# 高级语言程序设计

（理工类各专业大专段）

夏宽理 编

# 全国高等教育自学考试指导与训练

QUANGUO GAODENG JIAOYU ZIXUE KAOSHI ZHIDAO YU XUNLIAN

主审单位

复旦大学成人教育学院

上海第一电子信息应用教育中心

华东师范大学理工学院

上海电子信息应用教育中心

复旦大学出版社

全国高等教育自学考试指导与训练

# 高级语言程序设计

（理工类各专业大专段）

夏宽理　　编

复旦大学出版社

**图书在版编目(CIP)数据**

高级语言程序设计/夏宽理编．—上海：复旦大学出版社，2001.5
（全国高等教育自学考试指导与训练）
ISBN 7-309-02820-1

Ⅰ．高… Ⅱ．夏… 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(2001)第 13476 号

**出版发行** 复旦大学出版社
上海市国权路 579 号 200433
86-21-65102941(发行部) 86-21-65642892(编辑部)
fupnet@fudanpress.com http://www.fudanpress.com
**经销** 新华书店上海发行所
**印刷** 上海锦佳装璜印刷发展公司
**开本** 787×1092 1/16
**印张** 12
**字数** 300 千
**版次** 2001 年 5 月第一版 2001 年 5 月第一次印刷
**印数** 1—6 000
**定价** 20.00 元

## 内 容 提 要

本书介绍全国高等教育自学考试《高级语言程序设计》课程(大专段)(理工类,包括计算机学科和非计算机类学科各专业)自学考试的应试方法和技巧,以帮助自学考试考生能正确按照本课程的自学考试大纲的要求,全面地复习该课程教材的内容,掌握本课程的考核要求。

本书的内容包括以下几部分。第0章介绍如何正确理解考试大纲、熟悉试卷的结构、分析试卷的知识点分布,以及应试方法。第1章至第9章对教材中的各章重要知识点、难点等给出综合的论述,并对各章习题给出详细的分析和解答。第10章是上机自测试题选编和解答,可作为考生上机测试的自习实例。最后是模拟试卷及对"2000年下半年全国高等教育自学考试《高级语言程序设计》"试卷试题进行详细的分析和解答。

本书是自学考试计算机相关各专业学习"高级语言程序设计"课程的自学参考书,也可作为大专院校各类专业学习计算机程序设计课程的辅导教材。

《全国高等教育自学考试指导与训练》

**主 审 单 位**

复旦大学成人教育学院
上海第一电子信息应用教育中心
华东师范大学理工学院
上海电子信息应用教育中心

编 委 会

# 前　言

因种种原因应届高中毕业生还不能全部进入普通高等院校接受高等教育，自学考试为希望得到高等教育的青年自学成才提供了一条行之有效之路。每年有数以百万计的自学青年参加高等教育各专业的国家考试，有数以万计的自学者通过国家考试获得大专和本科学历文凭，这些事实足以证明高等教育自学考试制度是我国高等学历教育体系的重要组成部分。

高等教育自学考试“高级语言程序设计”课程是与计算机相关各专业的必修课程。高级语言是人与计算机通信的主要语言，用程序语言描述计算问题的求解算法的方法和技巧是程序设计课程的主要内容，所以高级语言程序设计课程是计算机相关专业的入门课程之一。随着国家国民经济信息化的迅速推进和信息产业的迅猛发展，每年越来越多的自学青年报考与计算机相关的专业，学习程序设计课程。在计算机学科的许多课程中，有不少是介绍计算机的基本理论知识的，但更多的是实用课程，实用课程通常要求掌握上机操作。自学者普遍对有实践操作的课程感到学习难度较大。为了帮助自学者能系统地根据大纲要求，掌握教材内容，复习迎考，给计算机课程编写自学辅导书，帮助考生自学是必要的。

在本书的编写过程中，坚持自学辅导书的编写原则要贯彻全国高等教育自学考试大纲的精神，反映自学考试的特点，综合分析教材中的知识要点，给予自学考试应考者以正确指导。

本书介绍“高级语言程序设计”课程（大专段）自学考试的应试方法和技巧，帮助考生按照自学考试大纲的要求，全面复习课程教材内容，深入理解课程的考核目的和要求。本书以提高学生的解题能力为主要目的，力求讲解准确、由浅入深、层次分明，并提供详细的习题解答，供考生参考。

