全国高等教育自学考试

高级语言程序设计自学辅导

组编／全国高等教育自学考试指导委员会
主编／迟成文

字出版社

全国高等教育自学考试

高级语言程序设计自学辅导

全国高等教育自学考试指导委员会组编

主编　迟成文

经济科学出版社

图书在版编目（CIP）数据

高级语言程度设计自学辅导/迟成文主编. —北京：经济科学出版社，2000. 10
ISBN 7—5058—2333—7

Ⅰ. 高… Ⅱ. 迟… Ⅲ. 程序语言—程序设计—高等教育—自学考试—自学参考资料
Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2000）第 53633 号

责任编辑：范　莹
责任校对：杨晓莹
版式设计：代小卫
技术编辑：董永亭

高级语言程序设计自学辅导
全国高等教育自学考试指导委员会组编
主编　迟成文
经济科学出版社出版
社址：北京海淀区万泉河路 66 号　邮编：100086
总编室电话：62541886
网址：www.esp.com.cn
电子邮件：esp@public2.east.net.cn
涿州市星河印刷厂印刷
787×1092　16 开　13.5 印张　340000 字
2001 年 2 月第一版　2001 年 3 月第 2 次印刷
印数：5 001—13 500 册
ISBN 7-5058-2333-7/F·1725　定价：18.30 元
（图书出现印装问题，请与当地教材供应部门调换）
（版权所有　翻印必究）

前　言

为了完善高等教育自学考试教育形式，促进高等教育自学考试的发展，我们组织编写了全国高等教育自学考试自学辅导书。

自学辅导书以全国考委公布的课程自学考试大纲为依据，以全国统编自考教材为蓝本，旨在帮助自学者达到学习目标，顺利通过国家考试。

自学辅导书是高等教育自学考试教育媒体的重要组成部分，我们将根据专业的开考情况和考生的实际需要，陆续组织编写、出版文字、音像等多种自学媒体，由此构成与大纲、教材相配套的、完整的自学媒体系统。

全国高等教育自学考试指导委员会

2000 年 5 月

编者的话

本书是全国高等教育自学考试“计算机及应用”专业（专科）《高级语言程序设计》课程的指定辅导材料。

提高考生的自学效果和应试能力，是本书的惟一宗旨。而如何使用本书，是达到上述宗旨的关键。下面就这个问题提出几点思路供参考。

1. 阅读教材前可以先阅读本书第 1 部分，了解课程的知识体系、内容结构和自学计划。

2. 自学过程中可以参考本书的第 2 部分，在初步了解自学方法的基础上，浏览教材的同时，注意重点内容的掌握和难点内容的理解。自学后可以利用其中的“基本概念、基本问题与方法”，进行知识方面的自我考核。

3. 每章自学结束后，要独立认真地完成所有章末习题，利用本书第 3 部分提供的章末习题参考答案进行自我评价。

4. 全部章节学习结束后，阅读本书第 4 部分，了解各类题型的解题方法。

5. 阅读本书第 4 部分中的典型题解，进一步总结各类题型的解题思路。

6. 在掌握各类题型解题思路的基础上，重做各章章末习题，以锻炼和提高自己的解题和应试能力。

7. 考前可以选用合适的时间和地点，利用本书第 5 部分提供的两套模拟试卷，进行考前训练。

8. 在每次自我测试后，必须进行认真的总结，对于没有掌握的知识，或掌握不牢固的知识，必须进行复习或重新自学。

最后，有两点说明，务必请考生注意：

1. 本书不能代替教材。教材是按照自考大纲的规定，全面介绍各个考核知识点的内容，并通过例题加深对概念和方法的理解。本书部分内容是供考生在自学教材的过程中参考的；部分内容是在基本掌握教材内容的基础上，进行应试准备的。

