

计算机系列教材

# 编译原理与技术 (第2版)

李文生 编著

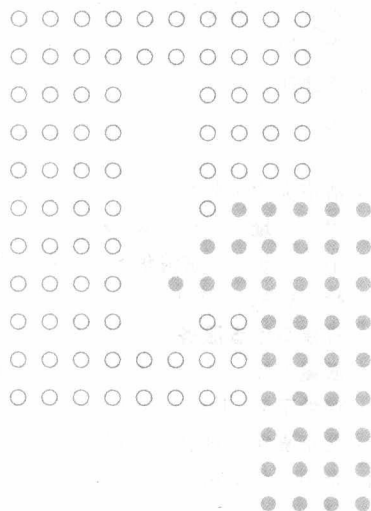
清华大学出版社



计算机系列教材

李文生 编著

# 编译原理与技术 (第2版)



清华大学出版社  
北京

## 内 容 简 介

本书系统地介绍了编译程序的设计原理和基本实现技术。主要内容包括词法分析、语法分析、语义分析、中间代码生成、代码生成和代码优化等,还重点介绍了用于实现语义分析和中间代码生成的语法指导翻译技术,以及程序运行时存储空间的组织与管理。

本书在介绍基本理论和方法的同时,也注重实际应用,介绍了 LEX 和 YACC 的使用方法 & 原理,剖析了 PL/0 语言的编译程序,介绍了 GCC 编译程序的基本结构。配合理论教学,给出了一些实践题目,旨在培养学生分析和解决问题的能力。

本书内容充实、图文并茂、各章节内容循序渐进,并注重理论与实践的结合。

本书可作为高等学校计算机科学与技术专业的本科生教材或参考书,也可供其他专业的学生或从事计算机工作的工程技术人员阅读参考。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

### 图书在版编目(CIP)数据

编译原理与技术/李文生编著. --2 版. --北京:清华大学出版社,2016

计算机系列教材

ISBN 978-7-302-44141-0

I. ①编… II. ①李… III. ①编译程序—程序设计—教材 IV. ①TP314

中国版本图书馆 CIP 数据核字(2016)第 139085 号

责任编辑:张瑞庆 柴文强

封面设计:常雪影

责任校对:李建庄

责任印制:沈 露

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, [c-service@tup.tsinghua.edu.cn](mailto:c-service@tup.tsinghua.edu.cn)

质量反馈:010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

课件下载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京嘉实印刷有限公司

经 销:全国新华书店

开 本:185mm×260mm

印 张:27

字 数:659 千字

版 次:2009 年 1 月第 1 版

2016 年 10 月第 2 版

印 次:2016 年 10 月第 1 次印刷

印 数:1~1000

定 价:49.00 元

产品编号:068058-01

## 《编译原理与技术(第2版)》前言

根据作者在教学实践中的心得和体会,以及在第1版教材使用过程中得到的积极反馈和建议,结合编译技术的发展和应用,本书对第1版教材的部分内容进行了修订。

本书仍以 Pascal 语言作为教学语言来讲解编译原理和实现技术,但其中的算法描述采用类 C 语言代码。作此选择的主要原因有两个:一是 Pascal 语言和 C 语言体现了结构化程序设计思想的代表性语言,但二者有明显的区别:Pascal 语言更注重抽象,注重简明严格,是程序设计理论专家的研究成果,最初也是为了教学和编写系统软件而设计的,并且这两个目标都已达到;C 语言更注重实用,注重表达式的简捷高效,在很多场合受到了软件开发人员的欢迎,逐渐成为软件开发的主流语言。二是 Pascal 语言规模适中,其全部语法规则采用形式化的语法描述形式 BNF 和语法图,作为计算机科学与技术专业相关课程的教学语言仍非常具有代表性,Pascal 语言编译程序所用原理和技术对 C 语言同样适用。

