C 语言程序设计

第 3 单元　顺序结构程序设计

Contents

任务 4　简单加密

任务描述

本任务设计完成一个简单密码加密软件。程序运行后，按照提示输入 8 位字母密码，然后进行简单加密操作，并输出加密后的密码。加密方法为，将采用从键盘输入字母后的第 2 个字母来替代原字母实现加密，例如，输入 a ，则加密后变为 c 。

知识准备

一、结构化程序设计基础

1 ．程序构成

一个程序应包括以下两方面内容：

（ 1 ）对数据的描述。在程序中要指定数据的类型和数据的组织形式，即数据结构（ data structure ）。

（ 2 ）对操作的描述。即操作步骤，也就是算法（ algorithm ）。

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操作步骤（即算法）。因此，著名计算机科学家沃思（ Nikiklaus Wirth ）提出一个公式：

数据结构 + 算法 = 程序

实际上，一个程序除了以上两个主要要素外，还应当采用结构化程序设计方法进行程序设计，并且用某一种计算机语言表示。因此，可以这样表示：

程序 = 算法 + 数据结构 + 程序设计方法 + 语言工具和环境

上述四方面是一个程序设计人员所应具备的知识，在设计一个程序时要综合运用这几方面的知识。

知识准备

一、结构化程序设计基础

2 ．算法

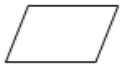
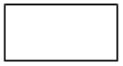
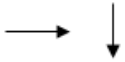
做任何事情都有一定的步骤。在日常生活中，由于人们已养成习惯，所以并没有意识到每件事都需要事先设计出“行动步骤”，如吃饭、上学、打球、做作业等，但事实上，这些活动都是按照一定的规律进行的，只是人们不必每次都重复考虑它。

不要认为只有“计算”的问题才有算法。广义地说，为解决一个问题而采取的方法和步骤统称为“算法”。一首歌曲的乐谱，也可以称为该歌曲的算法，因为它指定了演奏该歌曲的每个步骤，按照它的规定才能演奏出预定的曲子。

知识准备

一、结构化程序设计基础

流程图是用一些图框来表示各种操作。用图形表示算法，直观形象，易于理解。美国国家标准化协会 ANSI 规定了一些常用的流程图符号（如表 3-1 所示），已为世界各国程序工作者普遍使用。

符　号	名　称	说　明
	起止框	表示算法的开始和结束
	输入输出框	表示完成某种操作，如初始化
	判断框	表示根据条件成立与否，决定执行哪种操作
	处理框	表示数据的输入输出操作
	流程线	表示程序执行的流向
	连接点	用于流程分支的连接

说明：一个流程图应该包括以下几个部分

（ 1 ）表示相应操作的框

（ 2 ）带箭头的流程线

（ 3 ）框内外必要的文字说明

知识准备

一、结构化程序设计基础

4 ．结构化程序设计方法

一个结构化的程序就是用高级语言表示的结构化算法。这种程序便于编写、阅读、修改和维护，可以减少程序出错的机会，提高程序的可靠性，保证程序的质量。

结构化程序设计强调程序设计风格和程序结构的规范化，提倡清晰的结构。怎样才能得到一个结构化的程序呢？如果我们面临一个复杂的问题，是难以一下子写出一个层次分明、结构清晰、算法正确的程序的。结构化程序设计方法的基本思路是，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。

二、字符输入、输出函数

1．字符输出函数

（1）标准字符输出函数 putchar() 的一般形式：

putchar(ch)

（2）putchar() 函数的作用：向终端输出一个字符。

（3）参数 ch 通常为字符型变量、整型常量、字符本身，也可以输出控制字符，如 putchar('\n') 输出一个换行符，其函数类型是整型。

（4）putchar() 是标准 I/O 库中的函数，在使用时应在程序前加上预编译命令 #include <stdio.h> 。

知识准备

二、字符输入、输出函数

2 ．字符输入函数

（ 1 ）getchar() 函数没有参数，其一般调用形式为：

getchar()