2. 本书提供的典型题解、模拟试卷不能作为参加考试前的猜题依据，本书中出现的所有例题与正式考题无关，只能起到应试前自我考查的作用。

参加本书编写的人员有：第 1 部分由迟成文同志编写，第 2 部分由陈兵同志编写，第 3 部分由迟琳同志编写，第 4、5 部分由夏毅飞同志编写。全书由迟成文同志统稿。

由于编写过程中本课程的全国考试尚未正式开始，无法见到正式的试卷，只能依据已经编写好的自学考试大纲和教材。因此，对考试命题的范围、难易程度、考试试卷的格式、试卷中各类题型的题量和分数分配都很难准确把握，本书第 5 部分所提供的模拟试卷只能作为参考。

由于应试类辅导材料的特殊性，本书能否对应试者起到提高自学效果和应试能力，还有待于读者的评说。希望读者和应试人员不吝赐教，以便再版时修订。

主编　迟成文

2000 年 2 月

此为试读，需要完整PDF请访问：www.ertongbook.com

目　录

第1部分　概　　述

自学一门新的课程，首先要了解该课程的知识体系和教材的内容结构。知识体系可以使考生了解课程包含的知识及知识间的联系；可以使考生和已经掌握的知识体系类比，减轻自学过程中的难度；可以使考生将新课程的知识体系融入自己的知识体系中。教材内容结构可以使考生将教材各章内容和知识体系之间建立对应关系，自学时可以高瓴建树，区分基本知识和重点内容。

自学一门课程还要了解课程的特点，选取合适的自学方法，以便获得事半功倍的效果。合适的自学计划和自学时间的安排是自学者管理自己的一种手段，自学过程中一定会遇到各种困难，除了毅力之外，克服困难，坚持继续学习是需要一个计划的。

本部分简要介绍《高级语言程序设计》课程的概况，包括课程的知识体系与内容结构，对自学方法给出一些参考意见，同时排出了自学计划和自学时间安排表供考生参考。

1.1　课程的知识体系

《高级语言程序设计》课程的主要宗旨是学习并掌握一种计算机高级语言——C语言，用这种语言编写的程序，计算机能识别并能执行。

既然是一种语言，我们可以和大家非常熟悉的汉语进行类比。

汉语是一种语言，学习汉语，首先是学习汉字；然后学习组词、学习造句；最后学习写文章。一篇文章是由若干个语句组成的，一个语句是由若干个词组成的，一个词是由若干个汉字组成的。当然用汉字组成词需要组词的规则，用词造句需要语句的语法规则，用语句组成文章需要文章的格式规则。这些就构成了学习汉语的知识体系。

其实，汉语中的文章就相当于高级语言中的程序；汉语的语句就相当于高级语言中的语句；汉语的词就相当于高级语言中的词汇；汉语的汉字就相当于高级语言中的字符。因此，《高级语言程序设计》课程的知识体系就包括：

(1) C语言的字符集（可用来组成单词的所有字符的集合）。

(2) C语言的词汇（可用来组成语句的所有词汇）。

C语言的每种词类都有固定的格式，主要分为下列几种：

运算对象：常量、变量、数组及元素、函数调用等。

运算符号：运算符。

运算：表达式（用运算符连接运算对象构成的计算式）。

(3) C语言的语句（可用来组成程序的各种语句）。

C语言的每种语句都有固定的格式（称语法）和功能（称语义），主要分为下列几种：

数据定义语句（用来定义各种运算对象的）；

表达式语句（用来进行指定运算的，包括赋值语句、函数调用语句）；

程序控制语句（用来控制程序中语句执行顺序的，包括选择控制语句、循环控制语句、中止循环语句、继续循环语句、返回语句、转移语句等）。

（4）C语言的程序。

C语言的程序是有固定格式的，称为程序的结构。只有按照约定格式书写出的若干个语句才能构成正确的程序。

了解了课程的知识体系，就可以知道学习C语言，必须掌握的基本知识是：字符集；词汇的分类及每类词的构成规则；语句的分类及每种语句的语法（造句规则）和语义（语句的功能）；程序结构。

