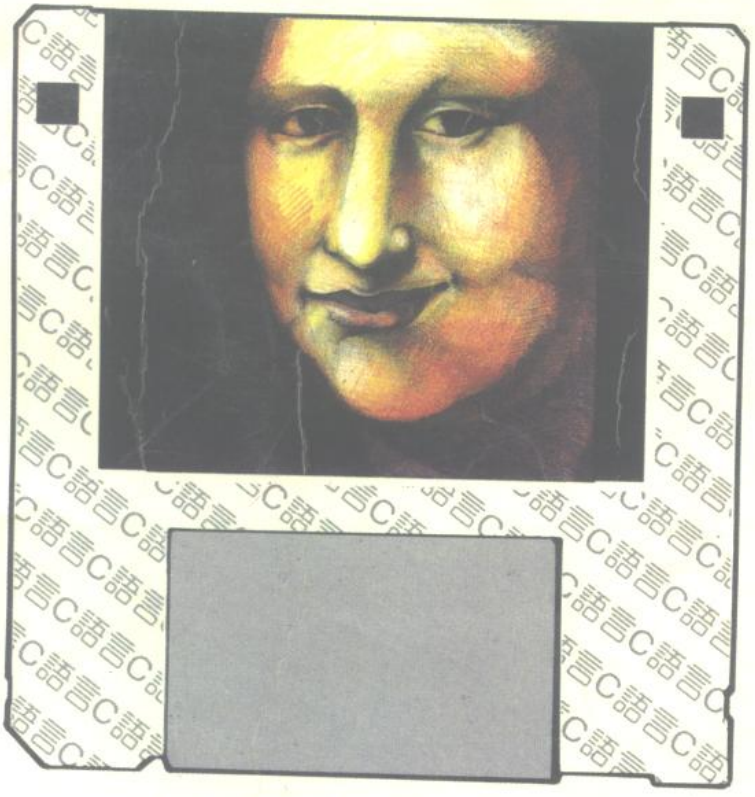
计算机语言技术系列丛书

C语言轻松学会



学苑出版社

第3波

希望

李至钦 著

- 结构化分析C语言设计元素
- 内容扼要并附习题供自我评量
- 本书附大量程序范例

计算机语言技术系列丛书(二)

轻松学会C语言

李玉钦　著
张　如　改编
希　望　审校

学苑出版社

(京)新登字 151 号

内 容 提 要

本书为读者循序渐进的剖析 C 语言程序设计的过程，对每一个主题均从基本概念深入浅出的介绍，内容新颖富有启发性，可以节省许多摸索时间。

本书每一章节中，均给出数个重要的程序范例，以层层剖析的方法来说明程序的要诀，并有效的了解 C 语言，进而掌握程序设计的技巧。本书每章均附有习题及解答，帮您找出学习的盲点，以便更进一步了解 C 语言。

欲购本书的用户，请直接与北京 8721 信箱书刊部联系，邮编：100080，电话：2562329。

版 权 声 明

计算机语言技术系列丛书(二)

轻松学会 C 语言

编　　著：李至钦
改　　编：张　如
审　　校：希　望
责任编辑：甄国宪
出版发行：学苑出版社　　邮政编码：100036
社　　址：北京市海淀区万寿路西街 11 号
印　　刷：北京四季青印刷厂印刷
开　　本：787×1092　1/16
印　　张：17.875　　字数：415 千字
印　　数：1～5000 册
版　　次：1994 年 8 月北京第 1 版第 1 次
ISBN7－5077－0905－1/TP・29
本册定价：28.00 元

学苑图书印、装错误可随时退换

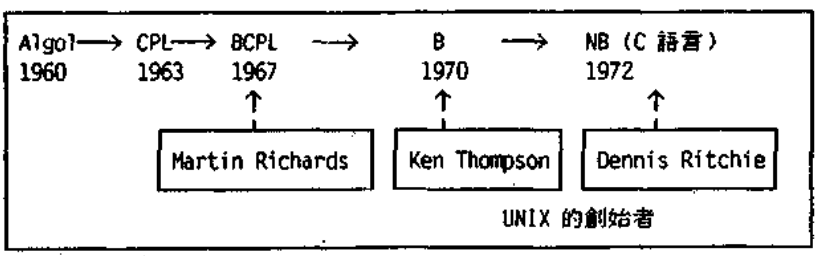
目 录

第一章 C语言简介

本章将告诉读者:C语言的历史、特点、基本结构及编写程序所需的步骤,读者也会学到编译器、连接器的基本概念及了解简单的C程序。本章讨论主题如下:

