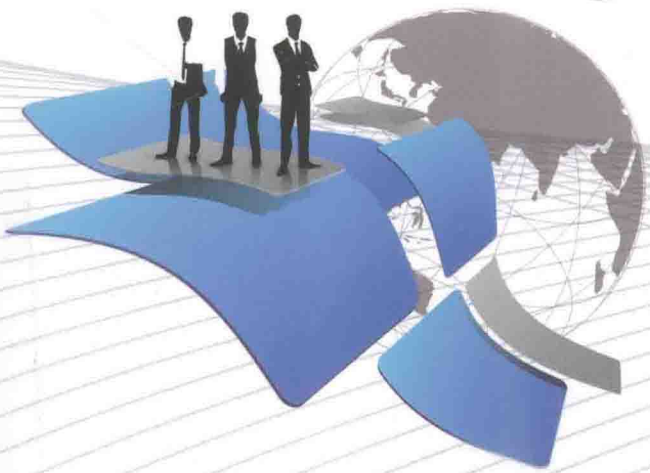
21世纪应用型本科规划教材

# C语言程序设计教程

C YUYAN CHENGXU SHEJI JIAOCHENG

邵雪航　王春明　主　编
杨　迎　副主编
杜　凯　主　审



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

21 世纪应用型本科规划教材

# C 语言程序设计教程

邵雪航　王春明　主　编
杨　迎　副主编
杜　凯　主　审

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

## 内 容 简 介

本书是为适应应用型本科发展新形势需要，为后续学习其他程序设计语言奠定基础而编写的，是一本既有理论基础，又注重操作技能实用性的程序设计教程。

全书共分 11 章。第 1 章简要介绍 C 语言及开发环境；第 2 章介绍变量、数据类型和运算符；第 3 章介绍顺序结构程序设计；第 4 章介绍选择结构程序设计；第 5 章介绍循环结构程序设计；第 6 章介绍数组；第 7 章介绍函数；第 8 章介绍指针；第 9 章介绍结构类型与联合类型；第 10 章介绍文件；第 11 章介绍位运算。本书以突出应用、强调技能为目标，同时覆盖全国计算机等级考试（二级）相关内容。本书还配有配套教材《C 语言程序设计实验与习题》，对本教材的知识点、技术和方法进行提炼、概括和总结，设计了大量的习题、实验、综合实训，便于学生巩固复习。

本书适合作为应用型本科各专业教材，也可作为全国计算机等级考试的复习用书，以及各类计算机培训班教材或初学者的自学用书。

**图书在版编目（CIP）数据**

C 语言程序设计教程 / 邵雪航，王春明主编. — 北京：中国铁道出版社，2016.2

21 世纪应用型本科规划教材

ISBN 978-7-113-21544-6

Ⅰ. ①C… Ⅱ. ①邵… ②王… Ⅲ. ①C 语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2016）第 041458 号

书　　名：C 语言程序设计教程
作　　者：邵雪航　王春明　主编

策　　划：张　铁　刘丽丽　　读者热线：（010）63550836
责任编辑：周　欣　徐盼欣
封面设计：大象設計·小威
封面制作：白　雪
责任校对：汤淑梅
责任印制：郭向伟

出版发行：中国铁道出版社（100054，北京市西城区右安门西街 8 号）
网　　址：http://www.51eds.com
印　　刷：北京明恒达印务有限公司
版　　次：2016 年 2 月第 1 版　　2016 年 2 月第 1 次印刷
开　　本：787 mm×1 092 mm　1/16　印张：15　字数：356 千
书　　号：ISBN 978-7-113-21544-6
定　　价：35.00 元

# 前　言

C 语言是针对应用型本科专业开设的第一门程序设计基础课程。本书是为了适应应用型本科发展新形势的需要，为后续学习其他程序设计语言奠定基础而编写的，主要是为学生提供一本既有理论基础，又注重操作技能实用性的程序设计教程。本书针对应用型本科教学的特点，注重基础知识的系统性和基本概念的准确性，尤其强调应用性和实用性。