1.2 教材的内容结构

由于教材的编写必须循序渐进，所以课程的教材不能完全按照知识体系的顺序来编写。因此，考生在阅读教材时，应该了解每章内容是属于知识体系中的哪个部分的。

第1章 了解C语言的字符集和简单的C程序结构。

第2章 了解最基本的词类：常量、变量、运算符、表达式。

第3章 了解C语言的语句，重点是程序控制类语句。

第4章 了解新的词类：数组和数组元素。

第5章 了解新的词类：指针与指针变量。

第6章 了解新的词类：函数定义及其调用。

第7章 了解新的词类：结构型、共用型、枚举型。

第8章 了解新的词类：文件及其处理。

第9章 了解新的语句：编译预处理命令。

从上述的介绍中可以看出，学完第3章后，关于C语言的基本知识体系中的内容已全部掌握了，可以进行C语言的程序设计。后续章节是在已有的基本知识体系中不断增添新的词类和处理新词类的方法。所有的新词类不过是C语言能加工处理的新对象而已，这些新对象大大地扩充了C语言的整体功能。

1.3 自学方法指导

从前面关于知识体系的分析中，可以看出学习的关键是要掌握好知识体系中的基本内容，下面分10个部分来介绍相应的学习方法，供考生参考。

1. 先学习教材前3章，宏观上了解课程的知识体系

学习时，请抓住下列5条线条：

（1）基本数据类型。带符号整型（整型、短整型、长整型）、无符号整型（整型、短整型、长整型）、实型（单精度、双精度）、字符型、字符串型。

要了解每种类型数据的特征、能表示数据的范围、在内存中占用的字节数、数据在内存中具体存放方式（数据值的表示形式）。

（2）基本运算符。算术运算符、关系运算符、逻辑运算符、赋值运算符、条件运算符、长度运算符、逗号运算符、位运算符。

C语言的运算符非常丰富，上述运算符系列是最基本的、也是最常用的。对于每种运算符必须了解：运算符的书写方法和位置（前缀、后缀、中缀）、运算对象数（单目、双目、三目）、运算对象的数据类型、运算规则、运算结果的数据类型、运算符的优先级、同级运算符的结合性等。要注意混合运算中的自动转换原则。

（3）基本词类。常量（符号常量）、变量、函数调用、运算符、表达式。

常量要注意区分数据类型，常量的数据类型是由书写格式自动确认的。

变量也要区分数据类型，变量可以进行初始化（赋初值），还可以选取存放地点（存储类型）。变量的上述特点都是在“变量定义语句”中一次性确定的。

函数调用的格式为“函数名(实际参数表)”。如果有返回值,则可以参加运算。要会使用系统函数解决数据的输入和输出,使用系统函数要注意把“头文件”包含在源程序清单中。

运算符也是要区分数据类型的，不同数据类型的运算符加工处理的运算对象有不同的数据类型。要特别注意逻辑型数据的表示方法。

表达式也分为不同类型，通常表达式的数据类型和运算结果（表达式值）是一致的。要注意每个表达式不但能完成某个计算过程，而且还一定有一个值。

（4）基本语句。赋值语句、函数调用语句、表达式语句、选择语句（if、if－else、switch）、循环语句（while、do－while、for、break、continue）、转移语句（goto）。

赋值语句是“赋值表达式”后跟一个“分号”组成的；函数调用语句是“函数调用”后跟一个“分号”组成的；“表达式”语句是“任何表达式”后跟一个“分号”组成的。由于赋值表达式和函数调用都是表达式，所以前两种语句是“表达式”语句的特例。3种选择语句是针对3种选择结构的，要掌握每种选择结构语句的格式与功能。3种循环语句是针对3种循环结构的，要掌握每种循环语句的格式与功能。break语句只能用在循环体中或switch语句中；continue语句只能用在循环体中。这两条语句通常是和单分支语句配合使用。goto语句破坏了程序结构，在程序中很少使用。