本书的内容由以下几部分组成。第0章介绍如何正确理解考试大纲、熟悉试卷的结构、分析试卷的知识点分布，以及应试方法。第1章至第9章对教材中的重要知识点、C语言的难点、用C语言编写小程序的方法等作扼要的叙述，并对教材中的习题给出详细的分析和解答。按自学考试的规定，“高级语言程序设计”的考试分笔试和上机考试两部分。本书的第10章是上机自测试题选编和解答，作为考生上机自习的参考，也可作为考生准备上机考试的自习实例。最后是模拟试卷及对“2000年下半年全国高等教育自学考试《高级语言程序设计》”试卷试题进行详细的分析和解答。

在本书编写过程中，得到多位老师的建议，并吸取了他们编写类似书籍的经验，在此深表谢意。也恳请本书的读者对本书提出宝贵意见和建议，以便在再版中臻于完善。

**编　者**

2001年4月

# 目　录

第0章　考试大纲、试卷结构和应试方法 …… 1
0.1　考试大纲 …… 1
0.2　试卷结构 …… 1
0.3　应试方法 …… 2
第1章　程序设计基础知识 …… 9
1.1　程序设计和程序设计语言 …… 9
1.2　C语言基础 …… 10
1.3　习题详解 …… 12
第2章　基本数据类型和运算 …… 15
2.1　C语言的数据类型 …… 15
2.2　常　量 …… 16
2.3　变　量 …… 17
2.4　运算符 …… 19
2.5　表达式 …… 25
2.6　数据类型转换 …… 26
2.7　习题详解 …… 27
第3章　结构化程序设计基础 …… 34
3.1　顺序结构 …… 34
3.2　常用输入输出库函数 …… 34
3.3　选择结构 …… 39
3.4　循环结构 …… 41
3.5　习题详解 …… 44
第4章　数　组 …… 56
4.1　一维数组 …… 56
4.2　二维数组和多维数组 …… 57
4.3　字符数组与字符串 …… 59
4.4　习题详解 …… 61
第5章　指　针 …… 72
5.1　指针和指针变量 …… 72
5.2　指针变量的应用 …… 73
5.3　指针数组和多级指针 …… 78
5.4　习题详解 …… 80
第6章　函　数 …… 92
6.1　函数定义 …… 92

6.2 函数调用 …… 92
6.3 函数说明 …… 94
6.4 函数调用中的数据传递方式 …… 94
6.5 返回指针的函数 …… 97
6.6 函数递归调用 …… 97
6.7 习题详解 …… 99
第7章 结构型、共用型和枚举型 …… 107
7.1 结构型和结构变量 …… 107
7.2 共用型和共用型变量 …… 112
7.3 枚举型和枚举型变量 …… 113
7.4 用户自定义类型 …… 114
7.5 习题详解 …… 115
第8章 文 件 …… 122
8.1 C文件概述 …… 122
8.2 文件处理程序结构和文件输入输出常用库函数 …… 125
8.3 习题详解 …… 130
第9章 编译预处理命令与带参数的主函数 …… 137
9.1 宏定义 …… 137
9.2 文件包含 …… 139
9.3 条件编译 …… 139
9.4 带参数的主函数 …… 141
9.5 习题详解 …… 142
第10章 上机自测试题选编和解答 …… 147
10.1 简单变量应用程序 …… 147
10.2 数组应用程序 …… 149
10.3 函数应用程序 …… 152
10.4 结构应用程序 …… 153
10.5 文件应用程序 …… 156

模拟试卷 …… 160

2000年下半年全国高等教育自学考试 **高级语言程序设计**试卷详解 …… 169

# 第0章　考试大纲、试卷结构和应试方法

## 0.1　考试大纲

全国高等教育自学考试教材《高级语言程序设计》(迟成文主编,经济科学出版社)一书中附有该课程的自学考试大纲。大纲有三部分组成:课程性质与设置目的、课程内容与考核目标、有关说明和实施要求。

