普通高等教育“十二五”规划教材

程序设计初步入门教程

（C语言）

◎ 高东日　张英宣　主编

中国石化出版社
HTTP://WWW.SINOPEC-PRESS.COM

普通高等教育“十二五”规划教材

程序设计初步入门教程

（C语言）

高东日　张英宣　主编

中国石化出版社

内容提要

本书主要介绍C语言的入门基础知识，由浅入深、循序渐进地介绍了C语言程序设计，同时结合程序设计理论基础，使学生的程序设计基本方法和基本技能进一步提升。本书内容主要包括软件技术基础、信息存储基本知识、C语言常量、变量、程序结构、数组、字符串、指针、函数和文件等一些非常重要的知识。通过阅读本书，读者可以在较短的时间内理解C程序设计的各个重要概念和知识点，为进一步学习打好基础。

本书适合C语言入门新手阅读；对于有一定基础的读者，可通过本书进一步理解C语言的各个重要知识点和概念；可作为高等学校各专业的程序设计入门教材，也可作为培训和自学教材。

图书在版编目(CIP)数据

程序设计初步入门教程(C语言)／高东日，张英宣主编．
—北京：中国石化出版社，2014.1
ISBN 978-7-5114-2598-0

Ⅰ.①程… Ⅱ.①高… ②张… Ⅲ.①C语言-程序设计-教材
Ⅳ.①TP312

中国版本图书馆CIP数据核字(2014)第008637号

中国石化出版社出版发行
地址：北京市东城区安定门外大街58号
邮编：100011　电话：(010)84271850
读者服务部电话：(010)84289974
http://www.sinopec-press.com
E-mail：press@sinopec.com
北京富泰印刷有限责任公司印刷
全国各地新华书店经销
*
787×1092毫米 16开本 11.75印张 213千字
2014年2月第1版　2014年2月第1次印刷
定价：32.00元

前　言

本书是根据教育部非计算机专业计算机课程教学指导委员会制定的《非计算机专业计算机基础课程教学基本要求》和《关于进一步加强高等学校计算机基础教学的意见暨计算机基础课程教学基本要求(试行)》中提出的要求，在总结多年来从事计算机程序设计教学的经验基础上编写的。本书的特点是引导入门、培养程序设计思想、注重教材的理论与实践相结合，为学生的进一步学习奠定基础。

C语言是目前世界上应用非常广泛的高级程序设计语言，它是国内外各高等院校各专业的计算机基础课程，在人才培养中占有重要的地位和作用，C语言是一种结构化程序设计语言，不仅应用广泛，同时目前流行的面向对象程序设计语言，如C++、Java、C#等都是在C语言的基础上发展派生而来的。本书选用C语言作为高级语言程序设计的程序设计工具，全面、详细地介绍了C语言的程序设计思想。通过学习C语言，学生不仅能够掌握程序设计的基本思想，也可为今后学习打下良好的基础。

本书共分11章，其中包括软件技术基础、信息存储知识、C语言编程与基础知识、选择结构、循环结构、数组、函数、指针、文件和C语言上机指导。本书整体结构安排合理，循序渐进、由浅入深，采用理论与例题相结合的方式，使复杂问题变得简单化，有助于学生对知识的理解与掌握。通过对C语言程序设计方法的了解和掌握，领会高级语言程序设计的基本思想和基本方法。

本书由辽宁石油化工大学高东日、张英宣主编，魏海平、常东超、石元博、吕宝志参编，全书由高东日统稿。

本书在编写过程中参考了不少国内外文献和资料，在此谨向这些文献和资料的作者表示衷心的感谢。编写过程中得到了多位资深教授的支持和帮助，他们提出了许多宝贵意见和建议，在此表示衷心的感谢。

由于编者水平有限，书中难免有不妥之处，恳请读者指正。

目　录

第1章　软件技术基础

1.1　结构化和面向对象的程序设计

1.1.1　结构化程序设计

结构化程序设计是采用结构化技术(结构化技术分析、结构化设计、结构化实现)来完成软件开发的各项任务，并使用适当的软件工具或软件工程环境来支持结构化技术的运用。结构化程序中的任意基本结构都具有唯一入口和唯一出口，并且程序不会出现死循环。在程序的静态形式与动态执行流程之间具有良好的对应关系。