（5）程序结构。顺序结构、选择结构、循环结构。

顺序结构的特点是自上而下按照书写顺序执行的结构，是一种简单的程序结构。

选择结构是按照某个（或某些）条件，从若干个操作（语句或语句序列组成的复合语句）中选取一个操作执行的结构。选择结构分为：单分支结构，为一个条件控制一个操作是做还是不做；双分支结构，一个条件控制两个操作，选其中一个来做；多分支结构，是n个条件控制n+1个操作，选其中一个来做。注意每种选择结构都有对应的语句来实现，如何选取合适的语句是设计选择结构程序的关键。注意多分支结构除了可选取switch语句外，还可以选取多层嵌套的双分支选择语句来实现。

循环结构是按照条件是否成立，来控制某个操作（语句或语句序列组成的复合语句）是否重复执行。循环结构分为：允许0次循环的循环结构，先判断控制循环的条件，后执行循环体；不允许0次循环的循环结构，先执行循环体，后判断控制循环的条件；次数型循环，用控制变量设初值，每次循环加一个增量，控制循环结束的条件是到达终值的方式实现固定

次数的循环。注意，每种循环结构都有对应的循环语句来实现，如何选取合适的循环语句是设计循环结构程序的关键。注意每种循环语句都可以实现3种循环结构，其中for语句实质上也是通过条件来控制循环的，使用面较广。

2. 学习数组（教材第4章）

数组是用来存放若干个数据类型相同的基本类型数据，在后续章节中还会介绍，数组也可存放若干个数据类型相同的复杂类型的数据。

数组也要遵守先定义后使用的规则，定义数组也是使用数据定义语句，可以选取存储类型、数据类型、维数、数组元素的个数、赋初值。

定义数组后，就可以使用数组元素。每个数组元素都可以看成是一个存放该数据类型的简单变量，当然也就可以当成一个简单变量来使用。

处理数组通常采用次数型循环结构。一维数组的处理使用一重循环、二维数组处理使用二重循环，在循环体中依次处理每个数组元素。

字符数组是存放字符的数组，该数组定义时的数据类型是字符型。每个字符数组的元素只能存放一个字符。一个一维的字符数组既可以存放若干个字符，也可以存放一个字符串。两者的区别就在于存放若干个字符时，最后一个字符是任意字符；而存放一个字符串的字符数组中最后一个字符必须是“字符串结束标记('\0')”。对于前者，一般只能按数组元素的引用方式分别处理其中存放的若干个字符；对于后者，可以使用字符串处理函数来处理其中的字符串。

3. 学习指针（教材第5章）

指针就是地址，这是学习“指针”概念的关键。一般变量是用来存放数据的，但是也可以用来存放地址，用于存放地址的变量就称为指针变量，以便区别于存放数据的一般变量。注意，C语言中的地址（指针）不是任意内存单元的地址常数，而是由C语言编译系统给变量分配的内存单元地址，所以我们在谈到“地址”时，一般都要说明是哪个变量的地址。如果将某个变量的地址存入指针变量，我们就说该指针变量指向对应的变量。

由于在C语言中，凡是能存放数据的对象都有地址，因此都可以定义指针变量来指向这些对象。比如指向变量的指针变量，指向数组元素的指针变量，指向数组（首地址）的指针变量，指向字符串的指针变量，指向字符数组的指针变量，指向指针变量的指针变量等等。

使用任何变量前都要先定义，参加运算前都要有值。指针变量实质上也是变量，所以使用前要先定义。定义指针变量也是使用数据定义语句，可以选取存储类型、数据类型，可以赋初值，也可以定义指针型的数组并对其赋初值。注意定义指针变量或数组时的数据类型是该变量将要指向的变量或数组的数据类型。

