21世纪高等学校计算机学科系列教材

编译原理简明教程

崔冬华　冯秀芳　范 辉　编著

全国高等学校计算机教育研究会
课程与教材建设委员会推荐出版

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

21 世纪高等学校计算机学科系列教材

编译原理简明教程

崔冬华　冯秀芳　范　辉　编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书全面地介绍了编译程序的基本结构，系统地阐述了编译原理的一般理论和常用的有效方法与技术。

全书共分 12 章，包括：形式语言与自动机理论、词法分析、语法分析、语义分析及中间代码的生成、代码优化、目标代码生成及错误校正等。在内容的组织上，本书将编译的基本理论和具体的实现技术有机地结合起来，既准确清楚地阐述了相关的概念和原理，又给出了典型的实现程序流程图。在分析方法中介绍了 LL（K）方法、递归下降分析法、算符优先分析法和 LR（K）方法等。

本书理论和实践并重，叙述严谨、简明，富有启发性，内容深入浅出，便于自学。各章之后附有习题，有关部分配有上机练习题。

本书可作为大学计算机专业本科生的教材，也可作为教师、研究生或计算机科技人员的参考书籍。

图书在版编目(CIP)数据

编译原理简明教程/崔冬华等编著. —北京：电子工业出版社，2002. 12
21 世纪高等学校计算机学科系列教材
ISBN 7-5053-7627-6

Ⅰ. 编… Ⅱ. 崔… Ⅲ. 编译程序—程序设计—高等学校—教材 Ⅳ. TP314

中国版本图书馆 CIP 数据核字(2002)第 057637 号

责任编辑：张云怡
印　　刷：北京四季青印刷厂
出版发行：电子工业出版社　http://www.phei.com.cn
　　北京市海淀区万寿路 173 信箱　邮编 100036
经　　销：各地新华书店
开　　本：787×1092　1/16　印张：16.75　字数：428 千字
版　　次：2002 年 8 月第 1 版　2002 年 8 月第 1 次印刷
印　　数：5 000　册　定价：21.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。
联系电话：(010)68279077

序　言

这套教材是面向21世纪计算机学科系列教材。为什么要组织这套教材？根据什么编写这套教材？这些都是在这篇序言中要回答的问题。

计算机学科是一个飞速发展的学科，尤其是近十年来，计算机向高度集成化、网络化和多媒体化发展的速度一日千里。但是，从另一个方面来看，目前高等学校的计算机教育，特别是教材建设，远远落后于现实的需要。现在的教材主要是根据《教学计划1993》的要求组织编写的。这个教学计划，在制定过程中主要参照了美国IEEE和ACM的《教学计划1991》。

10年来，计算机学科已有了长足发展，这就要求高等学校计算机教育必须跟上形势发展的需要，在课程设置和教材建设上做出相应调整，以适应面向21世纪计算机教育的要求。这是组织这套教材的初衷。

为了组织好这套教材，全国高等学校计算机教育研究会课程与教材建设委员会在天津召开了“全国高等学校计算机学科课程与教材建设研讨会”，在北京召开了“教材编写大纲研讨会”。在这两次会议上，代表们深入地研讨了全国高校计算机专业教学指导委员会和中国计算机学会教育委员会制定的《计算机学科教学计划2000》以及美国IEEE和ACM的《计算机学科教学计划2001》，这是这套教材参照的主要依据。

IEEE和ACM的《计算机学科教学计划2001》是在总结了从《计算机学科教学计划1991》到现在，计算机学科十年来发展的主要成果的基础上诞生的。它认为面向21世纪计算机学科应包括14个主科目，其中12个主科目为核心主科，它们是：算法与分析(AL)、体系结构(AR)、离散结构(DS)、计算科学(CN)、图形学、可视化、多媒体(GR)、网络计算(NC)、人机交互(HC)、信息管理(IM)、智能系统(IS)、操作系统(OS)、程序设计基础(PF)、程序设计语言(PL)、软件工程(SE)、社会、道德、法律和专业问题(SP)。其中除CN和GR为非核心主科目外，其他12项均为核心主科目。

