第一章 C语言简介

C语言是1972年由AT&T贝尔实验室(Bell Laboratories)的工程师Dennis Rithchie为了在PDP-11机器上开发UNIX操作系统而创作设计出的一种程序语言。早期在PDP机器上的UNIX操作系统是用汇编语言写的。用汇编语言撰写操作系统常会因为汇编语言的局限性(不同机器使用不同汇编语言)、而限制了操作系统在不同机器上的流通。这也就是Dennis Ritchie开发C语言的动机。最明显的例子是UNIX操作系统，它几乎全是用C语言撰写的；在全部系统程序的13000行中，只有关于向量表、工作环境相关函数、内存管理等最低级部分的800行是用汇编语言写成的。

由于C语言不但在操作系统程序设计上会有杰出的表现，而且亦能用来作一般数据处理，科学计算，所以自从IBM PC出现之后，短短几年间，C语言的编译程序，就如雨后春笋般出现。如Turbo C、Quick C和Microsoft C等。由于C语言不但具有低级汇编语言的功能，同时，又拥有高级语言的通用性，所以它在计算机语言世界中的地位止日趋重要。这一点可以从大型计算机也都纷纷宣称它们可以执行UNIX，并具有C语言编译器等这些事实得到证明。

1.1 C语言程序的结构

本节通过一个简单的C语言程序，介绍C语言程序的基本结构。等读者了解程序的基本组织结构后，我们才接着继续以实例说明C语言程序的语法。

1. 程序范例

```
#include <stdio.h>    /*program header */

main()              /* main function declaration */
{
   int i, j, k ;  /* local variable declaration */
    i = 1;         /* assignment of vairable */


    j = 2;
    k = 3;
 /* output of 3 variable2 */
    printf("variable i = %d\n", i);
    printf("variable j = %d\n", j);
    printf("variable k = %d\n", k);
 }
```

上面是一个简易的C语言原始程序，它只用来在屏幕上打印出三个变量值1,2,3而已。有关程序中变量声明、设置，以及输出等各项细节，以后会慢慢说明。当前的重点是让

读者了解整个原始C语言程序的组织结构而已。

2. 包含文件

因为C语言没有提供任何输出的命令（如READ，WRITE等语句），所有这些高层次的功能都是通过调用函数来完成的。例如，范例中第9，10，11行中所用的printf（）函数（用来输出）是被定义在stdio.h（Standard I/O，标准输入及输出）包含文件内。所以，必须在程序一开头第一行加上#include指令，加下所示：

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

有了行语句，程序在被编译（Compile）之前，会由C的预处理器（Preprocessor）先行处理。Turbo C所使用的包含文件就包括在它的include文件内，读者可自己调出来参阅。

3. 主程序声明

C语言程序最少要由一个函数（function）组成，这个函数名称为main，它是程序的主体。main函数的基本结构是：

```
main ()
{
  : 程序主体
  :
}
```

左大括号“{”表示函数主体的开始，右大括号“}”，表示函数主体的结束。它类似于PASCAL语言程序主体的BEGIN和END。C语言程序主体中的每一个语句都必须用分号（;）作为结束。

4. 变量声明

C语言程序中所有变量，使用前一定要先声明其变量的数据类型，以便程序编译器为此变量安排一个适当的存储空间，供程序使用。范例程序中第4行：

```
int i, j, k;
```

就是用来声明i，j，k，为整数类型（int）。第5，6，7行：

```
i=1;
i=2;
i=3;
```

用来设置初值给这3个变量。有关更详细的变量声明，留待下一章再行讨论。

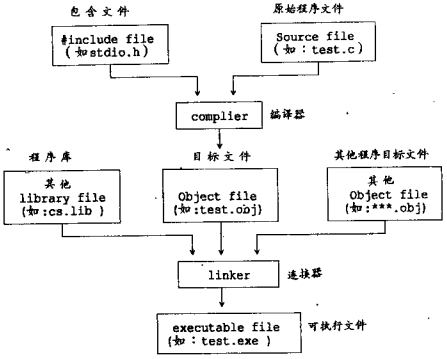
5. 注解

在程序中加注解以方便日后阅读，是设计程序的一种好习惯。C语言程序所用的注解符号是“/*”和“*/”，凡是介于“*/”和“*/”间的文字、符号，编译器都会略过不予编译。

