



高职高专“十一五”规划教材



YUYAN CHENGXU SHEJI

语言程序设计

● 孙 锋 主编
● 付兴宏 王庆桦 副主编

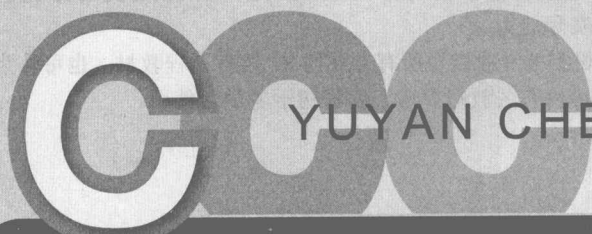


化学工业出版社





高职高专“十一五”规划教材



YUYAN CHENGXU SHEJI

语言程序设计

● 孙 锋 主编
● 付兴宏 王庆桦 副主编



化学工业出版社

· 北京 ·

本书以程序设计为主线,以程序设计的需要带动语言知识的学习,系统地介绍了 C 语言程序设计的基本知识、基本数据类型和数据运算、程序控制语句、函数定义及应用、数组和指针、结构体与共同体、预处理命令、文件操作以及结构化程序设计等内容。力求使读者通过学习,掌握 C 语言的基础知识,了解结构化程序设计的思想和方法,以及程序调试技巧和良好的编程习惯,从而更好地掌握相关技术与技能,并有助于读者通过相关水平考试、升学考试和职业资格证书考试。

本书主要根据作者长期教学和实践经验编写,内容丰富,语言准确,实用性强,每章除了有详细的讲解和实际项目举例,还配有常见错误、编程注意事项、实训项目和习题,帮助读者提高实际操作水平,以及阅读程序、调试程序和编写程序的实际能力。

本书可作为高等职业技术学院计算机相关专业以及所有专业的 C 语言课程教材,也可作为高等院校、中等专业学校的编程基础教材,还可供有关教师参考、编程初学者学习使用。

图书在版编目(CIP)数据

C 语言程序设计 / 孙锋主编. —北京: 化学工业出版社, 2008.6

高职高专“十一五”规划教材

ISBN 978-7-122-02901-0

I. C… II. 孙… III. C 语言-程序设计-高等学校: 技术学院-教材 IV. TP312

中国版本图书馆 CIP 数据核字 (2008) 第 070060 号

责任编辑: 王听讲

文字编辑: 吴开亮

责任校对: 战河红

装帧设计: 刘丽华

出版发行: 化学工业出版社 (北京市东城区青年湖南街 13 号 邮政编码 100011)

印 刷: 大厂聚鑫印刷有限责任公司

装 订: 三河市延风装订厂

787mm×1092mm 1/16 印张 18¼ 字数 424 千字 2008 年 8 月北京第 1 版第 1 次印刷

购书咨询: 010-64518888 (传真: 010-64519686) 售后服务: 010-64518899

网 址: <http://www.cip.com.cn>

凡购买本书, 如有缺损质量问题, 本社销售中心负责调换。

定 价: 29.00 元

版权所有 违者必究

前言

在科学技术高速发展的信息时代,我国计算机产业也迅速发展,对专业人才的需求也日益迫切。程序设计是计算机专业领域中最核心的工作,在程序设计中,要对问题进行分析,考虑各种设计的可能性,选择适当的算法、数据结构及语言,编写代码,对代码进行测试,本书针对这些问题进行了详细的讨论。

程序设计课程有两个任务,一是学会一种语言,二是培养程序设计基本能力。相对而言,学习语言比较容易,因为它是一种工具,是一种技能,通过边学边用达到熟能生巧并不困难。而程序设计能力的培养需要花较大的力气,因为它需要较强的逻辑思维能力,要理解和掌握一批常用的算法,这对初学者来说比较陌生、比较抽象。任何语言学习的过程都是一种思维的训练过程,语言是思维的一种工具,C语言也不例外。应用C语言的基础知识,针对具体问题编写良好的程序,是需要经过积累和训练才能够达到的。本书编写的目的就是希望能帮助读者缩短这个过程的时间,迅速提高C语言程序设计的能力和水平。