（ 2 ）getchar() 函数：作用是从终端上输入一个字符，返回值为一个整型数，即被输入字符的 ASCII 码值。

（ 3 ）getchar() 与 putchar() 一样，是标准 I/O 库中的函数，在使用时应在程序前加上预编译命令 #include <stdio.h> 。

（ 4 ）getchar() 只能接收一个字符，该函数得到的字符可以赋给一个字符型变量或整型变量，也可作为表达式的一部分不赋给任何变量。一般先定义一个字符类型的变量，然后再引用 getchar() 函数，并将函数值赋给这个字符型变量。

（ 5 ）getchar() 与后面学到的循环结构配合使用，可以连续输入任何字符，而且输入的多个字符都能被接收。原因是输入的多个字符以行为单位进行处理，即输入的字符先放入内存缓冲区中，待需要时再逐个取出。

知识应用

一、字符输出函数

1．使用 putchar() 函数输出

程序如下：

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

运行结果：

```
g
o
o
d
```

说明：用八个 putchar() 函数输出字符，其中有四条语句用于输出控制字符 putchar('\n')，即换行，因此 good 一词纵向排列

知识应用

二、字符输入函数

１．使用 getchar() 函数接收任意字符，并输出

程序如下：

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

   char  c;       /* 定义字符型变量 c*/
   c=getchar( );   /* 接收一个字符赋值给变量 c*/
putchar(c);     /* 输出字符型变量 c*/
}
```

若输入： k ↙

则运行结果：

k

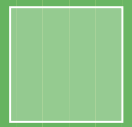
若输入： F↙

则运行结果：

F

说明：本程序中用 getchar() 接收任意一个字符，用 putchar() 将其输出。

任务实施



一、任务流程分解

1．变量定义分析：本任务共定义 8 个字符变量，分别为 ch1 、 ch2 、 ch3 、 ch4 、 ch5 、 ch6 、 ch7 、 ch8 ，分别对应 8 位字母密码。

2．变量赋值分析： ch1 、 ch2 、 ch3 、 ch4 、 ch5 、 ch6 、 ch7 、 ch8 都用 getchar() 函数赋值。录入的 8 个字母就是原密码，要求字符只能是 A-X 或者 a-x 。

3．加密过程分析：通过 getchar() 函数给 8 个变量赋值后，把每个变量的 ASCII 值加 2 ，以便完成加密。

4．密文输出分析：加密后的 8 位密码用 8 个 putchar() 函数在同一行上输出。

任务实施

二、代码实现

```
#include <stdio.h>
main()
{
        char
ch1,ch2,ch3,ch4,ch5,ch6,ch7,ch8;
        printf(" 请输入 8 个字母的密码（字母
范围 a-x 或 A-X):\n");
        printf(" 原密码 :");
        ch1=getchar();    /* 以下 8 行语句用
getchar() 输入密码 */
        ch2=getchar();
        ch3=getchar();
        ch4=getchar();
        ch5=getchar();
        ch6=getchar();
        ch7=getchar();
        ch8=getchar();
        printf(" 开始加密 .....\n");
        ch1+=2;  /* 以下 8 行是对输入的密码加
密 */
ch2+=2;
        ch3+=2;
        ch4+=2;
        ch5+=2;
        ch6+=2;
        ch7+=2;
        ch8+=2;
    printf(" 加密结果如下 ： \n");
        putchar(ch1);  /* 以下 8 行是
用 putchar() 输出加密后的密码 */
        putchar(ch2);
        putchar(ch3);
        putchar(ch4);
        putchar(ch5);
        putchar(ch6);
        putchar(ch7);
        putchar(ch8);
    putchar('\n');
}
```

任务实施

三、结果演示