1.2 程序编辑、编译和连结

设计一个C语言程序的步骤是：编辑（Edit）原始程序文件，编译（Compile）原始程序文件成为目的文件（Object）最后使用连结器（linker）将此目的文件所用到程序库（library），

甚至其他目的文件连结成一个可执行的执行文件(EXE 文件)。下面是 C 语言程序建立过程关系图：



1. 程序编辑

原始程序的编辑（Edit）可利用各种文本处理器（PE Ⅱ，WORD 等）来建立，亦可使用集成环境中本身所提供的编辑器来撰写。例如 UNIX 系统和 Microsoft C 没有集成环境系统，必须使用一般文本处理器来编辑原始程序。

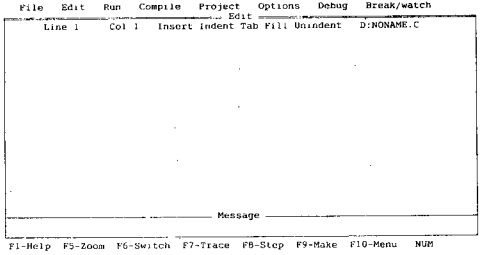


图 1.1　Turbo C 集成环境窗口

Turbo C 是一种集成环境系统，附有编辑器可供编辑程序。上面图 1.1 是进入 Turbo C 集成环境后，屏幕上所显示的编辑窗口画面。

Turbo C 集成环境中包括有下列几个菜单：

- File：输入文件，清除文件，存文件，及改变磁盘目录等。
- Edit：进入编辑状态。
- Run：执行程序。
- Compile：编译，连结程序。
- Project：项目制作。
- Option：编译器，连结器，环境，目录等模式选择。
- Debug：除错设置。
- Break/Watch：中断点或监视功能的设置。

光标移动命令键

命令键	功　能
←键或 Ctrl+S	左移一字符
→键或 Ctrl+D	右移一字符
Ctrl+A	右移一单字
Ctrl+F	右移一单字
↑键或 Ctrl+E	上移一行
↓键或 Ctrl+X	下移一行
Ctrl+W	上卷
Ctrl+Z	下卷
PgUP 键或 Ctrl+R	上移一页
PgDn 键或 Ctrl+C	下移一页

插入或删除命令键

命令键	功　能
Ins 键或 Ctrl+V	插入模式 启动/关闭
Ctrl+N	插入一行
Ctrl+Y	删除一行
先按 Ctrl+Q 再按 Y	删除光标后面的该行
Backspace 键或 Ctrl+H	删除光标左边字符
Del 键或 Ctrl+G	删除字符
Ctrl+T	删除光标右边单字

记号块命令

命令键	功　能
先按 Ctrl+K 再按 B	标记块的起点
先按 Ctrl+K 再按 K	标记块的终点
先按 Ctrl+K 再按 T	标记光标所在的单字
先按 Ctrl+K 再按 C	拷贝标记块
先按 Ctrl+K 再按 V	移动标记块
先按 Ctrl+K 再按 Y	删除标记块
先按 Ctrl+K 再按 R	读取文件到标记块
先按 Ctrl+K 再按 W	将标记块写入文件
先按 Ctrl+K 再按 H	隐藏或显示标记块
先按 Ctrl+K 再按 P	打印标记块内容，若设有标记块，则印整个编译器内的程序码。

2. 程序编译和连结

Turbo C 可利用其集成环境窗口中的“Run”功能选择项下的“Run”命令进行编译、连结、并且执行当前正在编辑的原始程序文件。或者分开使用“Complie”功能选择项下的“Complie”命令做编译工作，用“Link”命令做连结工作，以编制出可执行程序文件（.EXE）。

Turbo C 还有另一项选择，就是在 DOS 操作系统下使用 TCC 命令直接同时进行编译和连结工作。下面语句说明在 DOS 中如何使用 TCC 命令：

```
>TCC  原始文件名称
```

