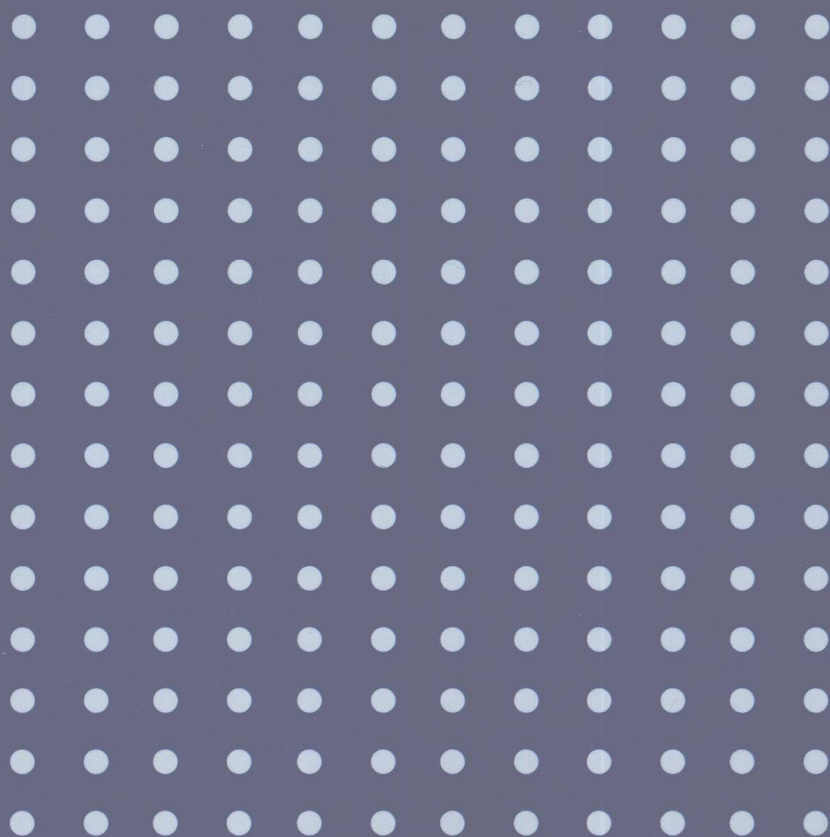


重点大学计算机专业系列教材

C++项目实战： 解释程序的面向对象设计与实现

郑晓娟 任晶磊 韦永隆 等 编著



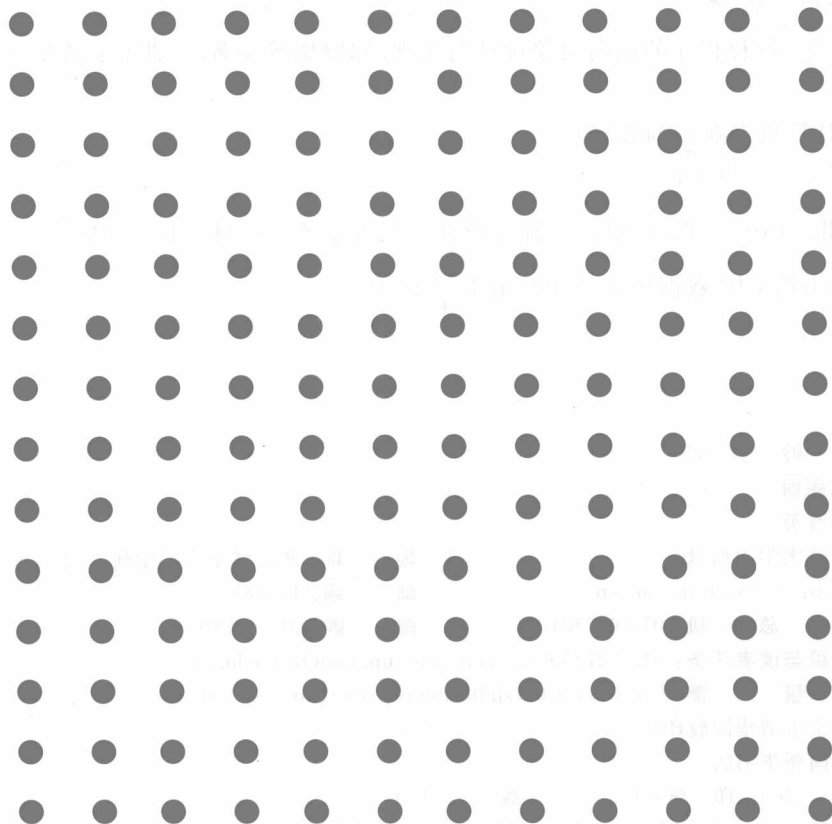
清华大学出版社



重点大学计算机专业系列教材

C++项目实战： 解释程序的面向对象设计与实现

郑晓娟 任晶磊 韦永隆 等 编著



清华大学出版社
北京

内 容 简 介

本教材为编译原理及其实践课程的工程化教学课群中的前导课程教学用书。本教材通过全面改革编译原理及其实践课程的内容和教学方法,将软件工程知识由浅入深、循序渐进地贯穿课程中,构成阶梯状上升的教学过程。

本教材以一个简单的过程式语言 C0 作为教学载体,详细介绍了该语言解释程序的面向对象设计和实现方法,同时对项目开发过程中涉及的软件工程的基本理念以及统一建模语言 UML 的使用进行了配套讲解。本教材可作为计算机专业本科生 C++ 语言和面向对象程序设计课程的实践项目,是后续编译原理和软件工程课程理论学习难得的前导实践环节。

本教材还可作为广大计算机软件开发爱好者学习实践和项目开发的参考书。

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

C++项目实战:解释程序的面向对象设计与实现/郑晓娟等编著. —北京:清华大学出版社, 2009.1

(重点大学计算机专业系列教材)

ISBN 978-7-302-18909-1

I. C… II. 郑… III. C 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2008)第 179218 号

责任编辑:丁 岭 李 晔

责任校对:焦丽丽

责任印制:李红英

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

<http://www.tup.com.cn>