```
请输入8个字母的密码（字母范围a-x或A-X）:
原密码:ABCDabcd
开始加密.....
加密结果如下:
CDEFcdef
Press any key to continue
```

Contents

任务 5　数学公式

任务描述

本任务设计完成一个一元二次方程求根软件。程序运行后，按照提示输入一元二次方程二次项系数、一次项系数、常数项，要求输入的二次项系数、一次项系数、常数项构成的一元二次方程必须有根，然后进行求根计算，最后输出此一元二次方程的根。

知识准备

一、格式输出函数的使用

1 ．格式输出函数

printf() 函数的一般格式： printf(格式控制 , 输出表列)

功能：向终端输出若干任意类型的数据。

说明：

（ 1 ）格式控制：是用双引号括起来的字符串，又称“转换控制字符串” ，它规定了输出表列中各项的输出形式。它包括三种信息：

① 格式转换控制符：可将输出的数据转换为指定的格式输出。由 “ %” 和格式字符组成，例如， %d 、 %c 等，但 “ %” 与格式字符之间不能留有空格。

② 转义字符：输出一些操作行为。例如， \n 、 \t 等。

③ 提示串：是除了格式转换控制符和转义字符之外的其他字符，这些字符可原样输出。例如， printf("a 为字符 %c,b 为字符 %c\n",a,b) ；语句中 “ a 为字符” 和 “ b 为字符” 都属于提示串，输出时原样输出。

（ 2 ）输出参数是需要输出的一批数据，可以是变量或表达式表列，输出参数的个数必须与控制参数中的格式转换控制符个数相同。

知识准备

一、格式输出函数的使用

2 ．格式字符功能及其用法

（ 1 ）d 格式符。以十进制数形式输出整数。有以下几种用法：

① %d ，按整型数据的实际长度输出。例如：

printf("%d,%d",a,b);

若 a=123,d=12345 ，则输出结果：

123,12345

② %md ， m 为指定的输出字符的宽度。如果输出数据的实际位数小于 m ，则左端补以空格，若大于 m ，则按实际位数输出。例如：

printf("%4d,%4d",a,b);

若 a=123,d=12345 ，则输出结果：

␣123,12345

③ %ld ，用于输出长整型数据。例如：

long a=135790;

printf("%ld",a);

知识准备

一、格式输出函数的使用

（ 2 ）o 格式符，以八进制数形式输出整数。由于是将内存单元中各位的值（ 0 或 1 ）按八进制形式输出，即符号位也作为八进制的一部分输出，因此不会输出带负号的形式。长整型数据也可以用 "%lo" 格式输出。 "%mo" 表示按指定宽度输出八进制整数。

（ 3 ）x 格式符，以无符号十六进制数形式输出整数。 %lx 输出长整型数， %mlx 输出指定宽度的十六进制整数。

（ 4 ）u 格式符，用于输出 unsigned 型数据，以无符号十进制形式输出。

一个有符号整数可以用 %u 格式输出，相反， unsigned 型数据也可以用 %d 、 %o 、 %x 格式输出。按相互赋值的规则处理，即将非 unsigned 型数据赋给长度相同的 unsigned 型变量，原则是原样照赋（原有的符号位也作为数据的一部分一起传送）；将一个 unsigned 类型数据赋给一个占字节数相同的整型变量，将 unsigned 型变量的内容原样送到非 unsigned 型变量中，但如果数据范围超过相应整型的范围，则会进行相应的转化。

（ 5 ）c 格式符，用于输出一个字符。在 C 语言中，字符型数据和整型数据之间可以通用。对于整数，只要它的值在 0 ~ 255 范围内，也可以用字符形式输出；同样，一个字符数据也可以转成相应的整型数据，即以 ASCII 码值输出。用 %mc 输出指定宽度的字符型数据。若指定的宽度大于实际宽度，则左端补空格。

知识准备

一、格式输出函数的使用