其中课程性质与设置目的指出:该课程是为专业课程奠定程序设计基础而设置的,该课程还作为其它专业课程的程序设计工具得到应用,所以学好这门课程将直接关系到后继课程的学习。该课程将介绍C语言的语法规则、高级语言的主要概念和设施,编写程序的基本方法。其目的是要求考生掌握用C语言进行简单程序设计的技能,为以后参与实用程序开发奠定良好的基础。课程的基本要求有知识和技能两个方面。

在课程内容与考核目标中,对教材的各章节分别指出了具体的课程内容、学习目的与要求、考核知识点与考核要求。考生在应试前,应紧扣大纲所指定的考核要求和要掌握的知识点进行学习和复习。自学考试规定,命题必须严格遵照大纲的考核要求。课程的基本要求有知识点和程序设计技能两个方面。对于知识点,考核重要的概念、程序语言成分的有关规定、性质和使用方法。这些内容就是各章考核要求的"识记"和"领会"部分。学习程序设计的目的就是能用程序语言编写程序,这是多数考生认为最难掌握的部分。但编程能力的考核主要通过考核考生的阅读程序能力和编写简单程序的能力。考生编写程序的熟练程度将直接影响这部分的得分。提高考生编写简单程序的熟练能力的唯一途径是考生要多阅读教材中的例题,一定要自己编写习题程序,并通过上机练习正确纠正程序中的错误,提高程序编写和调试能力。

## 0.2　试卷结构

大纲第三部分中有关于命题考试的若干规定,其中第二条款指出,大纲在各章的考核要求中所列的所有细目都是考试的内容,但试题将会加大重点内容的覆盖密度。对不同能力层次的要求在试卷中有确定的比例,大致为:

"记识"占20%,约20分;"领会"占30%,约30分;"简单应用"占30%,约30分;"综合应用"占20%,约20分。

试卷中试题的难易程度的分布比例大约为:易:较易:较难:难 = 2:3:3:2。

试卷的题型有:单项选择题、填充题、程序分析题、程序设计题。对考生来说,试卷由各种题型的试题组成,熟悉各种题型及它们的解答方法,对提高考生的成绩有直接影响。

**1. 单项选择题**

每小题1分,采用"四选一"形式选出试题要求的答案。试题的内容集中在考核要求中有

"识记"标识的内容。小题涉及的具体内容一般包括:表达式计算、语言概念和C语言的规定(如变量定义及其初始化、各种数据类型的特殊性质、函数和程序结构)的正误判断、完全实现简单功能的函数或程序等。进行单项选择时,除考生非常熟悉的内容,立即能指出正确答案外,一般可以采用排除法,即首先去除显然不是答案要求的选择,然后在剩下的多个供选择的解答中作出判定选择。

**2. 填 充 题**

每小题1分,由考生直接填写正确的解答。主要考核的内容与单项选择题的考核内容类似,要求填入教材中相关的基本概念、计算表达式的值、完全实现简单功能的函数或程序等。

**3. 程序分析题**

由若干道小试题组成,每道试题的分值可能不全相等。每道小试题通常是一个小程序,要求考生回答程序的输出结果,或指出程序的功能。考生应对各小试题自己模似计算机执行程序,给出程序运行的输出结果,或根据自己掌据的程序语言知识,识别出其中语句的工作内容,指出程序的功能。这类程序或含有较复杂的表达式计算,或有条件语句,多数是有循环语句,以及函数调用时实参与形参之间传递信息和函数返回值等,数组的生成、文件的生成和输出等。

**4. 程序设计题**

程序设计题直接给出要计算机解决的问题,由考生完整编写程序或函数。解决这类问题要求考生有一定的编写简单程序的能力。试题涉及生成特定要求的数组,输入数据完成指定的判断,找出符合要求的结果输出,从文件读入数据完成一定的分类或计算等。