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

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者:北京市清华园胶印厂

经 销:全国新华书店

开 本:185×260 印 张:12.5 字 数:305 千字

版 次:2009 年 1 月第 1 版 印 次:2009 年 1 月第 1 次印刷

印 数:1~3000

定 价:20.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770177 转 3103 产品编号:031043-01

出版说明

随着国家信息化步伐的加快和高等教育规模的扩大,社会对计算机专业人才的需求不仅体现在数量的增加上,而且体现在质量要求的提高上,培养具有研究和实践能力的高层次的计算机专业人才已成为许多重点大学计算机专业教育的主要目标。目前,我国共有16个国家重点学科、20个博士点一级学科、28个博士点二级学科集中在教育部部属重点大学,这些高校在计算机教学和科研方面具有一定优势,并且大多以国际著名大学计算机教育为参照系,具有系统完善的教学课程体系、教学实验体系、教学质量保证体系和人才培养评估体系等综合体系,形成了培养一流人才的教学和科研环境。

重点大学计算机学科的教学与科研氛围是培养一流计算机人才的基础,其中专业教材的使用和建设则是这种氛围的重要组成部分,一批具有学科方向特色优势的计算机专业教材作为各重点大学的重点建设项目成果得到肯定。为了展示和发扬各重点大学在计算机专业教育上的优势,特别是专业教材建设上的优势,同时配合各重点大学的计算机学科建设和专业课程教学需要,在教育部相关教学指导委员会专家的建议和各重点大学的大力支持下,清华大学出版社规划并出版本系列教材。本系列教材的建设旨在“汇聚学科精英、引领学科建设、培育专业英才”,同时以教材示范各重点大学的优秀教学理念、教学方法、教学手段和教学内容等。