将2001教学计划与1991教学计划比较可看出：

(1) 在1991年计划中，离散结构只作为数学基础提出，而在2001计划中，则作为核心主科目提出，显然，提高了它在计算机学科中的地位。

(2) 在1991计划中，未提及网络计算，而在2001计划中，则作为核心主科目提出，以适应网络技术飞速发展的需求。

(3) 图形学、可视化与多媒体也是为适应发展要求新增加的内容。

除此之外，2001计划在下述5个方面做调整：

将程序设计语言引论调整为程序设计基础，将人－机通信调整为人机交互，将人工智能与机器人学调整为智能系统，将数据库与信息检索调整为信息管理，将数值与符号计算调整为计算科学。

显然，这些变化使2001计划更具有科学性，也更好地适应了学科发展的需要。

在组织这套教材的过程中，充分考虑了这些变化和调整，在软件和硬件的课程体系、界面划分方面均做了相应的调整，使整套教材更具有科学性和实用性。

另外，还要说明一点，教材建设既要满足必修课的要求，又要满足限选课和任选课的要求。

因此，教材应按系列组织，反映整个计算机学科的要求，采用大拼盘结构，以适应各校不同的具体教学计划，使学校可根据自己的需求进行选择。

这套教材包括：《微机应用基础》、《离散数学》、《电路与电子技术》、《电路与电子技术习题与实验指南》、《数字逻辑与数字系统》、《计算机组成原理》、《微机接口技术》、《计算机体系结构》、《计算机网络》、《计算机网络实验教程》、《通信原理》、《计算机网络管理》、《网络信息系统集成》、《多媒体技术》、《计算机图形学》、《计算机维护技术》、《数据结构》、《计算机算法设计与分析》、《计算机数值分析》、《汇编语言程序设计》、《Pascal 语言程序设计》、《VB 程序设计》、《C 语言程序设计》、《C + + 语言程序设计》、《Java 语言程序设计》、《操作系统原理》、《UNIX 操作系统原理与应用》、《Linux 操作系统》、《软件工程》、《数据库系统原理》、《编译原理》、《编译方法》、《人工智能》、《计算机信息安全》、《计算机图像处理》、《人机交互》、《计算机伦理学》。对于 IEEE 和 ACM 的《计算机学科教学计划 2001》中提出的 14 个主科目，这套系列教材均涵盖，能够满足不同层次院校、不同教学计划的要求。

这套系列教材由全国高等学校计算机教育研究会课程与教材建设委员会主任李大友教授精心策划和组织。编者均为具有丰富教学实践经验的专家和教授。所编教材体系结构严谨、层次清晰、概念准确、论理充分、理论联系实际、深入浅出、通俗易懂。

教材组织过程中，得到了哈尔滨工业大学蒋宗礼教授，西安交通大学董渭清副教授，武汉大学张焕国教授，吉林大学张长海教授，福州大学王晓东教授，太原理工大学余雪丽教授等的大力支持和帮助，在此一并表示衷心感谢。

李大友

2000 年 6 月

前　言

本教材由全国高等学校计算机教育研究会课程与教材建设委员会选定并推荐出版。

“编译原理”是计算机科学和技术专业一门重要的专业课程，在计算机专业的学科课程中占有非常重要的位置，它是每个优秀的计算机专业人员必修的一门课程。设置本课程的目的在于系统地向学生讲述编译程序设计的基本理论、编译系统的结构及编译程序各部分的设计原理和实现技术。通过对这些方面知识的学习，能使学生既掌握编译理论和方法方面的基本知识，又具有设计、实现、分析维护编译程序等方面的初步能力。

全书共分 12 章，第 1 章对编译过程、编译程序逻辑结构及编译程序各组成部分的主要功能进行了概括说明；第 2、3 章介绍了文法和形式语言、自动机理论，它为学习后续各章奠定了理论基础；第 4 章讨论了词法分析程序的设计原理；第 5、6 章讲述了语法分析程序的设计技术，其中主要介绍 LL 分析法、递归下降分析法、算符优先分析法及 LR 分析法；第 7 ~ 12 章分别讨论了语义分析及中间代码的生成、符号表、目标程序运行时的存储组织与分配、代码优化、目标代码生成和源程序的错误校正等。