本书继承和发扬了第1版教材理论充实、注重实践的特色,并对原教材需要改善的内容进行了修订,主要体现在:

(1) 第4章,对 LR 分析程序进行了修改,主要涉及分析栈的组织和相应的分析动作,修改后的算法更便于理解和编码实现。

(2) 第5章,根据第4章的修改,对相应内容进行了修正,增加 5.5 节,介绍通用的语法制导翻译函数的构造方法。

(3) 第6章,内容组织做了调整,统一了示例语言文法,将声明部分的翻译方案和各语法成分的类型检查方案的内容组织在一节中,使得内容上下衔接更自然,便于理解。

(4) 第8章,增加了访问记录中域的翻译方案,重新组织了 8.5 节的内容,从实现的角度细化了 8.6 节的内容,根据第6章的文法,重写了 8.7 节过程调用语句的翻译。

(5) 第9章,修正了所用目标机器的指令格式,更便于熟悉 Intel 汇编语言的读者理解;给出了代码生成算法的伪码表示。

(6) 第10章,重新组织了 10.3 节的内容,并给出了基本块 DAG 构造算法的伪码表示。

(7) 第11章,这是新增加的一章,以 C++ 语言为例介绍面向对象的编译方法。主要

## 前言 《编译原理与技术（第2版）》

介绍了面向对象语言的核心概念类和对象，及其主要特征继承性、封装性和多态性，介绍了方法的编译和继承的编译方法，以及面向对象语言程序运行时环境。

限于学识水平和时间，书中难免存在不妥或错误之处，欢迎大家批评指正。读者对本书有任何意见或建议，请发送电子邮件至：[wenshli@bupt.edu.cn](mailto:wenshli@bupt.edu.cn)。

作者

2016年8月

## F O R E W O R D

本书是《编译原理与技术》的第2版。本书的第1版于2008年出版，自出版以来，受到了广大读者的欢迎和好评。在编写第2版的过程中，参考了国内外许多优秀的编译原理教材，并结合了近年来编译原理领域的最新研究成果。本书在保持原有框架的基础上，对部分内容进行了修订和补充。本书共分10章，主要内容包括：第1章 编译原理概论；第2章 词法分析；第3章 语法分析；第4章 语义分析；第5章 中间代码生成；第6章 寄存器分配；第7章 基本块优化；第8章 控制流图优化；第9章 数据流图优化；第10章 目标代码生成。本书可作为高等院校计算机专业及相关专业的教材，也可供从事编译原理工作的工程技术人员参考。

## 《编译原理与技术（第1版）》前言

“编译原理与技术”是计算机科学与技术专业的专业基础课程,通过本课程的学习,不仅可以提高编程技巧,掌握软件设计的一般方法,而且对计算机系统软件有一个比较清楚的认识和理解,为进一步的学习和研究打下良好的基础。

本书的前身是北京邮电大学出版社出版的《编译程序设计原理与技术》,主要介绍编译程序的设计原理和基本实现技术。根据多年的教学实践,对原书的内容进行了调整、补充和完善,并加强了实践环节。本书主要以 Pascal 和 C 语言为背景、就编译原理和技术有关的主要课题进行了系统和深入的讨论。