（ 6 ）s 格式符，用于输出一个字符串。 %s 按实际长度输出字符串。 %ms 输出指定宽度为 m 的字符串，若实际字符串长度小于 m ，则左端补足空格；若实际串长大于 m ，则按实际字符串长度输出该字符串。也可用 %-ms ，当实际串长小于 m 时，字符串左对齐，右端补相应的空格。 %m.ns 输出指定宽度为 m 的、从字符串左端取出的 n 个字符（ n 代表一个正整数）。若 n 小于 m 则左补足空格，若 n 大于 m 则以 n 为主，输出 n 个字符。 %-m.ns 同 %m.ns 类似，不同的是，当 n 小于 m 时右端补足空格。 %.n 只指定了 n 未指定 m ，此时自动使 m 等于 n ，即输出占 n 列，只取字符串左端的 n 个字符。

（ 7 ）f 格式符，用于输出实型数据，以小数形式输出单精度和双精度型数据。 %f 按系统规定的格式输出，即整数部分全部输出，小数部分取 6 位。不要以为所有打印出来的数字都是准确的，在一般系统下，单精度实数的有效位数为 7 位，双精度实数的有效位数为 15 位（不同的系统在实现格式输出时，输出结果可能会有一些小的差别）。 %m.nf 输出指定宽度为 m 列保留 n 位小数的实数，在 m 列中，小数点也占一位宽度。若输出数据实际长度小于 m ，则左端补空格，数字右对齐。 %-m.nf 与 %m.nf 类似，不同的是，若输出数据实际长度小于 m ，则右端补空格，数字左对齐（当格式符为 %0m.n 时，所空的格以 0 填充）；若实际长度大于 m ，则按实际长度输出，并保留 n 位小数。 %.nf 也是按实际长度输出，并保留 n 位小数。

知识准备

格式字符	说　明
d	以带符号的十进制形式输出整数（正数不输出符号）
o	以八进制无符号形式输出整数（不输出前导 0 ）
x	以十六进制无符号形式输出整数（不输出前导 0x ）
u	以无符号十进制形式输出整数
c	以字符形式输出，只输出一个字符
s	输出字符串
f	以小数形式输出单、双精度数，隐含输出 6 位小数
e	以指数形式输出实数
g	选用 f% 或 e% 格式中输出宽度较短的一种格式，不输出无意义的 0

字 符	说　明
l	用于长整型整数，可加在格式符 d 、 o 、 x 、 u 前面
m （整数）	数据最小宽度
n （整数）	对实数，表示输出 n 位小数；对字符串，表示截取的字符个数
—	输出的数字或字符在域内向左靠

知识准备

一、格式输入函数的使用

1．格式输入函数 scanf()

scanf() 函数的一般格式如下：

scanf(控制参数，地址表列)

功能：用来输入任何类型的多个数据。

说明：

（ 1 ）控制参数的含义与 printf() 函数中的控制参数的含义相同。

（ 2 ）在给多个输入项输入数据时，在控制参数中，若 % 格式字符与 % 格式字符之间没有其他字符，输入数据时，两个数据之间可以用一个或多个空格间隔，也可以用回车符、 Tab 符间隔。（ %c 输入格式除外 ）

例如，

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

以下的三种输入方法均是合法的：

① 10␣␣20↙

② 10↙

20↙

③ 10（按【 Tab 】键）20↙

（ 3 ）格式转换控制符之间有非格式字符，则把非格式字符当成普通字符，原样输入，当该字符是一个空格时，输入一个或多个空格均合法。

例如，

scanf("i=%d,a=%d",&i,&a) ；

若输入 i=45,a=67 是合法的，若输入 45␣67 则不合法。

（ 4 ）地址表列是由若干个地址组成的表列，可以是变量的地址或字符串的首地址。例如， scanf("%d %d",&a,&b) 中 &a 和 &b 就是变量在内存中的地址。 & 是地址运算符。初学者在使用该语句时，经常丢掉地址运算符 & ，应特别注意。

知识准备

一、格式输入函数的使用

2 ．格式字符的用法

（ 1 ）d 格式符，用于输入十进制整数。

（ 2 ）o 格式符，用于输入八进制整数。

（ 3 ）x 格式符，用于输入十六进制整数。

（ 4 ）c 格式符，用于输入单个字符。在用 %c 格式输入字符时，空格字符和转义字符都作为有效字符输入。

（ 5 ）s 格式符，用于输出字符串。

（ 6 ）f 格式符，用于输入实数，可以用小数形式或指数形式输入。

（ 7 ）e 格式符，与 f 格式符的作用相同，均可用来输入实型数据，输入时既可以用小数形式也可以用指数形式输入， e 与 f 可以互相替换。

知识应用

一、格式输出函数的应用

1 ． d 格式符的应用

程序如下：