本系列教材在规划过程中体现了如下一些基本组织原则和特点。

1. 面向学科发展的前沿,适应当前社会对计算机专业高级人才的培养需求。教材内容以基本理论为基础,反映基本理论和原理的综合应用,重视实践和应用环节。

2. 反映教学需要,促进教学发展。教材要能适应多样化的教学需要,正确把握教学内容和课程体系的改革方向。在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养,为学生知识、能力、素质协调发展创造条件。

3. 实施精品战略,突出重点,保证质量。规划教材建设的重点依然是专业基础课和专业主干课;特别注意选择并安排了一部分原来基础比较好的优秀教材或讲义修订再版,逐步形成精品教材;提倡并鼓励编写体现重点大学

计算机专业教学内容和课程体系改革成果的教材。

4. 主张一纲多本,合理配套。专业基础课和专业主干课教材要配套,同一门课程可以有多本具有不同内容特点的教材。处理好教材统一性与多样化的关系;基本教材与辅助教材以及教学参考书的关系;文字教材与软件教材的关系,实现教材系列资源配套。

5. 依靠专家,择优落实。在制订教材规划时要依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主编。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平的以老带新的教材编写队伍才能保证教材的编写质量,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

教材编委会

前言

软件产业作为信息产业的核心和国民经济信息化的基础,日益受到国家的高度重视。权威数据显示,我国软件产业每年的人才需求量近一百万。与此同时,许多软件企业反映招聘不到真正适合的软件人才。信息社会不断增长的软件人才需求与优质计算机专业教育供给不足的矛盾已成为计算机教育发展面临的一个重要问题。解决问题的方法之一就是將计算机科学的核心课程进行工程化改造,从而形成一个融基础理论、实验教学、工程实践为一体的整体化培养机制,使计算机专业教学质量得到不断提升,使计算机专业教育得到健康稳步发展,为社会培养真正满足产业需求的软件人才。

本教材为编译原理及其实践课程的工程化教学课群中的前导课教学用书。本教材通过全面改革编译原理及其实践课程的内容及教学方法,将软件工程知识由浅入深、循序渐进地贯穿在课程中,构成阶梯状上升的教学过程。

本教材的特色在于通过一个简单的教学实践项目,实现了4个“结合”:

- (1) 结合了C++编程语言的学习。
- (2) 结合了面向对象的程序设计方法与技术。
- (3) 结合了编译课程的解释程序原理。
- (4) 结合了软件工程与UML建模思想。

本教材以一个简单的过程式语言C0作为教学载体,详细介绍了该语言解释程序的面向对象设计和实现方法,同时对本项目开发过程中涉及的软件工程的基本理念以及统一建模语言UML的使用给予了配套讲解。

本教材中的项目可作为本科生C++语言和面向对象程序设计课程的实践项目,同时为编译原理和软件工程课程打下基础,作为难得的前导实践环节,为后续理论学习增加实践体验。

本书内容既不是空洞的理论指导,也不是泛泛的项目介绍,而是真正以一个解释器项目作为实例,进行深入浅出的细致讲解,内容贯穿软件开发的全流程,包含了系统分析、设计、编码、测试和部署,使读者获得完整的项目开发体验。

读者只需具备C++语言基础,便可轻松入手,逐步进阶,最终达到:

- (1) 掌握并实践C++程序设计语言,学会面向对象的分析与设计方法,获

得使用 C++ 编程语言解决实际问题的能力。

(2) 初步了解和掌握工程化的软件开发方法和统一建模语言 UML, 为后续编译原理及其实践课程工程化教学课群中的《C-语言编译程序的面向对象设计与实现》实践项目的开发打下一定的实践基础。

(3) 理解程序设计语言的解释执行过程, 为编译原理及其实践课程的工程化教学课群中的《编译原理及其实现技术》课程的学习打下一定的理论基础。

C++ 语言是一门广泛用于工业软件研发的大型语言, 支持面向对象的程序设计, 在产业界有着广泛的应用。C++ 语言有根本性优势的开发领域包括低级系统程序设计、高级大规模高性能应用设计、嵌入式程序设计、通用程序设计以及数值科学计算等。