使用指针变量主要有两种方式。

一是给其赋值，使用格式为：指针变量＝地址值。

注意该值必须是已定义过的某个变量的地址（& 变量名）、数组的首地址（数组名）、指针变量的地址（& 指针变量名）。可以形象地说，给指针变量赋值就是让指针变量和某个存放数据的对象挂钩，也就是将指针变量指向某个变量、数组。由于被指向的对象有不同的数据类型，所以指针变量也必须分为不同的数据类型，即某种数据类型的指针变量只能指向同种类型的变量、数组。

二是利用指针变量来引用所指向的变量，使用格式为：＊指针变量。

注意该指针变量必须已指向某个对象。若已指向变量，则代表该变量；若已指向数组元

此为试读，需要完整PDF请访问：www.ertongbook.com

素，则代表该数组元素；若已指向数组首地址，则代表该数组的下标为0的元素；若已指向另一个指针变量，则代表另一个指针变量。

学习指针要特别注意下列几个问题：

（1）指针变量的定义和引用所指向的变量，都是在指针变量名前加一个“*”，但含义是不同的。

（2）数组首地址就是数组名，而变量或数组元素的地址必须通过取地址运算符“&”来获得。

（3）可以用指向一维数组首地址的指针，通过单重循环来依次处理一维数组元素。其下标为i的数组元素的引用方法为：*（指针变量+i）。

（4）对二维数组，可以使用指针变量来处理其中的元素。处理方法有3种。

方法一　使用指向二维数组首地址的指针变量按照一维数组方式通过一重循环来依次处理数组元素。例如有指针变量p和二维数组a[k1][k2]，则常见的处理方法如下：

```
for (p=a;p<a+k1*k2;p++)
   {…      /*  *p依次代表数组元素a[0][0]、a[0][1]...a[k1-1][k2-1]*/
   }
```

方法二　使用指向二维数组首地址的指针变量按照二维数组方式通过二重循环来依次处理数组元素。例如有指针变量p和二维数组a[k1][k2]，则常见的处理方法如下：

```
for (i=0;i<k1;i++)
   for (j=0;j<k2;j++)
      {…                          /*  *(p+i*k2+j)代表数组元素a[i][j]*/
      }
```

方法三　使用指向具有m个元素一维数组的指针变量按照二维数组方式通过二重循环来依次处理数组元素。例如有指向具有k2个元素一维数组的指针变量p和二维数组a[k1][k2]，则常见的处理方法如下：

```
for (i=0;i<k1;i++)
   for (j=0;j<k2;j++)
      {…                          /*  *(*(p+i)+j)代表数组元素a[i][j]*/
      }
```

4. 学习函数（教材第6章）

C语言的程序实际上是由若干个函数组合而成的，其中有且仅有一个为主函数，其余均为一般函数。

函数是学习C语言程序设计的关键。学习函数需要注意掌握下列几点：

（1）函数的一般结构。

函数的一般结构如下：

函数头

函数体

（2）函数头的设计。

设计函数头时要注意区分有参函数和无参函数。

有参函数的函数头一般如下：

存储类型　数据类型　函数名（形式参数表）

形式参数的说明

无参函数的函数头一般如下：

存储类型　数据类型　函数名（　）

（3）形式参数的说明。

形式参数的说明类似于变量、数组的定义，惟一的区别是可以省略第一维数组长度。

（4）函数体的设计。

通常函数体是一个复合语句。复合语句的结构一般如下：

数据定义部分

数据加工部分

其中数据定义部分是由数据定义语句定义出本函数需要使用的各种数据对象，如变量、数组、指针变量等，包括它们的存储类型、数据类型、赋初值。

数据加工部分是由可执行的各种语句组成，完成对定义的数据和形式参数的加工。

（5）函数返回值。

函数可以有返回值，也可以没有返回值。

有返回值的函数在设计函数头时，要注意给出函数返回值的数据类型。在设计函数体时要注意使用“return（表达式）;”语句获得返回值，返回值就是 return 语句的表达式值。调用有返回值的函数要注意接受该返回值或参与计算。

