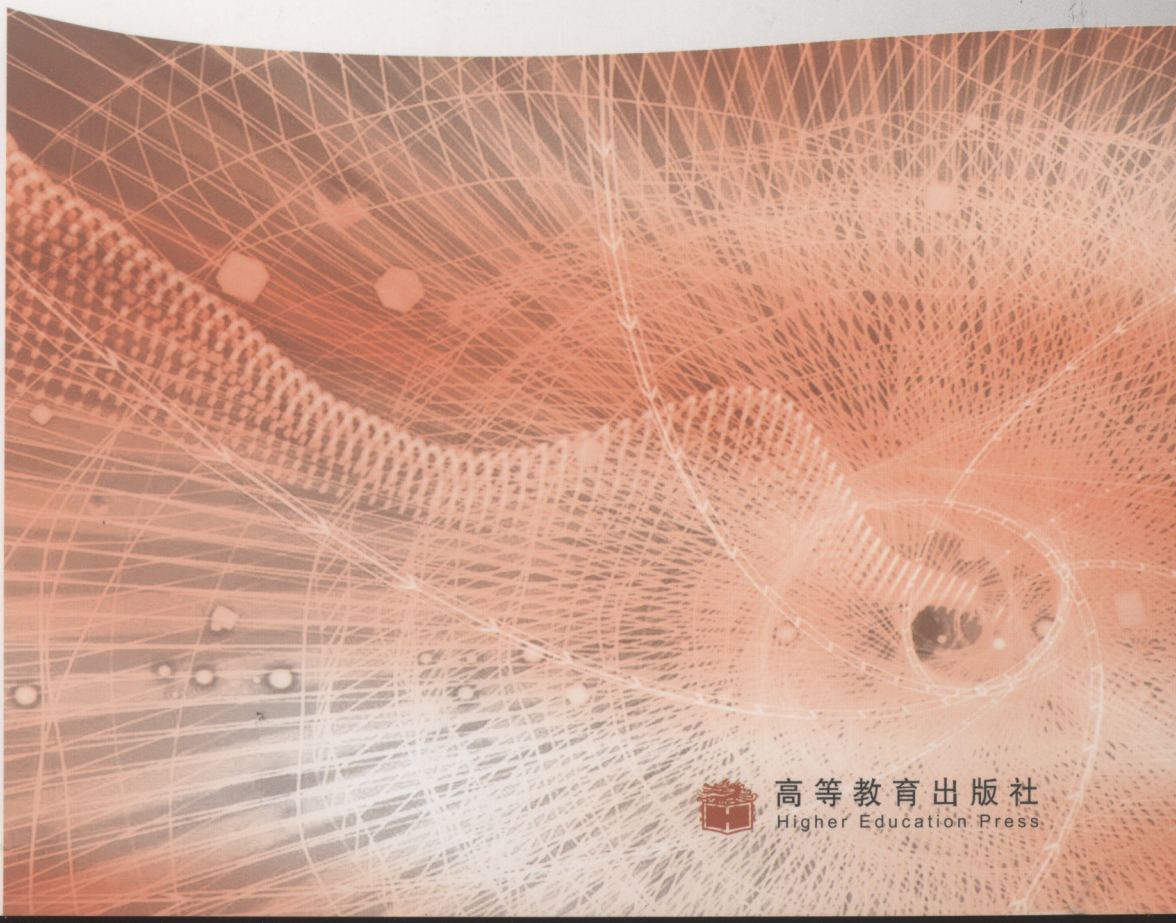
普通高等教育“十一五”国家级规划教材

国家精品课程主讲教材

编译原理

Compiler Principles and Techniques

蒋宗礼　姜守旭　编著



高等教育出版社
Higher Education Press

普通高等教育“十一五”国家级规划教材
国家精品课程主讲教材

编译原理

Bianyi Yuanli

蒋宗礼　姜守旭　编著

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容提要

“编译原理”是计算机科学与技术专业重要的专业(基础)课程。本书是普通高等教育“十一五”国家级规划教材,也是国家精品课程主讲教材,是作者结合近三十年在哈尔滨工业大学、北京工业大学讲授该课程的经验和体会,根据本科生教学的实际需要选择和组织有关内容撰写而成的,包含了“编译原理”课程所需涵盖的知识。本书将以知识为载体,对本学科问题求解的典型思想和方法进行探讨,致力于学生四大专业基本能力的培养,为“能力导向”的课程教学提供有力支持。为了便于读者学习和掌握有关内容,面向工程应用型学生的培养,在附录中给出了相应的课程设计。