面向对象软件开发方法与统一建模语言 UML 是当今软件工程的主流思想。学生应当初步了解和应用面向对象的软件工程方法进行面向对象软件需求分析(OOA)、面向对象系统设计(OOD)、面向对象编程(OOP)和面向对象测试(OOT), 并通过项目开发实践灵活地运用面向对象软件开发思想和技术, 学会运用面向对象的思想分析问题和解决实际问题, 同时学习软件项目管理和团队开发的工作方法。

编译(解释)程序是计算机系统不可缺少的部分, 是程序设计者的必备工具。学习并掌握编译程序的构造原理和实现技术, 能够增强对程序设计语言的理解, 提高程序设计尤其是大型软件设计的能力。然而, 编译原理课程理论性较强, 在本科教学中尤为困难。近年来的教学实践表明, 在讲授编译原理课程之前开发一个小语言的解释程序的方法不但锻炼了学生的实践能力, 同时预备了编译的初步知识, 部分解决了编译原理教学难的问题, 得到了同学们的广泛喜爱和好评。

本教材是一本难得的集 C++ 语言项目开发实例分析、解释程序原理和构造、面向对象软件开发方法和统一建模语言 UML 于一体的计算机专业教材。本教材还可以作为计算机专业低年级学生以及广大计算机软件开发爱好者的参考用书。

本书分章节系统全面地阐述了基于 UML 的 C0 解释程序的面向对象分析、设计与实现:

第 1 章详细介绍 C0 语言及其解释程序的基本概念、基本原理和相关技术, 包括表达式的处理技术, 并给出了 C0 语言解释程序的构造方法。

第 2 章简要介绍面向对象方法的基本概念和原理, 以及面向对象的工程化软件开发的不同阶段。结合实例分析传统的开发方法与面向对象的开发方法的不同。

第 3 章简要介绍统一建模语言 UML, 介绍面向对象分析与设计各个阶段使用的简单模型及其基本图形表示。

第 4 章对 C0 解释程序进行面向对象的分析, 将各种概念组织起来, 介绍其用例模型和概念模型。

第 5 章对 C0 解释程序进行面向对象的设计, 包括类及类间关系的确定, 抽象出静态结构, 设计出交互模型; 同时介绍命名空间的划分和异常处理。

第 6 章基于 C++ 语言具体构造解释程序各个类, 使用规范的注释和风格编写代码。对于软件测试部分, 不探讨具体的测试理论, 仅展示软件系统测试和集成的基本方法和途径。

第 7 章提供了 C0 语言集成开发环境的一套实现方案, 从需求分析到系统设计进行了扼要讲解。同时介绍了使用 Visual Studio 2008 创建 WinForm 应用程序的方法, 特别对 Windows 应用程序的部署进行了讲解。

CONTENTS

目录

第 1 章 项目总体介绍	1
1.1 项目需求	1
1.2 C0 语言介绍	1
1.3 基本概念	2
1.4 C0 词法分析器	6
1.4.1 Token 设计	6
1.4.2 词法分析器	8
1.4.3 算法原理	9
1.5 C0 语法分析器	10
1.5.1 语法分析器	10
1.5.2 算法原理	10
1.6 C0 解释器	11
1.6.1 虚拟存储器	11
1.6.2 虚拟输入器	12
1.6.3 虚拟输出器	12
1.6.4 运算分量栈	12
1.6.5 运算符号栈	13
1.6.6 算法原理	14
1.7 小结	15
第 2 章 面向对象方法	16
2.1 面向对象方法概述	16
2.2 面向对象的基本概念	17
2.2.1 对象	17
2.2.2 类	17
2.2.3 消息	18
2.2.4 封装	20