## 0.3 应 试 方 法

由于自学考试的社会性,为了体现公正和公平,试题不能像本科院校中常见的那样重复使用。自学考试的试卷内容有广泛且分散的特点,这要求考生必须全面地复习教材的内容。

**1. 单项选择题**

试题中的单项选择题中的多数试题涉及整本教材介绍的概念和知识点。为解答这类试题,要求考生要熟练地掌握和熟记大纲中指出的"识记"和"领会"的内容。考生应摘出教材中有关C语言的重要概念、定义、有关语言成分性质的叙述,对它们深入的理解和熟记,并对语言的一些基本规定能作简单的应用。请考生注意,理解、熟记和大段地背诵的区别。由于考题表现形式的多样性,理解是最重要的,仅对关键性的概念才有准确熟记的必要,多数的概念因是理解,并要求能熟练应用。试题通常要求对某个概念、术语或计算结果作出判断,或对一些规定作简单的应用等。由于计算机科学是一门新兴学科,许多概念还没有唯一性的定义,不同书籍由于出发点或论述领域不同,同一概念会有不同的说法,为此考生在复习迎考时,不要脱离指定自学考试教材,而从其它教材出发进行复习。为此,本书在叙述有关概念时力求与自学考试的教材保持一致。

【例0.1】设有以下代码定义字符数组c和字符指针变量pc:

```
char c[10] = "abcd", *pc = c;
```

问 *(pc+4)的值。供选择的答案有:

① "abcd"　② '\0'　③ 'd'　④ 不能确定

上述代码使字符数组 c 的前 5 个元素依次为:c[0] = 'a', c[1] = 'b', c[2] = 'c', c[3] = 'd', c[4] = '\0'。初始化 pc = c 使字符指针变量 pc 指向 c 数组的首元素 c[0]。而表达式 pc+4 的值是 c[4]的指针。因此,表达式 *(pc+4)就是引用 c[4]。所以问题的解答为②。

【例 0.2】指出下列说法中错误的叙述。

① 构成数组的所有元素的数据类型必须是相同的

② 用指针法引用数组元素允许数组元素的下标越界

③ 一维数组元素的下标为 1,2,3,…

④ 定义数组时的长度可以是整型常量表达式

由数组的概念知,数组的全部元素有相同的数据类型,另在定义数组时,需指出数组的元素个数,指定数组元素个数的表达式必须在编译时可计算的,即只允许是常量表达式,不可以含有变量。所以①和④是正确的叙述,不是问题要求的解答。在 C 语言中,当指针指向数组的某元素(不一定是数组的首元素)时,可利用该指针加减一个整表达式,构成指针表达式指向数组的某元素,然后用取内容运算符 * 间接引用指针表达式所指的数组元素。如有代码:

```
int a[100], *p;
```

表达式 p = &a[20]使 p 指向 a[20],欲通过 p 引用数组 a[1],可用表达式 *(p-19)。由 C 语言的约定,当指针指向数组某元素时,用指针表达式引用它所指的数组的某元素也可写成等价的下标引用形式,如表达式 *(p-19)可以等价地写成 p[-19]。这里 -19 是一个负整数,所以叙述②也是一个正确叙述。这种表示方法是借用下标表示法,与指针加减的整表达式引用数组元素,其中加减的整数实际不是数组元素的下标。还需要指出一点,指针与整表达式和的新指针不应该指向数组之外的别的地址。如前述的例子中指针变量 p 指向 a[20],表达式 *(p+n)中的 n 要求不能小于 20,也不能大于 79。最后,C 语言规定数组元素的下标从 0 开始顺序编号,所以选择③才是错误的叙述。

### 2. 填充题

填充题要考核的内容与选择题的考核内容基本相同,但考核的形式不同。填充题的试题多数是从基本概念、C 语言关于数据类型、程序对象、程序结构等的规定、C 程序设计基本技巧等引伸的具体应用。如 C 语言规定每个字符占一个字节,每个字符串除存储它所包含的字符外,在字符串最后一个字符之后还存有一个字符串结束符。对于这样两个基本概念和规定,填充题可能是问具体的一个字符和一个字符串各占多少个字节等。因填充题是概念或规定的具体应用,解答的难度也就比选择题的要大,不可能有猜得分的机会。

