第1章　C语言概述

C 语言是一种计算机程序设计语言，它由美国贝尔实验室的 Dennis Ritchie 于 1972 年推出。C 语言简单、可靠和易于使用，是学习其他语言的基础。如果没有 C 语言的知识，学习 C++或 JAVA 等语言就会变得很难。现在流行的操作系统，如 Windows、UNIX、Linux 等，基本上都是由 C 语言开发的，因为 C 语言的执行效率很高。本章首先介绍程序设计语言及其发展历史，C 语言的历史，C 语言的标准。然后通过剖析简单的 C 源程序，阐述 C 程序的基本构成，这也是本章的学习重点。同时本章还简单介绍了 C 程序的运行步骤及常见的 C 集成编译器。本章仅仅是 C 语言的初步介绍，大部分内容浅显易懂，具体和深入的技术细节将在后续章节中介绍。

1.1　程序设计语言及其发展

人们经常使用语言或文字来表达思想、交流和互通信息，如汉语、英语等。人类相互交流信息所用的语言被称为自然语言，但是计算机目前还不能识别人类的自然语言。计算机能够识别的是计算机程序。计算机程序(Computer Programs)是为完成一项特定任务而用计算机语言编写的一组指令序列。把解决一项任务的思路、方法和步骤最终落实为计算机程序的编写就是程序设计。用于书写计算机程序的语言叫程序设计语言(Programming Languages)，它是人与计算机之间进行信息交流的工具。

程序设计语言的种类非常多，总的来说可以分为机器语言、汇编语言和高级语言这 3 大类。

1.1.1　机器语言

机器语言(Machine Languages)是用二进制代码表示的计算机能直接识别和执行的一种机器指令的集合。它是计算机的设计者通过计算机的硬件结构赋予计算机的操作功能。机器语言具有灵活、直接执行和速度快等特点。

1. 机器指令

机器指令是指挥计算机完成某一基本操作的命令，由硬件电路设计决定，因而也被称为硬指令。机器指令是由一组能被计算机接受的 0 和 1 组成的二进制代码。机器指令由操作码和地址码组成，规定了要求计算机完成的操作及其操作的对象(数据或存储单元地址)。

2. 指令系统

每台计算机所具有的特有的、全部指令的集合构成该 CPU 的指令系统。不同的 CPU 具

有不同的指令系统。

3. 机器语言程序

机器指令的集合构成了机器语言，用机器语言编写的程序就是机器语言程序。计算机所能识别的语言只有机器语言，但机器语言非常难于记忆和识别。通常人们编程时，不采用机器语言，而采用汇编语言和高级语言。

1.1.2　汇编语言

汇编语言(Assembly Languages)是面向机器的程序设计语言。在汇编语言中，用助记符代替机器指令的操作码，用地址符号或标号代替指令或操作数的地址，如此就增强了程序的可读性并且降低了编写难度。像这样符号化的程序设计语言就是汇编语言，因此亦称为符号语言。使用汇编语言编写的程序，机器不能直接识别，还要由汇编程序或者叫汇编语言编译器转换成机器指令。汇编程序将符号化的操作代码组装成处理器可以识别的机器指令，这个组装的过程称为组合或者汇编。因此，有时候人们也把汇编语言称为组合语言。

1. 汇编指令

汇编指令是用助记符号表示的机器指令，它与机器指令一一对应。

2. 汇编程序

计算机不能直接识别汇编指令，要让机器接受汇编指令还需要有一个将汇编指令翻译为机器指令的过程，这个过程称为汇编。汇编程序就是把汇编语言源程序翻译成机器语言程序的一种系统软件。IBM PC 机中的汇编程序包括 ASM 和 MASM 这两种。ASM 称为小汇编程序，它只需要较小的存储区。MASM 称为宏汇编程序，它需要的存储区较大，但功能较强，且具有宏汇编能力。ASM 则不具备这种能力。

3. 伪指令

伪指令就是向汇编程序提供如何进行汇编工作的命令，也叫汇编控制命令。伪指令没有对应的机器指令，汇编时不产生机器码。

4. 汇编语言

汇编指令、伪指令、宏指令和汇编程序一起组成了汇编语言。汇编语言直接面向机器，用汇编语言编制的程序简洁、快速，常用于对运行速度要求较高的实时控制等场合。用汇编语言编制的用户程序称为汇编语言源程序。汇编语言的实质和机器语言是相同的，都是直接对硬件操作，但指令采用了英文缩写的标识符，更容易识别和记忆。而其所占用的存储空间和执行速度与机器语言相仿。