2.2.5 继承	21
2.2.6 多态性	22
2.2.7 结构与连接	22
2.3 面向对象的软件工程方法	23
2.4 面向对象方法与过程式程序设计	25
2.4.1 从面向过程到面向对象程序设计	25
2.4.2 面向对象方法的优势	27
2.5 小结	29
第3章 统一建模语言 UML	30
3.1 软件建模的原因	30
3.2 UML 的发展历程	30
3.3 什么是 UML	31
3.4 UML 建模工具	33
3.4.1 Rational Rose	33
3.4.2 Visio	33
3.5 UML 的主要内容	37
3.5.1 UML 的视图	38
3.5.2 UML 的图	39
3.5.3 UML 的基本模型元素	40
3.6 用例图	41
3.6.1 用例图模型	41
3.6.2 角色与用例	41
3.7 类图	42
3.7.1 类图模型	42
3.7.2 类图模型的语法表示	43
3.8 对象图	45
3.9 包图	45
3.10 状态图	46
3.11 活动图	46
3.12 时序图	47
3.13 协作图	48
3.14 组件图	48
3.15 部署图	49
3.16 小结	50
第4章 C0 解释程序的面向对象分析	51
4.1 概述	51
4.2 软件开发的一般步骤	52

4.3	软件生命周期模型	53
4.4	分析、设计与编程的关系	54
4.5	面向对象分析、设计与 UML 建模的关系	55
4.6	面向对象分析的基本过程	56
4.7	建立用例模型	57
4.7.1	角色的识别	57
4.7.2	用例的识别	58
4.7.3	用例之间的关系	59
4.7.4	用例的描述	60
4.7.5	C0 解释程序的用例模型	60
4.8	建立概念模型	62
4.8.1	识别类	62
4.8.2	确定类之间的联系	63
4.8.3	C0 解释程序的初步概念模型	64
4.9	小结	64
第 5 章	C0 解释程序的面向对象设计	66
5.1	类的设计	66
5.1.1	确定类的服务	66
5.1.2	确定类的属性	67
5.1.3	C0 解释程序的类图	67
5.2	面向数据设计	70
5.2.1	映射表类	70
5.2.2	链表类	71
5.2.3	解释环境类 Context	72
5.3	面向实体设计	73
5.3.1	词法分析类 Lexer	73
5.3.2	语法分析类 Parser	73
5.3.3	解释器类 Interpreter	74
5.4	面向接口设计	74
5.4.1	文件类 File	75
5.4.2	错误记录类 ErrorLog	75
5.5	面向关系设计	76
5.5.1	Token 类	76
5.5.2	语句类	77
5.5.3	表达式类	77
5.6	命名空间划分	79
5.6.1	命名机制	79
5.6.2	模块化机制	79

5.6.3 使用其他命名空间的元素	79
5.7 异常处理	80
5.8 小结	81
第6章 C0 解释程序的面向对象实现	82
6.1 注释、命名与代码规范	82
6.1.1 注释的内容	82
6.1.2 XML 文档注释	82
6.1.3 命名和代码规范	84
6.2 类的构造	85
6.3 类和源文件组织	85
6.4 静态类	86
6.5 接口类	87
6.6 单核类	88
6.7 面向对象的测试	91
6.7.1 原型测试	91
6.7.2 单元测试	96
6.7.3 系统测试	96
6.8 使用 Microsoft Visual Studio 2008 编写 C++ 控制台应用程序	97
6.8.1 Microsoft DreamSpark 计划: 学生免费获得正版 Visual Studio 2008 Professional Edition	97
6.8.2 使用 Visual Studio 2008 编写控制台应用程序	97
6.8.3 使用 Visual Studio 2008 自动生成类图	100
6.9 C0 解释器安装和使用说明	103
6.9.1 安装与卸载	103
6.9.2 使用说明	106
6.10 小结	106
第7章 C0 集成开发环境的设计与实现	107
7.1 Windows 窗口程序基本运行原理	107
7.1.1 Windows 窗口程序的驱动方式	107
7.1.2 Windows 消息机制	108
7.1.3 .NET Framework 与 WinForm 应用程序开发模型	110
7.2 需求分析	112
7.2.1 俘获需求	112
7.2.2 从需求出发建立概念模型	113
7.2.3 从需求出发建立用例模型	114
7.3 设计方案	117
7.3.1 用例分析	117