【例 0.3】下列函数的功能是统计并返回形参指针 s 所指向的字符串所含字符'A'的个数。试完成程序,写出应填写在程序空框中的代码。

```
int counts(char *s)
{  int n;
   for(n = 0; _____; s++)
      if(*s == 'A') n++;
   return n;
}
```

为统计字符指针 s 所指字符串包含的某字符的出现次数，必须用一个循环顺序考察整个字符串。由从指针 s 所指字符串的首字符开始，每考察一个字符后，指针 s 后移一个字符位置，考察循环直至字符串结束终止。所以填写在空框中的正确代码可写成 * s ! = '\0'。由于字符串束符'\0'的代码为 8 位全 0，其值为 0，正确解答也可写成 * s ! = 0，或更简洁地写成 * s。

【例 0.4】在内存中存储'A'要占用_____字节，存储"A"又要占用_____字节。

由于 C 语言规定字符只占 1 个字节，一个具体的字符当然也只占 1 个字节。字符串"A"要有 1 个字节用于存储字符'A'，另需要 1 个字节存储字符串的结束符，所以它要占用连续的 2 个字节。

【例 0.5】设整型变量 a、b 的值均为 3，执行语句：

```
b = a++, b++, ++a;
```

后，a 的值为_____，b 的值为_____。

该试题的表达式书写形式一般不会直接出现在实际应用程序中，但作为考核考生对有关表达式的计算规则，也不失为是一个很有意义的试题。赋值表达式自右至左计算，而逗号运算符的优先级最低，并且逗号表达式自左至右逐一计算，并以最后子表达式的值为逗号表达式的结果。上述表达式的计算顺序可用以下 3 个表达式语句等价表示：

```
b = a++; b++; ++a;
```

由以上一系列表达式知，变量 b 的最终值与其原来值无关，表达式 b = a++ 是先计算 a++。表达式 a++ 的值是变量 a 的原先值 3，但又让变量 a 增 1 后变为 4。然后表达式 b++ 又使变量 b 增 1，变成 4。而计算 ++a 的值，是让 a 增 1，使 a 的值变为 5。所以上述表达式使变量 a 的值变为 5，b 的值变为 4。

如上述表达式改写为：

```
b += (a++, b++, ++a);
```

请读者回答执行该表达式后，变量 a 和 b 的值又分别为多少。

### 3. 程序分析题

程序分析题要求考生阅读程序，回答程序的输出结果，或指出程序的功能。回答这类问题，要求考生将自己当作一台假想的计算机，模拟执行程序。

对于这类试题常有两种可用的方法。一是从程序的初值、循环结构、条件等发现程序的规律；二是完全从模拟执行出发读程序，求出程序的输出结果。如采用后一种方法，由于程序执行的动态性，程序中的有关变量，随着程序的执行，变量的值就会不断变化。一般来说，随时记住全部变量的当前值是非常困难的。一个行之有效的方法是用一个变量表，将程序中的全部变量罗列在该表中，某个变量值的变化记录在该变量当前值的栏中，这样就能方便地列出各个变量的动态变化过程。在这里，考生要当心函数形参及函数的局部变量与实参变量及程序的外部全局变量同名的情况。为了区别它们，对于函数形参和局部变量可以标上它所属的函数名，以与同名的实参变量及外部全局变量相区别。

由于试题程序总是完成某种有一定意义的计算工作。一般来说，程序的执行过程会有某种规律存在。如能找出程序的规律，就不需要逐句阅读程序的语句，能直接导出程序的结果。程序的规律从以下几个方面着手：有关变量的初值，特别是数组的初值；程序的循环控制结构，特别是遍历数组的循环，它的循环控制变量将控制数组元素下标的变化；循环体中的语句的条

件，一般条件有两种形式，一种是由数组元素值的大小描述，另一种是由元素的下标值描述，前者用于对其值满足某种条件的元素进行指定的计算，后者用于对满足条件的某些位置上的元素进行指定的计算。