1.1.3　高级语言

高级语言(Higher-level Languages)主要是相对于汇编语言而言，它是以较接近自然语言和数学公式的形式编程，基本脱离了计算机的硬件系统，用人们更易于理解的方式编写程序。

高级语言并不是特指某一种具体的语言，而是包括了很多编程语言，如Fortran语言、Basic语言、C语言、C++、JAVA、FoxPro等，这些语言的语法和命令格式各不相同。

高级语言所编制的程序不能直接被计算机识别，必须经过转换才能被执行。按照转换方式可将高级语言分为两类。

(1) 解释(Interpret)类。执行方式类似于人们日常生活中的“同声翻译”，应用程序源代码一边由相应语言的解释器“翻译”成目标代码(机器语言)，一边执行。因此效率比较低，而且不能生成可独立执行的可执行文件，应用程序不能脱离其解释器。但这种方式比较灵活，可以动态地调整、修改应用程序，如 Basic 语言便是采用这种方式。

(2) 编译(Compile)类。编译是指在应用源程序执行之前，就将程序源代码一次性“翻译”成目标代码(机器语言)，然后再在机器上运行目标程序，因此其目标程序可以脱离其语言环境独立执行，使用比较方便且效率较高。但应用程序一旦需要修改，必须先修改源代码，再重新编译生成新的目标文件才能执行。如果只有目标文件而没有源代码，修改会很不方便。现在大多数的编程语言都是编译型的，如 C 语言。

1.2　C 语言的历史

C 语言是一种结构化程序设计语言。结构化程序设计方法主要由以下 3 种逻辑结构组成。

(1) 顺序结构：顺序结构是一种线性、有序的结构，它依次执行各语句模块。

(2) 选择结构：选择结构是根据条件成立与否选择程序执行的通路。

(3) 循环结构：循环结构是重复执行一个或几个模块，直到满足某一条件为止。

采用结构化程序设计方法，程序结构清晰，易于阅读、测试、排错和修改。由于每个模块执行单一功能，模块间联系较少，使程序编制比过去更简单，程序更可靠，而且增加了可维护性，每个模块可以独立编制、测试。与大部分现代程序设计语言类似，C 语言来源于 ALGOL 语言，ALGOL 语言是最先使用块结构的程序语言。ALGOL 没有在美国得到普遍认可，但在欧洲却得到了广泛的应用。ALGOL 语言给计算机科学界带来了结构化程序设计的概念。20 世纪 60 年代，计算机科学家 Corrado Bohm、Guiseppe Jacopini 和 Edsger Dijkstra 使这一概念进一步大众化。

在 C 语言诞生之前还存在着一系列相关的程序语言。1967 年，Martin Richards 开发了一种称为 Basic Combined Programming Language 的语言，简称为 BCPL(基本组合程序设计语言)的计算机语言。Ken Thompson 在 1970 年开发了一种名为 B 的类似语言。B 语言是 UNIX 操作系统的第一个版本的开发语言。随后，1972 年，Dennis Ritchie 设计了 C 语言，它继承了 ALGOL、BCPL 和 B 语言的许多思想，并加入了数据类型的概念以及其他功能强大的特性。由于 C 语言是与 UNIX 操作系统一起被开发出来的，因此它与 UNIX 有着很强的关联。

多年以来，C 语言主要用于科研环境下，但最终随着多种商用 C 编译器的发布以及 UNIX 操作系统的不断流行，在计算机界开始获得广泛支持。今天，C 语言可以运行在多种操作系统和硬件平台下。

1980 年，Bjarne Stroustrup 开始用一种新的语言工作，这种语言被称作“带类的 C 语言”。

它增加了大量的新特性，全面改进了 C 语言，其中最重要的特性就是类。这种语言经过改进和扩充，最终成为 C++。

1.3　C 语言的标准

传统的 C 语言是 1972 年的版本。1978 年，Brian W. Kernighan 和 Dennis M. Ritchie 编写了著名书籍《The C Programming Language》，该书对 C 语言进行了文档化和推广。1983 年，美国国家标准协会(ANSI)开始制定 C 语言的标准，并于 1989 年 12 月通过。1990 年，国际标准化组织(ISO)接受了 ANSI 提出的标准，C 语言的这个版本称为 C89。C89 在 1995 年进行了一些微小的调整，改进后的版本称为 C95。后来一些重要的版本更新发生在 1999 年，新标准命名为 C99(ISO/IEC 9899：1999)。我国于 1994 年 12 月 7 日发布了程序设计语言 C 标准 GB/T 15272-1994。