全书共分 11 章。第 1 章对编译程序的组成、功能及有关的前后处理器等进行了介绍,读者可以从了解编译程序的概况。第 2 章介绍了有关形式语言与自动机的基本概念,这是学习编译原理必备的基础理论知识。第 3 章引入了词汇、模式等概念,介绍了利用状态转换图手工编写词法分析程序的方法和步骤,并对词法分析程序自动生成工具 LEX 的使用和工作原理作了介绍。第 4 章详细讨论了常用的语法分析技术,如适于手工实现的递归调用预测分析方法、适合利用分析程序自动生成工具实现的完备的 LR 分析技术等,介绍了语法分析程序自动生成工具 YACC 的使用方法等。第 5 章讨论了语法制导翻译技术,介绍了语法制导定义和翻译方案的概念,以及根据语法制导定义和翻译方案设计相应的翻译程序的基本方法,后继的语义分析和中间代码生成就是基于这种技术实现的。第 6 章介绍了语义分析的基本概念和要求,讨论了编译程序所用的重要数据结构——符号表的组织和管理,详细介绍了借助符号表、利用语法制导翻译技术实现类型检查的方法。第 7 章讨论了程序运行时的存储组织与管理问题,介绍程序运行相关的问题及解决方案,有助于读者理解程序设计中的问题,如非局部名字访问、参数传递机制等。第 8 章介绍了中间语言,讨论了如何利用语法制导翻译技术把一般的程序设计语言结构翻译成中间代码。第 9 章介绍了目标代码生成的思想和一个简单的实现算法。第 10 章简单讨论了常用的代码优化技术。最后一章介绍了编译程序实现的一般方法,剖析了 PL/0 语言的编译程序,介绍了 GCC 编译程序的基本结构,并提供了一个课程设计题目,

## 前言 《编译原理与技术（第1版）》

按照软件工程的思想,对课程设计提出了基本要求,希望通过实际操作,有助于加深读者对编译原理的理解及对编译技术的掌握。

由于作者水平所限,书中难免存在缺点和不妥之处,真诚地希望得到广大读者和同行、专家的批评指正。

作者

2008年7月

F O R E W O R D

本书共分11章,第1章介绍编译原理的发展概况,第2章介绍编译原理的组成,第3章介绍编译原理的体系结构,第4章介绍编译原理的编译过程,第5章介绍编译原理的编译系统,第6章介绍编译原理的编译系统,第7章介绍编译原理的编译系统,第8章介绍编译原理的编译系统,第9章介绍编译原理的编译系统,第10章介绍编译原理的编译系统,第11章介绍编译原理的编译系统。

第1章 编译概述 /1

1.1 翻译和解释 /1

1.1.1 程序设计语言 /1

1.1.2 翻译程序 /2

1.2 编译的阶段和任务 /4

1.2.1 分析阶段 /4

1.2.2 综合阶段 /7

1.2.3 符号表管理 /10

1.2.4 错误处理 /10

1.3 和编译有关的其他概念 /11

1.3.1 编译的前端和后端 /11

1.3.2 “遍”的概念 /11

1.4 编译程序的伙伴工具 /13

1.4.1 预处理器 /14

1.4.2 汇编程序 /14

1.4.3 连接装配程序 /16

1.5 编译原理的应用 /16

习题1 /18

第2章 形式语言与自动机基础 /19

2.1 语言和文法 /19

2.1.1 字母表和符号串 /19

2.1.2 语言 /20

2.1.3 文法及其形式定义 /21

2.1.4 推导和短语 /23

2.1.5 分析树及二义性 /25

2.1.6 文法变换 /27

2.2 有限自动机 /31

2.2.1 确定的有限自动机 /32

2.2.2 非确定的有限自动机 /34



|       |                             |     |
|-------|-----------------------------|-----|
| 2.2.3 | 具有 $\epsilon$ -转移的非确定的有限自动机 | /36 |
| 2.2.4 | DFA 的化简                     | /40 |
| 2.3   | 正规文法与有限自动机的等价性              | /42 |
| 2.4   | 正规表达式与有限自动机的等价性             | /45 |
| 2.5   | 正规表达式与正规文法的等价性              | /48 |
| 2.5.1 | 正规定义式                       | /48 |
| 2.5.2 | 表示的缩写                       | /49 |
| 2.5.3 | 正规表达式转换为等价的正规文法             | /50 |
|       | 习题 2                        | /51 |

