高等院校教材

高级语言程序设计

(C语言版)

刘加海　主编

浙江大學出版社

高等院校教材

高级语言程序设计（C语言版）

主编　刘加海

浙江大學出版社

内容提要

本书根据C语言的特点，力求突出系统性、实用性，简明易懂，由浅入深地讲授C语言的基本内容及编程特点。主要内容包括：C语言基本概念、控制结构、模块化程序设计、变量的存储类型及编译预处理、指针与数组、指针与函数、结构与共用体、位运算、文件、程序设计常用函数算法简介、实用程序设计等。本书内容精炼，编排合理，对读者可能遇到的难点做了十分系统、详细的阐述，对重要知识点配有大量例题，便于理解和掌握。

本书可作为高等院校各专业本专科学生高级语言程序设计课程的教材。

图书在版编目(CIP)数据

高级语言程序设计：C语言版/刘加海主编. —杭州：浙江大学出版社，2002.2
高等院校教材
ISBN 7-308-02516-0

Ⅰ.高… Ⅱ.刘… Ⅲ.C语言—程序设计—高等学校—教材 Ⅳ.TP312

中国版本图书馆CIP数据核字(2002)第006191号

出版发行 浙江大学出版社
(杭州浙大路38号 邮政编码310027)
(电话：0571—88273329，88273761(传真))
(E-mail：zupress@mail.hz.zjcn)
(网址：http://www.zjupress.com)
责任编辑 阮海潮
排　　版 浙江大学出版社电脑排版中心
印　　刷 浙江大学印刷厂
开　　本 787mm×1092mm 1/16
印　　张 21.75
字　　数 529千字
版 印 次 2002年2月第1版 2002年2月第1次印刷
印　　数 0001—4000
书　　号 ISBN 7-308-02516-0/TP·226
定　　价 30.00元

前　言

随着计算机应用的进一步普及和深入，大多数高等学校都把C语言程序设计列为高级语言程序设计课程的必修课。作者根据多年的教学经验和科研成果，在修改作者本人编著的同类教材的基础上编写了这本《高级语言程序设计(C语言版)》教材。

在C语言程序设计这门课程中，指针的学习及应用是重点及难点，而学生对指针的理解又往往感到困难，对指针的应用更无法灵活掌握，而大多数教材又把指针与数组分隔教学，使得学生对数组概念先入为主，造成对C语言的精华——指针的应用更为困难。因而作者在讲授C语言程序设计时，先讲指针，后讲数组，并且这些内容都在第1章讲授，使指针的概念及应用融合在整个教材中，达到了非常好的效果。本书共分11章，内容包括：C语言基本概念、控制结构、模块化程序设计、变量的存储类型及编译预处理、指针与数组、指针与函数、结构与共用体、位运算、文件、程序设计常用数据算法简介、实用程序设计等。本书的编写力求在体系结构上安排合理、重点突出、便于掌握；在语言叙述上注重概念清晰、逻辑性强、通俗易懂、便于自学。书中各章后均配有一定数量的选择题、填空题、程序阅读题及编程题，这些题目的参考答案安排在与其配套的习题集上，学生在经过自己的思考后，可以参照答案，以便正确地理解和掌握基本概念和各知识点。

在本书的编写过程中参阅了大量有关C语言程序设计的书籍与文献，编者在此表示诚挚的感谢。

由于编者水平有限，书中难免存在缺点和错误，真诚希望广大读者批评指正。

编著者

目　录

第1章　C语言基本概念

【重点提示】 本章论述C语言程序的组成特点，标识符的分类及取名规则，常量与变量、运算符、表达式、输入输出语句、地址、指针与数组等基本概念。

本章的难点：自加、自减运算符，运算符的优先级别，输入输出语句的格式，指针的概念及指针的运算，指针对数组的引用。

要求掌握C语言的特点及书写规则，掌握基本类型中的整型、实型和字符型常量的表示和变量的定义及初始化，掌握运算符的优先级与结合性，各种表达式及求解规则，掌握表达式、语句、复合语句、空语句等基本语句的用法和输入输出函数的使用方法，掌握地址与指针的基本概念，掌握指针对数组的初步引用方法，会上机调试、运行简单的顺序结构的程序。