本书还包含 3 个附录：附录 A 列出了 PL/0 编译程序文本，附录 B 和附录 C 是对编译程序构造工具 LEX 和 YACC 的介绍。

“编译原理”是一门理论性和实践性都比较强的课程，在本书的编写过程中，作者力图将其中的基本概念、基本编译技术和实现方法的思路阐述清楚，由浅入深，循序渐进，并对各种编译方法和技术适当配有相应的处理步骤或流程图。此外，各章都配有一定数量的习题，有关部分还配有上机实习题，学生通过完成这些练习可进一步加深对课堂教学内容的理解。

“编译原理”课程蕴涵着计算机学科中解决问题的思路和抽象问题的解决方法。在这门课程中所接受的训练很难在其他地方获得。可以这样说，像“高等数学”课程影响每一个理工科学生一生的工作和学习一样，学好“编译原理”课程会让计算机专业学生“享用一辈子”。

本书第 1、2、3、12 章及附录 B 由范辉编写，第 4、5、7、11 及附录 A 由崔冬华编写，第 6、8、9、10 及附录 C 由冯秀芳编写。在本书的编写过程中，我们得到北京工业大学李大友教授、太原理工大学余雪丽教授、段富教授和烟台大学陈守孔教授的大力帮助和支持；研究生李晋江、郑丽伟和杨红等同学为本书的图表设计制作、录入及校对做了大量的工作，在此谨向他们表示诚挚的感谢。

由于编者水平有限，书中难免还存在一些缺点和错误，恳请广大读者批评指正。

编　者

2002 年 2 月

目　录

第1章 引 言

电子计算机的诞生是科学发展史上的一个里程碑，它以其处理数据容量大、速度快、精度高而且具有判别功能等显著特点，作为一种工具被广泛应用于各个领域。计算机之所以能如此广泛地被应用，应当归功于高级程序设计语言，而编译程序的存在，使高级语言具有永恒的生命力。高级程序设计语言的引进，使人们能用接近于数学用语的表示法去表达算法，让计算机做人们想做的事，从而为计算机的推广应用打开了局面。没有高级程序设计语言，计算机要推广应用是不可思议的。高级语言只有经过编译程序的编译，才能生成机器能够识别的语言。编译程序可看做是程序设计语言的支持环境，在编译程序的支持下，高级语言程序才能运行。

1.1 编译程序、汇编程序、解释程序

1.1.1 什么是编译程序

编译程序是将高级语言写的源程序翻译成目标语言的程序。人类要用计算机解决问题，首先要告诉计算机解决什么问题，或许还要告诉计算机如何解决这个问题。这就牵涉到用什么样的语言来描述想要解决的问题。因而，计算机科学家设计一些比较习惯的语言来描述要解决的问题。这种语言，表达力强，易于使用，易于为人理解和接受，称为高级程序语言。相反，能被计算机直接理解与执行的语言，即机器指令，却不易被人们理解和接受，因而被称为低级机器语言。不管用什么语言来表达对问题的描述，它们都被称为程序。例如，用FORTRAN 语言、ALGOL 语言、Pascal 语言、C 语言、ADA 语言、C++语言等编写的程序，都称为高级语言程序。这种程序不能直接被计算机理解与执行，必须经过等价的转换，变成机器能理解与执行的机器语言才能执行。进行这种等价转换的工作，就是编译程序的任务。一般来说，编译程序就是这样一种程序，它将用一种语言写的程序，等价地转换为另一种语言写的程序。因此，也叫翻译程序。前一个程序，即被翻译的程序，叫做源程序；后一个程序，即翻译的程序，叫做目的程序或目标程序。

因此，翻译程序与编译程序这两个名词并无大的区别。但是，通常把从高级语言编写的源程序到机器语言表示的目标程序的转换程序称为编译程序。而把从一种语言编写的源程序到另一种语言写的目标程序的转换程序称为翻译程序。例如，从 ADA 语言到 C 语言的转换，从 ADA 语言到 Pascal 语言的转换，从 C 语言到 ADA 语言的转换都是翻译程序。