1. 程序设计的原则

程序设计方法的主要原则可以概括为：自顶向下，逐步求精，模块化，限制使用 goto 语句。

(1) 自顶向下：即先考虑总体，后考虑细节；先考虑全局目标，后考虑局部目标。这种程序结构按功能划分为若干个基本模块，这些模块形成一个树状结构。

(2) 逐步求精：对复杂问题，应设计一些子目标作过渡，逐步细化。

这种设计方法的过程是将问题求解由抽象逐步具体化的过程。用这种方法分解复杂问题，直到把复杂问题分解为可以直接用程序语言的基本语句结构表达出来为止。这种方法就叫做“自顶而下，逐步求精”。在向下一层展开之前应仔细检查本层设计是否正确，只有上一层设计是正确的才能向下细化。如果每一层设计都是正确的，则整个算法就是正确的。

(3) 模块化：是把程序要解决的总目标分解为分目标，再进一步分解为具体的小目标，把每个小目标称为一个模块。

(4) 限制使用 goto 语句。

2. 结构化程序的基本结构与特点

1966 年，Boehm 和 Jacopini 证明了程序设计语言仅仅使用顺序、选择和重复(循环)三种基本控制结构就足以表达出各种形式结构的程序设计方法。采用结构化程序设计方法编写程序，可使程序结构良好、易读、易理解、易维护，从而

可以提高编程工作的效率，降低软件开发的成本。

（1）顺序结构：顺序结构是一种简单的程序设计结构，顺序结构自始至终严格按照程序中语句的先后顺序逐条执行，是最基本、最常用的结构形式；它是程序设计中的必备，如图1-1(a)所示。

（2）选择结构：又称为分支结构，它包括简单选择和多分支选择结构，如图1-1(b)所示。

（3）重复结构：重复结构又称为循环结构，它根据给定的条件，判断是否需要重复执行某一相同功能的程序段。在程序设计语言中，重复结构对应两类循环语句，对先判断后执行的循环体称为当型循环结构，对先执行后判断的循环体称为直到型循环结构，如图1-2所示。

