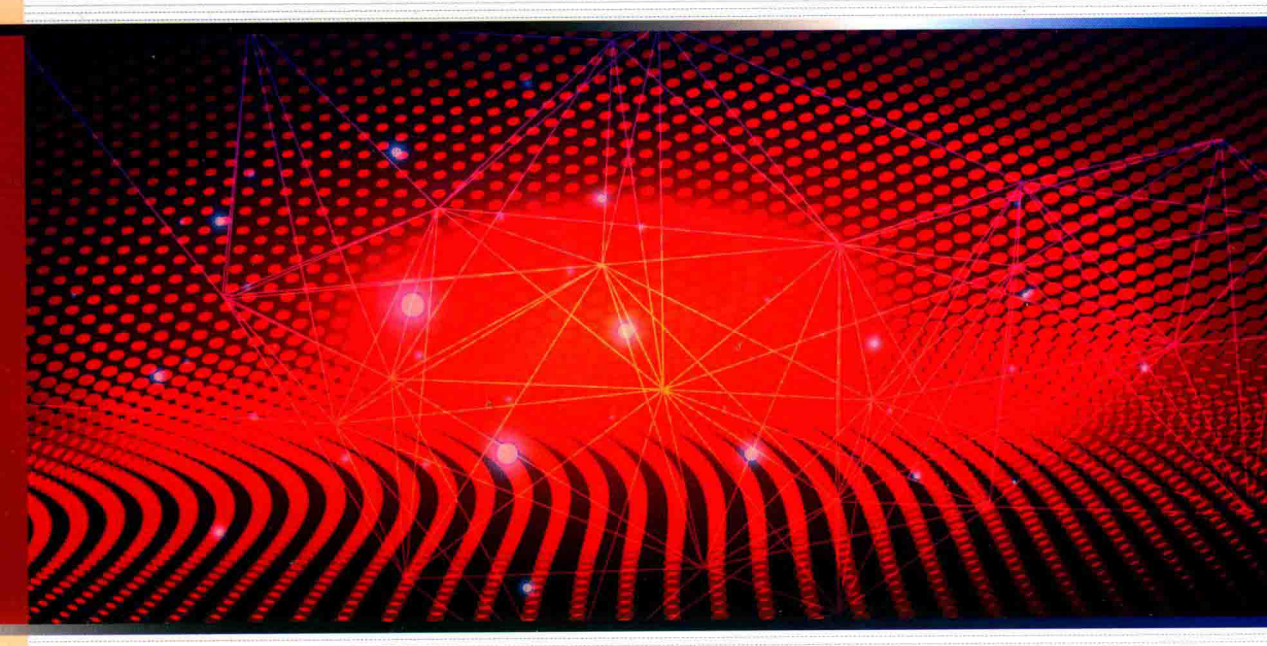
普通高等院校“十三五”精品规划教材

# 实用C语言程序设计

主编◎曾建成



哈尔滨工程大学出版社
Harbin Engineering University Press

普通高等院校“十三五”精品规划教材

# 实用C语言程序设计

主　编　曾建成
副主编（以汉语拼音为序）
蔡元宵　韩超　邵璐　王轶群　马　亮

哈尔滨工程大学出版社
Harbin Engineering University Press

## 内容简介

本书是为高等院校程序设计课程编写的教材，主要包括C程序设计概述，算法和程序，数据类型、运算符和表达式，程序结构，数组，函数，指针，结构体与共用体，文件等知识。

本书可供计算机专业的本科、职业院校学生使用，也可作为全国计算机等级考试参考书和对C语言程序设计感兴趣的读者的自学用书。

**图书在版编目（CIP）数据**

实用C语言程序设计 / 曾建成主编. -- 哈尔滨 : 哈尔滨工程大学出版社，2019.7

ISBN 978-7-5661-2351-0

Ⅰ. ①实… Ⅱ. ①曾… Ⅲ. ①C语言－程序设计－高等学校－教材 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2019)第137907号