本书适合于高等学校计算机科学与技术学科本科生“编译原理”课程教学使用,也可供有关专业的学生、教师和科研人员参考。

图书在版编目(CIP)数据

编译原理 / 蒋宗礼,姜守旭编著. -- 北京 : 高等教育出版社, 2010.2
ISBN 978-7-04-029058-5

Ⅰ. ①编… Ⅱ. ①蒋… ②姜… Ⅲ. ①编译程序—程序设计—高等学校—教材 Ⅳ. ①TP314

中国版本图书馆CIP数据核字(2010)第016500号

策划编辑	刘 艳	**责任编辑**	康兆华	**封面设计**	张志奇
责任绘图	尹 莉	**版式设计**	马静如	**责任印制**	朱学忠

出版发行	高等教育出版社	**购书热线**	010-58581118
社 址	北京市西城区德外大街4号	**咨询电话**	400-810-0598
邮政编码	100120	**网 址**	http://www.hep.edu.cn
总 机	010-58581000		http://www.hep.com.cn
		网上订购	http://www.landraco.com
经 销	蓝色畅想图书发行有限公司		http://www.landraco.com.cn
印 刷	保定市中画美凯印刷有限公司	**畅想教育**	http://www.widedu.com
开 本	787×1092 1/16	**版 次**	2010年2月第1版
印 张	28.25	**印 次**	2010年2月第1次印刷
字 数	640 000	**定 价**	33.10元

物料号 29058-00

蒋宗礼

北京工业大学教授，1978年3月至1984年7月在哈尔滨工业大学计算机学科学习，先后到美国、加拿大进修，1984年、1985年先后在哈尔滨工业大学和北京工业大学主讲编译原理、形式语言与自动机理论等课程。从事网络计算、操作系统、人工神经网络和计算机科学与技术学科的教育教学等研究，发表了大量研究论文，参加起草计算机科学与技术专业规范、认证文件、计算机技术工程领域硕士研究生培养基本要求等。

北京市教学名师，国家精品课程和北京市精品课程“编译原理”负责人，主编的《形式语言与自动机理论(第2版)》为国家2008年度普通高等教育精品教材，国家优秀教学团队负责人。获国家教学成果二等奖2项，北京市教学成果一等奖3项，二等奖1项，另有多项省部级教学、科研成果一、二、三等奖。曾获中国高校优秀青年学者、宝钢优秀教师、航天部优秀青年教师等荣誉称号。

主要学术兼职有全国工程教育专业认证专家委员会计算机类专业认证试点工作组成员、教育部高校计算机专业教学指导分委员会秘书长、中国计算机学会教育专委主任兼本科组组长、中国计算机学会教育工委副主任、全国高校计算机教育研究会副理事长。

姜守旭

哈尔滨工业大学教授，国家优秀教学团队骨干成员，1986年9月至1990年7月在哈工大计算机系学习，1993年以来在哈工大主讲编译原理、数据库系统、离散数学等课程。主要从事数据库、对等计算与信任管理、操作系统、无线传感器网络、容迟网络等方面的研究，主持或参加国家973计划、国家自然科学基金、国防预研及省部级科研项目等10余项，获省部级科技进步三等奖2项，获国家级教学成果二等奖1项、省级教学成果一等奖2项、二等奖1项，在VLDB、ICDE、VLDB Journal、软件学报等国内外重要学术会议或学术刊物上发表学术论文30余篇，出版普通高等教育“十一五”国家级规划教材1部(该教材被评为2008年度国家精品教材)、译著1部，主持建设的“编译原理”课程被评为教育部-微软精品课程和黑龙江省精品课程，作为骨干成员参与建设的“形式语言”课程被评为教育部双语教学示范课程和黑龙江省精品课程。主要学术兼职有2006-2010年教育部高等学校计算机科学与技术专业教学指导分委员会专家工作组成员等。

前　言

根据《中华人民共和国高等教育法》第二章第十六条高等学历教育学业标准，本科教育应当使学生比较系统地掌握本学科、专业必需的基础理论、基本知识，掌握本专业必要的基本技能、方法和相关知识，具有从事本专业实际工作和研究工作的初步能力。法律规定了高等教育对知识、能力、素质三方面的要求。专业基本能力在学生的可持续发展和创新精神、创新意识与创新能力的形成中具有非常重要的作用。所以，教育不仅要强调知识基础，更要强调能力基础。