### 第 3 章 词法分析 /53

|       |                  |     |
|-------|------------------|-----|
| 3.1   | 词法分析程序与语法分析程序的关系 | /53 |
| 3.2   | 词法分析程序的输入与输出     | /54 |
| 3.2.1 | 输入缓冲区            | /54 |
| 3.2.2 | 词法分析程序的输出        | /56 |
| 3.3   | 记号的描述和识别         | /57 |
| 3.3.1 | 词法与正规文法          | /58 |
| 3.3.2 | 记号的文法            | /58 |
| 3.3.3 | 状态转换图与记号的识别      | /61 |
| 3.4   | 词法分析程序的设计与实现     | /62 |
| 3.4.1 | 文法及状态转换图         | /63 |
| 3.4.2 | 词法分析程序的构造        | /65 |
| 3.4.3 | 词法分析程序的实现        | /65 |
| 3.5   | LEX 简介           | /71 |
| 3.5.1 | LEX 源程序的结构       | /71 |
| 3.5.2 | LEX 源程序举例        | /74 |
|       | 习题 3             | /76 |
|       | 程序设计 1           | /77 |

### 第 4 章 语法分析 /78

|       |           |     |
|-------|-----------|-----|
| 4.1   | 语法分析简介    | /78 |
| 4.1.1 | 语法分析程序的地位 | /78 |

|       |                       |      |
|-------|-----------------------|------|
| 4.1.2 | 常用的语法分析方法             | /78  |
| 4.1.3 | 语法错误的处理               | /79  |
| 4.2   | 自顶向下分析方法              | /80  |
| 4.2.1 | 递归下降分析                | /81  |
| 4.2.2 | 递归调用预测分析              | /82  |
| 4.2.3 | 非递归预测分析               | /88  |
| 4.3   | 自底向上分析方法              | /95  |
| 4.3.1 | 规范归约                  | /97  |
| 4.3.2 | “移进-归约”方法的实现          | /98  |
| 4.4   | LR分析方法                | /100 |
| 4.4.1 | LR分析程序的模型及工作过程        | /100 |
| 4.4.2 | SLR(1)分析表的构造          | /104 |
| 4.4.3 | LR(1)分析表的构造           | /112 |
| 4.4.4 | LALR(1)分析表的构造         | /119 |
| 4.4.5 | LR分析方法对二义文法的应用        | /124 |
| 4.4.6 | LR分析的错误处理与恢复          | /129 |
| 4.5   | 软件工具 YACC             | /131 |
| 4.5.1 | YACC 源程序              | /132 |
| 4.5.2 | YACC 对二义文法的处理         | /134 |
| 4.5.3 | 用 LEX 建立 YACC 的词法分析程序 | /136 |
|       | 习题 4                  | /137 |
|       | 程序设计 2                | /141 |

## 第5章 语法制导翻译技术 /142

|       |                  |      |
|-------|------------------|------|
| 5.1   | 语法制导定义及翻译方案      | /143 |
| 5.1.1 | 语法制导定义           | /143 |
| 5.1.2 | 依赖图              | /146 |
| 5.1.3 | 计算次序             | /147 |
| 5.1.4 | S属性定义及L属性定义      | /148 |
| 5.1.5 | 翻译方案             | /149 |
| 5.2   | S属性定义的自底向上翻译     | /151 |
| 5.2.1 | 为表达式构造语法树的语法制导定义 | /151 |