**责任编辑** 王俊一
**封面设计** 赵俊红

**出版发行** 哈尔滨工程大学出版社
**社　　址** 哈尔滨市南岗区南通大街145号
**邮政编码** 150001
**发行电话** 0451-82519328
**传　　真** 0451-82519699
**经　　销** 新华书店
**印　　刷** 唐山唐文印刷有限公司
**开　　本** 787 mm×1 092 mm　1/16
**印　　张** 16.5
**字　　数** 422千字
**版　　次** 2019年7月第1版
**印　　次** 2019年7月第1次印刷
**定　　价** 48.00元

http://www.hrbeupress.com
E-mail：heupress@hrbeu.edu.cn

# 前　言

C 语言是当今软件开发领域里广泛使用的计算机语言之一。C 语言具有概念简洁，数据类型丰富，运算符多样，表达方式灵活，程序结构性和可移植性好等特点。C 语言既可以有效描述算法，也可以直接对硬件进行操作，适合编写系统程序和应用程序。

随着高校应用型转型的飞速发展，产教融合的进一步深化，按照普通高等教育 C 语言程序设计课程教学大纲的基本要求，本书充分体现了“必需、够用”的原则，知识叙述简明扼要、通俗易懂，内容安排由浅入深、循序渐进，同时注意突出重点、分散难点。本书全面介绍了 C 语言的基本概念、基本语法、数据类型、程序结构及计算机高级语言程序设计的方法和常规算法。

本书共 9 章，分别为 C 程序设计概述，算法和程序，数据类型、运算符和表达式，程序结构，数组，函数，指针，结构体与共用体，文件。本书中所有例题均在 Turbo C 2.0 及 Win-TC 中调试通过，可以直接引用。

本书由宁夏大学的曾建成教授任主编，由宁夏大学新华学院的蔡元宵、韩超、邵璐、王轶群和马亮任副主编，各自承担了部分章节的撰写工作。全书由曾建成教授编写大纲并统稿。本书为宁夏大学新华学院应用型转型教改项目成果。本书在编写过程中，得到了宁夏大学新华学院的大力支持与帮助，在此表示诚挚的感谢。本书的相关资料和售后服务可与 QQ（2436472462）联系获得。

本书可供计算机专业的本科生、大专生使用，也可作为全国计算机等级考试参考书和对 C 语言程序设计感兴趣的读者的自学用书。

本书在编写过程中，难免有疏漏和不当之处，敬请各位专家及读者不吝赐教。

编　者

2019 年 6 月

# 目 录

# 第 1 章　C 程序设计概述

**本章知识点**

- 程序设计语言
- C 程序的产生与发展
- C 程序的结构与上机步骤

**重点与难点**

- C 语言标识符
- C 程序的结构

## 1.1　程序设计语言

程序设计语言按照书写形式和思维方式的不同，可分为低级语言和高级语言两大类。低级语言包括机器语言和汇编语言。

### 1.1.1　低级语言

**1. 机器语言**

我们已经知道，要使用计算机解决一个问题，必须先编制好程序。程序是由一系列指令组成的。机器语言是以二进制代码的形式来表示这些基本指令集合的。它是计算机系统唯一能够直接识别和执行的程序设计语言。它的优点是运算速度快，每条指令均为由 0 和 1 组合起来的代码串，由操作码和操作对象两部分组成。

其中，操作码用来指出运算种类，如“加”“减”“乘”“除”“跳转”等，操作对象用来指示参与运算数据保存的位置，如存储器的某个地址或某个寄存器等。该语言的缺点是每一条指令相当于一个单词或短语，缺乏表达复杂长句子的语法结构和能力，可读性差，难查错，难修改。就如同一个咿咿呀呀学说话的婴儿，难以沟通和交流信息。鉴于此，研究工作中很快就发明和产生了比较易于阅读和理解的汇编语言。

**2. 汇编语言**

汇编语言实际上是由一组汇编指令构成的语言，与机器语言相比，它可以用指令英文名称的缩写字符串来表示其所代表的操作，用标号和符号来代表地址、常量和变量。如：“ADD AX, BX;”实现将两个寄存器 AX 与 BX 中的数相加的功能。这种方式便于识别和

记忆，执行效率也较高。因为计算机只能识别机器指令，所以汇编语言指令写的程序不能在计算机系统上直接执行，需要借助汇编语言翻译程序（简称汇编程序），将这种符号化的语言转换成可以直接执行的机器指令程序，才能被执行。

### 1.1.2 高级语言

高级程序设计语言（简称高级语言）是指用于描述计算机程序的类自然语言。它是程序设计发展的产物，它屏蔽了机器的细节，提高了语言的抽象层次。高级语言采用接近自然语言和数学语言的语句，易学、易用、易维护，并且在一定程度上与机器无关，给编程带来了极大方便。