- ☐ C语言的来源
- ☐ 为何用C语言
- ☐ 结构化程序
- ☐ C程序基本结构图
- ☐ 在C语言中如何使用程序库函数
- ☐ 编写C语言过程七大步骤
- ☐ 程序的编译
- ☐ C语言的未来
- ☐ 第一个C程序
- ☐ 第二个C程序
- ☐ C语言的关键字

1.1 C语言的来源



C语言历史简介表

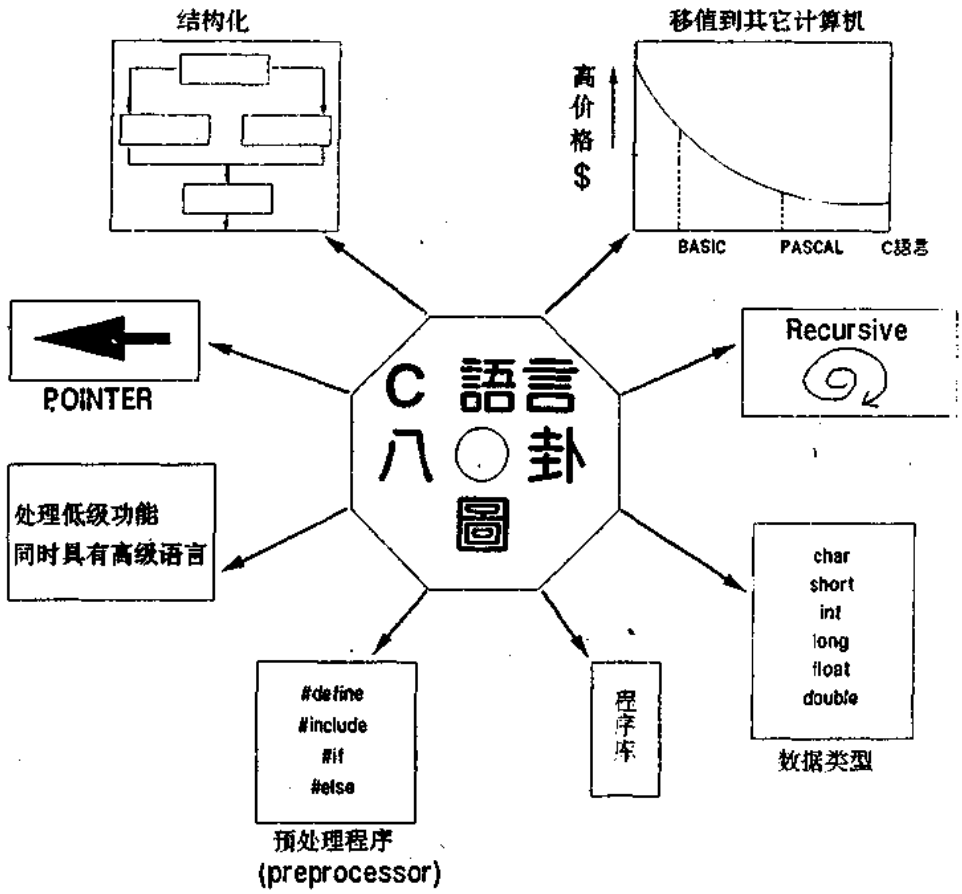
C语言是由贝尔实验室的Dennis Ritchie在1972年所发明的,当时他正与Ken Thompson共同设计UNIX操作系统。当然C语言并不是Ritchie凭空想象出来的,而是源于Thompson的B语言,此又源自BCPL(Basic Combined Programming Language)的组合语言,由Martin Richards所开发。至于BCPL是如何产生的呢?可以追溯至1960年C语言最早的前身Algol Programming Language,可Algol在美国没有广泛地流行。CPL(Combined Programming Language)修改Algol,使其能够直接作较低层次的操作。跟Alogl一样,CPL有太多特性,使得它很难了解,因此BCPL的出现,乃是针对此问题而产生的语言。