本书使用环境：Visual C++ 6.0，书中所有源程序已经在此环境下调试并成功运行。

通过本书的学习，学生可以掌握 C 语言程序设计的基本思想、方法和解决实际问题的应用技巧。

本书内容的主要特色是用学生成绩管理系统案例贯穿始终 ，并覆盖全国计算机等级考试二级 C 语言程序设计的相关内容。除第 1 章外，每章基本设有学习目标、完成任务、现场练习、示例、小结和作业，能够满足不同专业的要求。本书力求通过实际问题的讲解，融理论于实践当中，加强操作技能的实用性，逐步提高学生编写程序的能力。

本书由邵雪航、王春明任主编，由杨迎任副主编。具体编写分工如下：第 1 章、第 2 章、第 6 章、第 7 章、第 10 章、第 11 章由邵雪航编写；第 3 章、第 4 章、第 9 章由王春明编写；第 5 章、第 8 章由杨迎编写。全书由邵雪航统稿，由杜凯教授主审。

本书还配有配套教材《C 语言程序设计实验与习题》，对本教材的知识点、技术和方法进行提炼、概括和总结，设计了大量的习题、实验、综合实训，便于学生巩固复习。本书以突出应用、强调技能为目标，同时覆盖全国计算机等级考试二级 C 语言程序设计相关考试内容。

本书适合作为应用型本科各专业教材，也可作为全国计算机等级考试（二级）的复习用书，以及各类计算机培训班教材或初学者的自学用书。

限于编者水平，书中难免存在疏漏与不足之处，恳请广大读者批评指正。

编　者

2015 年 12 月

# 目　录

# 第 1 章　C 语言简介及基础

学习目标：

- 了解程序、算法和流程图的概念。
- 熟练掌握 C 程序的结构。
- 熟练使用 Visual C++ 6.0 编辑和运行 C 程序。
- 熟悉使用 Visual C++ 6.0 调试程序。

完成任务：

本书以学生成绩管理系统作为项目案例贯穿始终，结合每章涉及的知识点不断对此项目进行完善。学生成绩管理系统主要负责统计学生某几门课程的成绩情况，并可以根据需要对学生成绩进行查询，显示所有符合条件的学生成绩。

## 1.1　第一个 C 语言程序

【示例 1.1】要求输出“hello world!”这一行文字。

```
/*第一个 C 语言程序举例*/
#include <stdio.h>                    //包含有关标准输入/输出库函数的信息
void main()                           //main()函数
{
    printf("hello world!\n");         //输出库函数
}
```

程序运行结果：

```
hello world!
Press any key to continue
```

## 1.2　什么是程序

程序一词来自生活，通常指完成某些事务的一种即定方式和过程。可以将程序看作对一系列动作的执行过程的描述。日常生活中可以找到许多“程序”实例。例如，去银行取钱的行为可以描述为：

（1）带上存折去银行。

（2）填写取款单。

（3）将存折和取款单递给银行职员。

（4）银行职员办理取款事宜。

（5）拿到钱。

（6）离开银行。

这个过程是一个非常简单的程序。实际上，生活中去银行取钱还可以细化一下，例如：若银行职员办理取款事宜时发现取款单填写有误，或者填写好取款单后已经到了下班时间等。细化后的程序要复杂得多，不再是一个平铺直叙的动作序列，其步骤会更多，还出现了分情况处理和可能出现的重复性运作。日常生活中程序性活动的情况与计算机里程序执行很相似。这一情况可以帮助我们理解计算机的活动方式。

人们使用计算机，就是要利用计算机处理各种不同的问题。不要忘记计算机是机器，需要由人告诉它们工作的内容和完成工作的方法。为使计算机能按人的指挥工作，计算机提供了一套指令，其中的每一种指令对应着计算机能执行的一个基本动作。为让计算机完成某项任务而编写的逐条执行的指令序列就称为程序。

## 1.3 程序算法及流程图

### 1. 算法

为了让计算机能准确无误地完成任务，必须事先对各类问题进行分析，确定解决问题的具体方法和步骤，再编制好一组让计算机执行的指令，交给计算机，让计算机按人们指定的步骤有效地工作。这些具体的方法和步骤，其实就是解决一个问题的算法。由此可见，程序设计的关键之一，是解决问题的方法与步骤，即算法。算法也是程序设计的灵魂。