针对不同的应用领域，人们设计出几百种各具特点的高级语言，目前常用的也有几十种。例如：适合初学者的 BASIC 语言，易学易用；适用于科学计算的 FORTRAN 语言，具有强大的数值计算功能；适用于商业和管理领域的 COBOL 语言；第一个系统体现结构化程序设计思想的 PASCAL 语言适用于教学之中。在这些计算机语言中，C 语言既具有其他高级语言的优点，又具有低级语言的许多特点，它功能丰富，移植性强，编译质量高，被称为适用最广泛的计算机语言之一。

## 1.2 C 语言基本知识

### 1.2.1 C 语言的产生和发展

#### 1. 起源

C 语言的历史可以追溯到 20 世纪 60 年代末期，可以说 C 语言是在贝尔实验室 Ken Thompson、D. M. Ritchie 及其他同事在开发 UNIX 操作系统的过程中的副产品。Thompson 独自用汇编语言编写了 UNIX 操作系统的最初版本，但用汇编语言编写的程序往往难以调试和改进，UNIX 系统也不例外。Thompson 意识到需要用一种更加高级的编程语言来完成 UNIX 系统未来的开发，于是他设计了一种小型的 B 语言。Thompson 的 B 语言是在 BCPL（basic combined programming language）语言的基础上开发的（BCPL 语言是 20 世纪 60 年代中期产生的一种系统编程语言），而 BCPL 语言的起源又可以追溯到一种最早的且影响最深远的 ALGOL60 语言。

不久，Ritchie 也加入到 UNIX 项目，并且开始着手用 B 语言编写程序。1970 年，贝尔实验室为 UNIX 项目争取到一台 PDP-11 计算机。当 B 语言经过改进并且运行在了 PDP-11 计算机上时，Thompson 就用 B 语言重新编写了部分 UNIX 代码。到了 1971 年，B 语言已经暴露出非常不适合 PDP-11 计算机的问题，于是 Ritchie 开始开发 B 语言的升级版。他最初将新开发的语言命名为 NB 语言（意为“New B”），但是后来，新语言越来越脱离 B 语言，于是他决定将它改名为 C 语言。到 1973 年，C 语言已经足够稳定，可以用来重新编写 UNIX 系统了。

**2. 标准化**

整个 20 世纪 70 年代，特别是 1977 年到 1979 年之间，C 语言一直在持续发展。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 程序移植到其他机器时所需做的工作大大简化了。1978 年，Brian Kernighan 和 Dennis Ritchie 合作编写并出版了影响深远的名著 *The C Programming Language*。此书一经出版就迅速成为了 C 程序员的宝典。由于当时缺少 C 语言的正式标准，因此这本书就成为了事实上的标准，编程爱好者把它称为“K&R”或者“白皮书”。

1983 年，美国国家标准化协会（American National Standards Institute，ANSI）开始编制 C 语言标准。经过多次修订，C 语言标准于 1988 年完成，并且在 1989 年 12 月正式通过，成为 ANSI 标准 X3.159-1989。1990 年，国际标准化组织（International Organization for Standardization ISO）通过此项标准，将其作为 ISO/IEC 9899-1990 国际标准。我们把这些标准中描述的 C 语言称为“ANSI C”（标准 C）。虽然经常把“K&R”第 1 版中描述的 C 语言称为 K&R C。

## 1.2.2　C 语言的特点

由于 C 语言预期的用途（编写操作系统和其他系统软件）和 C 语言自身的基础理论体系，C 语言有自己的优缺点。概括来讲，其主要特点如下。

（1）高效性。高效性是 C 语言与生俱来的优点之一。因为 C 语言原来就用于编写传统的由汇编语言编写的应用程序，所以快速运行并占用有限内存就显得至关重要了。实验表明，针对同一问题，C 语言的代码效率只比汇编语言低 10%～20%。

（2）可移植性。移植是指程序从一个环境不加改动或稍加改动就可以在另一个环境中运行。C 语言编译器规模小且容易编写，这使得此种编译器得以广泛应用。

（3）功能强大。C 语言拥有一个数据类型和运算符的庞大集合，这个集合使得 C 语言具有强大的表达能力，往往寥寥几行代码就可以实现许多功能。

（4）系统级操作。C 语言可以直接访问物理地址，能进行位（bit）操作，能实现汇编语言的大部分功能，可以直接对硬件进行操作，是一种适合系统编程的语言，而这些却是其他编程语言试图隐藏的内容。