DOS 操作系统下，亦允许分开进行编译和连结工作。下面语句说明如何做编译工作：

```
>TCC-C  原始文件名称
```

上面命令中，“-C”选择项，告诉编译器只进行编译工作。以产生“.Obj”文件。Obj 文件必须经过连结过程才能产生可执行文件（EXE）。

```
>Tlink  目的文件名，执行文件名，图像文件名，程序库文件名
```

上面语句是用来将目的文件（Obj 文件）连结成可执行文件（EXE）。

第二章　基本数据

2.1　变量

变量（Variable）是用来在计算机主存中暂时保留数据的基本元素，它可以代表不同类型的数据值（如，数值、字符）。

1. 变量名称

变量名称是由英文字母，下划线（_）和阿拉伯数字所组成，而且它的开头必须是英文字母或下划线。大小写字母是有区别的。习惯上普通变量名称用小写字母构成，而符号常数则用大写字母。

变量名称最好是有意义的字或是与使用目的有关连，这样在一个复杂的大程序中才不会使众多变量互相混淆，而分辨不清。

变量名称不能使用系统本身的保留字（关键字），如 float，if，switch，int，return，……等。

合法的变量名称：

```
total
average_english
year12_math
_physics
```

不合法的变量名称：

```
Cost-2
Subtotal·3
12 Years
Cost$
PC+Pont
```

2. 变量声明

变量使用前必须先给合法的声明程序，以使编译程序安排适当的存储器空间。

格式

数据类型　　变量 1，变量 2，……；

数据类型可以是 char，int，float 和 double 等 4 种基本数据类型，它们分别代表字符、整数、单精度浮点数、双精度浮点数等 4 种。如：

```
int total1,total2,total3,TOTAL;
float average;
```

变量声明的同时，也可赋予该变量初值。如：

```
int sum = 0;
char test_char = 'A';
```

3. 变量的种类

A. 自动变量（auto，Automatic Variable）

自动变量的使用有效范围仅限于声明此变量的函数之内，换句话说，其有效性只限于该函数所包含的大括号（｛｝）内。所以它又被称为局域变量（local variable）。

格式

auto 变量类型　变量 1，变量 2，……；

如：

```
main()
{
   auto int total1,total2,total3;
   int average;
   .
   .
   .
   .
}
```

上面例子中的 total1，total2，total3 皆为整数类型的自动变量。第 4 行的“int average”声明，前面没有加“auto”亦认为是自动变量。

自动变量的特点是每次进入该段（以｛　｝包括）时，都将该变量定义一次。一旦离开该段后，就将其所占的内存释放给系统。

例：

```
main()
{
   auto int testno = 99;
   printf("testno =  %d\n",testno);
   testno++;
   {
     auto int testno = 1;
     printf("re-assignment testno =  %d\n",testno);
   }
```

```
    printf("after add 1: testno = %d\n",testno);
}
```

C 语言没有输出指令，上例中的 printf（）函数用来作为数据的输出，它是 C 语言中最常用的函数之一。详细用法留待后面 2-5 讨论。第 5 行中的："testno＋＋；"与"testno=testno＋1；"同义，它们的详细用法亦留待后面讨论。

第 3 行："auto int testno=99；"声明整数变量 testno，并设置初值为 99。第 4 行语句打印出变量内含 99。第 5 行将 testno 变量加 1（变成 100），第 6 行到第 9 行中，虽然重新声明变量 testno 并设置为 1，因为其有效范围仅限｛　｝内，即占用不同内存，所以并不会影响前一个（第 5 行）已经加 1 的 testno 变量值（100）。

下面是执行输出的结果：

```
testno = 99
re-assignment testno = 1
after add 1: testno = 100
```

B. 静态变量（static，static variable）

静态变量主要特点是它所占的内存在程序执行过程中都会继续存在，换句话说，static 变量只分配内存一次直到程序结束。而 auto 变量是每进入该变量所在段时都会重分配（定义）该变量。static 变量如果没有设置初值，则系统会自动设置初值为 0 或空白。而 auto 变量若没有设置初值，则内存内可能会出现乱七八糟的东西。