在对知识基础和能力基础的追求上，东西方教育存在着一定的差异。相对而言，东方教育表现出更注重夯实知识基础的倾向，而西方教育则更注重夯实能力基础。实际上，“知识基础”和“能力基础”并不矛盾，两者相辅相成：以知识为载体，通过对知识的学习，掌握恰当的问题求解思想和方法，培养学生的（专业）能力；能力的增强又可以促进学习、掌握甚至发现更多的知识。所以，倡导的是研究型“教”与“学”，尊崇的是“能力导向”。

在大学里，不能简单、肤浅地将学习一门课程看成是未来要从事这门课程所含内容的研究、设计和开发，而要关注是否在有限时间内最有利于实现对受教育者专业能力的培养。所以，我们反对面向系统的教育，更反对产品教育。由于计算学科还是一个较新的学科，其专业教育从总体上来说还不够成熟，所以才有了今天的“操作系统”、“数据库系统”、“网络系统”、“编译系统”等面向系统的课程。相信随着计算学科的发展，计算机专业教育会不断成熟，会形成更能体现专业教育需要的课程。目前应该努力做到“使用工具、探索规律”、“实现具体系统、研究基本原理”，也就是“使用工具，不忽略规律；学习系统，未冷落原理”。

那么，作为计算机专业的学生，应该具有什么样的基本能力呢？除了交流、获取知识与信息的基本能力、应用数学和自然科学知识的能力、创新能力、工程实现能力、团队合作能力外，作为接受专业教育的专业人员，更应该具备专业基本能力。自2002年起，作者就将计算机专业人才的专业基本能力归纳成计算思维能力、算法设计与分析能力、程序设计与实现能力、系统能力。这四大基本能力有着自己丰富的内涵，大约有83个“能力点”，它们的培养需要落实到各个教学环节中，特别是各门主干课程的教学中。

例如，系统能力要求学生站在系统的全局去看问题、分析问题和解决问题，并实现系统优化。对计算机专业人才来说，狭义的系统能力包括对一定规模系统的“全局掌控能力”（从全局上掌控一定规模的系统）和在构建系统时能够系统地考虑问题的求解方法。要培养学生的系统能力，就需要在基本思想的指导下，从教学的点滴入手。例如，自顶向下是系统设计的重要思想方法，用于引导学生分层次考虑问题，逐步求精；鼓励学生由简到繁，进行复杂程序的设计，是一个逐渐深入、逐渐扩展规模的过程；结合计算机硬件系统、编译系统、操作系统等教学，可以使学生学着关注并努力去掌握系统逻辑，引导学生从宏观到微观去分析、理解和把握系统；通过参与较

大型系统的设计与实现，鼓励学生在工作过程中努力掌握系统的总体结构，关心本人承担的工作在系统中的地位及其与其他部分的关系等，以此增强系统观和合作能力。在教学中要不断提升学生的眼光，以使学生从系统级上对算法和程序进行再认识。

“编译原理”是一门非常好的课程。Alfred V. Aho 编著的 Compilers: Principles, Techniques, and Tools 被认为是编译领域里的经典教材，加上其封面上“龙”的造型，被人们尊称为“龙书”。作为第 1 章的第一句话，作者这样写道“编写编译器的原理和技术具有十分普遍的意义，以至在每个计算机科学家的研究生涯中，本书中的原理和技术都会被反复用到。”这句话给出了这门课程的真正教学定位。

因为计算学科是对信息描述和变换算法的系统研究，包括理论、分析、效率、实现和应用，问题求解的基本思路是“问题、形式化描述、计算机化”，以抽象、理论、设计为其学科形态。编译原理涉及的是一个比较适当的抽象层面上的数据变换，既抽象，又实际；既有理论，又有实践，而且这些实践还是理论指导下的实践。课程还包含一个具有一定规模的系统的设计，非常适于对学生进行系统能力的培养。此外，还含有基本问题求解的典型思想、技术和方法。因此，课程教学还应使学生掌握程序变换的基本概念、问题描述和处理方法（自顶向下、自底向上、逐步求精、递归求解、目标驱动、问题的抽象与形式化描述、算法设计与实现、数据结构的选取与实现、系统构建、模块化）等。所含的问题描述与处理方式，有利于进一步培养学生的形式化描述能力：给出问题的形式化描述，基于这种描述设计出自动化处理的过程，最后实现“自动计算”，体验实现自动计算的乐趣，使学生养成“问题、形式化描述、计算机化”的问题求解习惯，推进从“实例计算”到“类计算”和“模型计算”的跨越。