（5）灵活性。虽然 C 语言最初的设计目的是系统编程，但是没有固有的约束将它限制在此范围内。C 语言现在可以用于编写从嵌入式系统到商业数据处理的各种应用程序。C 程序比其他程序简练，源程序短，采用的表达方式简洁，书写形式自由。虽然灵活性可能会让某些错误溜掉，但是它却使编程变得更加轻松。

（6）标准库。C 语言的一个突出优点就是它的标准库，包含了数百个函数，这些函数可以用于输入/输出、字符串处理、存储分配以及其他一些实用的操作。

## 1.2.3　C 语言的字符集

C 语言的字符集是用来书写源程序清单时允许出现的所有字符的集合，字符是组成语言的最基本的元素。C 语言字符集由字母（小写字母 a～z 共 26 个，大写字母 A～Z 共 26 个）、数字（0～9 共 10 个）、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。

### 1.2.4 C 语言的标识符

在编写程序时，需要对变量、函数、数组、宏和其他实体进行命名，这些名字称为标识符（identifier）。简单地说，标识符就是一个名字。在 C 语言中，标识符可以含有字母、数字和下划线，但是都必须以字母或者下划线开头。

几种合法和非法的标识符：

| 合法 | 非法 |
| --- | --- |
| student | 5student（以数字开头） |
| _ok | ok?（含有特殊字符“?”） |
| student_num | student.num（标识符中不能含“.”，只能含下划线） |

C 语言是区分大小写的。也就是说，在标识符中 C 语言区别大写字母和小写字母。例如，下列所示的标识符全是不同的。

job joB jOb jOB Job JoB JOb JOB

上述 8 个标识符可以全部同时使用，且每一个都有完全不同的意义。因为 C 语言是区分大小写的，许多程序员都会遵循在标识符命名时只使用小写字母的规则（宏命名除外）。标准 C 对标识符的最大长度没有限制，所以不用担心不能使用过长的描述性名字。

### 1.2.5 C 语言的关键字

关键字又称保留字，是一种预先定义的、具有特殊意义的标识符。用户不能重新定义关键字，也不能把关键字定义为一般的标识符，如关键字不能作变量名、函数名等。表 1-1 是标准 C 的所有关键字。C 语言的关键字有类型标识符、控制流标识符、预处理标识符等。所有的关键字均用小写字母。

表 1-1 标准 C 的所有关键字

| | | | |
| --- | --- | --- | --- |
| auto | double | int | struct |
| break | else | long | switch |
| case | enum | register | typedef |
| char | extern | return | union |
| const | float | short | unsigned |
| continue | for | signed | void |
| default | goto | sizeof | volatile |
| do | if | static | while |

## 1.3 C 程序的结构

下面以 1 个简单的 C 程序为例，分析 C 程序的组成特性。

【例 1.1 】在计算机屏幕上输出一行文字“Welcome!”。

实现这个功能的 C 语言源程序如下：

```
# include <stdio.h>
```

```
int main（）
{
    printf（"Welcome! \n"）;
    return 0;
}
```


图 1-1　例 1.1 运行结果

运行结果如图 1-1 所示。

即使是最简单的 C 程序也依赖 3 个关键的语言特性：命令行（在编译操作前修改程序的编辑命令），函数（被命名的可执行代码块，例如 main 函数）和语句（程序运行时执行的命令）。

在编译 C 程序之前，预处理器会首先对 C 程序进行编辑。我们把预处理器执行的指令称为命令。后面会详细介绍这部分内容，这里只关注#include 命令。在上面程序中的第 1 行# include <stdio.h>通常称为命令行，这条命令说明，在编译前把<stdio.h>中的信息“包含”到程序中。<stdio.h>包含了关于 C 标准输入/输出库的信息。C 语言拥有大量类似于<stdio.h>的头文件。每个头文件都包含一些标准库的内容。这段程序中包含<stdio.h>的原因是：C 语言不同于其他的编程语言，它没有内置的“读”和“写”命令，因此，进行输入/输出操作就需要用标准库中的函数来实现。所有命令行必须用“#”开头，这个字符可以把 C 程序中的指令和其他代码区分开来。默认情况下，命令行是一行，命令行的结尾既没有分号也没有其他特殊标记。