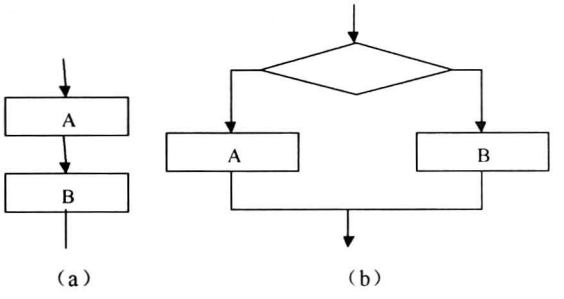


图1-1 顺序与选择结构

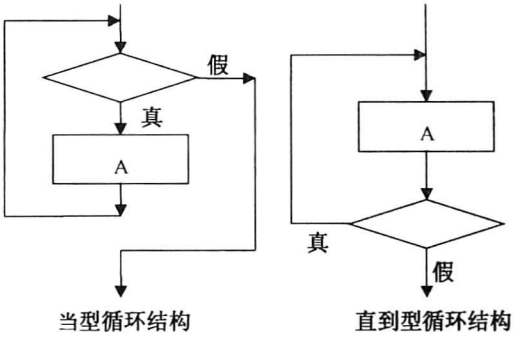


图1-2 循环结构

结构化程序设计的优点如下：

（1）程序的可读性好，易于维护。

（2）提高了编程效率，降低了开发成本。

3. 结构化程序设计的原则和方法的使用

在结构化程序设计的具体实施中，要注意把握如下要素。

（1）使用程序设计语言中的顺序、选择、循环等有限的控制结构表示程序的控制逻辑。

（2）选用的控制结构只允许有一个入口和一个出口。

（3）程序语句组成容易识别的块，每块只有一个入口和一个出口。

（4）复杂结构应该用嵌套的基本控制结构进行组合嵌套来实现。

（5）语言中没有的控制结构，应该采用前后一致的方法来模拟。

（6）严格控制非结构化语句(如goto、break、continue语句)的使用。除非以下情况：

① 使用后可以极大提高程序的效率，而且不但不影响程序可读性，反而使程序机构更加清晰，才考虑使用。

② 用一个非结构化的程序设计语言去实现一个结构化的构造，当然目前此类情况已是微乎其微。

1.1.2　面向对象的程序设计

面向对象程序设计是一种计算机编程架构。面向对象程序设计的一条基本原则是计算机程序是由单个能够起到子程序作用的单元或对象组合而成。它达到了软件工程的三个主要目标：重用性、灵活性和扩展性。为了实现整体运算，每个对象都能够接收信息、处理数据和向其他对象发送信息。

1. 关于面向对象方法

客观世界中任何一个事物都可以被看成是一个对象，对象是现实世界事物或个体的抽象表示，抽象的结果不仅包括事物个体的属性，还包括事物的操作。属性值表示了对象的内部状态。面向对象方法的本质就是主张从客观世界固有的事物出发来构造系统，提倡用人类在现实生活中常用的思维方法来认识、理解和描述客观事物，强调最终建立的系统能够映射问题域，也就是说，系统中的对象以及对象之间的关系能够如实地反映问题域中固有事物及其关系。从计算机的角度来看，面向对象就是运用对象、类、继承、封装、消息、结构与连接等面向对象的概念对问题进行分析、求解的系统开发技术。

面向对象方法有以下几个主要优点：

（1）与人类习惯的思维方法一致。

（2）稳定性好。

（3）可重用性好。

（4）易于开发大型软件产品。

（5）可维护性好。

2. 面向对象方法的基本概念

面向对象的程序设计方法中涉及的对象是系统中用来描述客观事物的一个实体，是构成系统的一个基本单位，它由一组表示其静态特征的属性和它执行的一组操作组成。面向对象方法学中的对象是由描述该对象属性的数据以及可以对这些数据施加的所有操作封装在一起构成的统一体。对象可以做的操作表示它的动态行为，在面向对象分析和面向对象设计中，通常把对象的操作称为方法或服务。属性在设计对象时确定，一般只能通过执行对象的操作来改变。对象有一些基本特点，即标识唯一性，分类性，多态性，封装性，模块独立性好。

1）对象

对象(Object)是面向对象方法中最基本的概念。它可以用来表示客观世界中的任何实体，也就是说，应用领域中有意义的、与所要解决的问题有关系的任何

事物都可以作为对象。总之，对象是对问题域中某个实体的抽象。

2）类和实例

类(Class)是对具有共同特征的对象的进一步抽象。将属性和操作相似的对象归为类，也就是说，类是具有共同属性、共同方法的对象的集合。所以，类是对象的抽象，它描述了属于该对象类型的所有对象的性质，而一个对象则是其对应类的实例(Instance)。如杨树、柳树、枫树等是具体的树，抽象之后得到“树”这个类。类具有属性，属性是状态的抽象，如一棵杨树的高度是10m，柳树是8m，树则抽象出一个属性“高度”。类具有操作，它是对象行为的抽象。

3）继承

继承(Inheritance)是使用已有的类定义作为基础来建立新类的定义技术。已有的类可当作基类来引用，则新类相应地可当作派生类来引用。面向对象软件技术的许多强有力的功能和突出的优点都来源于把类组成一个层次结构的系统：一个类的上层可以有父类，下层可以有子类。这种层次结构系统的一个重要性质是继承性，一个类直接继承其父类的描述或特性，子类自动地共享基类中定义的数据和方法。

4）聚合

聚合(Aggregation)模拟了现实世界的部分与整体的关系。它允许利用现有的类组成新类。比如说汽车是由发动机、变速箱、底盘等组成，那么就可以利用发动机、变速箱、底盘等类聚合成一个新的类——汽车类。