|                        |                  |      |
|------------------------|------------------|------|
| 5.2.2                  | S 属性定义的自底向上翻译    | /155 |
| 5.3                    | L 属性定义的自顶向下翻译    | /158 |
| 5.3.1                  | 消除翻译方案中的左递归      | /158 |
| 5.3.2                  | 预测翻译程序的设计        | /162 |
| 5.4                    | L 属性定义的自底向上翻译    | /165 |
| 5.4.1                  | 移走翻译方案中嵌入的语义规则   | /166 |
| 5.4.2                  | 直接使用分析栈中的继承属性    | /166 |
| 5.4.3                  | 变换继承属性的计算规则      | /169 |
| 5.4.4                  | 改写语法制导定义为 S 属性定义 | /172 |
| 5.5                    | 通用的语法制导翻译方法      | /173 |
|                        | 习题 5             | /176 |
| <b>第 6 章 语义分析 /180</b> |                  |      |
| 6.1                    | 语义分析概述           | /180 |
| 6.1.1                  | 语义分析的任务          | /180 |
| 6.1.2                  | 语义分析程序的位置        | /181 |
| 6.1.3                  | 错误处理             | /181 |
| 6.2                    | 符号表              | /182 |
| 6.2.1                  | 符号表的建立和访问时机      | /182 |
| 6.2.2                  | 符号表内容            | /184 |
| 6.2.3                  | 符号表操作            | /187 |
| 6.2.4                  | 符号表组织            | /189 |
| 6.3                    | 类型检查             | /193 |
| 6.3.1                  | 类型表达式            | /194 |
| 6.3.2                  | 类型等价             | /197 |
| 6.4                    | 一个简单的类型检查程序      | /204 |
| 6.4.1                  | 语言说明             | /204 |
| 6.4.2                  | 符号表的建立           | /205 |
| 6.4.3                  | 表达式的类型检查         | /210 |
| 6.4.4                  | 语句的类型检查          | /213 |
| 6.4.5                  | 类型转换             | /214 |

|       |                     |             |
|-------|---------------------|-------------|
| 6.5   | 类型检查有关的其他主题         | /216        |
| 6.5.1 | 函数和运算符的重载           | /216        |
| 6.5.2 | 多态函数                | /217        |
|       | 习题 6                | /220        |
|       | 程序设计 3              | /223        |
|       | <b>第 7 章 运行环境</b>   | <b>/225</b> |
| 7.1   | 程序运行时的存储组织          | /225        |
| 7.1.1 | 程序运行空间的划分           | /226        |
| 7.1.2 | 活动记录与控制栈            | /227        |
| 7.1.3 | 名字的作用域及名字绑定         | /230        |
| 7.2   | 存储分配策略              | /231        |
| 7.2.1 | 静态存储分配              | /231        |
| 7.2.2 | 栈式存储分配              | /233        |
| 7.2.3 | 堆式存储分配              | /237        |
| 7.3   | 非局部名字的访问            | /239        |
| 7.3.1 | 程序块                 | /239        |
| 7.3.2 | 静态作用域规则下非局部名字的访问    | /241        |
| 7.3.3 | 动态作用域规则下非局部名字的访问    | /248        |
| 7.4   | 参数传递机制              | /250        |
| 7.4.1 | 传值调用                | /250        |
| 7.4.2 | 引用调用                | /252        |
| 7.4.3 | 复制恢复                | /253        |
| 7.4.4 | 传名调用                | /255        |
|       | 习题 7                | /255        |
|       | <b>第 8 章 中间代码生成</b> | <b>/259</b> |
| 8.1   | 中间代码形式              | /259        |
| 8.1.1 | 图形表示                | /259        |
| 8.1.2 | 三地址代码               | /260        |

|       |                 |      |
|-------|-----------------|------|
| 8.2   | 赋值语句的翻译         | /265 |
| 8.2.1 | 仅涉及简单变量的赋值语句的翻译 | /265 |
| 8.2.2 | 涉及数组元素的赋值语句     | /268 |
| 8.2.3 | 记录结构中域的访问       | /273 |
| 8.3   | 布尔表达式的翻译        | /274 |
| 8.3.1 | 翻译布尔表达式的方法      | /274 |
| 8.3.2 | 数值表示法           | /275 |
| 8.3.3 | 控制流表示法及回填技术     | /276 |
| 8.4   | 控制语句的翻译         | /282 |
| 8.5   | goto 语句的翻译      | /287 |
| 8.6   | CASE 语句的翻译      | /289 |
| 8.7   | 过程调用语句的翻译       | /292 |
|       | 习题 8            | /294 |