本书编者由长期从事程序设计教学、软件系统开发的教师组成,大量教学实例来自于实际开发项目,既有较强的理论性,又具有鲜明的实用性。本书以初次学习程序设计者的认知规律为主线,以程序设计思想为核心,从C语言基本语法、结构化程序设计思想和良好的编程习惯三个方面,循序渐进地对C语言程序设计进行全面剖析,对各种典型算法进行多视角诠释,让读者全面了解结构化程序设计的内涵,从而掌握基本设计方法。为使读者更好地理解、掌握并运用编程技术,本书在知识的讲解上,紧紧地与实际项目、实训练习相结合,使读者不仅有理性认识,还有相当的感性认识。通过大量实例说明每个知识点如何应用在实际项目中,并且每章都提供常见错误分析和怎样培养良好编程习惯的提示,帮助读者提高程序设计的职业技能,章后习题都是一些非常实用的算法,考虑到不同读者的需求,还增加了拓展能力实训题目,为进一步学习程序设计打下良好基础。

本书内容以程序设计为主线,以程序设计的需要带动语言知识的学习。按照程序由简单到复杂的顺序,主要介绍了C语言程序设计基本概述;数据类型、运算符和表达式;简单的C程序设计;自定义函数设计;程序的控制结构;数组;指针;结构体与共用体;函数的高级应用;文件操作;项目开发实例等内容。本教材中的源程序代码均已通过上机调试。

本书语言浅显易懂、结构科学合理,符合高职院校学生学习的特点,能够由基础到提高,再到综合应用,切实锻炼学生的实践动手能力和程序设计思想,真实体验知识的有效性和实用性,提高学习兴趣,有利于培养学生的程序设计思想,为学习后续课程打下良好

的基础。

本书由孙锋主编，付兴宏和王庆桦副主编。第1章由解皓编写，第2、3章由罗雨滋编写，第4、5、9章和附录1~4由孙锋编写，第6、7章由王庆桦编写，第8、10、11章由付兴宏编写。由于编者水平所限，如有疏漏和不妥之处，恳请读者提出宝贵意见。

编 者

2008年4月

目 录

第 1 章 C 语言程序设计基本概述	1
1.1 C 语言概述	1
1.2 一个简单 C 语言程序	2
1.2.1 注释	3
1.2.2 预处理	3
1.2.3 定义 main() 函数	4
1.2.4 关键字	4
1.2.5 main() 函数的定义	5
1.2.6 输出信息	5
1.2.7 参数	5
1.2.8 控制字符	6
1.3 C 程序的运行	6
1.3.1 Turbo C 的开发环境	6
1.3.2 编辑	7
1.3.3 编译	8
1.3.4 连接	8
1.3.5 执行	8
*1.4 用 C 语言开发程序	9
1.4.1 理解问题	9
1.4.2 详细设计	9
1.4.3 实现	9
1.4.4 测试	10
1.5 项目举例	10
1.6 实训项目——熟悉 C 语言程序开发环境	11
1.7 常见错误	11
1.8 编程注意事项	11
本章小结	11
习题 1	12
第 2 章 数据类型、运算符和表达式	13
2.1 C 语言的常见符号	13
2.2 数据类型	14
2.2.1 引入数据类型的原因	14
2.2.2 C 语言的数据类型	14

2.2.3	基本数据类型及类型说明符	15
2.3	常量	16
2.3.1	整型常量	16
2.3.2	浮点型常量	17
2.3.3	字符常量	17
2.3.4	字符串常量	18
2.3.5	符号常量	19
2.4	变量	20
2.4.1	变量的声明和初始化	20
2.4.2	使用变量的注意事项	21
2.5	常用运算符及表达式	22
2.5.1	C 语言运算符和表达式概述	22
2.5.2	算术运算符	22
2.5.3	关系运算符	23
2.5.4	逻辑运算符	24
2.5.5	赋值运算符	25
2.5.6	自加、自减运算符	27
2.5.7	条件运算符	27
2.5.8	位运算符	28
2.5.9	逗号运算符	30
2.6	表达式中的类型转换	31
2.6.1	自动转换	31
2.6.2	强制类型转换	32
2.7	项目举例	33
2.8	实训项目二——运算符和表达式的应用	34
2.9	常见错误	35
2.10	编程注意事项	36
	本章小结	36
	习题 2	36
第 3 章	简单的 C 程序设计	38
3.1	C 语句分类	38
3.2	基本输入/输出函数的实现	38
3.2.1	输入/输出概述	38
3.2.2	字符数据的输入/输出	39
3.2.3	格式输出函数	39
3.2.4	格式输入语句	42
3.3	项目举例	44
3.4	实训项目三——简单 C 程序设计	45
3.5	常见错误	47