5）消息

消息(Message)是一个实例与另一个实例之间传递的信息，它请求对象执行某一处理或回答某一要求的信息，它统一了数据流和控制流。消息中包含传递者的要求，它告诉接受者需要做哪些处理，但并不指示接受者应该怎么样完成这些处理。消息完全由接受者解释，接受者独立决定采用什么方式完成所需的处理，发送者对接受者不起任何控制作用。一个对象能接受不同形式、不同内容的多个消息；相同形式的消息可以送往不同的对象，不同的对象对于形式相同的消息可以有不同的解释，能够作出不同的反映。一个对象可以同时向多个对象传递消息，两个对象也可以同时向某个对象传递消息。

消息是对象之间交互的唯一途径，一个对象要想使用其他对象的服务，必须向该对象发送服务请求消息。而接收服务请求的对象必须对请求作出响应。

6）多态性

多态性(Polymorphism)是指在一般类中定义的属性或行为，被特殊类继承之后，可以具有不同的数据类型或表现出不同的行为。

多态性机制不仅增加了面向对象软件系统的灵活性，进一步减少了信息冗

余，而且显著提高了软件的可重用性和可扩充性。当扩充系统功能增加新的实体类型时，只需派生出与新实体类相应的新的子类，完全无须修改原有的程序代码，甚至不需要重新编译原有的程序。利用多态性能够发送一般形式的消息，而将所有的实现细节都留给接受消息的对象。

1.2　软件工程基础

1.2.1　软件工程的基本概念

1. 软件的定义、特点

1）软件的定义

计算机软件是指计算机系统中与硬件相互依赖的另一部分，包括程序、数据和有关的文档，是与计算机系统的操作有关的计算机程序、规程、规则，以及可能有的文件、文档及数据。

2）软件的特点

软件是逻辑产品，具有抽象性；硬件是物理产品，具有可见性。软件的具体特点如下：

（1）软件开发更依赖于开发人员的业务素质、智力、人员的组织、合作和管理。软件开发、设计几乎都是从头开始，成本和进度很难估计。

（2）软件存在潜伏错误，硬件错误一般能排除。

（3）软件开发成功后，只需对原版进行复制。

（4）软件在使用过程中维护复杂。

（5）软件不会磨损和老化。

2. 软件工程过程与软件生命周期

1）软件工程过程

软件工程过程是将软件工程的方法和工具综合起来，以达到合理、及时地进行计算机软件开发的目的。通常把用户的要求变成软件产品的过程也叫作软件开发过程。

软件工程过程包含软件规格说明、软件开发、软件确认和软件演进四种基本活动。

2）软件生命周期

软件产品从定义开始，经过开发、使用和维护，直到最终退役的全过程称为软件生存周期。可将软件生存周期划分为三个过程，共九个阶段。

三个过程是：软件定义过程、软件开发过程、软件使用与维护过程。

九个阶段是：可行性研究、需求分析、概要设计、详细设计、实现、组装测试、验收测试、使用与维护、退役。对每个阶段都明确规定了该阶段的任务、实施方法、实施步骤和完成标志，其中特别规定了每个阶段需要产生的文档。它们之间的关系如图 1 -3 所示。