1.1　C语言的发展简史及程序设计的一般概念

当今，计算机已广泛应用于社会生活的各个领域，成为大众化的现代工具。计算机是一种具有内部存储能力、由程序自动控制的电子设备。人们将需要计算机做的工作写成一定形式的指令，并把它们存储在计算机的内部存储器中，当人们给出命令之后，它就按指令操作顺序自动进行。人们把这种可以连续执行的一条条指令的集合称为程序。可以说，程序就是人与机器进行“对话”的语言，也就是我们常说的程序设计语言。

目前，正在使用的计算机程序设计语言有上百种，有些语言是面向机器的，如二进制语言，而多数是面向问题的语言。面向问题的语言都被称为计算机的高级语言，如C语言、Pascal语言、Foxbase等，当然我们也把Visual C++、Visual Basic等称为面向对象的语言。这些语言都是用接近人们习惯的自然语言和数学语言作为语言的表达形式，所以人们学习和操作起来感到十分方便。

在各种计算机语言中，C语言可以说是较为成功的。C语言是在UNIX操作系统的研制和开发过程中诞生的。1970年，在美国电话与电气公司(AT &T)的贝尔实验室中工作的丹斯·里奇(Dennis Ritche)先生，在B语言的基础上写成了C语言，并于1972年投入了使用。他同肯·汤普森(Ken Thompson)于1973年用C语言重写了UNIX操作系统，并且成了以后UNIX各种版本的发展基础。

对于计算机本身来说，它并不能直接识别由高级语言编写的程序，它只能接受和处理由代码0和1构成的二进制指令或数据。由于这种形式的指令是面向机器的，因此也称为机器语言。

我们把由高级语言编写的程序称为源程序，把由二进制代码表示的程序称为目标程序。如何把源程序转换成机器能够接受的目标程序，软件工作者编制了一系列软件，通过这些软件可以把用户按规定语法写出的语句一一翻译成二进制的机器指令。这种具有翻译功能的软件称为编译程序。每一种高级语言都有与它对应的编译程序。

目前的程序设计一般可分为非结构化程序设计和结构化程序设计、面向对象的程序设计；C语言是结构化程序设计语言。程序设计的过程一般包括：

(1)问题的提出、要求及所采用的数据结构；

(2)算法的确定,程序的编制;

(3)程序的调试及修改;

(4)整理并写出文档资料。

结构化程序设计由三种结构组成:顺序结构、选择结构、循环结构。

我们所写的每条C语句序列称C源程序,它的后缀为c。C源程序经过编译(compile)后生成一个后缀为obj的二进制文件,最后由连接程序(link)把此obj文件与C语言提供的各种库函数连接起来生成一个exe文件,在DOS状态下就可运行该文件。

程序的设计过程如图1.1所示。

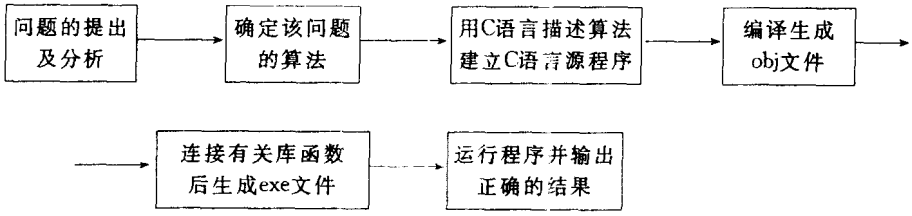

图1.1 程序设计过程

1.2 C语言的优点及地位

现在人们之所以广泛地采用C语言来完成各种编程任务,是因它具有下列独特优点:

(1)C语言是一种通用的结构化程序设计语言,用C语言编写的程序由多个模块(函数)组成,每个模块完成一项功能。

(2)C语言的语句简洁、灵活,使用起来得心应手,便于初学者掌握。

(3)C语言具有丰富的运算符,使用起来简单精炼,生成的机器代码质量高,内存占用少,运行速度快。

(4)C语言的数据类型丰富,可进行复杂的数据结构运算,C语言的指针数据类型的使用具有更大的优点,它使参数传递更简单、迅速,节省内存。

(5)C语言具有位(bit)操作能力,可直接对计算机硬件的物理地址进行访问,它既是高级语言又兼有低级语言的功能。

(6)简洁、高雅、灵活,移植性好,应用面广。

一般来说,汇编语言处于硬件与系统之间,而数据库语言Cobol等处于用户与软件之间,Pascal语言更适合于系统与软件之间,C语言偏向于硬件与系统方面,并且能完成其他高级语言能完成的任何工作。因而,更确切地说,C语言是一种中级语言。

1.3 C语言的特点

C语言的特点主要有以下几点:

(1)C语言具有汇编语言的精华,引入结构、指针、地址、位运算,可直接访问硬件,可对位、地址进行操作。

(2)C语言是一种结构化程序设计语言,即顺序、选择(if…else、switch…case)、循环(for、while、do…while)这三种结构,易于理解,便于软件的维护。

(3)C 语言是一种模块化结构的程序设计语言，例如程序由函数构成、模块间可相互调用，为集体开发软件工程技术提供方便。

(4)C 语言具有丰富的运算符，如＋、－、＊、/、%、&&、||、!、&、|、～、∧、＞＞、＋＋、－－ 等。

(5)可使用#define、#include 等编译预处理语句、宏定义、条件编译。

(6)C 语言具有丰富的数据类型(整型、实型、双精度型、字符、数组、指针、结构体、共用体)，具有很强的数据处理能力。

(7)C 语言具有较高的移植性，描述准确、目标程序质量高、语言紧凑、存储空间小。

1.4 C 语言的结构与书写要求

一、C 语言的结构

(1)C 语言是结构化的程序设计语言。

(2)C 语言程序由函数组成，函数间可以嵌套、递归调用。程序中必须有一个函数名为 main()的函数，且只能有一个 main()函数，其中“{”和“}”分别表示程序执行的起点与终点或程序块的起点与终点。

(3)C 语言程序可由多个文件组成。

(4)C 语言程序书写格式自由，一行内可写几个语句，一个语句也可写在多行上，但 C 语言程序中区分大小写字母。用 C 语言写成的函数结构如图 1.2 所示。

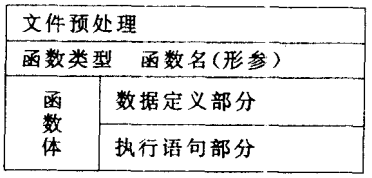

图 1.2 C 语言的函数结构示意图

【例 1-1】 程序结构示例。

```
#include<stdio.h>              /*文件预处理*/
int add( int x,int y)          /*函数名*/
{                              /*函数体开始*/
  int z;                       /*数据定义部分*/
  z=x+y;                       /*执行语句部分*/
  return z;                    /*函数返回值*/
}                              /*函数体结束*/
```

在程序中可以对程序进行注释，注释部分必须用符号“/*”和“*/”括起来。“/”和“*”之间不可以有空格。注释可以用西文，也可以用中文。注释可以出现在程序中任意合适的地方。注释部分对程序的运行不起作用。在注释中可以说明变量的含义、程序段的功能，以帮助人们阅读程序。因此，一个好的程序应该有详细的注释。

在上例中，如要在机器上运行必须有一个 main()函数，通过 main()调用函数 add()，例如完整的程序代码为：

```
#include<stdio.h>
int add( int x,int y)
{
  int z;
  z=x+y;
  return z;
}
void main()
{  int a=1,b=2,z;                    /* 变量的定义及赋初值 */
   z=add(a,b);                       /* 函数调用 */
   printf("%d+%d=%d\n",a,b,z);       /* 把结果显示在屏幕上 */
}
```