无返回值的函数在设计函数头时，要注意数据类型应是“void”。在设计函数体时要注意使用“return;”语句完成函数返回的操作。调用无返回值的函数要注意使用“函数调用”语句的形式。

（6）函数间数据传递方法。

设计与调用函数时，要注意设计好主调函数与被调函数之间数据传递的方法。主要有 4 种：

函数返回值方式，该方法只能从被调函数带回一个值。

全局外部变量方式，该方式很简单，但是破坏了程序的结构，不建议使用。

值传递方式，这是利用形参与实参结合的一种单向传递方式，只能在调用时，将实参的值传递给对应的形参。调用结束后，形参的值将不会传递给实参。因此，这种传递数据方式只能将主调函数的数据传递给被调函数。所以称为“单向传递”。

地址传递方式，这是利用形参与实参结合的一种双向传递方式，在调用时，将实参的地址传递给对应的形参作为形参的地址。由于实参和形参的地址相同，即共用相同的内存。所以在调用时，已自动将实参的值传递给形参；调用结束后，自动将形参的值传递给实参。因此，这种传递数据方式可以将主调函数的数据传递给被调函数，被调函数的数据也可以传递给主调函数，所以称为“双向传递”。

5. 学习结构型（教材第 7 章）

数组中只能存放数据类型相同的若干个有联系的数据。而现实世界中有很多数据的类型并不相同，但是它们之间却有紧密的联系。如工资表中的姓名、部门是字符串，年龄是整型，工资是实型。这些数据是不能用一个数组来存放的，结构型数据正好具有这样的特点，可以把不同类型的数据存放在一起。结构型数据是十分重要的一种数据类型，在 C 语言中使用很广泛。

学习结构型数据的处理需要掌握下列几点：

（1）不同的结构型数据可能含有不同名称或类型的成员，所以某种结构型需要用户在程序中进行定义，这个定义不是定义变量、数组等，而是定义一种类型，以后可以用这种（结构）类型来定义具体的变量、数组等。

（2）当已定义了某个结构型，就可以使用这种数据类型来定义对应的变量、数组、指针变量等，也可以给其赋初值。具体的定义方法就是使用普通的数据定义语句，只不过其中的数据类型符是“struct 结构型名”。

（3）当定义了某个结构型的变量、数组后，就可以在程序中使用它们的成员。每个结构型数据的成员就相当于一个普通变量，可以赋值或参与运算。引用结构型变量的成员方法有两种：一是利用成员运算符“.”，二是利用指向运算符“->”。注意成员运算符的前面是结构型变量名，后面是成员名；而指向运算符的前面是已经指向某个结构型变量的指针变量名，后面也是成员名。

（4）在结构型数据的处理中使用指针变量时，需要注意区分指向结构型的指针变量和指向结构型数据成员的指针变量。一般情况下，这两种变量是不能混用的，因为它们的数据类型不同。

6. 学习共用型（教材第 7 章）

共用型是为了节省内存而设置的一种用户定义的数据类型，它是把几个变量当做成员定义在一起。从表面上看来，和结构型类似，也是由若干个成员组成的，也是需要事先进行定义的，定义后的共用型也可以看成一种数据类型说明符来定义变量等；使用时也是只能引用其成员，不能直接使用共用型变量。和结构型不同的是，共用型中的所有成员是占用相同的一段内存单元，因此不能同时使用。

学习共用型时，要注意下列几点：

（1）共用型的定义方法。

（2）共用型变量的定义方法，注意不能赋初值。

（3）共用型成员的引用方法。

7. 学习枚举型（教材第 7 章）

枚举型数据像一个集合，集合中的所有元素都在定义时列出。如果某个变量的取值范围就是该集合，则可以把该变量定义成这种枚举类型的变量。

学习枚举型数据时，需要注意下列几点：

（1）枚举型的定义方法，以及枚举型常量的值的确定方法。

（2）枚举型变量的定义方法。

（3）枚举型变量的使用。

8. 学习文件（教材第 8 章）

文件处理是大部分实用程序设计中必然会遇到的问题。常见的文件处理需要解决的问题不外乎有以下几个：文件的打开、文件中数据的读写、文件的关闭。在 C 语言中，这些操作都是利用系统函数来实现的。