```
#include <stdio.h>
main ()
{
   int a,b;
   long c,d;
   a=32767;
   b=1;
   c=2147483647;
   d=1;
   printf("%d,%d\n",a,b);
   printf("%3d,%3d\n",a,b);
   printf("%ld,%ld\n",c,d);
   printf("%10ld,%10ld\n",c,d);
}
```

运行结果：

```
32767,1
32767,  1
2147483647,1
2147483647,         1
```

说明：定义 4 个变量并分别进行赋初值。采用了 d 格式符的 %d 、 %3d 、 %ld 和 %10ld 进行输出。其中， %d 是采用按整型数据的实际长度输出； %3d 输出 a 时，列宽不够，按实际长度输出； %ld 用于输出长整型数据，也可限定列宽。

知识应用

一、格式输出函数的应用

2 、 c 格式符的应用

程序如下：

```
#include <stdio.h>
main()
{
   char c='A';                                          /* 定义字符型变量并
赋初值 */
   int i=65;                                            /* 定义整型变量并赋初值 */
   printf("%c,%d\n",c,c);                    /* 以两种格式控制符输出变量 c*/
   printf("%c,%d\n",i,i);                    /* 以两种格式控制符输出变量 i*/
}
```

运行结果：

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

说明：整型变量 i 的值为 65 ，在 0 ~ 255 之内，以 %c 的形式输出，是输出 65 所对应的字符 A ，字符型变量以 %c 格式输出，是输出字符数据的 ASCII 码值。

知识应用

一、格式输出函数的应用

3 ．f 格式符的应用

程序如下：

```
#include <stdio.h>
main()
{
   float f=123.456;
   printf("%f_ _ %10f_ _ %10.2f_ _ %.2f_ _ %-10.2f\n",f,f,f,f,f);
}
```

运行结果： /*Turbo C 程序中的运行结果 */

123.456001_ _ 123.456001_ _ _ _ _ _ 123.46_ _ 123.46_ _ 123.46_ _ _ _

说明： %f 以小数形式输出实数，小数点后带 6 位小数，小数点本身占一位； %10f 指定列宽为 10 列， 123.456 再加上三位小数恰巧占列宽为 10 ； %10.2f 指定列宽为 10 列，但要取小数点后两位，加上小数和整数部分，一共是 6 列，因此左端补 4 个空格； %.2f 相当于 %2.2f 即 m=n 情况，指定输出共占 2 列，但要取小数点后两位，加上小数点和整数部分，共 6 列，因此这 6 列冲破 2 列的限定，原样输出； %-10.2f 与 %10.2f 类似，在右端补 4 个空格。

运行结果： /*V C 程序中的运行结果 */

123.456001VV123.456001VV_ _ _ _ 123.46VV123.46VV123.46

知识应用

二、格式输入函数的应用

1 ．输入两个十进制整数，求其和并输出

程序如下：

```
#include <stdio.h>
main()
{
   int a,b;                /* 定义两个整型变量 a 和 b*/
   scanf("%d,%d",&a,&b);  /* 从键盘读入两个整型数据 */
   printf("a+b=%d",a+b);   /* 输出二者的和 */
}
```

输入：

15,30↙

运行结果：

a+b=45

说明：输入格式中两个 %d 之间有逗号间隔，因此在输入两个数据时，中间也必须输入一个逗号作为间隔，否则接收数据时会出错。

知识应用

二、格式输入函数的应用

2、输入一个实型数据，输出它的平方值

程序如下：