在 C 语言中，函数仅仅是一系列组合在一起并且被赋予了名字的语句。函数类似于其他编程语言中的“过程”或“子程序”，它们是用来构建程序的构建块。事实上，一个 C 程序就是一个函数的集合。函数分为两大类：一类是用户编写的函数，另一类则是由 C 语言的编译器提供的库函数（library function）。

在上面的程序中，main 表示“主函数”，C 语言规定必须用 main 作为主函数名。其后的一对圆括号中间可以是空的，但这一对圆括号不能省略。程序中的第二行 main（）是主函数的起始行。在函数的起始行后面是函数体。函数体由大括号{}括起来。一个 C 程序可以包含任意多个不同名的函数，但必须有一个而且只能有一个主函数。一个 C 程序总是从主函数开始执行。

语句是程序运行时执行的命令。本例中主函数只有一个输出语句，printf 是 C 语言中的输出函数。双引号内的字符串原样输出。“\n”是换行符，即在输出“Welcome!”后回车换行。C 程序规定，每条语句都以分号结束。

【例 1.2】已知矩形的两条边长分别是 3 和 4，求矩形的面积。

参考程序如下：

```
# include <stdio.h>
int main（）
{
    int a,b,area;              /* 定义 a、b 和面积 area 为整型变量 */
    a=3;                       /* 给矩形两条边赋值*/
    b=4;
    area=a*b;                  /* 求出面积将值赋给 area */
    printf（"a=%d,b=%d,area=%d\n",a,b,area）; /* 输出矩形的两条边长和面积*/
```

```
    return 0;
}
```

```
a=3,b=4,area=12
```

图 1-2　例 1.2 运行结果

运行结果如图 1-2 所示。

上面程序的作用实际是求 a、b 两个数的乘积。“/* …… */”语句，其目的是增强程序的可读性。注释部分必须用“/*”和“*/”包围。“/*”和“*/”必须成对地出现，“/”和“*”之间不可以有空格。注释可以用英文，也可以用中文。注释可以出现在程序中任何合适的地方。它既可以单独占行也可以和其他程序文本出现在同一行中。注释部分对程序的运行不起作用

程序第 4 行是定义部分，定义变量 a 和 b，指定 b 为整型（int）变量。第 5 行起的两个语句是赋值语句，使 a 和 b 的值分别为 3 和 4。第 7 行使 area 的值为 a*b，第 8 行中“%d”是控制输入输出的“格式字符串”，用来指定输入输出时数据类型和格式，“%d”表示十进制整数类型。在执行输出时，“%d”将由 a、b 和 area 的值取代，“a=”“b=”和“area=”原样输出。

一般地，简单的 C 程序具有如下形式。

```
/*第一部分：编译预处理语句*/
/*自定义类型或全局变量定义*/
…
/*第二部分：子函数的声明或定义*/
…
/*第三部分：主函数*/
main（）
{
变量声明与定义语句;
可执行语句;
}
/*第四部分：子函数定义*/
……
```

通过以上几个程序例子，可以看到一个 C 程序的结构有以下特点。

（1）一个 C 程序至少包含一个 main 函数，或者包含一个 main 函数和若干个其他函数。也就是说，C 程序是由函数构成的，函数是 C 程序的基本单位。其他函数可以是系统提供的标准库函数，也可以是用户根据实际需要自己设计编写的函数。

（2）一个函数的基本结构如下。

```
函数类型函数名（函数参数类型函数参数名，…）
{声明部分;
执行部分;
}
```

其中的声明部分和执行部分合在一起又被称为函数体。

在 C 语言中，也允许函数体是空的，这种函数被称为空函数。如：

```
dump（）
{ }
```

空函数什么也不做，但它是合法的。我们在设计调试一些大型程序时可以在某些地方

放置一个空函数。

（3）C 程序的执行都是从 main 函数开始，并且一定结束于 main 函数，而不管 main 函数在程序中的位置如何。

（4）函数体中的每一个语句都要以分号结束。C 语言的书写格式是非常自由的，我们可以把多个语句写在一行上，也可以把一个语句分写在多行上，系统是以分号判断一个语句结束的。