此程序的执行结果为 1+2=3。

请读者仔细思考，有关问题将在以后学到。

二、编程时应养成良好的书写习惯

良好的程序书写习惯有助于对程序的书写、理解和记忆，因而编程时应养成良好的书写习惯，并注意以下各点：

①用“;”作为语句的结束标记，但循环体、main()、#include< >、#define 不是语句，因而后面不能用“;”号，例如，

```
int i;
for(i=1;i<=5;i++)                   /* 此位置无";"号 */
   printf("Hello\n");
```

②“/*”和“*/”、“{”和“}”必须成对使用。

③一定要有预处理语句，如 #include<stdio.h>、#include<math.h>、#include<graphics.h>等。

④程序用小写字母，标识符用大写，如#define PI 3.14159。

⑤为了方便阅读，书写最好规范。

⑥程序中必须有一个 main()函数，由 main()函数调用一般函数，且文件中只能有一个 main()函数。

⑦main()函数也可以递归。

1.5 简单的C语言程序

C语言程序可由系统函数及自定义函数组成，程序可以很复杂，也可以很简单，但即使很简单的程序也应有 main()及“{”、“}”。

【例 1-2】 最简单的C语言程序。

```
/* 最简单的C语言程序 */
#include<stdio.h>
void main()
```

```
{
    ;
}
```

此程序没有什么意义，执行后什么也没做，可以认为是最简单的C语言程序。

【例1-3】 C语言程序示例。

```
/* 功能：打印字符，在屏幕上输出：Hello，world */
#include<stdio.h>
void main()
{
 printf("Hello，world\n");          /* \n 为换行符 */
}
```

解 此程序执行后在屏幕上输出：

```
Hello，world
_
```

1.6 关键字与标识符

在C语言中，标识符可用作变量名、符号名、函数名和后面将要学到的数组名、文件名以及一些具有专门含义的名字。合法的标识符由字母、数字和下划线组成，并且第一个字符必须为字母或下划线，不能跨行书写，自定义标识符不能与关键字同名。下面的标识符都是合法的：

are、PI、liu_123、liu、e_array

以下都是非法的标识符：

456P、cade-y、w.w、a&b

在C语言的标识符中，大写字母和小写字母被认为是两个不同的字符，因此page和Page是两个不同的标识符。

对于标识符的长度(即一个标识符允许的字符个数)，一般的计算机系统规定取前8个字符有效，如果多于8个字符，多余的字符将不被识别。如num12345和num123456在计算机系统内则被认为是相同的标识符num12345。有些系统允许取较长的名字，读者在取名时应当了解所用系统的具体规定。

C语言的标识符可以分为以下3类。

一、关键字

关键字也叫保留字，是C语言中用来表示某种特殊含义的英文单词，如include、int、char、for等。关键字含义特定，不允许随意书写或拼写错误，它们也不能再用作变量名或函数名。C语言中的关键字有auto、break、case、char、const、continue、default、do、double、else、enum、extern、float、for、goto、if、int、long、register、return、short、signed、sizeof、static、struct、switch、typedef、union、unsigned、void、volatile、while。

二、预定义标识符

有些标识符在C语言中也有特定的含义，如C语言提供的库函数的名字(如printf)和预

编译处理命令(如 define)等。C 语言语法允许用户把这类标识符另作他用,但这将使这些标识符失去系统规定的原意。鉴于目前各种计算机系统的 C 语言都一致把这类标识符作为固定的库函数名或预编译处理中的专门命令使用,因此为了避免误解,建议用户不要把这些预定义标识符另作他用。

三、用户标识符

由用户根据需要定义的标识符称为用户标识符。用户标识符一般用来给变量、函数、数组或文件等命名。程序中使用的用户标识符除要遵循取名规则外,还应注意做到"见名知义",即采用具有相关含义的英文单词或汉语拼音,以增加程序的可读性。

如果用户标识符与关键字相同,程序在编译时将给出出错信息;如果与预定义标识符相同,系统并不报错,只是该预定义标识符将失去原定含义,代之以用户确认的含义,或者运行时发生错误。