7.3.2	交互设计	118
7.3.3	用户界面设计	118
7.3.4	体系结构设计 with 实现	121
7.3.5	类模型的建立	122
7.4	使用 Visual Studio 2008 编写 WinForm 应用程序	123
7.4.1	WinForm 应用程序的创建	123
7.4.2	WinForm 应用程序的部署	126
7.5	C0 IDE 安装配置和使用说明	136
7.5.1	系统要求	136
7.5.2	安装与卸载	137
7.5.3	配置与使用	140
7.6	小结	142
附录A	C0 解释器源代码	143
A.1	命名空间划分	143
A.2	基本类型和行为单元	143
A.2.1	单词语义编码 SemanCode	143
A.2.2	单词类型 TokenType	144
A.2.3	Token 类及其子类	144
A.2.4	Expression 类及其子类	147
A.2.5	Statement 类及其子类	149
A.2.6	错误类 Error	151
A.3	数据结构	152
A.3.1	记号链表 TokenList	152
A.3.2	语句链表 StatementList	155
A.3.3	栈 Stack	157
A.4	词法分析阶段	158
A.4.1	单词与语义编码映射类 SemanMapper	158
A.4.2	单词与记号类型的映射类 TypeMapper	159
A.4.3	标识符与语义编码映射 IDTable	160
A.4.4	文件类 File	160
A.4.5	词法分析器类 Lexer	163
A.5	语法分析阶段	166
A.5.1	操作符优先级类 OperatorPriority	166
A.5.2	自动操作符栈 OperatorStack	167
A.5.3	语法分析器类 Parser	168
A.6	解释执行阶段	174

A.6.1 解释环境类 Context	174
A.6.2 错误记录 ErrorLog	174
A.6.3 解释器类 Interpreter	175
A.7 主函数	178
参考文献	179

项目总体介绍

第 1 章

对程序设计语言的解释是一个很复杂的过程。本项目旨在通过一个非常简单的编程语言 C0, 介绍过程式语言解释程序的基本概念、基本技术以及面向对象的软件开发方法, 展示其面向对象的分析、设计和实现过程, 以使读者对解释程序有初步了解, 对软件工程的面向对象方法进行初步应用。

1.1 项目需求

C0 语言是一个自定义的小巧的教学语言, 其相应的解释程序规模不大, 构造原理简单, 适宜初学者入手, 同时也能展现出实现语言解释的基本思想和方法。

C0 语言解释程序需要完成如下功能:

- (1) 读入使用 C0 语言编写的源程序。
- (2) 对源程序进行词法分析生成词法正确的单词序列。
- (3) 对词法正确的单词序列进行语法分析, 生成语法正确的单词序列。
- (4) 对词法、语法正确的单词序列进行解释、执行以便得出程序的正确结果。
- (5) 能够识别词法、语法错误, 并指出错误所在位置。

1.2 C0 语言介绍

C0 语言程序是一组由“{”和“}”括起来的语句序列, 语句只有赋值语句、输入语句和输出语句。变量均定义为整型变量, 默认初始值为 0, 因此不需要变量的声明。表达式由整数、变量和运算符组成。运算符有两种——加法运算符和乘法运算符。C0 语言的程序结构如图 1.1 和图 1.2 所示。

下面给出一个 C0 程序的例子:

```
{  
    x = 10;  
    read(y);  
    x = x + y;  
    write(x * y);  
}
```