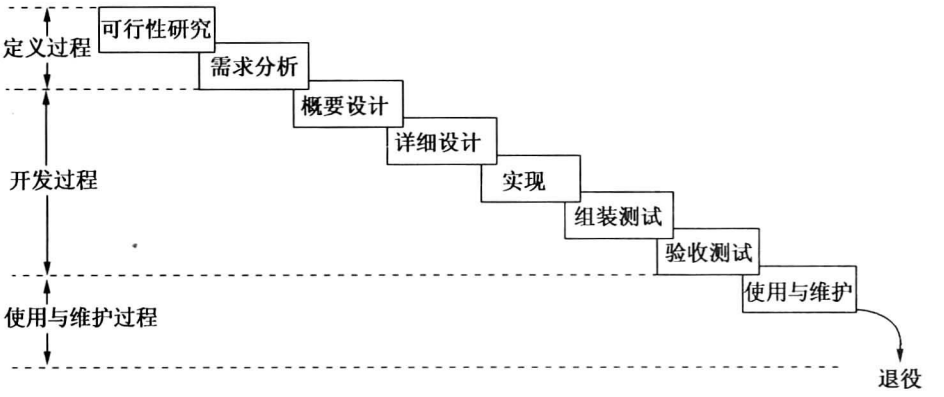


图 1 -3　软件生存周期阶段的划分

3. 软件工程的目标与原则

1）软件工程的目标

软件工程的目标是在给定成本和进度的前提下，开发出具有有效性、可靠性、可理解性、可维护性、可重用性、可移植性、可追踪性和可互操作性且满足用户需求的产品。

2）软件工程的原则

为了达到上述目标，在软件开发过程中必须遵循软件工程的基本原则，这些基本原则包括抽象、信息屏蔽、模式化、局部化、确定性、一致性、完备性和可验证性。

抽象、信息隐藏、模块化和局部化的原则支持可理解性、可修改性、可靠性等目标，并可提高软件产品的质量和开发效率；一致性、完全性和可验证性等原则可以帮助软件开发人员去实现一个正确的系统。

1.2.2　结构化分析方法

结构化方法是结构化分析、结构化设计和结构化编程的总称，其核心和基础是结构化程序设计理论。

结构化分析方法(Structured Analysis)是结构化程序设计理论在软件需求分析阶段的运用。

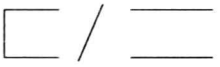
结构化分析方法的实质是着眼于数据流，自顶向下，逐层分解，建立系统的流程，以数据流图和数据字典为主要工具，建立系统的逻辑模型。

结构化分析的常用工具有：

1）数据流图

数据流图(Data Flow Diagram，DFD)是描述数据处理过程的工具，是需求理解的逻辑模型的图形表示，它直接支持系统的功能建模，从数据传递和加工的角度刻画了数据流从输入到输出的移动变换过程，数据流图基本符号的含义说明见表1－1。

表1－1　数据流图的基本符号的含义

符号	说　明
	加工，输入数据在此进行变换产生输出数据，中间要注明加工的名字
	数据输入的源点和数据输出的终点，在其中要注明源点或终点名字
	数据流，被加工的数据及数据流向，在箭头边要用名词或名词性短语给出数据流的名字
	数据存储文件。要用名词或名词性短语给出数据文件的名字

2）数据字典

数据字典(Data Dictionary，DD)是结构化分析方法的另一种有力工具，是结构化分析方法的核心。数据字典是对所有与系统相关的数据元素的一个有组织的列表，使用户和系统分析员对于输入、输出、储存成分和中间计算结果有共同的理解。