所以，该课程的内容对于培养学生的计算思维、算法设计与分析、程序设计与实现、计算机系统的认知、开发和利用等四大专业基本能力非常重要，是在程序设计、数据结构与算法等课程中受到一定的锻炼后，从系统的级别上对程序、算法的认识进行再提高，通过课程进一步提升学生的计算机问题求解水平，在系统的级别上重新认识算法和程序，增强系统能力。因此，“编译原理”课程是计算机专业最为恰当、有效的知识载体之一。

自 1986 年以来，我们在哈尔滨工业大学和北京工业大学为计算机科学与技术专业学生讲授该课程，这门课程在两校都是主干课，从原来的专业课变成现在的专业技术基础课。多年的研究和教学实践积累，使我们不断加深对该课程的认识和理解，两校的“编译原理”课程分别被评为国家、北京市、黑龙江省精品课程。

本书按能力培养的要求，结合作者 20 余年的经验和体会，根据本科生教学的实际需要选择和组织有关内容，包含了“编译原理”课程所需涵盖的知识，但是本书没有按照惯例着力追求知识的全、深、新，而是以知识为载体，对本学科问题求解的典型思想和方法进行探讨，致力于学生四大专业基本能力的培养。

全书共分为 11 章，第 1 章为编译原理引论，第 2 章讲文法，第 3 章讲词法分析，第 4、5 章分别讲述自顶向下和自底向上的语法分析方法，第 6 章给出语法制导翻译与属性文法，第 7 章具体讨论语义分析与中间代码生成，第 8 章是符号表管理，第 9 章为运行时的存储组织，第 10 章讨论

代码优化，第11章介绍代码生成。

作为教材，为了便于读者使用，在附录中按照56学时（理论授课44学时，课内实验12学时）给出了面向工程应用型学生的课程教学设计，也进一步体现作者撰写本书的目的。另外，对于科学型的学生，建议在授课中注意进一步突出对基本原理的研究；对于工程类的学生，可以给后4章多分配些学时。特别是在如果有更多学时的情况下，建议用于后4章的学习。当然，要想更好地理解和掌握本书内容，安排一个课程设计是很有意义的。

在过去的几十年中，专家们编著了很多与编译原理相关的教材，包括前面所说的“龙书”，都为本书的编写提供了很好的参考和借鉴。同时，“编译”作为理论基础好、形式化（自动化）水平高、对类计算体现好的技术和系统，与其相关的“形式语言与自动机”类教材也为本书的编写提供了参考，在这里向相关作者表示诚挚的谢意！

由于作者水平有限，书中的不当之处在所难免，敬请读者批评指正。如果读者有任何建议或意见，可以发送电子邮件至 jiangzl@ bjut. edu. cn 或 jsx@ hit. edu. cn。

作　者

2009年10月

目　录

第1章 引 论

程序设计语言的“进步”，给人们使用计算机提供了方便，随着它越来越接近人们的习惯，就会越来越远离机器。因此，在人和机器之间就需要一个系统来**自动地**完成人们用程序设计语言所编写的程序的翻译工作，以弥补这个越来越宽的鸿沟，这就是编译系统。

编译原理讨论编译系统的基本部分——编译程序的构造原理，同时讨论相关的技术和方法。对计算机科学与技术学科（简称计算机学科或计算学科）的学生来说，它是一门非常好的课程。Alfred V. Aho 曾在其编著的《编译原理、技术与工具》的开篇中写道，“编写编译器的原理和技术具有十分普遍的意义，以至在每个计算机科学家的研究生涯中，本书中的原理和技术都会被反复用到。”

计算机学科对信息描述和变换算法进行系统研究，主要包括相关的理论、分析、效率、实现和应用。编译原理涉及的是一个比较适当的抽象层面上的数据变换（既抽象，又实际），在这里除了给出一些问题的具体表示和变换算法外，还给出了自顶向下、自底向上、逐步求精、递归求解、目标驱动、问题的归纳与分析、抽象与形式化描述，以及相应算法的设计与实现、模块化等一系列系统构建的基本方法和问题求解思想。同时还清晰地给出了一个具有一定规模的系统的设计，包括非常重要的系统总体结构，这些对于培养学生的基本学科能力非常重要，所以它是计算机学科最为恰当、有效的知识载体之一。