自从贝尔实验室利用C语言编写UNIX的系统控制程序后,便使得C语言一举成名。UNIX和相关的公共程序共包含30万行以上的C源代码。C语言高级语句和函数的功能相当强,并包含处理位的运算,所以应用范围涵盖甚广。C语言最重要的特点是兼备了高级及汇编语言的特点。

1.2 为何用C语言

过去几年来,C程序语言已成为IBM系列计算机上的程序设计者的第一选择,为什么它能在众多程序语言中立于不败之地呢?就因为它提供如PASCAL或BASIC等高级语言的方便性,而且也提供了如汇编语言之类而足以控制计算机硬件及外部设备的能力,且很多要用汇编语言才可完成的事,通常可以方便用C语言来做到,这或许是它如此受欢迎的的主要理由。

C语言正迅速地变成最通用而重要的语言之一。其日益风行是因为人们喜欢它,使用它。等您学过C语言后,您将认识它的许多优点。以下就是它的基本优点。



C语言的优点

- C语言让用户很自然的使用自顶向下(top—down)的规划,结构化的程序组织和模块化(modular)设计。因此产生可靠的、容易理解的程序,使程序易懂易维护。
- C语言是一种可移植的语言。在某一系统下写的C程序,不用修改或只须略加修改,就可以在其他系统上运行。若需修改的话,通常也只须修改头文件(Header File)内的少数几项而已。
- C语言提供递归(Recursive)处理能力及提供预处理器(Preprocessor)功能,使程序具

有 macro(宏)指令。

- C 语言提供多种而具有弹性的数据类型,让程序设计者可以不受规划上的束缚。
- C 语言能与汇编语言密切搭配。必要时可利用此特性调配程序使其达到最高的效率。
- C 语言提供指针变量,对内存做直接管理。指针变量可以使来做算术运算。
- C 语言具有对位处理的低级功能,同时具有高级语言的程序语句。
- C 语言提供了丰富的程序库(library),供程序设计者来使用。并且可以让程序计者自己来设计程序库,以加强他们对应用程序的开发。

1.3 结构化程序设计

结构化程序设计(Structured Programming)一词,首先由荷人 Edsgerw Dijkstra 提出,他于 1969 年发表了"Notes on Structured Programming"以及"Structured Programming"两篇论文,从此打开结构程序的构想。而有关数学理论,则通过美国人 Corrado Bohrm 及意大利人 Giuseppe Jacopini 证明了任何程序均可以用简单的逻辑结构来表示。而任何逻辑线路,不管它如何复杂,均可由 AND、OR 及 NOT 三种逻辑所构成。

结构化程序的优点

(a)统一使用三种基本逻辑结构(顺序、IF_THEN,WHILE_DO)等),故只有一个入口,一个出口,逻辑极为清楚。

(b)易于除错(debug)及后续修改维护。

(c)理论上可以不用 Goto。

(d)使用缩进方式编写程序,故易于了解与阅读,可减少错误。

(e)可配合自顶向下(Top—Down)程序开发方法。

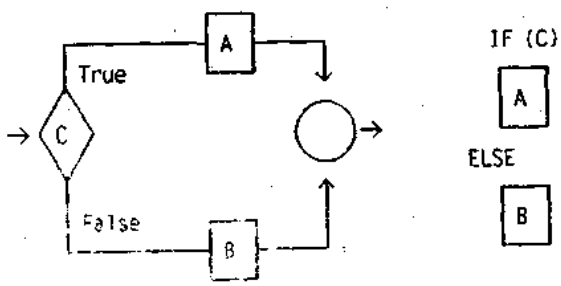
(f)提高程序设计者的生产力。

结构化程序的特性

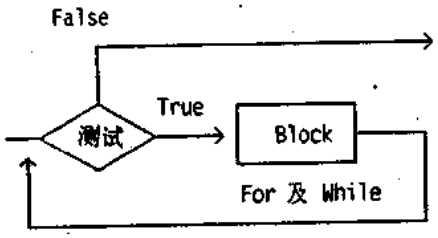
逻辑结构有四种方式:

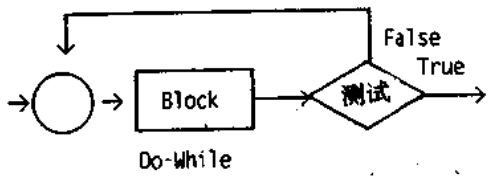
(a)顺序结构:在程序中,依顺序处理两条或两条以上的语句(statement)。如下所示:

(b)选择结构:IF_THEN 及 IF_THEN_ELSE,依某一条件 C,若 C 为真,则运行 A,若 C 为假,则运行 B。

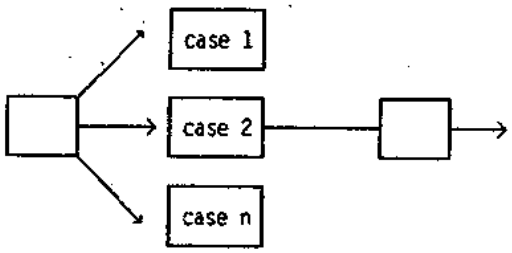


(c)循环结构(Loop Control):循环结构有前测试循环及后测试循环两种,如下图:





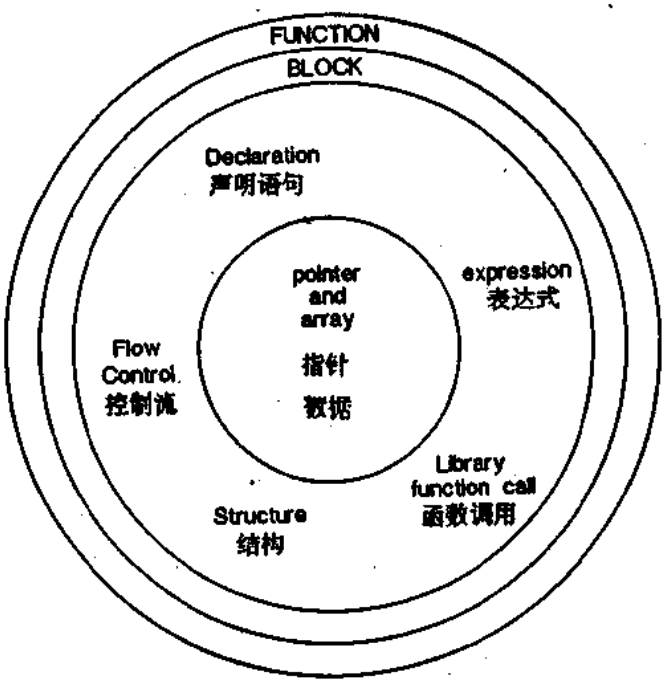
(d)多选择条件结构(Case):测试某一数值,再视诠值而选择其路线。如下图所示:



结构化程序的缺点

所需程序指令较多,所占内存空间亦较大。

1.4 C程序基本结构图



C程序由一个或多个函数(Function)所组成，其中一个函数的名称一定要命令为main，C程序运行流程由main函数开始，其他函数的名称可以自由选择，但一定要有main来启动。

例：

```
#include<stdio.h>->header
main() {/* a sample program */
Block{
    char x, y;->Declaration(Local variable)
    x=2;
    y=x+3;->expression
    printf ("x=%d,y=%d\n",x, y);->Library function call
}
```