思考现实生活中计算矩形面积的简单问题。要解决这个问题，需要执行以下步骤。

（1）了解此矩形的长和宽两个值。

（2）判断长和宽的值是否大于零。

（3）如果大于零，将长和宽两个值相乘得到面积。

（4）显示面积值。

由于同一个问题可以有不同的解决方法，所以不同的两个人也可能编写出不同的算法而得到相同的结果。

在实际应用中可以用自然语言、流程图、结构化流程图、伪代码、PAD图等形式来描述算法。通常情况下使用流程图。流程图是算法的一种图形化表示方式，直观、清晰，更有利于人们设计算法。它使用一组预定义的符号来说明如何执行特定任务。这些预定义的符号已标准化，从而让开发人员可以采用这些符号而不会引起混淆。

### 2. 流程图

美国国家标准学会（American National Standards Institute，ANSI）规定了一些常用的流程图符号，如表1-1所示。

表1-1 常用的流程图符号

| 符　号 | 描　述 | 符　号 | 描　述 |
| --- | --- | --- | --- |
|  | 程序的开始框或结束框 |  | 判断和分支框 |
|  | 计算步骤处理符号框 |  | 流向线 |
|  | 输入/输出框 | | |

前面给出的计算矩形面积的算法可用流程图表示，如图 1-1 所示。

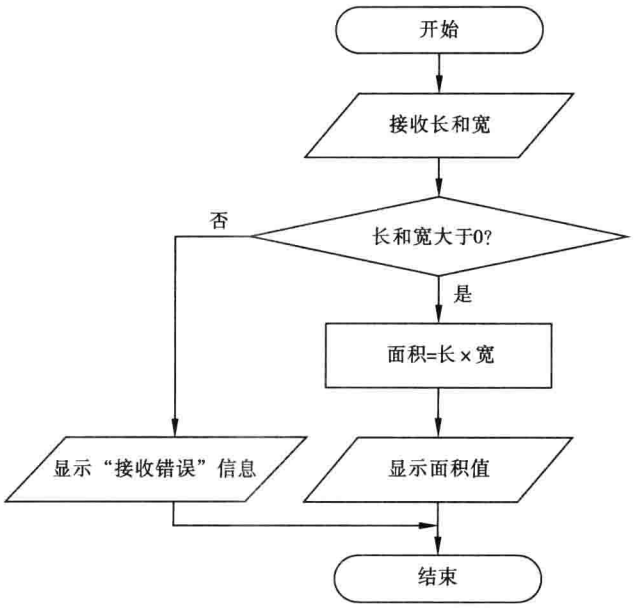



图 1-1　计算矩形面积的流程图

**【示例 1.2】**描述 5!的算法流程图。

分析：求 $1\times2\times3\times4\times5$。

步骤 1：先求 $1\times2$，得到结果 2；

步骤 2：将步骤 1 得到的乘积 2 乘以 3，得到结果 6；

步骤 3：将 6 乘以 4，得 24；

步骤 4：将 24 乘以 5，得 120。

如果要求 $1\times2\times\cdots\times1000$，则要写 999 个步骤，这样太烦琐。

可以设两个变量：一个变量代表被乘数，另一个变量代表乘数。不另设变量存放乘积结果，而直接将每一步骤的乘积放在被乘数变量中。设 p 为被乘数，i 为乘数。用循环算法来求结果，算法可改写为：

S1：使 p=1；

S2：使 i=2；

S3：使 p*i，乘积仍放在变量 p 中，可表示为：p*i→p；

S4：使 i 的值加 1，即 i+1→i；

S5：如果 i 不大于 5，返回重新执行步骤 S3 以及其后的步骤 S4 和 S5；否则，算法结束。最后得到的 p 值就是 5!的值。

以上算法的流程图如图 1-2 所示。

如果题目改为“求 $1\times3\times5\times\cdots\times11$”，算法只需做很少的改动：

S1：1→p；

S2：3→i；

S3：p*i→p；

S4：i+2→p；

S5：若 i≤11，返回 S3。否则，算法结束。

**【示例 1.3】**描述打印 50 名学生中成绩在 80 分以上者的学号和成绩的算法流程图。

分析：设 n 表示学号，n1 代表第一个学生学号，ni 代表第 i 个学生学号。用 g 代表学生成绩，gi 代表第 i 个学生成绩，算法的流程图如图 1-3 所示。