## 第 9 章 目标代码生成 /297

|       |               |      |
|-------|---------------|------|
| 9.1   | 目标代码生成概述      | /297 |
| 9.1.1 | 代码生成程序的位置     | /297 |
| 9.1.2 | 代码生成程序设计的相关问题 | /298 |
| 9.2   | 基本块和流图        | /300 |
| 9.3   | 下次引用信息        | /302 |
| 9.4   | 一个简单的代码生成程序   | /305 |
| 9.4.1 | 目标机器描述        | /305 |
| 9.4.2 | 代码生成算法        | /307 |
| 9.4.3 | 其他常用语句的代码生成   | /312 |
|       | 习题 9          | /315 |

## 第 10 章 代码优化 /317

|        |              |      |
|--------|--------------|------|
| 10.1   | 代码优化概述       | /317 |
| 10.1.1 | 代码优化程序的功能和位置 | /317 |
| 10.1.2 | 代码优化的主要种类    | /317 |
| 10.2   | 基本块优化        | /318 |

|                              |                |      |
|------------------------------|----------------|------|
| 10.2.1                       | 常数合并及常数传播      | /318 |
| 10.2.2                       | 删除公共表达式        | /320 |
| 10.2.3                       | 复制传播           | /321 |
| 10.2.4                       | 削弱计算强度         | /321 |
| 10.2.5                       | 改变计算次序         | /321 |
| 10.3                         | dag 在基本块优化中的应用 | /322 |
| 10.3.1                       | 基本块的 dag 表示    | /322 |
| 10.3.2                       | 基本块的 dag 构造算法  | /323 |
| 10.3.3                       | dag 的应用        | /327 |
| 10.3.4                       | dag 构造算法的进一步讨论 | /330 |
| 10.4                         | 循环优化           | /333 |
| 10.4.1                       | 循环展开           | /333 |
| 10.4.2                       | 代码外提           | /334 |
| 10.4.3                       | 削弱计算强度         | /334 |
| 10.4.4                       | 删除归纳变量         | /335 |
| 10.5                         | 窥孔优化           | /337 |
| 10.5.1                       | 删除冗余的传送指令      | /337 |
| 10.5.2                       | 删除死代码          | /337 |
| 10.5.3                       | 控制流优化          | /338 |
| 10.5.4                       | 削弱计算强度及代数化简    | /338 |
|                              | 习题 10          | /339 |
| <b>第 11 章 面向对象的编译方法</b> /341 |                |      |
| 11.1                         | 面向对象语言的基本概念    | /341 |
| 11.1.1                       | 类和对象           | /341 |
| 11.1.2                       | 继承             | /343 |
| 11.1.3                       | 信息封装           | /346 |
| 11.1.4                       | 多态性            | /347 |
| 11.2                         | 方法的编译          | /350 |
| 11.2.1                       | 静态方法           | /350 |
| 11.2.2                       | 动态方法           | /351 |
| 11.3                         | 继承的编译          | /354 |