3.6	编程注意事项	47
	本章小结	48
	习题 3	48
第 4 章	自定义函数设计	50
4.1	程序设计的艺术	50
4.2	结构化程序设计思想和方法	51
4.2.1	结构化程序设计的核心思想	51
4.2.2	结构化程序设计方法	51
4.3	函数的定义与应用	51
4.3.1	函数的分类	52
4.3.2	函数的定义	52
4.3.3	函数的调用	54
4.3.4	函数的参数和返回值	56
4.3.5	函数声明	58
4.3.6	函数的嵌套调用	59
4.4	项目举例	60
4.5	实训项目四——简单函数应用	63
4.6	常见错误	64
4.7	编程注意事项	65
	本章小结	65
	习题 4	65
第 5 章	程序的控制结构	68
5.1	算法及其描述方法	68
5.1.1	算法的概念	68
5.1.2	算法的描述	68
5.1.3	结构化程序设计的规则	70
5.2	顺序结构	71
5.2.1	顺序结构的流程图和 N-S 图表示	71
5.2.2	应用程序举例	71
5.3	选择结构	72
5.3.1	应用场合	72
5.3.2	选择结构的流程图和 N-S 图表示	72
5.3.3	if 条件语句	73
5.3.4	开关语句	77
5.4	循环结构	82
5.4.1	应用场合	82
5.4.2	循环结构的流程图和 N-S 图表示	82
5.4.3	循环语句	83
5.4.4	单重循环应用举例	86

5.4.5	嵌套循环及其应用举例	87
5.5	流程的转移控制	88
5.5.1	goto 语句	88
5.5.2	break 与 continue 语句	88
5.5.3	应用程序举例	91
5.5.4	几种循环的比较	92
*5.6	程序错误与调试	92
5.6.1	程序中常见的出错类型	92
5.6.2	程序测试与程序调试	93
5.7	项目举例	94
5.8	实训项目五——控制结构程序设计	99
5.9	常见错误	102
5.10	编程注意事项	104
	本章小结	104
	习题 5	105
第 6 章	数组	108
6.1	数组类型的应用场合	108
6.2	一维数组	109
6.2.1	一维数组的声明	109
6.2.2	一维数组的引用	110
6.2.3	一维数组的初始化	111
6.2.4	程序实例	112
6.3	二维数组的声明、引用和初始化	113
6.3.1	二维数组的声明	113
6.3.2	二维数组的引用	114
6.3.3	二维数组的初始化	115
6.3.4	程序实例	116
6.4	向函数传递数组	117
6.4.1	向函数传递一维数组	117
6.4.2	向函数传递二维数组	120
6.5	字符数组	121
6.5.1	字符数组与字符串的关系	121
6.5.2	字符数组的输入/输出	122
6.5.3	字符串处理函数	124
6.6	项目举例	126
6.6.1	排序问题	126
6.6.2	查询字符位置的问题	128
6.6.3	查询某字符开始的子串问题	129
6.7	实训项目六——数组应用	130

6.8 常见错误	131
6.9 编程注意事项	132
本章小结	132
习题 6	133
第 7 章 指针	136
7.1 指针概述	136
7.1.1 指针的概念	136
7.1.2 指针变量的使用	137
7.1.3 指针变量作为函数参数	139
7.2 指针和数组间的关系	142
7.2.1 指针与数组	142
7.2.2 一维数组指针	142
7.2.3 二维数组指针	149
7.2.4 指针与字符串	153
7.3 指针数组	157
7.4 指向指针的指针	159
7.5 项目举例	160
7.6 实训项目七——指针的应用	163
7.7 常见错误	164
7.8 编程注意事项	164
本章小结	165
习题 7	165
第 8 章 结构体与共用体	169
8.1 问题的提出	169
8.2 结构体类型与结构体变量	170
8.2.1 结构体类型的定义	170
8.2.2 结构体类型变量的定义	170
8.2.3 typedef 的用法	171
8.2.4 结构体变量的引用和初始化	172
8.3 结构体数组	174
8.3.1 结构体数组的声明	174
8.3.2 结构体数组的初始化	175
8.3.3 结构体数组的应用实例	176
8.3.4 结构体数组与指针	177
8.4 结构体与函数	177
*8.5 动态数据结构	180
8.5.1 动态存储分配概述	180
8.5.2 链表概述	182
8.5.3 链表的基本操作	183