S1：1→i；

S2：如果 gi≥80，则打印第 i 个学生的学号和成绩，否则不打印；

S3：i+1→i；

S4：如果 i≤50，返回 S2，继续执行。否则，算法结束。

变量 i 作为下标，用来控制序号（第几个学生，第几个成绩）。当 i 超过 50 时，表示已将 50 个学生的成绩处理完毕，算法结束。

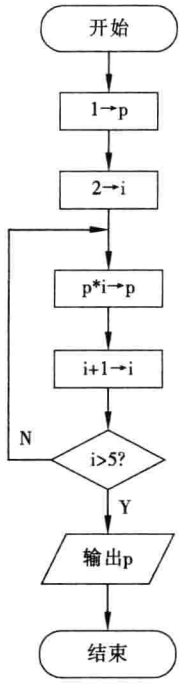



图 1-2　5!的流程图

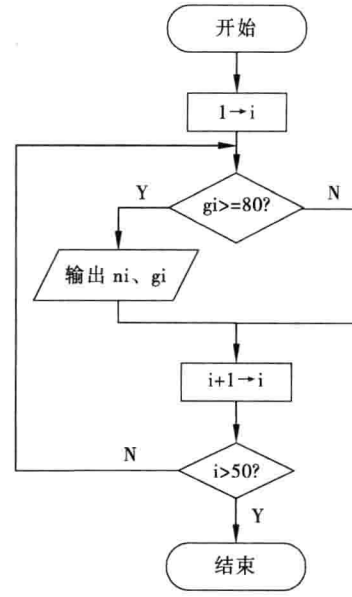



图 1-3　打印 80 分以上学生成绩的流程图

## 1.4　程序设计语言的发展历程

自 1946 年世界上第一台电子计算机问世以来，计算机科学及其应用的发展十分迅猛，计算机被广泛地应用于人类生产、生活的各个领域，推动了社会的进步与发展。特别是随着 Internet 日益深入千家万户，传统的信息收集、传输及交换方式正在发生改变，计算机已将人类带入了一个新的时代——信息时代。

新的时代对于人们的基本要求之一是：自觉、主动地学习和掌握计算机的基本知识和基本技能，并把它作为自己应该具备的基本素质。要充分认识到，缺乏计算机知识，就是信息时代的“文盲”。

对于大学生而言，掌握一门高级语言及其基本的编程技能是必需的。大学学习，除了掌握本专业系统的基础知识外，科学精神的培养、思维方法的锻炼、严谨踏实科研作风的养成，以及分析问题、解决问题能力的训练，都是日后工作的基础。学习计算机语言是一种十分有益的训练方式，而计算机语言本身又是与计算机进行交互的有力工具。

一台计算机是由硬件系统和软件系统两大部分构成的，硬件是物质基础，而软件可以说是计算机的灵魂。没有软件，计算机就是一台“裸机”，什么工作都无法完成，有了软件，计算机才能成为一台真正的“电脑”。所有的软件都是用计算机语言编写的。

计算机程序设计语言的发展经历了从机器语言、汇编语言到高级语言的历程。

### 1．机器语言

电子计算机所使用的是由“0”和“1”组成的二进制数，二进制是计算机语言的基础。计算机发明之初，人们只能用计算机的语言去命令计算机工作，也就是写出一串串由“0”和“1”组成的指令序列交由计算机执行，这种语言就是机器语言。使用机器语言编写程序十分烦琐且工作量巨大，特别是在程序有错误需要修改时更是如此。而且，由于每台计算机的指令系统往往各不相同，所以，在一台计算机上执行的程序，要想在另一台计算机上执行，必须另编程序，造成了重复工作。但是，机器语言使用的是针对特定型号计算机的语言，故而运算效率是所有语言中最高的。机器语言是第一代计算机语言。

### 2．汇编语言

为了减轻使用机器语言编程的工作量，人们进行了一种有益的改进：用一些简洁的英文字母、符号串来替代一个特定指令的二进制串，例如，用“ADD”代表加法，用“MOV”代表数据传递，等等，这样，人们很容易读懂并理解程序在干什么，纠错及维护工作都变得方便了。这种程序设计语言称为汇编语言，即第二代计算机语言。然而计算机是不认识这些符号的，这就需要一个专门的程序专门负责将这些符号翻译成二进制数的机器语言，这种翻译程序称为汇编程序。

汇编语言同样十分依赖于机器硬件，移植性不好，但效率仍十分高。针对计算机特定硬件而编制的汇编语言程序，能准确发挥计算机硬件的功能和特长，程序精练而质量高，所以至今仍是一种常用而强有力的软件开发工具。