2011 年 12 月 8 日，ISO 发布了新的 C 语言的新标准——C11，之前被称为 C1X，官方名称为 ISO/IEC 9899：2011。新的标准提高了对 C++的兼容性，并将新的特性增加到 C 语言中。新功能支持多线程，基于 ISO/IEC TR 19769：2004 规范下支持 Unicode，提供更多用于查询浮点数类型特性的宏定义和静态声明功能。

1.4　C 语言的程序结构

1.4.1　简单的 C 语言程序剖析

学习一门新程序设计语言的唯一途径就是使用它编写程序。下面引入 C 语言的设计者 Brian Kernighan 和 Dennis Ritchie 合著的《The C Programming Language》一书中的第一个示例程序，使用该程序打印出字符串“hello,world”。尽管这个编程练习很简单，但对于初学 C 语言的人来说，它仍然可能成为一大障碍。因为要实现这个目的，编程者首先必须编写程序文本，然后成功地编译，并加载、运行，最后输出结果。掌握这些操作细节以后，其他的事情就比较容易了。

【例 1.1】 问候程序，输出字符串“hello,world”。

源程序：

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

运行结果：

```
hello,world
```

程序分析：

一个 C 语言程序，无论大小，都是由函数组成的。函数中包含一些语句，以指定所要执行的操作，本例中函数的名字为 main。通常情况下，C 语言并没有限制函数必须取一个什么样的名字，但 main 是个特殊的函数。main 函数称为主函数，每个程序都以 main 函数为起点开始执行，这就意味着每个程序都必须包含一个 main 函数，函数体须由{　}括起来。

C 语言编译系统将一些常用的操作或计算功能定义成函数，如 printf、scanf、sqrt、fabs 等。这些函数称为标准库函数，其声明部分放在指定的以.h 为扩展名的头文件中。例如，存放标准输入/输出库函数声明的头文件名为 stdio.h，在使用系统库函数时须将对应的头文件包含进来。#include <stdio.h>是一个预处理命令，以#号开始，其功能是包含文件“stdio.h”。

printf("hello, world\n");语句中，printf 是一个用于打印输出的库函数，在本例中被主函数 main()所调用，它用于打印用双引号括住的字符串。用双引号括住的字符序列叫作字符串或字符串常量，“hello,world\n”就是一个字符串，是 printf 函数的参数。“hello, world!”是原样输出的字符串序列，printf 函数不会自动换行，\n 是个换行符。遇到它时输出将换行。

例如，语句

```
printf("I see, \n I remember!");
```

其输出如下。

```
I see,
I remember!
```

printf("hello, world\n");语句最后以分号(;)表示该语句结束。

在继续讨论其他更多的示例之前，应必须注意很重要的一点：C 语言是区分大小写字母的(即大小写敏感)。例如，printf 和 PRINTF 并不相同。

【例 1.2】 求两个整数 10 和 20 的和并输出结果。

源程序：

```
/*
功能：计算两个数的和，并输出
*/
#include <stdio.h>   /* 包含头文件 stdio.h */
main( )
{
     int a, b, sum; /* 定义变量 */
     a=10; /* 给变量 a 赋整数值 10 */
     b=20; /* 给变量 b 赋整数值 20 */
     sum=a+b; /* 求和 */
     printf("sum=%d\n", sum); /* 输出 sum 的值 */
}
```

运行结果：

```
sum=30
```

程序分析：

在本程序中，/*...*/是注释(Comments)，不是程序的必需部分，在程序执行时注释不起任何作用。注释的作用是增加程序的可读性，因此，适当地在程序中加以注释，是一种良好的程序设计风格。C 语言的注释方法有以下两种。

(1) 块注释

形式如下。

```
/*
注释内容
*/
```

跨多行的注释语句，适用于注释多行，“/*”和“*/”之间的内容为注释内容。

(2) 行注释

形式如下。

```
/*  注释内容  */
```

/* ....注释内容..... */放在一行上，通常放在语句之后。

C99 标准支持“//”行注释(这个特性实际上在 C89 的很多编译器上已经被支持了)。

形式如下。

```
//注释内容
```

作用范围是“//”后面开始至本行结束。

示例代码如下。

```
int a, b, sum;   // 定义变量
```

注释可以出现在程序中的任何位置，注释中的任何内容都不会被计算机执行。

程序中 int a，b，sum; 语句定义 a，b，sum 为 int 类型变量，在 C 语言和其他大部分编程语言中，使用变量(Variables)来存储数据。每个变量都由一个变量名来标识。每个变量必须有一种类型(Types)，用于表示它所存储的是哪种类型的数据。C 语言的基本数据类型分为整型、实型、字符型等。变量必须先定义，然后才能使用。