```
#include <stdio.h>
main()
{
   float  a,s;
   scanf("%f",&a);     /* 输入时不能指定精度，如 %5.2f 是错误的 */
   s=a*a;              /* 求其平方 */
   printf(" 输入的数据 %.2f 的平方为 %.2f\n",a,s);
}
```

若输入：

5.0↙ 或 5 ↙

运行结果：

输入的数据 5.00 的平方为 25.00

说明：本程序由键盘随机输入一个实型数据 5.0 赋值给单精度实型变量 a ，求其平方后输出。需注意的是：① 在输入实型数据时，可输入 5.0 ，也可直接输入 5 ，系统会自动对其进行转换。② 用 %f 进行输入时，不可指定输入数据的精度，而输出时可指定输出数据的精度。

任务实施

一、任务流程分解

1 ．变量定义分析：本任务共定义 5 个实型变量，分别为 a 、 b 、 c 、 x1 、 x2, 功能分别为二次项系数、一次项系数、常数项、第一个根、第二个根。

2 ．变量赋值分析： a 、 b 、 c 用 scanf() 函数赋值。一元二次方程的根有三种情况，我们这个程序只能求解有根的两种情况，无根情况无法求解，所以在输入 a 、 b 、 c 值的时候，一定要保证 b^2-4ac>=0 ，否则程序无法正确求解一元二次方程的根。

3 ．输出结果分析：通过使用 sqrt() 数学函数，求解一元二次方程的根，并输出一元二次方程的根，输出时保留两位小数。

任务实施

二、代码实现

```
#include <stdio.h>
#include <math.h>
main()
{
        float a,b,c,x1,x2;
   printf("**********************************\n");
        printf("    一元二次方程求解程序 \n");
        printf("**********************************\n");
        printf(" 请输入一元二次方程各项系数 ( a,b,c) ： \n");
        scanf("%f,%f,%f",&a,&b,&c);
        printf("--------------------------------\n");
        x1=(-b+sqrt(b*b-4*a*c))/(2*a);   /* 利用一元二次方程求根公式求解 */
   x2=(-b-sqrt(b*b-4*a*c))/(2*a);
printf("%.2fx^2+%.2fx+%.2f=0 的解为 ： \n",a,b,c);
   printf("--------------------------------\n");
        printf("x1=%.2f,x2=%.2f\n",x1,x2);  /* 输出时保留两位小数 */
        printf("**********************************\n");
}
```

任务实施

三、结果演示

1 ．有不相等的两个实根
程序执行时，如果输入 1.0,5.0,2.0 时，此一元二次方程的根为 x1= -0.44,x2= -4.56 。如图所示。说明： sqrt() 函数返回值为实型，所以此一元二次方程的根为近似值

```
**********************************
    一元二次方程求解程序
**********************************
请输入一元二次方程各项系数（a,b,c):
1.0,5.0,2.0
----------------------------------
1.00x^2+5.00x+2.00=0的解为：
----------------------------------
x1=-0.44,x2=-4.56
**********************************
Press any key to continue
```

2 ．有相等的两个实根
程序执行时，如果输入 1.0,2.0,1.0 时，此一元二次方程的根为 x1= -1.00,x2= -1.00 。如图所示。

```
**********************************
    一元二次方程求解程序
**********************************
请输入一元二次方程各项系数（a,b,c):
1.0,2.0,1.0
----------------------------------
1.00x^2+2.00x+1.00=0的解为：
----------------------------------
x1=-1.00,x2=-1.00
**********************************
Press any key to continue
```

总结

本章通过两个任务，讲解了 putchar( )、 getchar( ) 、 printf( )、 scanf( ) 四个输入 / 输出函数的使用，结构化程序设计的方法及顺序结构程序设计。在编写顺序结构程序时，要注意以下几点：

1 ．顺序结构设计的思路要清楚，执行语句的先后逻辑次序、条理要清晰。

2 ．表达式与计算公式一致。

3 ．输入 / 输出的格式说明符与输入 / 输出变量的类型一致。

尽管顺序结构程序设计较简单，但刚开始学习时应该注意培养良好的程序设计习惯及代码编写的规范。首先分析所给的问题，明确要求，找出解决问题的途径（即算法）；然后安排分配合适的变量，再一步一步地写出处理步骤；最后输出结果。算法设计是自顶向下进行的，复杂的算法要逐步求精。

C 语言程序设计

Thank You!