### 3．高级语言

从最初与计算机交流的经历中，人们意识到，应该设计一种这样的语言：接近于数学语言或人的自然语言，同时又不依赖于计算机硬件，编写出的程序能在所有机器上使用。经过努力，1954 年，第一个完全脱离机器硬件的高级语言——FORTRAN 问世了，60 多年来，共有几百种高级语言出现，有重要意义的有几十种，影响较大、使用较普遍的有 C++、VC、VB、Java 和 C#等。

高级语言的发展也经历了从早期语言到结构化程序设计语言，从面向过程到非过程化程序语言的过程。相应地，软件的开发也由最初的个体手工作坊式的封闭式生产，发展为产业化、流水线式的工业化生产。

20 世纪 60 年代中后期，软件越来越多，规模也越来越大，而软件的生产基本上是各自为战，缺乏科学规范的系统规划与测试、评估标准，其后果是大批耗费巨资建立起来的软件系统由于含有错误而无法使用，甚至带来巨大损失，软件给人的感觉是越来越不可靠，以致几乎没有不

出错的软件。这一切极大地震动了计算机界，史称“软件危机”。人们认识到：大型程序的编制不同于小程序的编写，它应该是一项新的技术，应该像处理工程一样处理软件研制的全过程。程序的设计应易于保证正确性，也便于验证正确性。1969 年，E.W.Dijkstra 提出了结构化程序设计方法，1970 年，第一个结构化程序设计语言——Pascal 语言出现，标志着结构化程序设计时期的开始。

20 世纪 80 年代初，在软件设计思想上又产生了一次革命，其成果就是面向对象的程序设计。在此之前的高级语言几乎都是面向过程的，程序的执行是流水线作业，在一个模块被执行完成前不能执行其他操作，也无法动态地改变程序的执行方向。这和人们日常处理事物的方式是不一致的，人们希望发生一件事就处理一件事，也就是说，不能面向过程，而应是面向具体的应用功能，也就是对象（Object）。其方法就是软件的集成化，如同硬件的集成电路一样，生产一些通用的、封装紧密的功能模块，称为软件集成块。它与具体应用无关，但能相互组合，完成具体的应用功能，同时又能重复使用。对使用者来说，只关心它的接口（输入量、输出量）及能实现的功能，至于如何实现则是它内部的事，使用者完全不用关心。C++就是典型代表。

高级语言的下一个发展目标是面向应用，也就是说：只需要告诉程序要做什么，程序就能自动生成算法，自动进行处理，这就是非过程化的程序设计语言。

## 1.5 C 语言发展历程

早期的操作系统等系统软件主要是用汇编语言编写的，如 UNIX 操作系统。汇编语言依赖于计算机硬件，程序的可读性和可移植性都比较差。为了提高可读性和可移植性，最好改用高级语言，但一般高级语言难以实现汇编语言的某些功能，如不像汇编语言可以直接对硬件进行操作 （对内存地址的操作、位（bit）操作等）。人们设想能否找到一种既具有一般高级语言特性，又具有低级语言特性的语言，集它们的优点于一身。于是，C 语言应运而生，之后成为国际上广泛流行的计算机高级语言。它适合于作为系统描述语言，既可用来编写系统软件，也可用来编写应用软件。

C 语言是在 B 语言的基础上发展起来的，它的根源可以追溯到 ALGOL 60。1960 年出现的 ALGOL 60 是一种面向问题的高级语言，它离硬件比较远，不宜用来编写系统程序。1963 年，英国的剑桥大学推出了 CPL（Combined Programming Language）。CPL 在 ALGOL 60 的基础上有所接近硬件，但规模比较大，难以实现。1967 年，英国剑桥大学的 Matin Richards 对 CPL 语言做了简化，推出了 BCPL（Basic Combined Programming Language）。1970 年，美国贝尔实验室的 K.Thompson 以 BCPL 为基础，又做了进一步简化，使得 BCPL 能在 8 KB 内存中运行，这个很简单而且很接近硬件的语言就是 B 语言（取 BCPL 的第一个字母），并用它编写了第一个 UNIX 操作系统，在 DEC PDP-7 上实现。1971 年，在 PDP-11/20 上实现了 B 语言，并编写了 UNIX 操作系统。但 B 语言过于简单，功能有限，并且和 BCPL 都是“无类型”的语言。1972—1973 年间，贝尔实验室的 D.M.Ritchie 在 B 语言的基础上设计出了 C 语言（取 BCPL 的第二个字母）。C 语言既保持了 BCPL 和 B 语言的优点（精练，接近硬件），又克服了它们的缺点（过于简单，数据无类型等）。最初的 C 语言只是为描述和实现 UNIX 操作系统提供一种工具语言而设计的。1973 年，K.Thompson 和 D.M.Ritchie 两人合作把 UNIX 的 90%以上用 C 语言进行了改写，即 UNIX 第 5 版。（原来的 UNIX 操作系统是