变量定义的一般形式：类型标识符、变量名列表。

变量定义后其初值一般是不确定的，不能直接使用。示例代码如下。

```
int a; printf("%d\n",a);       /*****错误!*****/
int a; a=10;printf("%d\n",a); /*****正确*****/
```

在定义变量的同时，也可以同时对变量赋初值，称为变量的初始化。示例说明如下。

语句 int sum=0; 定义了 sum 为 int 类型变量，为 sum 赋初值 0。

printf("sum=%d\n"，sum)；语句输出 sum 的值，C 语言中的 printf 是个用得很普遍的命令，称为格式输出函数。其基本命令格式如下。

```
printf(格式控制, 输出列表);
```

其中，格式控制部分用双引号引起来，里面通常包含两种信息。一种是普通字符，普通字符按原样输出；另一种就是以“%”开头的格式说明，它的作用是将数据按指定的格式输出。例如，“%d”表示对应的输出值(即 sum 的值)以十进制整数形式显示，其余都是普通字符。所以程序执行后输出的结果如下：

```
sum=30
```

对应关系如图 1.1 所示。

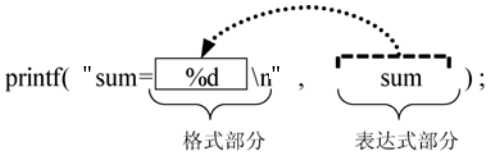

图 1.1　printf 函数输出示例图

如果程序执行后想得到如下形式的输出结果。

```
10+20=30
```

则程序中的 printf 语句可改写如下。

```
printf("%d+%d=%d\n", 10,20,sum);
```

其中 10，20，sum 为输出列表，各表项之间用逗号分隔。由于有 3 个输出值，所以用 3 个格式说明符与之一一对应，如图 1.2 所示。

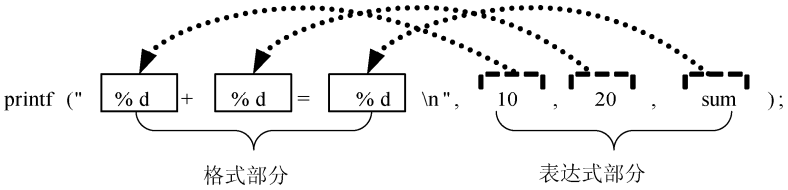

图 1.2　printf 函数输出示例图

【例 1.2】程序只能计算 10 加 20 的和，因为程序中规定了 a 和 b 的值。如果要计算其他两个数的和，则需要修改程序。程序在运行时通过键盘操作输入需要求和的两个数，然后进行计算求和输出最为理想。

【例 1.3】 求任意两个整数的和并输出结果。

源程序：

```
#include <stdio.h>
main( )
{
     int a, b, sum;       /* 定义变量 */
     scanf("%d", &a);     /* 输入第一个整数 */
     scanf("%d", &b);     /* 输入第二个整数 */
     sum=a+b;             /* 计算和 */
     printf("The sum of %d and %d is %d.\n", a,b,sum);/* 输出和 */
}
```

运行结果：

```
33↙
55↙
The sum of 33 and 55 is 88.
```

程序分析：

在 C 语言中，要想获得从键盘输入的值，可以使用 scanf 函数。scanf 是与 printf 相对应的格式输入函数，其基本命令格式如下：

```
scanf(格式控制, 地址列表);
```

语句 scanf("%d", &a);表示以十进制整数的形式(由格式说明符“%d”指定)输入数据，存放到变量 a 对应的存储单元中，这样变量 a 的值就是刚刚从键盘输入的值。&是取地址运算符，&a 指变量 a 在内存中的地址。变量 a 和 b 及其所存储的数值如图 1.3 所示。

图 1.3　变量及其值

1.4.2　C 语言程序的基本结构

(1) C 语言程序是由函数组成的。一个完整的 C 程序可以由一个或多个函数组成，其中 main 主函数必不可少，且只有唯一一个。C 程序执行时，总是从主函数 main 开始，与 main 函数在整个程序中的位置无关，其他函数都是为 main 函数服务的。函数是 C 程序的基本单位，用函数来实现特定的功能，所以说 C 是函数式的语言。C 语言的函数包括系统提供的库函数(如 printf 函数)，以及用户根据实际问题编制设计的函数。