我们希望通过本课程的教学，学生能够**掌握“编译原理”中的基本概念、基本理论、基本方法，在系统层面上再认识程序和算法，提升计算机问题的求解水平，增强系统能力，体验实现自动计算的乐趣。**

考虑到编译是关于程序设计语言的变换，所以本章首先简要介绍程序设计语言的发展，在此基础上讨论程序变换的基本概念、编译系统的构成，以及编译程序的生成技术，使读者能够从系统的角度对编译程序及编译过程有一个清晰、宏观的了解。

1.1 程序设计语言

众所周知，语言是用来进行信息交流的工具。要想让计算机按照人们的意愿去工作，就需要以适当的语言将这种意愿“告诉”计算机。就目前计算机系统的能力而言，需要告诉计算机的内容既包括要完成什么样的工作，又包括完成此工作的具体过程。相当于说，程序需要描述待完成的工作及其实现过程，所以人们称这种语言为计算机程序设计语言，简称为**程序设计语言**（programming language）。显然，这种用来刻画“意愿”的语言应该是人和计算机都能够理解的。为了

方便起见，当用某种语言表示（书写）程序的时候，称用此语言**描述**（describe）某一个程序，并称该程序是该语言程序。

自从计算机诞生以来，先后出现了许多程序设计语言。按照人们使用的方便程度划分，依次为机器语言、汇编语言、高级程序设计语言。随着它们越来越便于人们用来描述问题及其求解算法，它们离机器执行的“距离”就越来越远。下面分别对其进行简单介绍。

1. 机器语言

每一个具体的计算机系统都具有自己的指令系统，每条指令均用规定格式的0、1串来表示，可以被计算机直接理解并执行。这种可以被计算机直接理解的语言称为**机器语言**（machine language），它们是用0、1代码按照规定组成的指令序列的集合。在计算机出现的初期，由0、1表示的指令成为最早的程序设计语言。人们用这些由0、1组成的序列来表达机器可以执行的所有指令及其可以操作的数据。指令系统与机器的密切相关性决定了机器语言与机器紧密相关。

简单地说，机器语言就是以0、1代码表示的机器指令所构成的语言。用机器语言描述的程序称为**机器语言程序**（machine language program）。在机器语言程序中，每一条“语句”对应一条指令或者一个数据。

例1.1 一个简单的机器语言程序，其功能为：从内存中读入两个数值，并求出这两个数值的和。假设数据段的首地址为152B，被求和的两个数值位于数据段起始的两个字。

```
1011  1000  0010  1011  0001  0101(B82B15)
1000  1110  1101  1000(8ED8)
1010  0001  0000  0000  0000  0000(A10000)
1000  1011  0001  1110  0000  0010  0000  0000(8B1E0200)
1011  1001  0000  0000  0000  0000(B90000)
0000  0011  1100  1000(03C8)
0000  0011  1100  1011(03CB)
1000  1011  0000  1110  0000  0100  0000  0000(8B0E0400)
1011  1000  0000  0000  0100  1100(B8004C)
1100  1101  0010  0001(CD21)
```

□

由于机器语言程序是由机器指令组成的0、1代码，所以它的可读性非常差，给人们进行程序设计带来了诸多不便，通常只有计算机专业人员才使用，而且编写效率极其低下，特别容易出错。为了改变这种状况，计算机专家们考虑用适当的助记符来表示这些难记、难读、难理解的0、1代码，这就是汇编语言。

2. 汇编语言

汇编语言（assembly language）用一系列助记符来表示指令中的操作和操作数，同时用符号表示程序用到的一系列数据。例如，在汇编语言中分别用ADD、SUB、MUL、DIV表示加法、减法、乘法、除法运算，用AX、BX、AH、BH等表示计算机中的寄存器。用汇编语言描述的程序称为**汇编语言程序**（assembly language program）。

例 1.2 一个简单的汇编语言程序(与例 1.1 功能相同)。

```
ASSUME CS:CODE, DS:DATA
DATA SEGMENT
    DW 1234H,5678H
DATA ENDS
CODE SEGMENT
START:MOV AX, DATA
      MOV DS, AX
      MOV AX, DS:[0]
      MOV BX, DS:[2]
      MOV CX, 0
      ADD CX, AX
      ADD CX, BX
      MOV CX, DS:[4]
      MOV AX, 4C00H
      INT 21H
CODE ENDS
END START
```