【例 1-4】 下列各组字符序列中,可用作 C 语言程序标识符的一组是(　　)。

(A) S.b,sum,auerage,_above　　(B) Class,day,lotus_1,2day

(C) #md,&12x,month,student_n1　　(D) D56,r_1_2,name,_st_1

分析 C 语言规定标识符只能由字母、数字和下划线三种字符组成,而且第一个字符必须是字母或下划线,因而答案为(D)。

【例 1-5】 下列字符序列中,不可用作 C 语言标识符的是(　　)。

(A)b70　　(B)#ab　　(C)_symbo　　(D)a_1

分析 C 语言规定标识符只能由字母、数字和下划线三种字符组成,而且第一个字符必须是字母或下划线,因而答案为(B)。

四、常量

在程序运行过程中,其值不能被改变的量,称为常量。常量通常分为普通常量、无名常量及符号常量。在 C 语言中,常量有不同的类型,有整型常量(int)、实型常量(float 或 double)、字符常量(char)和字符串常量;即使是整型常量也还有短整型(short int)、长整型(long int)和无符号型(unsigned int)。

(一)普通常量

整型常量只用数字表示,不能带小数点,如 12、-1、0 等都是合法的整型常量。实型常量也称数值型常量,有正、负之分。实型常量通常用带小数点的数表示,如 3.14159、2.71828、0.0、0.54 等。而字符常量用单引号括起来,如'A'是字符型常量。字符串常量用双引号括起来,如"asfgTYN"是合法的字符串常量。

由此可见,常量的类型从字面形式来看即可区分是整型常量、字符常量还是实型常量。C 编译程序就是以此来确定数值常量的类型的。

可以用一个符号名来代表一个常量,但是这个符号名在程序中必须先进行特别的"指定",这种"指定"就叫做定义常量,即给常量分配存储空间。

- 整型常量定义举例:const int y=4;
- 实型常量定义举例:const float x=2.1;
- 字符常量定义举例:const char ch='A';

应注意常量值必须在常量定义时给定,常量的值一旦指定,在程序中就不能再改变。

(二)无名常量

在C语言中,整型常量可以用十进制、八进制和十六进制格式表示。十进制常量用一串连续的数字来表示,如32767、-32768、0等。八进制数用数字0开头(注意:是数字0,不是字母o),例如,05、012、01都是八进制数,它们分别代表十进制数5、10、1。因此,在C程序中不能在一个十进制整数前面加零。注意:八进制数只能用合法的八进制数字表示,如不能写成018,因为数字8不是八进制数字。十六进制数用数字0和字母x(或大写字母X)开头,例如0x10、0xff、0X8均是十六进制数,它们分别代表十进制数16、165、8。注意:十六进制数只能用合法的十六进制数字表示,字母a、b、c、d、e、f既可用大写字母也可用小写字母。

十进制常量表示方法示例:123;

八进制常量表示方法示例:0123;

十六进制常量表示方法示例:0x123。

应注意:常量的长度及范围与机器类型有关,例如,

16位机整数:-32768~32767

32位机整数:-2147483648~2147483647

【例1-6】 在C语言中,错误的int型的常数是(　　)。

(A)32768　　(B)0　　(C)037　　(D)0xAF

分析 int型常数的范围是-32768至32767,32768已超出int型数据的范围,(C)为八进制数,(D)为十六进制数,所以答案为(A)。

【例1-7】 下列常数中不能作为C语言常量的是(　　)。

(A)0xA5　　(B)2.5e-2　　(C)57　　(D)0582

分析 (D)为八进制表示的常数,但(D)包含了数字8,所以答案为(D)。

【例1-8】 下列形式的常数中,C程序不允许出现的是(　　)。

(A)45　　(B)±123　　(C)25.6e--2　　(D)4e3

分析 C程序允许出现的常数为一确定的整数、实数(可用小数形式或指数形式)等,但不允许出现类似±123的数,所以答案为(B)。

(三)符号常量

符号常量的定义格式为:

　　#define 符号名　符号常量

例:#define PI 3.14159

它的含义是指凡是在程序中出现符号名的地方,都用符号常量替换。

注意:①符号常量在程序中替换时,不作语法检查;

②符号名、符号常量之间无等号,符号常量后无分号;

③符号名一般大写。

【例1-9】 符号常量在程序中的应用:计算圆面积。

```
#include<stdio.h>
#define PI 3.14159          /*定义符号名 PI 为 3.14159*/
void main()
{
   float  s,r=5.0;
   s=PI*r*r;
```

```
    printf("s=%f\n",s);
}
```

解 程序的执行结果为 s=78.539749。

1.7 变　　量

变量是指在程序运行过程中其值可以发生变化的量；通常是用来保存程序运行过程中的输入数据、计算获得的中间结果和最终结果。

变量的取名规则与用户标识符相同，其中的英文字母常用小写。C 语言系统本身也使用变量名，一般都是以下划线"_"开头的，所以为了区别，用户程序中的变量名一般都不以下划线"_"开头。给变量取名时，为了便于阅读和理解程序，一般都用代表变量值或用途的标识符，可以是英文单词或缩写，也可以是中文拼音字母或缩写。

当程序运行时，每个变量都要占用连续的若干个字节，所占用的字节数由变量的数据类型确定。其中第 1 个字节的地址称为变量的地址。C 语言规定，程序中变量的地址是用"& 变量名"来表示的。

一、变量的数据类型及其定义

C 语言规定，变量既可以是任何一种数据类型，例如整型、短整型、长整型、无符号整型、无符号短整型、无符号长整型、单精度实型、双精度实型、字符型等基本类型，也可以是数组型、结构体型、共用体型等构造类型和指针型、枚举型。

通常把具有某种数据类型的变量就叫做该类型变量，例如短整型变量、长整型变量、单精度型变量、双精度型变量、字符型变量，等等。基本数据类型符及其含义如表 1.1 所示。

表 1.1　基本数据类型符

数据类型	数据类型符	占用字节数	数值范围
整型	int	2	−32768～+32767
短整型	short	2	−32768～+32767
长整型	long	4	−2147483648～2147483647
无符号整型	unsigned int	2	0～65535
无符号短整型	unsigned short	2	0～65535
无符号长整型	unsigned long	4	0～4294967295
单精度实型	float	4	-10^{38}～10^{38}
双精度实型	double	8	-10^{308}～10^{308}
字符型	char	1	−128～+127

需要注意的是，字符串只能是常量，C 语言中没有字符串变量。

每个变量在使用前都必须定义，定义的内容之一就是说明其数据类型。

定义变量数据类型的语句格式如下：

数据类型符　变量名 1，变量名 2，……，变量名 n；

表 1.1 中，整型和无符号整型所占用的字节数是随计算机的不同而不同的，在大部分计算机上都是和短整型占用相同的字节数(2 个字节)。

例如，定义两个整型变量及一个字符变量可写为：

```
int x,y;
char ch;
```

对变量进行定义时，要注意下列几点：

①对变量的定义可以放在函数之外，也可以放在函数体中或复合语句中。如果是放在函数体或复合语句中，则必须集中放在最前面。

②被定义为整型（包括 int、short、long）的变量，若其值在－128～127 之间，可以当做字符型变量使用；被定义为无符号整型（包括 unsigned int、unsigned short、unsigned long）的变量，若其值在 0～255 之间，也可以当做字符型变量使用；被定义为字符型的变量，既可以当做整型（包括 int、short、long）变量使用，其值在－128～ 127 之间，也可以当做无符号整型（包括 unsigned int、unsigned short、unsigned long）变量使用，其值在 0～255 之间。

二、变量的存储类型及其定义

当我们说明了一个变量的数据类型后，C 语言编译系统就要给该变量安排若干个字节，用来存放该变量的值。我们知道，在计算机的寄存器和内存中都可以存放数据，而内存中又可以分为一般数据区和堆栈区。我们把变量存放在何处称为变量的存储类型。用户可以通过说明变量的存储类型来选择变量的具体存储地点。