|                             |                         |      |
|-----------------------------|-------------------------|------|
| 11.3.1                      | 单一继承的编译                 | /354 |
| 11.3.2                      | 多继承的编译                  | /355 |
| 11.4                        | 程序运行环境                  | /358 |
|                             | 习题 11                   | /359 |
| <b>第 12 章 编译程序构造实践 /360</b> |                         |      |
| 12.1                        | 编译程序的表示及实现方法            | /360 |
| 12.1.1                      | 表示方法                    | /360 |
| 12.1.2                      | 实现语言                    | /360 |
| 12.1.3                      | 自展法                     | /361 |
| 12.1.4                      | 移植法                     | /362 |
| 12.2                        | PL/0 语言及其编译程序介绍         | /364 |
| 12.2.1                      | PL/0 语言                 | /365 |
| 12.2.2                      | PL/0 编译程序的结构            | /368 |
| 12.2.3                      | PL/0 编译程序的词法分析          | /369 |
| 12.2.4                      | PL/0 编译程序的语法分析          | /371 |
| 12.2.5                      | PL/0 编译程序的出错处理          | /373 |
| 12.2.6                      | PL/0 编译程序的执行环境及<br>代码生成 | /375 |
| 12.2.7                      | PL/0 程序编译和运行示例          | /379 |
| 12.3                        | GCC 编译程序                | /381 |
| 12.3.1                      | GCC 简介                  | /382 |
| 12.3.2                      | GCC 编译程序的结构与处理<br>流程    | /383 |
| 12.3.3                      | GCC 的分析程序               | /384 |
| 12.3.4                      | GCC 的中间语言及中间代码<br>生成    | /385 |
| 12.3.5                      | GCC 的代码优化               | /389 |
| 12.3.6                      | GCC 的代码生成               | /391 |
| 12.4                        | 编译实践                    | /392 |

- 12.4.1 Pascal-S 语言说明 /392
- 12.4.2 课程设计要求及说明 /398
- 12.4.3 编译程序的测试 /400

附录 PL/0 编译程序源程序 /402

参考文献 /416



# 第1章 编译概述

编译程序是现代计算机系统中重要的系统软件之一，是高级程序设计语言的支撑软件。众所周知，用高级程序设计语言(比如 C/C++) 书写的源程序是不能直接在计算机上运行的(起码现有的计算机不支持)，要想运行它并得到预期的结果，首先必须把源程序转换成与之等价的目标程序，这个过程就是所谓的编译。编译原理与技术是计算机科学技术中的一个重要分支，现在已经基本形成了一套比较系统的、完整的理论和方法。编译原理与技术是计算机工作者所必须具备的专业基础知识。

编译程序的设计涉及程序设计语言、形式语言与自动机理论、计算机体系结构、数据结构、算法分析与设计、操作系统，以及软件工程等各个方面。本章通过描述编译程序的组成及编译程序的工作环境来介绍编译程序相关的基本概念，以便大家了解本课程的主要内容。

## 1.1 翻译和解释

### 1.1.1 程序设计语言

正如语言是人们进行交流的媒介和手段一样，在计算机应用领域，程序设计语言充当了人与问题，以及协助解决问题的计算机之间的通信工具。一种高效的程序设计语言应该能够提高计算机程序的开发效率、具有较好的问题表达能力，并在人们通常的思维方式与计算机执行所要求的精确性之间架起桥梁。程序设计语言同时也是开发人员之间的交流工具，在许多大型软件项目中，项目成功的关键在于程序员是否能读懂别人写的程序代码。

在计算机发展初期，程序员直接用机器语言编写程序。机器语言程序很不直观，难写、难读、难修改，并且对机器硬件的依赖性很强、移植性差。程序设计人员必须受过一定的训练并且熟悉计算机硬件，这在很大程度上限制了计算机的推广应用。

之后，出现了符号语言，即用比较直观的符号来代替纯粹数字表示的机器指令代码，这样使程序便于记忆、阅读和检查，在此基础上又进一步发展为汇编语言。在汇编语言中，除了用直观的助记符代替操作指令以对应一条条的机器指令外，还增加了若干宏指令，每条宏指令对应一组机器指令、完成特定的功能。这些宏指令构成指令码的扩展。汇编语言仍然是依赖于机器的，使用起来还是很不方便，并且程序开发效率也很低。

为进一步解决这些问题，John Backus 等人参照数学语言设计了第一个描述算法的语言并于 1954 年正式对外发布，这就是 FORTRAN I，1957 年第一个 FORTRAN 编译器在 IBM704 计算机上实现，并首次成功运行了 FORTRAN 程序。之后，又相继出现了得到广泛应用的过程性语言，如图灵奖获得者 Peter Naur 在 1960 年主编的《算法语言