（5）在程序的任何地方都可以加入以“/*”和“*/”包围起来的注释，注释的作用是增加程序的可读性，它并不被系统执行。

（6）C 语言中大小写字母是严格区分的。例如：main 如果任何一个字母写成大写就是错的。这一点我们在使用时要特别注意。

## 1.4　C 程序的上机步骤

通过前面的学习，我们了解到要使计算机能按照人的意图工作，就要根据问题的具体要求，编写相应的程序。程序是一组计算机可以识别和执行的指令，每一条指令使计算机执行特定的操作。程序可以用高级语言或汇编语言编写，用高级语言或汇编语言编写的程序称为源程序。

C 程序源程序的扩展名为“.c”。因为计算机只能识别和执行由 0 和 1 组成的二进制指令，所以源程序不能直接在计算机上执行，需要用“编译程序”将源程序翻译为二进制形式的“目标程序”。目标程序的扩展名为“.obj”。目标代码尽管已经是机器指令，但是还不能运行，因为目标程序还没有解决函数调用问题，需要将各个目标程序与库函数连接，才能形成完整的可在操作系统下独立执行的程序，称为“可执行程序”。可执行程序的扩展名为“.exe”。图 1-3 表示了一个 C 语言程序经过编辑、编译、连接到运行的全过程。

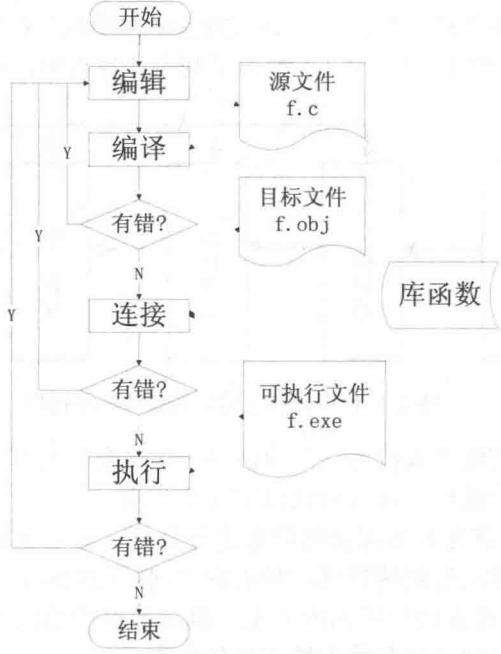



图 1-3　C 程序的上机步骤

# 第 2 章　算法和程序

**本章知识点**

- 程序设计的基本步骤
- 算法及算法的特性
- 算法的描述方法
- 结构化程序设计方法

**重点与难点**

- 用流程图表示算法
- 常用算法举例

## 2.1　程序设计的基本步骤

程序设计是运用计算机解决问题的一种方式，通常从对实例问题的分析着手，设计合适的算法，进而转化成某种计算机语言编写的程序并输入到计算机，经调试后执行这个程序，最终达到解决问题的目的。图 2-1 给出了利用计算机解决问题的基本过程。

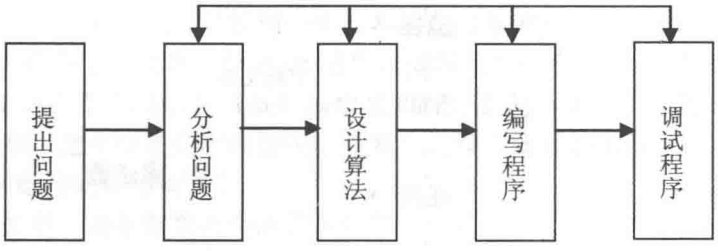



图 2-1　利用计算机解决问题的基本过程

程序设计语言是实现“人机对话”的桥梁，程序设计的基本过程是指从确定任务到得到结果、写出文档的全过程，可以分成以下几个步骤。

（1）分析问题。首先对要解决的问题进行分析，对要处理的对象进行调查，研究所给定的条件，分析最后应达到的目标，找出解决问题的规律，选择解题的方法。

（2）设计算法。根据用户提出的要求，确定数据的组织形式和数据结构，针对存放数据的数据结构来确定解决问题的方法和具体步骤。

（3）编写程序。使用选定的计算机语言编写程序代码，把流程图描述的算法用计算机语言描述出来，变成能由计算机运行的目标程序。

（4）调试程序。调试程序就是对送入计算机的程序进行编译、排错、试运行的过程，调试的结果是得到一个能正确运行的程序。程序中常见的错误有两种：一种是语法错误；另一种是逻辑错误。语法错误一般会在源程序被编译时由编译程序及时发现，因此相对比较容易排除，而程序的逻辑错误常常是潜在的。