最容易出题，变化也最多的是数组（包括字符串）处理程序，正确解答这类试题要熟练掌握两点：一是引用数组元素的两个等价方法，即用数组首元素指针（数组名）和下标引用数组元素，及通过指向数组元素的指针间接引用数组的元素；二是一些常用的简单算法，如数组或字符串遍历、插入元素或删除元素，以及常用的排序方法等。

对于文件处理程序，要注意文件当前的读/写位置，即对于读文件，注意当前读入的数据及当前读头位置；对于写文件，要注意当前写入的数据。

另外要特别指出的是，通过读程序，能发现程序执行的规律是非常有用的技术。但这个技术的掌握是建立在熟读大量的程序和自己编写过大量程序的基础上的。如一个程序是对数组的前 n 个元素执行某种操作。考生在阅读这种程序时，不妨假定输入的 n 值为 4 或 5，将 n 等于 4 或 5 的结果类推到任意的 n。

【例 0.6】阅读下列程序，写出程序运行后的输出结果。

```
#include <stdio.h>
main()
{   int a[][3] = {1,2,3,4,5,6,7,8,9};
    int i, j, s1 = 0, s2 = 0;
    for (i = 0; i < 3; i++)
       for (j = 0; j < 3; j++) {
          if (i == j) s1 += a[i][j];
          if (i + j == 2) s2 += a[i][j];
       }
    printf("s1 = %d s2 = %d \n", s1, s2);
}
```

首先将二维数组 a 的初值写成每行 3 个元素，由于程序只提供 9 个初值，a 只有 3 行：

```
1   2   3
4   5   6
7   8   9
```

程序用 a[i][j]引用 a 的元素，所以外循环 i 是控制行的循环，内循环 j 是控制列的循环，这两重循环控制遍历整个数组 a 的全部元素。条件 i == j 表示当行下标与列下标相等时，即是 a 的主对角线上的元素时，将它们累计到变量 s1，所以 s1 的值为 15。

条件 i + j == 2 用于控制行下标与列下标之和为 2。行列下标之和为某个常数的元素是同在某条右高左低斜线上的元素。对于 3 行 3 列的二维数组来说，就是副对角线上的元素。将这些元素累计于变量 s2，所以 s2 的值也是 15。

【例 0.7】阅读下列程序，简述程序的主要功能。

```
main()
{   int i, s[10], *p = &s[9];
    for(i = 0; i < 10; i++) scanf("%d", &s[i]);
    for(; p >= s; p--) printf("%d", *p);
```

```
    printf("\n");
}
```

程序中定义的变量 i 用于循环控制，数组 s 用于存储读入的整数，指针变量 p 的初值指向数组 s 的末元素。程序的第一个循环用于顺序输入数组 s 的 10 个元素的值。从程序的第二个循环代码知，每循环一次，指针 p 减 1，即指向数组的前一个元素，循环条件直至循环处理了数组的首元素后结束，循环体只是简单地输出指针当前所指的数组元素。所以该循环实现从数组的末元素开始逆序遍历数组输出。这样程序的功能可简述如下：

“顺序输入 10 个整数，并逆序输出它们的值。”

**4. 程序设计题**

程序设计题是给出问题，要求考生自己独立编写程序。考生平时认真参加上机实习，自己编写程序，是能解答这类试题的基本条件。多数考生学了程序设计以后，能基本了解教材的内容，能解答大部分前述三种类型的试题，平时还能指出别人程序的错误，但由于很少自己动手，或不知道从何着手编写程序等原因，自己还一直不会编程序。

简单程序的设计通常要包含两个步骤：首先是设想计算方法，即用什么方法来解决给定的计算问题；其次是将求解方法告诉计算机，命令计算机怎么做。第一步工作人们采用常人的思维习惯，而第二步工作必须采用计算机的思维习惯。对于程序设计的初学者来说，最困难的可能还是很难适应计算机程序的思维习惯，人们几乎无法承受程序必须将要计算机完成的计算过程描述得几乎绝对的精细和精确。但对计算机来说，这又是非常必要的。编写程序就是在向计算机讲话，非常精确地告诉计算机怎么做。

