变量，也可以不赋给任何变量，作为表达式的一部分。如上面这个程序的第 5、6 行可

以用下面一行代替：

```
putchar(getchar( ));
```

因为 getchar()的值为'A'，因此输出'A'。也可以用在 printf 函数中：

```
printf("%c", getchar());
```

【例 3.5】 输入两个字符并回显这两个字符。

```
#include <stdio.h>
void main( )
{
    char a,b;
    a = getchar( );
    b = getchar( );
    putchar(a);
    putchar(b);
  }
```

程序运行结果：

```
XY↙
XY
```

输入两个字符 XY 后，按回车键，它们才被送到内存标准输入缓冲文件中，标准输入函数实际上是从内存标准输入缓冲文件中读取数据。

注意，不能按如下形式输入：

```
X↙
Y↙
```

如果输入 X↙，则第一个 getchar 输入的是'X'赋给 a，第二个 getchar 输入的是'\n'(即换行符)赋给 b，也就不会再要求继续输入 Y↙。此时的输出结果为：

```
X
```

注意：调用 getchar 函数时，必须用#include "stdio.h"或#include <stdio.h>编译预处理命令，将 stdio.h 文件包含到用户源文件中去。

3.4.5　getche()函数和 getch()函数

getche()函数和 getch()函数直接从键盘读数据，而不通过输入缓冲区，每次读一个字符，不需要按回车键就能读输入的字符，使用格式为：

```
ch = getche( );
ch = getch( );
```

注意：

(1) getche()函数和 getch()函数不是标准输入函数，而是键盘输入函数，需要包含头文件 conio.h。

（2）由于getch()函数不回显输入的字符，因而使用getch()函数时，输入的字符看不见。

【例3.6】 输入一串字符，以'0'结束，统计输入的字符数。不包括'0'。

```
#include <stdio.h>
#include <conio.h>
void main( )
{
    char ch = 'a';
    int n = 0;
    while(ch != '0')
    {
        ch = getche( );
            n++;
    }
    printf("\nn = %d\n", n - 1);
}
```

输入及程序运行过程：

```
123450
n = 5
```

说明：

（1）输入时没有按回车键，getche()函数直接从键盘的输入获得字符。

（2）while是循环语句，只要条件(ch! ='0')为真，循环体就会执行，直到用户输入字符'0'，条件为假，才结束。

```
{
    ch = getche( );
    n++;
}
```

如果把getche()函数换成getch()函数会是什么结果？

3.5 良好结构的程序

一个程序应该层次分明，具有必要的注释，才便于理解和查找错误。从逻辑上可以把一个程序的语句分成若干级别，在书写程序的时候，同级别的语句要按列对齐，下一级别的语句要右退若干列并对齐。这样写出的程序称为良好结构的程序，如例3.7所示。一个程序如果没有层次，就变成了不良结构的程序，如例3.8所示，这样的程序不便于理解，也不便于查找程序的错误。写程序者应避免编写不良结构的程序。

【例3.7】 求1+3+…+15及2×4×6×…×16。

```
#include <stdio.h>
void main( )
{
     int i, s, t;
```

```
    s = 0;                          //累加变量 s 初始化为 0
    t = 1;                          //累乘变量 t 初始化为 1
    for(i = 1; i <= 16; i++)
    {
        if(i % 2 == 1)              //如果是奇数则累加
         {
            s += i ;
         }
         if( i % 2 == 0)            //如果是偶数则累乘
         {
            t * = i;
         }
    }
printf("s = %d,t = %d\n", s, t);  //输出结果
}
```

【例 3.8】 不良结构的程序。

```
void main( )
{
int i, s, t;
s = 0;                              //累加变量 s 初始化为 0
t = 1;                              //累乘变量 t 初始化为 1
for(i = 1; i <= 16; i++)
{
if(i % 2 == 1)                      //如果是奇数则累加
{
s += i ;
}
if( i % 2 == 0)                     //如果是偶数则累乘
{
t * = i;
}
}
printf("s = %d,t = %d\n", s, t);   //输出结果
}
```

说明：在 VC++ 6.0 编程环境中，可按 Ctrl＋A 组合键选中所有程序行，然后按 Alt＋F8 组合键来调整结构。

3.6　顺序结构程序设计举例

【例 3.9】 已知圆的半径为 2，编程计算圆的周长和圆的面积。

算法：

(1) 说明实型变量 r 为半径，l 为圆周长，s 为圆面积；

(2) 调用格式输入函数输入半径 r；

(3) 分别利用公式：$l=2\pi r$，$s=\pi r^2$ 计算；

(4) 调用格式输出函数输出结果。

程序：

```
#include <stdio.h>
void main( )
{
   float pi,r,l,s;
   pi = 3.14159;
   printf("Please input radius: \n");                    /* 输入提示 */
   scanf("%f",&r);                                       /* 从键盘上输入半径 2,按回车键 */
   l = 2 * pi * r;
   s = pi * r * r;
   printf("The circle length: l = %.2f\n",l);            /* 输出圆的周长 */
   printf("The circle area: s = %.2f\n",s);              /* 输出圆的面积 */
   }
```