## 2.2　算法的基本知识

### 2.2.1　算法的概念

做任何事情都有一定的步骤。为解决一个问题而采取的方法和步骤，就称为算法。计算机算法是为实现某个任务而构成的简单指令集，是有穷的计算过程。它规定了在程序中对数据进行正确处理的运算步骤。世界著名的计算机科学家 Nikiklaus Wirth 提出的公式为

数据结构+算法=程序

本书认为

程序=算法+数据结构+程序设计方法+语言工具和环境

这 4 个方面是一个程序设计人员所应具备的知识。在程序设计当中，算法是灵魂，数据结构是加工对象，算法解决的是“做什么”和“怎么做”的问题。程序中的操作语句，就是对算法的体现。计算机算法可分为以下两大类。

（1）数值运算算法：求解数值。

（2）非数值运算算法：事务管理领域。

下面通过实例说明如何根据问题给出确定的算法。

【例 2.1】输入 10 个数，找出其中最大的数，并输出。

分析：解决此类问题的一般思路是引入一个变量 max 保存最大数。先将输入的第一个数存入 max，然后输入第二个数并与 max 比较，如果大于 max，则用它取代 max 的原值。再输入第三个数，做同样的操作。依次进行下去，直到所有数据输入完为止。

除变量 max 外，还要引入一个变量 i 累计已输入数据的个数，一个变量 x 暂时存放当前输入的数据，算法描述如下。

Step1：输入一个数，存放在一个变量 max 中。

Step2：设置用来累计比较次数的计数器 i（也是一个变量），并给 i 赋初值 1，即 1→i。

Step3：输入一个数，存放在另一个变量 x 中。

Step4：比较 max 和 x 中的数，若 x>max，则将 x 的值送入 max；否则，max 的值不变。

Step5：i 增加 1，即 i+1→i。

Step6：若 i<9，则返回 Step3，继续执行；否则，输出 max 中的数，此时 max 中的数即为最大数。

（说明：在算法描述中经常使用“=>”或“→”表示赋值。）

【例 2.2】求 1×2×3×4×5 的值。

方法 1：

Step1：先求 1×2，得到结果 2。

Step2：将步骤 1 得到的乘积 2 乘以 3，得到结果 6。

Step3：将 6 再乘以 4，得 24。

Step4：将 24 再乘以 5，得 120，这就是最后的结果。

以上步骤是按照常规数学方法计算得到的结果，这样的算法虽然是正确的，但表述起来过于烦琐。如果要计算 100 以内的所有数的积，则要写 99 个步骤，这显然是不可取的。

对于此类问题需要进一步寻找规律。如果设两个变量，一个变量代表被乘数，一个变量代表乘数，每一步的乘积放在被乘数变量中，每做一次乘法运算后，使乘数的值增加 1，则可以使用循环算法求出结果。设 t 为被乘数，i 为乘数，具体表述如下。

方法 2：

Step1：1→t

Step2：2→i

Step3：t×i→t

Step4：i+1→i

Step5：若 i≤5，返回步骤 Step3；否则，算法结束。

如果采用此方法来计算 100!，只需将 Step5 中 i≤5 改成 i≤100 即可。

该算法不仅正确，而且是计算机较好的算法，因为计算机是高速运算的自动机器，实现循环轻而易举。在确定算法时，对同一个问题，可能有不同的解决方法和步骤，即有多种算法。有的算法可能只需要很少的步骤而有些算法则需要较多的步骤。一般而言，应该选择易于理解、简单、步骤少的算法。所以在进行算法设计时，不仅要保证算法的正确性，还要考虑算法的质量，选择合适的算法。

## 2.2.2 算法的特性

算法是指为解决某个特定问题而采取的确定且有限的步骤。一个算法应当具有以下五个特性。

（1）有穷性：一个算法包含的操作步骤应该是有限的。

（2）确定性：算法中每一条指令必须有确切的含义，不能有二义性，对于相同的输入必须能得到相同的执行结果。

（3）可行性：算法中指定的操作，都可以通过已经验证过可以实现的基本运算执行有限次后实现。

（4）有 0 个或多个输入：在计算机上实现的算法是用来处理数据对象的，在大多数情况下这些数据对象需要通过输入来得到。

（5）有一个或多个输出：算法的目的是为了求解，这些解只有通过输出才能得到（注意：算法要有一个以上的输出）。