【例 0.8】编一个程序，从名为“text.txt”的文本文件中读取一个字符显示在屏幕上。

本例题要求实现最简单的文件处理。如考生知道文件处理程序的编写要点，就能方便地写出程序。

文件处理程序有以下几个要点：

(1) 在程序的开始处，用包含预处理命令，包含标准文件 stdio.h。定义文件指针变量和存储文件名的字符数组。如以下代码所示：

```
#include <stdio.h>
FILE *fp;  /* 定义文件指针变量 fp */
char fname[40] = "某文件名";
```

(2) 如文件名在程序执行时输入，可用以下代码：

```
printf("请输入文件名(包括文件的目录路径、文件的扩展名)\n");
scanf("%s%*c", fname);  /* 输入文件名及其随后的回车符 */
```

(3) 使用文件前，必须先打开文件，常用的有两种打开方式：

若文件打开为了让程序从正文文件输入数据，用读方式打开，则用以下代码：

```
if ((fp = fopen(fname, "r")) == NULL) { /* 为读打开 */
  printf("%s 文件不能打开，结束程序的执行\n", fname);
  return;
}
```

若文件打开为了让程序向正文文件输出数据，则用以下代码：

```
fp = fopen(fname, "w"); /* 为写打开 */
```

读打开时,要求被打开文件已存在。写打开时,若被打开文件不存在,则建立一个以 fname 内容命名的新文件;若被打开文件已存在,则该文件上的数据被删除。

(4) 文件使用结束后,要及时关闭,如以下代码所示:

```
fclose(fp); /* 以后 fp 又可用于打开文件 */
```

(5) 调用有关文件输入输出库函数。最经常使用的有:

调用函数 fgetc()从文件输入下一个字符,如:

```
ch = fgetc(fp); /* 将输入字符存于变量 ch */
```

调用函数 fscanf()从文件按指定格式输入数据,如:

```
fscanf(fp,"%d%d", &k, &j);  /* 从文件输入两个整数分别存于 k 和 j */
```

除在第一位置增加一个文件指针变量实参外,其余与函数 scanf()的用法全相同。

调用函数 fputc()向文件输出一个字符,如:

```
fputc(ch, fp); /* 将变量 ch 中的字符输出到文件 */
```

调用函数 fprintf()向文件按指定格式输出数据,如:

```
fprintf(fp,"%d %d \ n", k, j);
```

该函数调用是按格式要求将 k 和 j 的值输出到文件。除在第一位置增加一个文件指针变量实参外,其余与函数 printf()的用法全相同。

(6) 从正文文件逐一输入字符,作某种处理的程序结构为:

```
int c; /* 若要用 EOF 测试文件结束,则不能为 char 类型 */
FILE * fp;
… /* 说明有关变量和设置初值等 */
if ((fp = fopen(文件名, "r")) = = NULL) { /* 以输入方式打开 */
  printf("不能打开文件 %s。 \ n", "文件名字符列");
  return;
}
while((c = fgetc(fp))! = EOF)  {
  … /* 这里对刚读入的字符信息 c 作某种处理 */
}
fclose(fp);
…  /* 输出处理结果 */
```

(7) 字符逐一生成输出,形成新文件程序的一般结构形式有:

```
int c; /* 也可以是 char 类型 */
FILE * fp;
… /* 说明有关变量和设置初值等 */
fp = fopen(文件名,"w");
while(还有字符)  {
  … /* 生成字符(或字节)存于变量 c */
  fputc(c, fp); /* 将生成的字符输出 */
}
fclose(fp);
…  /* 输出程序结束报告 */
```

对于本例题，只要包含上述(1)、(3)、(5)和(4)即可，写成完整程序如下：

```
#include <stdio.h>
FILE *fp; /* 定义文件指针变量 fp */
char fname[40] = "text.txt";
main()
{   char c;  /* 或 int c */
    if ((fp = fopen(fname, "r")) == NULL) { /* 为读打开 */
      printf("%s 文件不能打开，结束程序的执行 \n", fname);
      return;
    }
    c = fgetc(fp); /* 将从文件输入的字符存于变量 c */
    printf("%c \n", c);
    fclose(fp); /* fp 所指文件关闭 */
}
```