概括地说，数据字典的作用是对DFD中出现的被命名的图形元素作确切解释。通常数据字典包含的信息有名称、别名、何处使用/如何使用、内容描述、补充信息等。

同时，数据字典也是软件维护时使用的一种重要资料。如果要求所有开发人员都根据公共的数据字典描述数据和设计模块，则能避免许多麻烦的接口问题，提高开发的效率和质量。

1.2.3　结构化设计方法

1. 软件设计的基本内容

1）软件设计的任务和目标

软件设计是软件工程的重要阶段，是一个把软件需求转换为软件表示的过程。

软件设计包括概要设计和详细设计两部分。

（1）概要设计：将软件需求转换为软件结构和数据结构，并编写概要设计说明书。

（2）详细设计：通过对软件结构的细化，得到软件的详细算法和数据结构，产生描述软件的详细设计文档。

2）软件设计的基本原理

软件设计遵循软件工程的基本目标和原则，建立了适用于在软件设计中应该遵循的基本原理和软件设计有关的概念。

（1）抽象：是一种思维工具，就是把事务的本质的共同特性提取出来，而不用考虑其他细节。

（2）模块化：是指把一个待开发的软件分解成若干小的简单部分。

（3）信息隐蔽：是指在一个模块内包含的信息（过程或数据），对于不需要这些信息的其他模块来说是不能访问的。

（4）模块的独立性：是指在每一个模块只完成系统要求的独立的子功能，并且与其他模块的联系最少且接口简单。

模块的独立程度是评价设计好坏的重要度量标准。衡量软件的模块独立性使用耦合性和内聚性两个定性的度量标准。其中内聚性是指一个模块内部各个元素间彼此结合的紧密程度的度量；耦合性是指模块间相互连接的紧密程度的度量。在程序结构中，各模块的内聚性越强，则耦合性越弱。一般较优秀的软件设计应尽量做到高内聚，低耦合，即减弱模块之间的耦合性和提高模块内的内聚性，有利于提高模块的独立性。

2. 概要设计

1）概要设计（又称为初步设计或总体设计）的主要任务

（1）设计软件系统结构。

（2）设计数据结构及数据库。

（3）编写概要设计文档。

（4）概要设计文档评审。

2）设计软件结构的图形工具

（1）层次图用来描绘软件的层次结构，适于在自顶向下设计软件的过程中使用。

（2）结构图（Structure Chart，SC）也叫程序结构图、软件结构图、系统结构图，它的符号如图 1 -4 所示。其中方框代表模块，框内注明模块的名字或主要功能；方框之间的大箭头或直线表示模块的调用关系；带注释的小箭头表示模块调用时传递的信息及其方向。尾部加空心圆的小箭头表示传递数据信息，加实心圆的小箭头表示传递控制信息。

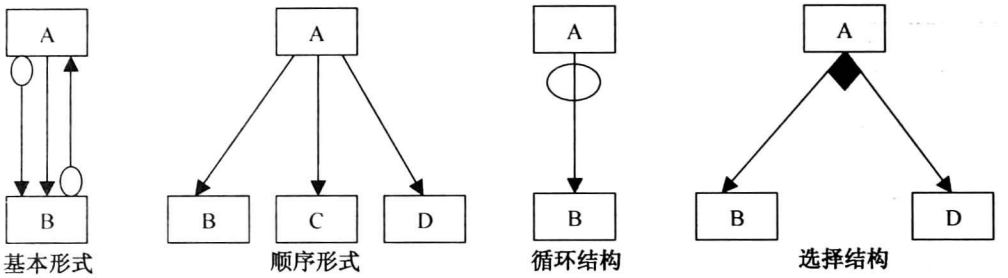


图 1－4　结构图构成的基本形式

3）面向数据流的设计方法

所有系统的 DFD 结构可分为两种典型形式，即变换型结构和事务型结构。相应地，由数据流图转换成的软件结构图也有变换型和事务型两种结构。

（1）变换型结构。变换型数据流图中数据的处理过程大致可分为三步，即输入数据、变换数据和输出数据。

将变换型数据流图映射为软件结构图的方法是先将数据流图的输入、变换和输出部分分别转换为输入、变换和输出模块；然后在输入、变换和输出模块上增加总控模块，以调度这三个模块，协调完成任务。

（2）事务型结构。事务型数据流图有一个明显的事务中心，它接受一项事务，根据该事务的特点和性质，选择分配一个适当的处理单元，然后输出结果。所以，事务型数据流图由接受事务、事务中心和若干处理单元输出结果部分组成。

将事务型数据流图映射为软件结构图的方法是先将数据流图中的各个部分转换成软件结构图中的相应模块；然后增加调度模块，让它调度 n 个处理单元，最后让事务中心模块调度接受事务、调度和输出结果模块。

4）设计准则

面向数据流设计方法的设计准则包括提高模块独立性；模块规模适中；深度、宽度、扇出和扇入适当；使模块的作用域在该模块的控制域内；应减少模块的接口和界面的复杂性；设计成单入口、单出口的模块；设计功能可预测的模块。

3. 详细设计

在概要设计阶段完成了软件系统的总体设计，规定了各个模块的功能及模块之间的联系后，就要进一步考虑实现各个模块规定的功能，也就是进行软件的详细设计，也称为过程设计。

程序流程图（Program Flow Chart）又称为程序框图。常用标准程序流程图符号见表 1－2。