1969 年由美国的贝尔实验室的 K.Thompson 和 D.M.Ritchie 开发成功的，是用汇编语言写的）这样，UNIX 使分散的计算系统之间的大规模联网以及互联网成为可能。

后来，C 语言做了多次改进，但主要还是在贝尔实验室内部使用。直到 1975 年 UNIX 第 6 版公布后，C 语言的突出优点才引起人们普遍关注。1977 年出现了不依赖于具体机器的 C 语言编译文本“可移植 C 语言编译程序”，使 C 移植到其他机器时所需做的工作大大简化了，这也推动了 UNIX 操作系统迅速地在各种机器上实现。例如，VAX、AT&T 等计算机系统都相继开发了 UNIX。随着 UNIX 的日益广泛使用，C 语言也迅速得到推广。C 语言和 UNIX 可以说是一对孪生兄弟，在发展过程中相辅相成。1978 年以后，C 语言已先后移植到大、中、小、微型机上，如 IBM System/370、Honeywell 6000 和 Interdata 8/32，已独立于 UNIX 和 PDP。现在 C 语言已风靡全世界，成为世界上应用最广泛的计算机语言之一。

以 1978 年由美国电话电报公司（AT&T）贝尔实验室正式发表的 UNIX 第 7 版中的 C 编译程序为基础，Brian W.Kernighan 和 D.M.Ritchie 合著了影响深远的名著 *The C Programming Language*，常常称它为 K&R，也有人称之为“K&R 标准”或“白皮书”（White Book），它成为后来广泛使用的 C 语言版本的基础，但在 K&R 中并没有定义一个完整的标准 C 语言。为此，1983 年，美国国家标准学会 X3J11 委员会根据 C 语言问世以来各种版本对 C 的发展和扩充，制定了新的标准，称为 ANSI C。ANSI C 比原来的标准 C 有了很大的发展。K&R 在 1988 年修改了他们的经典著作 *The C Programming Language*，按照 ANSI C 标准重新写了该书。1987 年，ANSI 又公布了新标准——87 ANSI C，目前流行的 C 编译系统都是以它为基础的。当时广泛流行的各种版本 C 语言编译系统虽然基本部分是相同的，但也存在一些不同之处。在微型机上使用的有 Microsoft C（MS C）、Borland Turbo C、Quick C 和 AT&T C 等，它们的不同版本又略有差异。到后来的 Java、C++、C#都是以 C 语言为基础发展起来的。

C 语言是一种国内外广泛流行的、已经得到普遍应用、很有发展前途的程序设计语言，它既可以用来编写系统软件，又可以用来编写应用软件。

对操作系统以及需要对硬件进行操作的场合，C 语言明显优于其他高级语言，许多大型应用软件都是用 C 语言编写的。通常情况下，学习程序设计语言的最佳途径是尽早地编写程序、调用程序，进而解决实际问题。

## 1.6　C 语言特点

C 语言简洁、紧凑，使用方便、灵活。总体来说，C 语言具有以下特点：

（1）具有 32 个关键字、9 种控制语句，程序形式自由。

（2）运算符丰富，具有 34 种运算符。

（3）数据类型丰富，具有现代语言的各种数据结构。

（4）具有结构化的控制语句，是完全模块化和结构化的语言。

（5）语法限制不太严格，程序设计自由度大。

（6）允许直接访问物理地址，能进行位操作，能实现汇编语言的大部分功能，可直接对硬件进行操作，兼有高级和低级语言的特点。

（7）目标代码质量高，程序执行效率高。只比汇编程序生成的目标代码效率低 10%～20%。