8.5.4	链表的创建	183
8.5.5	链表结点的删除	185
8.5.6	链表结点的插入	187
8.6	共用体	190
8.6.1	共用体的定义	190
8.6.2	共用体变量的引用	191
8.7	项目举例	192
8.8	实训项目八——结构体应用	195
8.9	常见错误	196
8.10	编程注意事项	197
	本章小结	198
	习题 8	198
第 9 章	函数的高级应用	201
9.1	递归	201
9.1.1	递归问题的提出	201
9.1.2	函数的递归调用	201
9.2	变量的作用域和存储类型	206
9.2.1	变量的作用域	206
9.2.2	变量的存储类型	209
*9.3	内部函数和外部函数	213
9.3.1	内部函数	213
9.3.2	外部函数	213
*9.4	返回指针值的函数	214
*9.5	函数指针	215
9.6	预处理指令	216
9.6.1	文件包含命令	216
9.6.2	宏命令	217
9.6.3	条件编译命令	219
9.7	模块化程序设计	220
9.8	项目举例	222
9.9	实训项目九——复杂函数调用程序设计	229
9.10	常见错误	230
9.11	编程注意事项	231
	本章小结	231
	习题 9	231
第 10 章	文件操作	236
10.1	文件概述	236
10.1.1	文件的概念	236
10.1.2	文件的分类	236

10.1.3	文件的存取方式	237
10.2	文件的打开和关闭	237
10.2.1	文件指针	237
10.2.2	文件操作的基本步骤	237
10.2.3	文件的打开	238
10.2.4	文件的关闭	239
10.3	文件的读写操作	239
10.3.1	文件的字符读写操作	239
10.3.2	文件的块读写函数	242
10.3.3	文件的字符串读写函数	243
10.3.4	其他文件读写函数	244
10.3.5	随机文件的读写	245
10.3.6	出错检测	247
10.4	项目举例	247
10.5	实训项目十——文件应用	251
10.6	常见错误	252
10.7	编程注意事项	253
	本章小结	253
	习题 10	253
第 11 章	C 语言项目开发实例	255
11.1	项目概述	255
11.1.1	项目背景	255
11.1.2	学生信息管理概述	255
11.2	系统分析	256
11.2.1	系统功能分析	256
11.2.2	系统数据分析	257
11.3	系统设计与实现	257
11.3.1	系统登录模块	257
11.3.2	系统录入模块	258
11.3.3	数据查询模块	260
11.3.4	数据修改模块	261
11.3.5	数据删除模块	263
11.3.6	数据打印模块	265
11.3.7	数据统计模块	265
11.3.8	数据初始化模块	267
11.3.9	其他数据模块	268
11.4	系统设计说明	270
11.4.1	关于 C 语言知识的综合应用	270
11.4.2	关于系统的开发工具	270

11.4.3	关于本系统的功能	270
11.4.4	关于程序的测试	271
附录 1	C 语言运算符优先级表	272
附录 2	ASCII 表	273
附录 3	常用 Turbo C 2.0 库函数	274
附录 4	部分习题答案	276
参考文献		278

第1章 C语言程序设计基本概述

1.1 C语言概述

人们为了用计算机来解决实际问题，一般总是要编写程序。所谓程序，是指以某种程序设计语言为工具对解决问题的动作序列的描述，它表达了人们解决问题的过程，用于指挥计算机进行一系列操作，从而实现问题的解决。程序设计语言就是用户用来编写程序的语言，它是人与计算机进行信息交流的工具。

在C语言诞生以前，系统软件主要是用汇编语言编写的。由于汇编语言程序依赖于计算机硬件，其可读性和可移植性都很差；但一般的高级语言又难以实现对计算机硬件的直接操作（这正是汇编语言的优势），于是人们盼望有一种兼有汇编语言和高级语言特性的新语言。

C语言是用英语和数学表达式这样一种接近自然语言的语句来描述程序过程的计算机程序设计语言。C语言允许直接访问物理地址，可以直接对硬件进行操作。其既具有高级语言的功能，又具有低级语言的许多功能，能够像汇编语言一样对位、字节和地址进行操作，可以用来编写系统软件。

它是伴随着UNIX操作系统成长起来的，历史上曾用C语言改写了UNIX操作系统大约70%的代码，创造了计算机历史上用高级语言设计操作系统的第一个成功范例。此后，计算机科学家们又用C语言成功开发了包括Windows在内的诸多系统软件和大量应用软件，并在C语言的基础上，结合面向对象技术，成就了面向对象的程序设计语言C++，还结合了网络环境的需求，催生了Java语言。