函数(Function)通常由二大部分组成：

(1)程序头(header)——#号表示在C程序被编译运行之前，需先由C的预处理程序来处理。

(2)Block(块)——又称为程序体(Body)，由左括弧"{"来当块之起始，右括弧"}"来表示块的结束。

而语句/* a sample progam */表示一个注解，在程序中并不运行。

至于在C程序中，每个语句都以分号(;)来作结尾，表示一个语句的结束。

1.5 在C语言中如何使用程序库函数

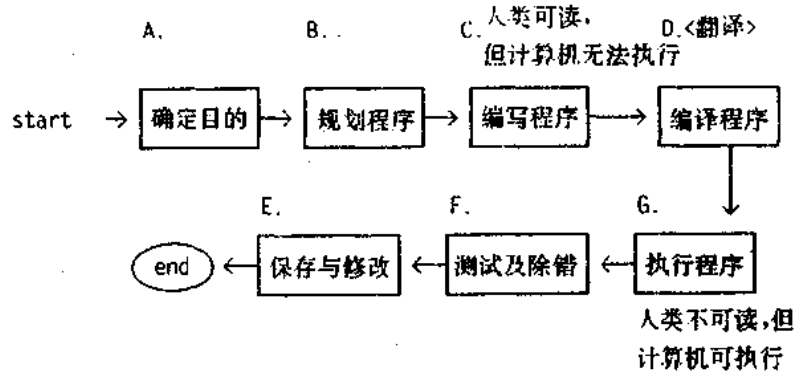
标准程序库并不是C语言本身的一部分，而是C语言系统(或称C语言环境)的组合部分，这个语言环境自然支持标准C语言，且提供标准程序库的若干函数声明，类型定义以及宏定义等。而程序库函数的使用与在C语言中进行的直接函数调用一样，只不过在使用前，在程序开始应当把对这些函数进行声明和类型定义的头文件包括进去，否则就不能正常使用。比如，在C语言程序中要用到函数getchar()，此时就必须在程序开头把包含对getchar()声明的头文件<stdio.h>包含进去，即#inchude <stdio.h>，否则就不能正常使用。因为getchar在stdio.h文件中被定义成getc(stdin)。

标准程序库中的函数声明，类型和宏定义通常是放在一些称为标准头文件(header)的.h文件中，这些文件是

<assert.h>	<math.h>	<float.h>	<stdarg.h>	<stdilb.h>
<ctype.h>	<limits.h>	<setjmp.h>	<stddef.h>	<String.h>
<time.h>	<locale.h>	<crrno.h>	<stdio.h>	<signal.h>

在一般的C的程序中stdio.h使用次数比较多。

1.6 编写C语言过程七大步骤



大部分初学者常会忽略前两个步骤而直接跳到步骤C去编写程序。当然刚开始写程序都很简单，脑子一想便可了解整个过程，有错误也很容易改，但是随着程序愈来愈长或愈复杂，脑海中开始无法掌握全局，错误愈来愈难找。最后那些忽视步骤A及B的人，终将偿到浪费时间、困惑及受挫这些苦头。因此我们得到的教训：读者编写程序之前应该有规则的习惯，用纸笔简单写下程序的目的及初步设计，如此既能省时间，又能写出令人满意的程序。

阶段A：确定目的

一开始第一件事要很清楚程序目的是什么。然后把此问题转成所需的数据并要求此程序最后所要输入的结果。在这阶段中，读者应用一般的语句思考，而不是以其特定的计算机语言来编写。

阶段B：规划程序

当对于要这个程序做些什么事有了概念之后，便应决定如何设计程序来完成这个工作，这包括要有怎样的用户界面？如何组织程序？如何表示程序及辅助文件的数据，以及用什么方法来处理数据？做这个程序需花多少时等等。

阶段C：编写程序

程序规划好之后，便可以开始编写程序。也就是将程序"翻译"成C语言。要注意的是，所有C源程序文件都要有.c的文件扩展名，不然将不被编译程序接受。一般来说，读者需要借助文字编辑器(text editor)来建立C的源程序(source program)文件。下列程序便是C源程序的一个例子：

```
main() {
    printf("with my book, you read less and learn more\n");
}
```

阶段D：编译程序

所谓编译(compile)亦即"翻译成计算机可以了解的语言"的意思。而进行编译的程序，称为编译程序(compiler)。编译程序是先翻译人类可读的程序，再变换为计算机可读的(可运行的)程序。此处，人类可读的程序，称为源程序，计算机可运行的程序称为运行程序。而编