定义变量存储类型的语句格式如下：

存储类型符　数据类型符　变量名 1，变量名 2，……，变量名 n；

存储类型符及其含义见表 1.2 所示。

表 1.2　存储类型符表

存储类型	存储类型符	存储地点
自动型	auto	内存堆栈区
寄存器型	register	CPU 的通用寄存器
静态型	static	内存数据区
外部参照型	extern	

（一）自动型

自动型又称堆栈型。自动型变量分配在内存的堆栈区，堆栈区内存在程序运行中是重复使用的。当某个函数中定义了自动型变量，C 语言就在堆栈区给该变量分配字节用于存放变量的值。当退出该函数时，C 语言就释放该变量，即从堆栈区中收回分配给该变量的字节，以便重新分配给其他自动型变量。这样做的目的是节省内存。

用户对于只在某函数中使用的变量应该将其定义成自动型变量，以便充分利用内存。当定义某个或某些变量时，省略了存储类型符，系统将默认为自动型变量。

（二）寄存器型

寄存器型变量分配在 CPU 的通用寄存器中。由于 CPU 具有的通用寄存器数量有限，所以 C 程序中允许定义的寄存器型变量一般以 2 个左右为宜。如果定义为寄存器型变量的数目超过所提供的寄存器数目，编译系统会自动将超出的变量设为自动型变量。对于占用字节数多的变量，如 long、float、double 类型的变量不能定义为寄存器型变量。

寄存器型变量一般是在函数中定义的，退出该函数后就释放它所占用的寄存器。

（三）静态型

静态型变量分配在内存的数据区中，它们在程序开始运行时就分配了固定的字节，在程序

运行过程中不释放。只有程序运行结束后，才释放所占用的内存。

（四）外部参照型

C 语言允许将一个源程序清单分放在若干个程序文件中，采用分块编译方法编译生成一个目标程序。其中每个程序文件称为一个编译单位。

外部参照型变量专用于多个编译单位之间传递数据。当编译单位甲中要使用在编译单位乙中定义的变量时，编译单位甲就要说明该变量是外部参照型，以便 C 语言编译系统在编译单位甲之外的其他编译单位中寻找该变量的定义。将某个变量说明为其他编译单位定义的变量的方法就是将它说明为外部参照型。被说明成外部参照型的变量的存储类型和数据类型都由定义该变量的那个编译单位来说明。

1.8 整型变量

一、整型变量的定义

本节只介绍基本型的整型变量。

整型变量可以分为基本型(int)、短整型(short int)、长整型(long int)和无符号型(unsigned)四种。

基本型的整型变量在内存中占 2 个字节，用类型名关键字 int 进行定义。

定义形式如下：

```
int 变量名 1;int 变量名 2;……;int 变量名 n;
```

或

```
int 变量名 1,变量名 2 ,……,变量名 n;
```

例如：int i;int j;int k;

或

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

应注意：① 一个定义语句必须以一个“;”号结束。在一个定义语句中也可以同时定义多个变量，变量之间用逗号隔开。

一般计算机为基本型整型变量开辟 2 个字节（16 个二进制位）的内存单元，并按整型数的存储方式存放数据。整型的变量只能存放整型数值。基本型整型变量中允许存放的数值范围是：－32768～32767。

②当在程序中用以上方式定义变量 i、j 和 k 时，编译程序为变量 i、j 和 k 都开辟存储单元，但是并没有在存储单元中放置任何初值，因此在这些存储单元中，原有的信息并没有被清除，这时变量中的值是无意义的。

C 语言规定，可以在定义变量后再给变量赋值，如：

```
int x,k;                /*此处定义 x、k 为整型变量*/
x=2,k=3;                /*定义变量后才可给变量赋值*/
```

也可把这两个语句合成一个，定义的同时给变量赋值，这叫做变量的初始化，例如：

```
#include<stdio.h>
void main()
{ int x=2, y=3;     /*定义 x、y 为整型变量，同时分别给它们赋初值 2、3*/
```