C语言的原型是ALGOL 60语言，也称为A语言。1963年，剑桥大学将ALGOL 60语言发展成为CPL（combined programming language）语言。1967年，剑桥大学的Martin Richards对CPL语言进行了简化，于是产生了BCPL语言。1970年，美国贝尔实验室的Ken Thompson将BCPL进行了修改，并为它起了一个有趣的名字“B语言”，意思是将CPL语言煮干，提炼出它的精华，并且他用B语言写了第一个UNIX操作系统。1973年，B语言也被人“煮”了一下，美国贝尔实验室的D.M.RITCHIE在B语言的基础上最终设计出了一种新的语言，他取了BCPL的第二个字母作为这种语言的名字，这就是C语言。为了使UNIX操作系统推广，1977年Dennis M.Ritchie发表了不依赖于具体机器系统的C语言编译文本《可移植的C语言编译程序》，即是著名的ANSI C。1978年Brian W.Kernighan和Dennis M.Ritchie出版了名著《C语言程序》（*The C Programming Language*），从而使C语言成为当时世界上流行最广泛的高级程序设计语言。1988年，随着微型计算机的日益普及，C语言出现了许多版本。由于没有统一的标准，使得这些C语言之间出现了一些不一致的地方。为了改变这种情况，美国国家标准化协会（ANSI）为C语言制定了一套ANSI

标准，成为现行的 C 语言标准。

C 语言简洁紧凑、灵活方便，只有 32 个关键字，9 种控制语句，程序书写自由，主要用小写字母表示。C 语言字符集由字母、数字、空格、标点和特殊字符组成。在字符常量，字符串常量和注释中还可以使用汉字或其他可表示的图形符号。C 语言的运算符共有 34 个，括号、赋值、强制类型转换等都作为运算符处理，运算类型极其丰富，表达式类型多样化，灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

C 语言的数据结构丰富，数据类型有：整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等。能用来实现各种复杂的数据类型的运算，并引入了指针概念，使程序效率更高。

C 语言是结构化语言，以函数形式提供给用户，这些函数可方便地调用，并具有多种循环、条件语句控制程序流向，从而使程序完全结构化，层次清晰，便于使用、维护以及调试。

同时，C 语言还具有强大的图形功能，支持多种显示器和驱动器；强大的计算、逻辑判断功能；良好的可移植性。因此，C 语言适用范围很广。

用 C 语言规定的文字和规则书写的程序叫做 C 的源程序，存放 C 源程序的文件叫做 C 的源程序文件。一个 C 源程序主要是由若干个函数定义所组成的。每个函数定义建立了一个解决较大问题中一个子问题的函数。函数之间存在一种调用关系，通过调用建立起函数之间的相互联系。为了方便使用，C 标准提供了大量预先定义好的被称为标准库函数（也称系统函数）的通用函数，使用它们时只要按照每个库函数的要求调用即可。所以说 C 程序是由用户定义的函数（用户函数）和标准库函数（库函数）构成的。当问题比较简单时，通常用一个被称为主函数的用户函数调用一些库函数来完成。若干个函数定义构成 C 的源程序文件（*.c 文件）。

1.2 一个简单 C 语言程序

【例 1-1】 以下程序的功能是从键盘输入一个数 x ，求 x 的正弦值，然后输出结果，从中可以了解 C 语言的基本构成。

```
/*求 x 的正弦值*/
#include<math.h>

#include<stdio.h>
main()
{
    double x,s;

    printf("input number:\n");
    scanf("%lf",&x);
    s=sin(x);
    printf("sin of %lf is %lf\n",x,s);
}

/*include 称为文件包含命令扩展名为.h 的文件也称为头文件
或首部文件*/

/*定义主函数*/
/*main 函数体开始*/
/*函数声明部分：声明两个实型变量 x 和 s，以被后面程序
使用*/
/*下面开始都是函数体：显示提示信息 input number:*/
/*从键盘获得实数 x 的具体值*/
/*求 x 的正弦，并把它赋给变量 s，其函数定义在 math.h 中*/
/*显示程序运算结果 s 的值*/
/*main 函数结束*/
```

1.2.1 注释

程序注释是书写规范程序时很重要的一个内容，注释可以方便程序的阅读和维护。注释在编译代码时会被忽略，不会编译到最后的可执行文件中，所以注释不会增加可执行文件的大小。编写注释需注意的问题如下。