译程序同时也会检查程序的语法。如果编译程序发现错误，它会向您报告，并且不会产生运行文件；若没有错误，即可产生运行文件。

阶段 E：运行程序

传统上可运行文件本身便是一个可运行的程序。在 UNIX 或 MS-DOS 等一般环境运行时，只需键入可运行文件的名字即可。

阶段 F：测试与除错

程序能运行当然是件好事，但这并非表示它没有问题。因为我们应该检查其运行情形是否与预期的相符合。其次，读者在第一次编译时很可能不会成功，如输入时打错了字，而这些错误——计算机术语称为(bug)，而除错(debug)即找寻并除去程序的错误。

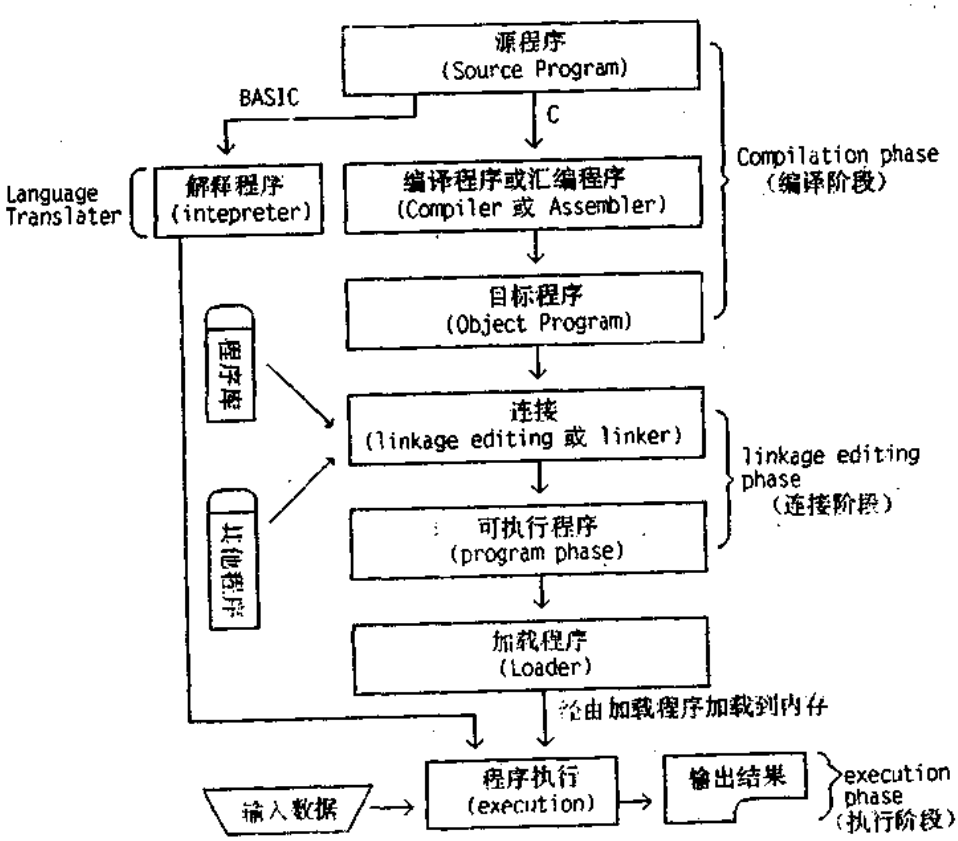
阶段 G：保存及修改

当为自己或他人编写程序时，此程序可能有广泛的用途，因此有时候可能要做某些修正。也许可能要修正程序而使它能在别的系统上运行。只要遵守正确方法来设计并将程序完整保存起来，上述各项工作都可以大大地简化。

注：写程序不会总是必须按照上面七个步骤来进行，有时必须在某两个步骤间反反复复好几次。譬如可能从阶段 D 回到阶段 C，反复做了几次，或也可能从阶段 F 又回到阶段 C 等等。

1.7 程序的编译

C 语言与 BASIC 语言翻译方式的比较图



解释执行程序(Interpreter)与编译程序(compiler)