□

显然用汇编语言表示的程序比起用 0、1 表示的机器语言程序更容易被人理解。但是由于计算机只能执行由 0、1 构成的程序,所以计算机是无法直接执行汇编语言程序的。因此,需要一种变换程序,它能够承担起把汇编语言程序转换成等价的机器语言程序的翻译工作。

实际上,在汇编语言中,只是用一些助记符来表示机器指令指定的内容,它必然严格地受到具体机型的限制。这不仅要求程序设计者了解机器,还需要按照机器的具体执行过程进行程序设计。为了提高程序设计效率,人们引入了"宏"的概念,用"宏"表示一些常用的基本处理过程,从而就有了**宏汇编**(macro assembly)。助记符的引入大大地方便了计算机的使用者。

虽然汇编语言比机器语言更便于人们进行程序设计,但是它仍然对程序设计人员有比较严格的要求。因为汇编语言与人们通常的描述习惯相比,还有较大的距离。在计算机主要用于进行科学计算的时代,对于使用计算机进行科学计算的非计算机专业的科学家和工程设计人员来说,有着极大的限制。因此,计算机专家们开始考虑寻找一种语言,这种语言可以像描述数学公式那样,允许直接在表示程序的语言中书写诸如 sum = A + B * 10.9、x = sin(y) * sin(y) - cos(z) * cos(z)之类的内容,以给用户提供更多的方便,于是就出现了高级程序设计语言,简称高级语言。

3. 高级语言

高级程序设计语言(high level programming language),简称**高级语言**(high level language)。这个概念是瑞典数学家 H. Rutishauser 在 1952 年首先提出来的。第一个实用的高级语言是 FORTRAN,由 IBM 公司在 1956 年研制。在高级语言的发展历史中,除了 FORTRAN 外,BASIC、

ALGOL、Pascal、COBOL、PROLOG、Smalltalk、C、Ada、C++、Java 等都是很重要的语言。

简单地讲,高级语言就是以人们容易理解的形式表达计算的要求和过程的语言。

高级语言通常不受具体机型的限制,而更接近人们的表达习惯,一条语句通常需要若干条机器指令去实现,这样就大大地提高了程序设计的效率。

用高级语言描述的程序称为**高级语言程序**(high level language program)。例 1.3 就是一个与例 1.1 中程序具有相同功能的 C 语言程序。

例 1.3 一个简单的高级语言程序(与例 1.1 功能相同)

```
int main
{
    int a,b,c;
    a = 1234h;
    b = 5678h;
    c = a + b;
    return 0;
}
```

□

高级语言的优点首先是接近人们的习惯,可读性好,便于理解和维护,因此比较容易保证程序的正确性。其次,其程序设计的效率远远高于汇编语言。再次,用这种语言编写的程序是和机器无关的,因此可以在不同的机器上运行。由此也确定了高级语言程序是不能被计算机直接执行的,要想执行这种程序,必须对其进行翻译,使之变成**等价**的机器语言程序。当然,这种翻译工作应该是由一个系统自动完成的。

按照不同的关注点,高级语言可以被分为不同的类。例如,命令式语言(imperative language)、申述式语言(declarative language)。前面提到的 FORTRAN、BASIC、ALGOL、Pascal、COBOL、C、Ada、Smalltalk、C++、Java 是命令式语言;PROLOG 是申述式语言。也有人将其分得更细一些,FORTRAN、BASIC、Pascal、C、COBOL、ALGOL 等是命令式语言,其中 BASIC、FORTRAN 为段结构的,Pascal、ALGOL 为嵌套结构的;LISP、ML 为函数式语言(functional language);PROLOG 为逻辑式语言(logical language)或称基于规则的语言;Smalltalk、Ada、C++、Java 等为面向对象语言(object - oriented language)。也有分成过程语言(procedural language)和非过程语言(nonprocedural language)的。

1.2 程序设计语言的翻译

在上一节中已经讲到,为了使用上的方便,人们设计出了汇编语言、高级语言。但是,用这些语言描述的程序是不能被计算机直接执行的,它们必须被翻译成机器可以执行的程序——机器语言程序才能被执行。这种翻译工作由一个翻译系统自动完成。