1.1.2 什么是汇编程序

汇编程序是把由汇编语言编写的源程序翻译成机器语言的目标程序。汇编语言十分接近机器语言。事实上，很多汇编语言的语句恰好就是机器语言语句的符号表示。况且，汇编

语言的语句通常具有固定格式，这种格式将使汇编程序更易于分析这些语句，在这些汇编语句中通常不包含嵌套语句、分程序等。

1.1.3 什么是解释程序

解释程序不同于编译程序，它不是直接将高级语言的源程序翻译成目标程序后再执行，而是一个语句一个语句读入源程序，即边解释边执行。其工作过程如图 1.1 所示。

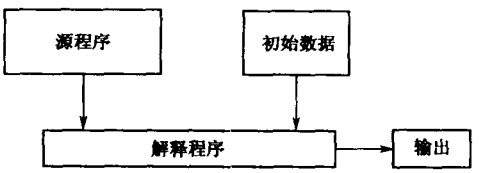

图 1.1 一个概念化的解释程序

解释程序把源程序看成是自己的输入，源程序原来的输入也是解释程序的一部分输入，因而可以对源程序进行处理，就像对另一部分数据一样。程序执行时的控制点在解释程序之中，而不在用户程序中，即用户程序是消极的，这就不同于编译程序产生的可执行的用户程序，在那里，它是积极的。

解释程序允许：

① 在执行用户程序时修改用户程序。因此，它提供一种直接的交互调试能力。这种修改对于像 APL、BASIC 这样的非分程序结构的语言是非常容易的，因为修改个别语句并不需要重新分析整个程序。

② 对象的类型可动态地修改。随着程序的执行，符号的意义可以变化，例如，在某一点，它可以是整型变量，而在另一点，它可以是一个字符数组。这种符号意义的动态确定叫做流动绑定（Fluid Binding），对于编译程序是很头痛的事，它使得编译程序很难对其进行翻译。

③ 提供良好的诊断信息。解释程序执行时，把程序的执行与源程序行文的分析交织在一起，因而可以在诊断信息中，给出出错点的源程序行号，变量的符号名，对变量交互赋值，这些工作对于编译程序是比较困难的。

④ 解释程序不依赖于目标机，因为它不生成目标代码。因此，其可移植性优于编译程序。

解释程序的缺点主要是开销大，速度慢。

开销大主要体现在两个方面。一是在执行时要连续多次重新对程序行文进行分析考察，包括标识符的绑定、类型及操作的确定。这种考察所费的开销是巨大的，对于极为动态的语言，如 APL，同编译程序相比，速度上可能是 1:100，甚至还要差。对于较静态的语言，如 BASIC，速度之比约为 1:10。二是空间上的开销，解释程序要保存大量支撑子程序，源程序不能按紧凑方式存放，符号表及程序行文的存放格式都要易于使用与修改，而不考虑空间上的节省。因此，一般解释性程序对于程序规模，变量个数，过程个数等都有一定的限制，超过这些内部限制的程序往往不能由解释程序处理。

有一些语言，如 BASIC，既有解释程序，又有编译程序。前者用于程序开发与调试，后者用于生产运行。事实上，编译与解释并非总有明显的分界线。例如，许多 BASIC 解释

程序把源程序转换成一种内部表示。在这种表示中，像“LET”及“GOTO”这样的关键字都表示为一个字节的操作码，标识符则用其在符号表中的入口号表示。这就是说，解释程序包含有某种形式的“翻译”。另一方面，编译程序也可包含有某种层次上的“解释”。

1.2 编译过程概述

编译程序的工作，从输入源程序开始到输出目标程序为止的整个过程，是非常复杂的。但就其过程而言，它与人们进行自然语言之间的翻译有许多相近之处。当我们把一种文字翻译为另一种文字，例如把一段英文翻译为中文时，通常需经下列步骤：

① 识别出句子中的一个个单词；

② 分析句子的语法结构；

③ 根据句子的含义进行初步翻译；

④ 对译文进行修饰；

⑤ 写出最后的译文。

类似地，编译程序的工作过程一般也可以划分为五个阶段：词法分析、语法分析、语义分析与中间代码的产生、优化、目标代码的生成。

第一阶段，词法分析。词法分析的任务是：输入源程序，对构成源程序的字符串进行扫描和分解，识别出一个个的单词（亦称单词符号或简称符号），如保留字（begin、end、if、for、while 等）、标识符、常数、特殊符号（标点符号、左右括号等等）。例如，对于 Pascal 的循环语句：

```
for i:=1 to 100 do
```