C语言是近年来极为流行,广为大家使用的语言。今天,谈到计算机,大多数人都会想到个人计算机。若是说到知名度最高的语言,可能还是非BASIC语言莫属。因解释执行对程序开发来说是十分方便,因为在输入程序后几乎就立刻运行,但是BASIC愈是广泛地流行,愈是有许多人发现到它的缺点。试举下列几点如下:

1. 运行速度慢
2. 非结构化语言,很难写大的程序
3. 很容易解密,无法保持机密性

这些缺点与其说是BASIC本身的问题,不如说是因解释执行程序(interpreter)而造成的。而近年来也有用BASIC的编译程序(compiler),所以稍微可以减少这些缺点。但是BASIC本身并非是合适编译程序的语言,实用上令人无法消除"慢"的感觉。

以下说明编辑程序是什么

所谓编译程序便是将源程序转换成可运行程序的代码(executable code)。可运行程序是以机器语言(计算机的"母语")写成的程序。不同计算机有不同的机器语言,而C的编译程序则将C翻译成特殊的机器语言。编译程序也会将程序库中的程序加入最终的程序;程序库里含有一大堆标准的功能以备使用,如printf()及scanf()。更精确地说,有个叫连接的程序(linker)负责处理程序库里的工作,但在大部分的系统中编译程序都会帮您运行连接的工作。最后的结果是产生可运行文件。

由上看来使计算机执行程序方法的根本概念的不同,在于"解释"与"编译"的不同。由计算机开发历史来看,先建立编译之后才出现解释,两者各具特色,各有优、缺点。而经常可听到有人说解释的"简便性"非编译所及,而编译的"速度"又非解释"所及。

操作系统的考虑

关于如何编写C程序,以及如何编译和运行的细节问题,往往会随着操作系统的不同而有所差异。不过,总体上来说仍然是极相似的。现在我们来描述一下在UNIX或(XENIX)系统下的操作步骤,编写并运行一个程序的步骤。

1. 建立一个文件扩展名为.c的正文文件,并键入一个C语言程序。例如:

```
vi program.c
```

vi为Unix的文字编辑器的名称。在使用编辑器之前,应该先对插入(insert)和修改(modify)等编辑命令有所了解。

2. 在键入program.c文件后,利用下列命令来编译它。

```
cc program.c
```

cc为C编译的命令,如果编译的程序没有错误的话,便会产生运行文件a.out,否则便需要再重新回到步骤1,修改源程序。

3. 利用下列命令运行程序:

```
a.out
```

如果该程序没有运行错误(run—time error),应该会完全地运行。

多个函数的程序编译

假如有二个以上的源文件(source file)如何连接(link)在一起呢?假设有三个源文件,progl.c、prog2.c及prog3.c。

(a)在UNIX系统下

直接用cc来编译及连接，以下指令可以同时编译此三个文件并产生a.out可运行文件。

```
cl progl.c prog2.c prog3.c
```

另外还有progl.o、prog2.o、prog3.o三个目标文件。如果改变了progl.c，而prog2.c及prog3.c仍保持不变，以下指令可以编译progl.c，并使它与prog2.0及prog3.0目标程序相连接而产生a.out运行文件。

```
cc progl.c prog2.0 prog3.0
```

(b)在PC系统下

与UNIX系统下的概念一样，只是在运行文件以第一个文件来命名。如在Microsoft C 4.5～5.1版。

```
cl progl.c prog2.c prog3.c
```

(c)在Turbo C下

```
tcc progl.c prog2.c prog3.c
```

编译后产生三个目标文件(progl.obj prog2.obj prog3.obj)及progl.exe可运行文件。然后在PC上键入progl即可。

1.8 C语言的未来

C语言的兴起，源自于UNIX操作系统。因为UNIX系统的百分之九十都是由C语言所写的。来自UNIX的C用户急欲把他们的程序带回家，现在已经有几种编译程序可实现他们的心愿。

C语言已成为UNIX系统下微型计算机世界的主流，现在已经蔓延到所有计算机的领域里。许多软件商正积极改用C语言生产其套装软件：文字处理程序，数据库管理系统，CAD/CAM应用软件，OA(办公式自动化)、电子表格，绘图软件，编译程序等等。这些厂商知道C语言能产生简洁而有效的程序，更重要的是，他们知道这种程序容易修改可以配合不断推出的新计算机硬件。