学习文件处理，需要注意下列几点：

（1）文件类型。

（2）文件名。

(3) 打开与关闭文件的系统函数的调用格式、参数、功能与返回值。
(4) 读写文件中字符的系统函数的调用格式、参数、功能与返回值。
(5) 读写文件中字符串的系统函数的调用格式、参数、功能与返回值。
(6) 读写文件中格式数据的系统函数的调用格式、参数、功能与返回值。
(7) 读写文件中数据块的系统函数的调用格式、参数、功能与返回值。
(8) 文件内部指针定位的系统函数的调用格式、参数、功能与返回值。
(9) 文件操作时的测试函数的调用格式、参数、功能与返回值。

9. 学习编辑预处理 (教材第 9 章)

符号常量实际上是一种最简单的无参宏定义和宏替换。本章介绍的宏定义和宏替换具有更广泛的应用。学习时要特别注意带参宏的定义和替换方法。

包含文件的主要作用是源程序清单的拼接,特别是存放系统函数有关说明的包含文件(头文件)可利用这种方式拼接到用户源程序中。当然,用户自己也可以建立自己的个人函数集和宏定义集,用包含文件方式拼接到自己开发的任何源程序中,以便简化程序设计工作。

条件编译在大型实用程序的开发和调试中很有用。

10. 学习带参数的主函数 (教材第 9 章)

带参数的主函数是编写实用程序的关键。学习时要注意以下几点:

(1) 主函数中的参数个数、名称、数据类型是固定的,用户不能改变。

(2) 调用主函数就是执行对应的程序,主函数被调用时的实参就是执行对应程序时给出的程序名,以及其后跟着的用“空格”分隔的若干个字符串。

(3) 在设计带参数的主函数时,关键是在主函数体中如何取得所有实际参数的值,以及如何对其进行加工。

1.4 自学计划与安排

下列给出的“自学计划与安排表”是按照总学时为 105 学时设计的 (表 1),其中的“顺次”是每次 1 学时,按学时顺序排列的。表中给出每次自学时的章节、重点、难点、章末必做习题。

读者可以按照自己的情况,参照执行。

表 1　　自学计划与安排表

第 1 章　概　　述				学时数: 5 学时
顺次	教材章节	重　　点	难　　点	必做习题
1	1.1 1.2	程序的概念;解决问题的方法步骤;计算机语言及其发展;C 语言的发展和特点。	C 语言的特点。	一 (1、2) 二 (1、2)
2	1.3	C 语言的字符集;保留字;标识符。	转义字符。	一 (3、4、5、6) 二 (3)
3	1.3 1.4	C 语言词类的划分;语句及其分类。		二 (4、5)