(2) 源程序中可以有预处理命令，预处理命令通常放在源文件或源程序的最前面。

(3) 每一个语句都必须以分号结尾，但预处理命令、函数头和右花括号“}”之后不加分号。

(4) 注释不是程序的必需部分，在程序执行时注释不起任何作用。注释的作用是增加程序的可读性，因此，适当地在程序中加以注释，是一种良好的程序设计风格。C 语言有块注释和行注释这两种注释方法。

(5) 在 C 语言中，虽然一行可写多个语句，一个语句也可占多行，但建议一行只写一个语句。

(6) 一般采用缩进格式书写程序，以提高程序的可读性和清晰性。

1.5　C 语言程序的运行

1.5.1　运行 C 语言程序的步骤

C 语言属于编译型的编程语言，如果要使 C 程序在计算机上执行，必须经过源程序的编辑、编译和连接等一系列步骤，最后得到可执行程序并运行，如图 1.4 所示。

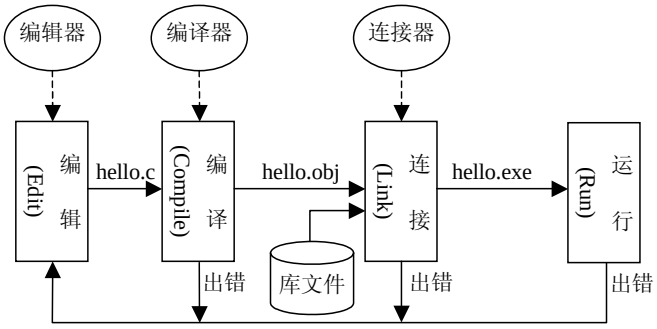

图 1.4　运行 C 程序的步骤

1. 编辑(Edit)

编辑是建立或修改 C 源程序文件的过程，并以文件的形式存储在磁盘上，C 源程序文件的扩展名为.c。

2. 编译(Compile)

C 语言编译器将 C 源程序转换为机器代码，生成目标程序。目标程序文件的扩展名为.obj。在 C 语言源程序的编译过程中，可以检查出程序中的语法错误。

3. 连接(Link)

编译生成的目标程序计算机还不能直接执行，还需将目标程序与库文件进行连接处理，连接工作由连接程序完成。经过连接后，生成可执行程序，可执行程序的扩展名为.exe。

4. 运行(Run)

C 源程序经过编译、连接后生成了可执行文件(.exe)。生成的可执行文件，既可在编译系统环境下运行，也可以脱离编译系统直接在操作系统下执行。

当编译时出现错误，说明 C 程序中有语法错误；若在运行时出现错误或结果不正确，说明程序设计上有错误(称为逻辑错误)，都需要修改源程序并重新编译、连接和运行，直至将程序调试正确为止。

1.5.2　集成开发环境

集成开发环境(Integrated Development Environment，简称 IDE)可提供编程时所必需的工具。这些工具包括编辑器、编译器和调试器，它们集成在一个软件包内供程序员使用。在早前的 DOS 环境下主要的 C 语言集成开发环境有 TC 2.0、Turbo C++ 3.0 和 Borland C++。在 Windows 环境下主要使用 Visual C++ 6.0。Visual C++ 6.0 既可以对 C++进行编译，也可以对 C 语言进行编译。本书的示例程序均可在 Visual C++ 6.0 集成环境下进行编辑、编译和运行。

1.6 本 章 小 结

C 语言是一种结构化程序设计语言，C 程序主要由函数组成，main 函数是程序的入口。C 语言中的输入/输出由标准输入/输出函数来完成，其中 printf 函数完成输出功能，scanf 函数完成输入功能。C 语言属于编译型的编程语言，需要通过编辑、编译、连接后确认没有错误方可运行，目前常用的集成开发环境是 Visual C++ 6.0。

1.7 习　　题

1. C 语言程序由哪几部分组成?

2. 填空。

(1) 在 C 语言中，每个语句必须以_____结束。

(2) _____函数称为主函数。

(3) 存放标准输入/输出库函数声明的头文件名为_____。

3. 参照本章例题，编写一个 C 语言程序，输出以下信息。

```
************************
Hello World!
************************
```

4. 参照本章【例 1.3】，编写一个 C 语言程序，求任意两个整数的差并输出结果。

5. 指出下面程序中的错误。

```
#include (stdio.h)
main( )
{
     int x , y , s ;
     x=10
     y=20;
     s=x + y ;
     printf("s = %d\n , s );
}
```