总之，C语言注定成为1980年至2000年间最重要语言之一。C语言使用于微型计算机和个人电脑方面，为软件厂商，计算机和信息学生，及各界的拥护者所热爱使用，可以说担负软件编写工作的必备条件之一就是能够使用C。

C语言的应用

1.9 第一个C程序

学程序设计第一件工作便是在屏幕上显示信息。现在让我们写一个程序将"Hello! word"输出到屏幕上,其完整的程序如下:

```
main() {
    printf("Hello. world\n");
}
```

我们利用文字编辑器,将此程序键入到扩展文件名为.c的文件。这里,我们假设文件名为hello.c当此程序编译运行后,便会在屏幕上显示Hello,world。

程序说明

(1)main()

每个程序都有一个main函数做为运行的开始。后面的小括号表示它是一个函数。

(2){

每个函数体均以左大括号开始,而且必有一个相对的右大括号做为函数的结束。

(3)printf()

C系统内包含了一个标准函数程序库,以供程序使用。printf()便是程序库内函数之一,它可在屏幕上显示信息。

(4)"Hello. world\n"

此字符串为函数printf()的一个参数(argument),用以控制所显示的内容。在字符串最后的两个字符\n,其实只代表单个字符,我们称为换行字符(newline)。换算字符是不可打印的字符,它的作用只是将光标移至另一新行而已。

(5)}

此右大括号和上述左大括号相对应,用来结束函数main(),下面程序的结果与上述的第一个程序完全一样。

```
#include<stdio.h>
 main(){
   printf("Hello,");
   printf("world");
   printf("\n");
 }
```

1.10 第二个C程序

此程序可用以说明编写C程序的一些基本特点:

```
#include<stdion.h>
main() {      /* a simple program */
    int number; /* define a variable called number */
```

```
    number=7;       /* assign a value to number */
    printf("A Simple");/* use the printf() function */
    printf("computer\n");
    printf("My favoite number is %d because it is lucky\n",number);
}
```

如果猜想这程序会在屏幕上显示一些东西来，那便猜对了！但究竟会显示什么东西？如果编译、运行一切顺利，应该有以下输出：

A simple computer

My favorite number is 7 because it is lucky

这个结果当然不会令人太惊讶。但其中的\n 和%d 是什么意思呢？程序中有几行看来也有些奇怪。我们在下面会加以解释。

程序解说

(1) #include <stdio.h>

把另一个文件包括进来又称为头文件，本行告诉计算机将 stdio.h 文件中所含数据包括进来。

stdio.h 是随着 C 编译程序的套装软件一起供应的，它包含有关输入、输出函数的信息，以供编译程序使用，如 printf()函数。有些程序需要包括 stdio.h，而有些则不用。C 语言在使用系统的文件时，应含有对程序库内函数的描述，该描述便提到要用到 stdio.h。当然省略适当的头文件对一特定程序并无影响，但最好不要对此依赖太深。每次用到程序库函数时，我们都将使用 ANSI 标准为这些函数所定义的头文件。

(2)main()→函数名

函数是 C 程序的基本模块(module)，C 程序便由一个或多个函数组成。

本程序由一个称为 main 的函数组成，括号用来标识 main()为一函数名。

(3)/* a simple program */→注解

在/*与*/表示法之间都是注解(comments)。注解可用来帮助说明程序。它专供程序阅读者使用，编译程序会忽略它们。

{→开始函数体

}←结束函数

(4)int number;→声明语句

此语句声明程序将使用一名为 number 的变量，其类型为 integer(整数)。

(5)number=7;←赋值语句

此语句将 7 赋给(assign)给变量 number。

(6)printf("a simle");←打印语句

此语句将把 a simple 显示于屏幕上，并将光标(cursor)留在该行。

(7)printf("Computer\n);←另一个打印语句

此语句把 computer 紧接在上一组字符后显示。\n 告诉计算机重新开始新的一行，亦即将光标移到下一行的开端。

printf("My favoite number is %d becaure it is lucky \n",number);最后这个 printf 语句