词法分析的结果是识别出如下的单词符号：

保留字　for
标识符　i
赋值号　:=
整常数　1
保留字　to
整常数　100
保留字　do

这些单词是组成上述 Pascal 语句的基本符号。单词符号是语言的基本组成成分，是人们理解和编写程序的基本要素。识别和理解这些要素无疑也是翻译的基础。如同将英文翻译成中文的情形一样，如果你对英语单词不理解，那就谈不上进行正确的翻译。在词法分析阶段的工作中所遵循的是语言的词法规则（或称构词规则）。描述词法规则的有效工具是正规式和有限自动机。

第二阶段，语法分析。语法分析的任务是：在词法分析的基础上，根据语言的语法规则，把单词符号串分解成各类语法单位（语法范畴），如“短语”、“句子”（“语句”）、“程序段”和“程序”等。通过语法分析，确定整个输入串是否构成语法上正确的“程序”。语法分析所依循的是语言的语法规则。语法规则通常用上下文无关文法描述。词法分析是一种线性分析，而语法分析是一种层次结构分析。例如，在很多语言中，符号串

```
Z:=X+2*Y
```

代表一个“赋值语句”，而其中的“X+2*Y”代表一个“算术表达式”。因而，语法分析的任务就是识别“X+2*Y”为算术表达式。同时，识别上述整个符号串属于赋值语句语法范畴。

第三阶段，语义分析与中间代码的产生。这一阶段的任务是：对语法分析所识别出的各类语法范畴，分析其含义，并进行初步翻译（产生中间代码）。这一阶段通常包括两个方面的工作。首先，对每种语法范畴进行静态语义检查，例如，变量是否定义、类型是否正确等。如果语义正确，则进行另一方面工作，即进行中间代码的翻译。这一阶段所依循的是语言的语义规则。通常使用属性文法描述语义规则。

“翻译”仅仅在这里才开始涉及到。所谓“中间代码”是一种含义明确、便于处理的记号系统，它通常独立于具体的硬件。这种记号系统或者与现代计算机的指令形式有某种程度的接近，或者能够比较容易地把它变换成现代计算机的机器指令。例如，许多编译程序采用了一种与“三地址指令”非常近似的“四元式”作为中间代码。这种四元式的形式如表 1.1 所示。

表 1.1　四元式形式

运算符	第一运算分量	第二运算分量	结果

它的意义是：对第一和第二运算分量按运算符进行某种运算，把运算所得的值作为“结果”保留下来。在采用“四元式”作为中间代码的情况下，中间代码产生的任务就是按语言的语义规则把各类语法范畴翻译成四元式序列。例如，下面的赋值语句

```
Z:=（X+3）*Y/W
```

可被翻译为如表 1.2 所示的四元式序列。

表 1.2　赋值语句的四元式序列

序号	运算符	第一运算分量	第二运算分量	结果
(1)	+	X	3	T_1
(2)	*	T_1	Y	T_2
(3)	/	T_2	W	Z

其中，T_1 和 T_2 是编译期间引进的临时工作变量；第一个四元式意味着把 X 的值加上 3 存放于 T_1 中；第二个四元式指将 T_1 的值和 Y 的值相乘存于 T_2 中；第三个四元式指将 T_2 的值除以 W 的值结果存于 Z 中。

一般情况下，中间代码是一种独立于具体硬件的记号系统。常用的中间代码，除了四元式之外，还有三元式、间接三元式、逆波兰式和抽象语法树等。

第四阶段，优化。优化的任务在于对前段产生的中间代码进行加工变换，以期在最后阶段能产生出更为高效（节省时间和空间）的目标代码。优化的主要方面有：公共子表达式的提取、循环优化、删除无用代码等。有时，为了便于“并行运算”，还可以对代码进行并行优化处理。优化所依循的原则是程序的等价变换规则。

例如，对于下列循环语句：

```
for K:=1 to 100 do
    begin
```

```
        M:=I+10*K;
        N:=J+2*K
    end
```

的中间代码如表 1.3 所示。

表 1.3　循环语句的四元式

序号	运算符	第一运算分量	第二运算分量	结 果
(1)	:=	I	-	K
(2)	<=	K	100	T_1
(3)	BF	10	T_1	-
(4)	*	10	K	T_2
(5)	+	I	T_2	M
(6)	*	2	K	T_3
(7)	+	J	T_3	N
(8)	+	K	1	K
(9)	BR	2	-	-
(10)				