续表 1

4～5	1.5	C 程序的基本结构；函数结构；书写要求；执行顺序。	函数结构。	一（7） 二（6、7）
第 2 章　基本数据类型和运算				学时数：12 学时
顺次	教材章节	重　点	难　点	必做习题
6	2.1 2.2	数据类型的概念；数据类型的分类；常量；整型常量及其分类；实型常量及其分类。		一（4）
7	2.2	字符常量；转义字符常量；字符串常量及其长度计算；符号常量定义方法；变量、变量名称、变量地址。	转义字符常量；含转义字符的字符串常量。	一（1、2、3、5） 二（1、3）
8	2.3	变量的数据类型；基本数据类型符及其数据特征；变量的存储类型；存储类型符及其数据特征；变量初始化方法；变量初始化与存储类型的关系；完整的变量定义语句及其功能。	变量的数据类型及其定义方法；变量的存储类型及其定义方法；不同存储类型变量的使用特点和初始化的不同结果。	二（4、5、6）
9	2.3 2.4	外部变量与内部变量；变量的生存期；全局变量与局部变量；变量的作用域；C 语言运算符及其分类。	如何确定一个变量的生存期和作用域。	
10	2.4	基本算术运算符、增 1、减 1 运算符的运算规则、优先级和结合性。		一（12、13、14、18） 二（11）
11	2.4	关系、逻辑运算符及其运算规则、优先级和结合性。		一（16、19） 二（2、9）
12	2.4	赋值、算术自反赋值运算符及其运算规则、优先级和结合性。		一（6、7、8、9、10、11）
13	2.4	逗号、条件、长度运算符及其运算规则、优先级和结合性。		一（15、23） 二（7）
14	2.4	位逻辑、位移位、位自反赋值运算符及其运算规则、优先级和结合性。		一（20、21、22、24） 二（8）
15	2.5	表达式及其一般构成规则；算术、关系、逻辑表达式的构成与计算。	逻辑值的表示方法。	一（17） 二（10、13、14）
16	2.5	赋值、逗号、条件表达式的构成与计算；数据类型自动转换原则和强制转换的方法。	数据类型自动转换原则。	二（12、15）
17	复习	常量、变量、运算符和表达式的构成及运算。		
第 3 章　顺序结构、选择结构和循环结构程序设计				学时数：14 学时
顺次	教材章节	重　点	难　点	必做习题
18	3.1	顺序结构、三种选择结构、三种循环结构的特点。		二（1）

续表 2

19	3.2	赋值语句的格式及功能、字符输入/输出函数的使用方法、格式输入函数的使用方法。	输入格式字符对输入格式的控制。	一（2、4、5）
20	3.2	格式输出函数的使用方法、顺序结构的程序设计。	格式输出字符对输出格式的控制。	一（1、3） 四（1）
21～22	3.3	Turbo－C 的启动方法，主菜单及主要菜单项的使用，编辑、编译连接、运行程序的方法。	调试 C 语言源程序的操作步骤和方法。	一（6）
23	3.4	单分支、双分支选择语句的格式和功能；嵌套双分支结构。	嵌套选择语句中的 if 和 else 的配套对原则。	一（7、8） 二（2、3）
24	3.4	多分支选择语句的格式与功能。	多分支语句使用中的注意事项。	
25	3.4	选择结构的程序设计。		三（1、2） 四（2、3、4）
26	3.5	while、do-while 语句的格式与功能。	while 和 do－while 语句的主要区别。	一（9、12、14）
27	3.5	for 语句的格式与功能。	for 语句使用中的注意事项。	一（10、11、13） 二（4）
28	3.5	break、continue 语句的格式与功能。	break、continue 语句在使用上的特点。	
29	3.5	多重循环结构的实现；循环结构程序设计。	多重循环结构的实现。	二（5） 三（3） 四（5、6、7、8）
30	3.6	goto 语句的格式与功能；如何用 goto 语句构成循环。	控制使用 goto 语句的原因。	一（15）
31	复习	顺序、选择、循环结构的程序设计方法；调试源程序的方法和步骤。		四（9、10）
第 4 章　数　　组				学时数：9 学时
顺次	教材章节	重　　点	难　　点	必做习题
32	4.1	一维数组的定义、初始化、元素引用方法。	一维数组在内存的存放，各种初始化方法。	一（1）
33	4.1	一维数组的程序设计。	冒泡排序法和选择排序法。	二（4） 四（1、2）
34	4.2	多维数组的定义、初始化、元素引用方法。	多维数组在内存的存放，各种初始化方法。	二（1、2）
35	4.2	多维数组的程序设计。	矩阵的和、差、转置等算法。	三（1、2） 四（5、7）
35	4.3	字符数组的定义和初始化。		一（2、3）
37	4.3	字符数组存放字符串。	字符数组赋予字符串初值的方法。	一（7） 二（5） 四（3、6）

此为试读，需要完整PDF请访问：www.ertongbook.com