程序运行结果：

```
Please input radius:
3 ↙
The circle length: l = 18.85
The circle area: s = 28.27
```

【例 3.10】 从键盘输入一个大写字母，要求输出小写字母及对应的 ASCII 码值。

程序：

```
#include <stdio.h>
void main( )
{
    char c1,c2;
    c1 = getchar( );
    c2 = c1 + 32;
    printf("\n%c, %d\n",c1,c1);
    printf("%c, %d\n",c2,c2);
  }
```

程序运行结果：

```
B ↙
B,66
b,98
```

分析：

getchar 函数得到从键盘输入的大写字母“B”，赋给字符变量 c1。经过运算得到小写字母“b”，赋给字符变量 c2。将 c1、c2 分别用字符形式和整数形式输出。

【例 3.11】 根据三角形的 3 条边长，求面积。

设三角形 3 条边长为 a、b、c，则三角形面积公式：

$$p = \frac{a+b+c}{2}$$

$$s = \sqrt{p(p-a)(p-b)(p-c)}$$

算法：

(1) 定义实型变量 a、b、c、p、s；

(2) 输入 a、b、c 的值；
(3) 根据公式计算 p 的值；
(4) 根据公式计算 s 的值；
(5) 输出三角形的面积。
提示：C 程序中求平方根，需调用函数 sqrt。
程序：

```
#include <math.h>
#include <stdio.h>
void main( )
{
   float a,b,c,p,s;
   printf("Please input a b c: ");                    /* 输入提示 */
   scanf("%f%f%f",&a,&b,&c);
   p=(a+b+c)/2;
   s=sqrt(p*(p-a)*(p-b)*(p-c));                       /* 调用数学函数计算面积 s */
   printf("a=%.2f,b=%.2f,c=%.2f\n",a,b,c);
   printf("s=%.2f\n",s);
}
```

程序运行结果：

```
3          4     5
a=3.00,b=4.00,c=5.00
s=6.00
```

习题 3

一、请写出下面程序的输出结果。

```
#include <stdio.h>
void main( )
{     int a=5,b=7;
      float x=67.8564,y=-789.124;
      char c='A';
      long n=1234567;
      unsigned u=65535;
      printf("%d%d\n",a,b);
      printf("%3d%3d\n",a,b);
      printf("%f,%f\n",x,y);
      printf("%-10f,%-10f\n",x,y);
      printf("%8.2f,%8.2f,%.4f,%.4f,%3f,%3f\n",x,y,x,y,x,y);
      printf("%e,%10.2e\n",x,y);
      printf("%c,%d,%o,%x\n",c,c,c,c);
      printf("%ld,%lo,%x\n",n,n,n);
      printf("%u,%o,%x,%d\n",u,u,u,u);
      printf("%s,%5.3s\n","COMPUTER","COMPUTER");
}
```

二、下面的 scanf 函数输入数据，使 a＝3，b＝7，x＝8.5，y＝71.82，c1＝'A'，c2＝'a'，在键盘上如何输入？

```
#include <stdio.h>
void main( )
{       int a,b;
        float x,y;
        char c1,c2;
        scanf("a = %d b = %d",&a,&b);
        scanf(" %f %e",&x,&y);
        scanf(" %c %c",&c1,&c2);
        printf("a = %d b = %d x = %f y = %e c1 = %c c2 = %c",a,b,x,y,c1,c2);
}
```

三、编程

1. 用下面的 scanf 函数输入数据，使 a＝20，b＝30，c1＝'B'，c2＝'d'，x＝2.5，y＝－6.66，z＝12.8，在键盘上如何输入？

```
scanf("%5d%5d%c%c%f%f% *f,%f", &a,&b,&c1,&c2,&x,&y,&z);
```

2. 编写程序，任意从键盘中输入 4 个整数，求出这 4 个数的平均值。

3. 试编写一个程序，任意输入一个小写字母，分别按八进制、十进制、十六进制、字符格式输出。

4. 输入一个华氏温度，要求输出摄氏温度。公式为：

$$c = \frac{5}{9}(F - 32)$$

输出要有文字说明，取 2 位小数。

5. 设圆半径 r＝3，圆柱高 h＝4，求圆周长、圆面积、圆球表面积、圆球体积、圆柱体积。用 scanf 输入数据，输出计算结果，输出时要求有文字说明，取小数点后 2 位数字。请编程实现。