循环语句优化以后的四元式如表 1.4 所示。

表 1.4　优化后循环语句的四元式

序号	运算符	第一运算分量	第二运算分量	结 果
(1)	:=	I	-	M
(2)	:=	J	-	N
(3)	:=	1	-	K
(4)	>	K	100	T_1
(5)	BT	10	T_1	-
(6)	+	M	10	M
(7)	+	N	2	N
(8)	+	K	1	K
(9)	BR	4	-	-
(10)				

那么，最终所得的目标程序的执行效率就肯定会提高很多。因为，对于前者，在循环中需做 300 次加法和 200 次乘法；对于后者，在循环中只需做两个 300 次加法。 尤其是在多数硬件中，计算乘法的时间比计算加法的时间要长得多。

第五阶段，目标代码的生成。这一阶段的任务是：把中间代码（或经优化处理后的代码）变换成特定机器上的低级语言代码。这阶段实现了最后的翻译，它的工作有赖于硬件系统结构和机器指令含义。这阶段工作非常复杂，涉及到硬件系统功能部件的运用，机器指令的选择，各种数据类型变量的存储空间分配，以及寄存器和后援寄存器的调度等。如何产生出足以充分发挥硬件效率的目标代码是一件非常不容易的事情。

目标代码的形式可以是绝对指令代码、可重定位的指令代码或汇编指令代码。如目标代码是绝对指令代码，则这种目标代码可立即执行。如果目标代码是汇编指令代码，则需汇编器汇编之后才能运行。必须指出，现代多数实用编译程序所产生的目标代码都是一种可重

定位的指令代码。这种目标代码在运行前必须借助于一个连接装配程序把各个目标模块（包括系统提供的库模块）连接在一起，确定程序变量在主存中的位置，装入内存中指定的起始地址，使之成为一个可以运行的绝对指令代码程序。

上述编译过程的五个阶段是一种典型的分法。事实上，并非所有编译程序都分为这五个阶段。有些编译程序对优化没有什么要求，优化阶段就可省去。在某些情况下，为了加快编译速度，中间代码产生阶段也可以去掉。有些最简单的编译程序是在语法分析的同时产生目标代码。但是，多数实用编译程序的工作过程大致都像上面所说的那五个阶段。有时编译过程还可以分为六个阶段，即把语义分析和中间代码产生分为两个阶段。

1.3 编译程序的结构框图

典型的编译程序如图 1.2 所示。

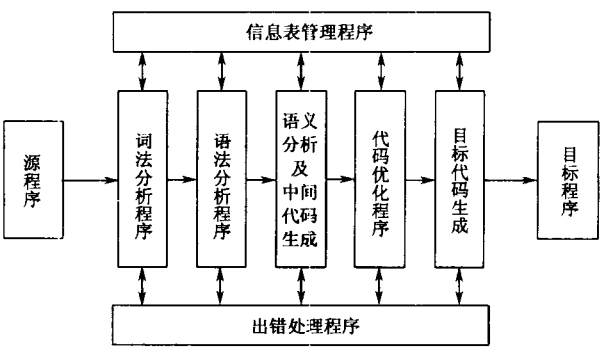

图 1.2 典型的编译程序结构框图

不同的编译程序结构不同，图中给出的只是编译程序各部分的逻辑结构图，它并不代表时间上的执行顺序。有些编译程序可能恰好按图中的顺序执行这些逻辑过程，另一些编译程序可能按平行、互锁方式执行这些逻辑过程。

采用哪种执行方式，视语种、机型等因素的不同而不同。

1.4 编译程序的开发

编译程序是一个非常复杂的软件系统，虽然编译理论和编译技术不断发展，已使编译程序的生产周期不断缩短。但是目前要研制一个编译程序仍需要相当长的时间，而且工作相当艰巨，因此如何高效地生成一个高质量的编译程序一直是人们追求的目标。本节将简单介绍编译程序的开发步骤、开发技术和程序的自动生成。

1.4.1 编译程序的开发步骤

从软件工程的理论和方法来分析，编译程序的开发过程大致分为以下几个阶段。

（1）认真做好需求分析并合理分工。编译程序是要把源程序翻译成某台计算机上的目标程序，用户首先要熟悉其种源语言，对源语言的语法和语义要有准确无误的理解；其次确定对编译程序的需求，同时很重要的一点是对目标机要有深刻的可行性研究，否则将会生成