格式

static　变量类型　变量 1，变量 2，……；

如：

```
main()
{
   static int var1, var2;
   .
   .
   .
   .
}
```

上面例子是将 var1 和 var2 声明为静态变量。

例：

```
main()
{
   int i;
   for (i=0;i<3;i++)
   {
      static int static_variable = 99;
      auto int auto_variable = 9;
```

```
        printf("static var: %d\n",static_variable);
        printf("auto var: %d\n",auto_variable);
        static_variable++;
        auto_variable++;
    }
}
```

上面例中的“for (i=0; i<3; i++)”是循环控制语句，用来使循环体内（{ }间语句）的语句连续执行3次，详细用法留待第3章再讨论。

第6行语句将static_variable整数类型变量声明静态变量，且设置初值为99。第7行语句将auto_variable整数类型变量声明为自动变量，且初值设置为9。第10，11行语句分别执行static_variable和auto_variable的累加1操作（每执行一次自动加1）。

因为static_variable变量被声明为静态变量，所以每次执行“static_variable++;”语句都会使结果值累加1。而auto_variable变量被声明为自动变量，虽然第11行“auto_variable++;”语句执行加1动作，但是，每次回到第7行时，auto_variable变量声明语句时，它又被重新定义，且设置值为9。

下面是执行输出的结果：

```
static var: 99
auto var: 9
static var: 100
auto var: 9
static var: 101
auto var: 9
```

C. 外部变量（extern, external variable）

外部变量又可称为全局变量（global variable），它能被不同函数或程序引用，甚至被独立编译的函数（存在于不同文件中）所引用。

声明外部变量的方式有二：

（1）在同一文件的主程序或函数外面直接声明变量。如：

```
int score1, score2, score3;

function1()
{
    :
    :
}

function2()
{
    :
    :
}

main()
{
    :
    :
}
```

此例中的 score1，score2，score3 三变量，因为被声明在函数外，属于外部变量。所以 function1 ()，function2 () 和 main () 函数都可以引用此变量。

(2) 在另一程序用 extern 关键字声明某一变量为外部变量，以供另一程序文件引用。

格式

extern　变量类型　　变量 1，变量 2，……；

如：

```
FILE 1:
-------

int external_variable;

main()
{
   static int local_variable;
       :
       :
   function1()
       :
       :
}


FILE 2:
-------

function1()
static int local_variable;
{
   extern int external_variable;
    :
    :
}
```

上面例子中的外部变量 external_variable 可以在二个文件 (FILE 1，FILE 2) 之间互相连系引用。另一方面，FILE 1 中的 local_variable 变量和 FILE 2 中的 local_variable 变量，虽然名相同，但其内存都是在各自定义中分配，属于局域变量，不会互相干扰。

D. 寄存器变量 (register，register variable)

寄存器变量与 auto 自动变量类似，但是，寄存器变量是将所声明的变量放入寄存器(register) 内，其目的是加快执行的速度。

格式

register　数据类型　变量 1，变量 2；……；

2.2　数据类型

C 语言有 4 种基本数据类型：整数 (int)、字符 (char)、单精度浮点数 (float)、双精度

浮点数（double）。

1. 整型数据类型（integer）

整型数据类型根据数据长度所占位数（bit）可分成4种。以Turbo C为例：

整数类型	占位数（Bit）	数值范围
int	16（2 Bytes）	−32768～32767
short int	16（2 Bytes）	−32768～32767
long int	32（4 Bytes）	−2147483648～2147483647
unsigned int	16（2 Bytes）	0～65535

2. 字符型数据类型（Character）

每个字符所占的内存是1 Bytes（8 Bit）。它们（例如，A～Z，0～9，＋，－，＊，!，……等符号）在计算机中都用一个正整数值来代表，其值为0～255（2^8=256）。以IBM PC为例，是以ASCII码来表示。

格式

Char　变量1，变量2，……；

C语言中有一些无法显示打印的字符，称之为“转义字符”（escape Character）。它的表示方式是字符前加上“\”（back slash）符号。当编译器遇到“转义字符”时，会把“\”后面的字符当成某种特别意义来处理。