【例 0.9】编写函数 f，该函数设有浮点型数组形参 float p[]和整型形参 n，函数的功能是计算并返回 p[]中前 n 个元素的平均值。

由于函数返回已知数组的平均值，函数的头有以下形式：

```
float f(float p[], int n)
```

函数为了计算平均值，需要两个计算步骤，首先是求出数组元素之和，然后将求得的和除以元素个数。严格地说，函数还因防止形参 n 小于等于 0 的情况，假定当 n 小等于 0 时，函数返回 0 值。为求数组元素和，需要一个存储和的变量(例如说 s)。求和通过遍历数组实现，有两种实现方法：

一是引入一个循环控制变量(例如说 i)，并让 i 作为引用数组元素的下标(如 p[i])。所以有以下代码：

```
float f(float p[], int n)
{ int i; float s;
  if (n <= 0) return 0.0;
  for(s = 0.0, i = 0; i < n; i++) s += p[i];
  return s/n;
}
```

二是由于函数的数组形参实际是一个指针变量，遍历数组直接可用指针形参 p 实现。循环次数可让变量 j 控制，j 的初值为 n，每次循环后让 j 减 1，循环直至 j 为 0 结束。写成 C 代码如下：

```
float f(float p[], int n)
{ float s; int j = n;
  if (n <= 0) return 0.0;
  for(s = 0.0; j > 0; j--) s += *p++;
  return s/n;
}
```

# 第 1 章　程序设计基础知识

教材内容按学习要求可分领会、识记、简单应用和综合应用等四个层次。其中识记知识点多数是要熟记的概念、应用规则和使用方法。简单应用知识点是指不但要求识记，而且能将这些知识应用于表达式计算和简单的程序设计中。如与简单应用知识点有关的内容直接出现在试题中，要求能没有任何困难地看懂、理解。综合应用的知识点要求考生对它们已非常熟悉，能自由地应用于自己的程序设计中。知识内容掌握程度可以分成多种层次，但因语言本身是一种综合性非常强的工具，所有知识点都是紧密连系着的，特别是综合应用的知识点都是建立在知识层次稍低的知识点上。总之，有了比较坚实的识记知识点，才会有简单应用，并通过反复实践，才能上升到综合应用的程度，达到掌握某门学问的目的。

现按教材章节顺序介绍各章节核心知识，并给出教材中习题的分析与解答。

## 1.1　程序设计和程序设计语言

### 1. 程　　序

从最一般的意义来说，程序是对解决某个计算问题的方法（算法）步骤的一种描述；而从计算机来说，计算机程序是用某种计算机能理解并执行的计算机语言作为描述语言，对解决问题的方法步骤的描述。计算机执行按程序所描述的方法步骤，能完成指定的功能。所以，程序就是供计算机执行后能完成特定功能的指令序列。

一个计算机程序主要描述两部分内容：描述问题的每个对象和对象之间的关系，以及描述对这些对象作处理的处理规则。其中关于对象及对象之间的关系是数据结构的内容，而处理规则是求解的算法。针对问题所涉及的对象和要完成的处理，设计合理的数据结构常可有效地简化算法，数据结构和算法是程序最主要的两个方面。

### 2. 程序设计的任务和主要步骤

程序设计的任务就是分析解决问题的方法步骤（算法），并将解决问题算法的方法步骤用计算机语言记录下来。程序设计的主要步骤包括：认识问题、设计解决问题的算法、按算法编写程序、调试和测试程序。在程序开发过程中，上述步骤可能有反复，如发现程序有错，严重情况可能会要求重新认识问题和重新设计算法等。

### 3. 机器语言和汇编语言

计算机能直接识别和执行的二进制代码称为计算机的机器语言。用有助于记忆的符号来代表二进制代码，称为汇编语言。汇编语言与机器语言几乎有一对一的关系。用汇编语言编写的程序称为“汇编源程序”，汇编源程序不能在计算机上直接执行，需要用汇编程序将汇编源程序翻译成机器语言程序，然后执行由汇编程序翻译出来的机器语言程序。机器语言和汇编语言是与具体计算机紧密相关的，称它们是面向机器的语言。