① 在 C 语言中，所有的注释由字符 “/ *” 开始，以 “* /” 结束。在星号及斜杠之间不允许有空格。编译程序自动忽略注释开始符到注释结束符间的任何文本。

运行下面程序在屏幕上只打印 “hello”。

```
#include<stdio.h>
main ()
{
    printf("hello");
    /*printf("This is a sample to print hello");*/
}
```

② 注释可出现在程序的任何位置，但它不能出现在关键字或标识符中间。例：
`x=10+/*add the numbers */ 5;` 是有效的，但 `swi/* this will not work */tch(c){...` 是不正确的，因为 C 的关键字 `switch` 当中不能含有注释。通常也不希望表达式中间出现注释，因为这会使意义含混不清。

③ 注释不可嵌套，即一个注释内不可含有另一个注释。例如，以下写法在编译时出错：

```
/*this is an outer comment
x = y / a ;
/*this is an inner comment -and causes an error */
*/
```

当需要解释程序的行为时，注释应简明扼要。除了最简单和最直观的函数外，都应有注释，在函数开始处说明其功能，如何调用以及返回何处。

1.2.2 预处理

本程序的前两行称为预处理命令（详见第 9 章）。`include` 称为文件包含命令，其意义是把双引号 “” 或尖括号 < > 内指定的文件包含到本程序来，成为本程序的一部分。被包含的文件通常是由系统提供的，其扩展名为 .h，因此也称为头文件或首部文件。C 语言的头文件中包括了各个标准库函数的函数原型。因此，凡是在程序中调用一个库函数时，都必须包含该函数原型所在的头文件。在本例中，使用了三个库函数：输入函数 `scanf`，正弦函数 `sin`，输出函数 `printf`。`sin` 函数是数学函数，其头文件为 `math.h` 文件，因此在程序的主函数前用 `include` 命令包含了 `math.h`。`scanf` 和 `printf` 是标准输入，输出函数，其头文件为 `stdio.h`，在主函数前也用 `include` 命令包含了 `stdio.h` 文件。需要说明的是，C 语言规定对 `scanf` 和 `printf` 这两个函数可以省去对其头文件的包含命令。所以在本例中也可以删去第二

行的包含命令#include。

1.2.3 定义 main()函数

一个完整的C语言程序，是由若干个函数组成的，但每个程序有且仅有一个main()函数（又称主函数）。而不论其在程序中的位置，一个C语言程序总是从main()函数开始执行。当main()函数执行完毕时，亦即程序执行完毕。

main 是系统定义的主函数的函数名，main 后面的一对圆括号是必须有的，其中可放置函数的参数列表，也可无参数。用{}括起来的部分，是主函数的函数体部分，{}一般与该结构语句的第一个字母对齐，并单独占一行。低一层次的语句可比高一层次的语句缩进若干格，以便看起来更加清晰，增加程序的可读性。

1.2.4 关键字

关键字是由C语言规定的具有特定意义的字符串，通常也称为保留字。用户定义的标识符不应与关键字相同。根据关键字的作用，可将其分为数据类型关键字、控制语句关键字、存储类型关键字和其他关键字四类。下面是C标准定义的全部32个关键字。

1. 数据类型关键字（12个）

- ① char: 声明字符型变量或函数。
- ② double: 声明双精度变量或函数。
- ③ enum: 声明枚举类型。
- ④ float: 声明浮点型变量或函数。
- ⑤ int: 声明整型变量或函数。
- ⑥ long: 声明长整型变量或函数。
- ⑦ short: 声明短整型变量或函数。
- ⑧ signed: 声明有符号类型变量或函数。
- ⑨ struct: 声明结构体变量或函数。
- ⑩ union: 声明联合数据类型。
- ⑪ unsigned: 声明无符号类型变量或函数。
- ⑫ void: 声明函数无返回值或无参数，声明无类型指针（基本上就这三个作用）。

2. 控制语句关键字（12个）

- ① for: 一种循环语句。
- ② do: 循环语句的循环体。
- ③ while: 循环语句的循环条件。
- ④ break: 跳出当前循环。
- ⑤ continue: 结束当前循环，开始下一轮循环。
- ⑥ if: 条件语句。
- ⑦ else: 条件语句否定分支（与if连用）。
- ⑧ goto: 无条件跳转语句。
- ⑨ switch: 用于开关语句。
- ⑩ case: 开关语句分支。