字符表示式	意　　义
\n	换新行（New line）
\t	制表（tab）
\b	退一格（backspace）
\r	RETURN键
\f	跳页（form feed）
\'	打印出单引号（Single quote）
\"	打印出双引号（double quote）
\\	打印出倒斜线（back slash）
\0	空格（null space）
\xdd	ASCII码用16进制表示
\ddd	ASCII码用8进制表示

例：

```
main()
{
   printf("\n");
```

```
    printf("escape character test:\n");
    printf("tab test:\t");
    printf("backspace test:\b");
    printf("abcdefghijk\n");
    printf("single quote test:\'\n");
    printf("double quote test:\"\n");
    printf("ASCII code of 41h(Hex):\x41\n");
    printf("ASCII code of 101(Oct):\101\n");
}
```

上面例子是各种转义字符的使用法，第3行语句中的“\n”符号，可以使下一个打印语句换新的一行。第5行语句中的“\t”可以使下一个打印语句跳格。第6行语句中的“\b”可以使下一个打印语句往左退回一格。所以它使“abcdefghijk”中的“a”重叠到“backspace test:”中的“:”。第8行语句的“\'”，可以打印出“'”符号。第9行语句中的“\"”可以打印出“"”符号。第10行语句中“\X41”可以打印出“A”(A的ASCII码以16进制数字系统表示为41)。第11行语句的“\101”印出“A”(A的ASCII码以16进制数字系统表示为101)。下面是执行输出的结果：

```
escape character test:
tab test:       backspace testabcdefghijk
single quote test:'
double quote test:"
ASCII code of 41h(Hex):A
ASCII code of 101(Oct):A
```

3. 单精度浮点数数据类型 (Single-precision floating point)

单精度浮点数在内存中占4字节 (Bytes) 空间，浮点数表示法有点类似于科学记法。例：

数	科学记法	浮点数
1000000	1.0×10^{6}	1.0e6
1234567.8	1.2345678×10^{6}	1.2345678e6
0.0012345	1.2345×10^{-3}	1.2345e-3

单精度浮点变量的声明：

格式

float 变量1，变量2，……；

4. 双精度浮点数数据类型

双精度浮点数内存中占8字节 (Bytes) 空间。

双精度浮点变量的声明：

格式

double 变量1，变量2，……；

2.3 常数

1. 整数常数

整数常数不但允许10进制数字，而且允许8进制及16进制数字。

凡是以0为开头的整数都会作为8进制数字处理。如：

8进制	10进制
023	19
036	30

凡是以0X为开头的整数都作为16进制数字处理。如：

16进制	10进制
0X16	22
0X17	23

如：

声明：int i=0X2D；

相当于：int i=45；

声明：int j=023；

相当于：int j=19；

长整数（long int）常数的写法是在数字后而加上L，例如，456L；同样，8进制和16进制常数后而加上一个L，亦会当作长整数处理。

2. 浮点常数

浮点常数可以使用小数表示法和指数表示法。如：

i=1.234e-3 相当于 i=0.001234

j=0.234e3　相当于 j=234.000000

3. 字符常数

字符常数是用单引号括起来的字符符号，如' A'。每一个字符符号在ASCII码中都有一个特定值。例如' A' 值65，' B' 值66，' a' 值97，' b' 值54。

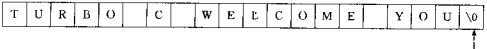
4. 字符串常数

字符串常数就是双引号间的任意个数字符符号的组合。如：

" Turbo C Welcome you"

" This is a book"

实际上，C语言没有字符串类型的定义。字符串是用一连串的字符数组（character array）以达到字符串的效果。如：



上面例子是由许多字符常数组合成的一个数组，此数组就形成一字符串。编译时，编译器会自动加上一个空字符（null character）在字符串最后，以表示整个字符串的结束。所以计算机在分配内存时，会比实际的字符数多出一个空字符（∅）。

双引号不属于字符串的内容，所以如果需要在字符串中出现双引号时，则必须使用“\”符号隔开表示。如：

字符串常数定义如下：

```
" Turbo \" c\" is one kind of C language"
```

结果会输出如下字符串：

```
Turbo " C" is one kind of c language
```

“\”符号亦可以用来作为字符串连结之用，使上一行与下一行的字符串相连在一起。

字符串常数定义如下：

```
stringl="Turbo C is\
the best one";
```

结果会输出如下字符串：

```
Turbo C is the best one
```

Turbo C 允许将一长字符串分成数个短字符串来表示。如：

字符串常数定义如下：

```
stringl="C language"
"is a very"
"Popular language";
```

结果会输出如下字符串：

```
C language is a very popular language
```

2.4　类型转换

设计C语言程序，时常会遇到不同数据类型的数学运算。例如，浮点数变量和整数变量两种不同数据类型的运算项相遇要作数学运算。C语言提供2种方式处理类型转换。

1. 自动转换

如果表达式中有不同数据类型的运算项时，C语言编译器会根据一些简单原则自动把它们转换成一个共同的类型。

（1）精确度低与精确度高的运算项一起运算时，精确度低者会被转换成高精确度。精确度高低顺序如下：

double
float
long
int
short
char

例：

```
main()
{
   int a=2;
   float b=3.9;
   float c;
   c=a+b;
   printf("%f\n",c);
}
```

上面例子中，a 是整数数据类型，b 是浮点数数据类型，c 是浮点数数据类型。因为整数精确度低，所以运算时（c=a+b），会被转换成浮点数与 b 相加。

下面是执行输出结果：

```
5.900000
```

（2）在语句中，等号右边的常数数据类型会被转换成等号左边的数据类型。如：

```
float i=123;
```

等号右边的 123 为整数类型，左边的 i 变量被声明为浮点数（float），所以 123 亦会被转换成浮点数输出（123.000000）。如：

```
int i=123.456789;
```

等号右边的 123.456789 为浮点数数据类型，右边的 i 变量被声明为整数，所以 123.456789 会被转换成整数输出（123）。

2. 强制转换

C 语言虽然会自动作类型的转换，但是，因为它会影响执行速度。所以 C 语言又提供另一种强制转换数据类型的方式，让设计者能在程序内直接作数据转换。

格式

（数据类型）表达式

如：

```
C= (int) a+ (int) b;
```

上面语句，假设 a，b 是浮点数类型，c 是整数类型变量。(int) a 会将 a 强制转换成整数，(int) b 会将 b 转换成整数。运算结果存入整数变量 c 中。

2.5 基本输入与输出

1. printf（）函数

C 语言本身没有输出指令，printf（）是程序库中的函数，它用来将数据输出到屏幕上。它是 C 语言中经常使用的函数之一。

格式

printf（" 规格参数"，变量 1，变量 2，……；

规格参数可分成两部分：

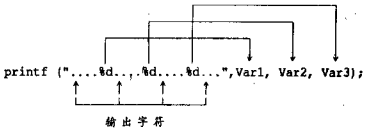
输出字符

凡是规格参数内的任何输出字符（例如：A～Z，0～9，＋，－，×，/，……等任何符号字符）都会完整地输出到屏幕上。

数据类型字符

数据类型字符用来控制输出参数的数据类型，它是以"%"符号作为前导符号。下面列出各种数据类型字符：

数据类型字符	输 出 数 据 类 型
%d	十进制整数
%f	浮点数（小数类型）
%c	单一字符
%s	字符串
%e	浮点数（指数类型）
%u	无符号十进制整数
%o	无符号八进制整数
%x	无符号十六进制整数
l	加在 d，u，o，x 前，加%ld，%lu，%lo，%lx，表示长整数



上面输出函数 printf 中，规格参数里第 1 个数据类型字符%d 控制后面输出参数中第 1 个变量 Var1 的输出数据类型．其他依此类推。例：

```
main()
{
   int i,j,k;
   i = 100;
   j = 200;
   k = 300;
   printf("\n");
   printf("i = %d, j = %d, k = %d",i,j,k);
}
```