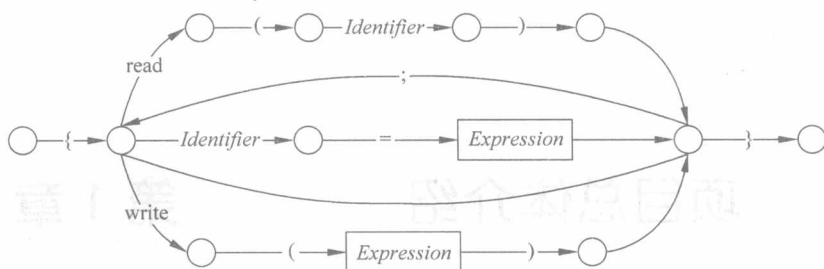


图 1.1 C0 程序结构图

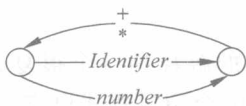


图 1.2 C0 表达式结构图

1.3 基本概念

1. 基本符号

每种程序设计语言都必须首先定义基本符号，它们一般是计算机键盘符号的子集。不同语言的基本符号不尽相同，比如“&”和“?”在C语言中是基本符号，但在Pascal中却不是。程序由基本符号组成。

2. 程序文本

程序文本是由对应语言的基本符号组成且符合语言语法的字符串，也称为源程序或源代码。源程序对程序员而言是真正意义上的符号序列，如 $x = a + 10$ 。

3. 程序文件

源程序通过键盘输入计算机，即变成另外一种形式的表示，称为源程序的机内表示，也称为源程序文件。在机内文件中可以有通过键盘输入的空白或回车换行符号。文件的结束标志通常表示为EOF。源程序的机内表示一般是ASCII码序列。每敲击一个键盘上的符号键，一个相应的ASCII码就进入计算机（每个ASCII码是8位二进制码）。例如，假设有表达式 $(10+20)$ ，则其ASCII码序列的二进制表示如下：

```
00101000 00110001 00110000 00101011 00110010 00110000 00101001
```

为了阅读方便，在每个ASCII编码后面加了空白。

4. 语义单位

源程序文本由语言的基本符号组成，例如“integer”由7个基本符号组成。但从语义角度看“integer”是不可分解的具有独立含义的基本符号序列，这种基本符号序列称为语义单

位,通常还称为单词或记号。

5. 单词分类

在处理源程序之前,首先要确定语义单位都有哪些,然后进行分类。这些工作一般和具体语言有紧密关系,甚至与设计者有关。通常可分为以下几大类:

- 标识符,如 `x1`、`y`、`length`。
- 保留字,如 `case`、`if`、`then`、`else`、`while`、`and`、`or`、`not`。
- 常数,如 `3`、`20`。
- 运算符,如 `+`、`-`、`*`、`/`、`<`、`<=`、`>=`、`=` 等。
- 界限符,如 `(`、`)`、`"`、`'`、`;`、`{`、`}`。
- 编辑符,如空格、回车、换行、制表符等。

6. 空格符号

在源程序文本中空格表现为空白,但在输入计算机时,要按下键盘上的空格键,这样代表空格的 ASCII 码将进入到源程序的机内表示,即源程序文件中。除了字符串内的空格外,其他空格一般都没有实际意义,因此在从源程序文件读进一个单词时,要把那些无意义的空格符全部过滤掉。

7. 换行符号

它和空格符类似,即在源程序中没有换行符号,但在把源程序输入计算机时,为了把后面的内容输入到下一行,需要敲键盘上的回车换行键,这样回车换行键的 ASCII 码便进入到源程序文件中。尽管换行符没有任何程序意义,但在语法分析时,它可用于指出程序的出错地点,因此对它不能像空格符那样删除掉而要保留到单词序列中。

8. 单词编码

在实现时需要对单词进行编码,以便使单词信息数据化。对如何编码并没有统一的说法。从形式上说,单词的编码(称为 Token)就是一个二进制数。例如:

单词	标识符	整常数	{	}	+	*	;
编码	0	1	2	3	4	5	6

标识符和常数个数无穷多,因此不可能把每个标识符或常数作为单独一类单词,而是把所有标识符作为一类单词,把所有常数也作为一类单词。单词的编码简称为词码。若使用 C++ 语言开发解释程序,可使用宏定义方法或者定义枚举类型,例如:

```
/* 单词语义编码的枚举类型 */
enum SemanCode
{
    BEGIN,          //{
    PLUS,            //{ +
    MULT,            //{ *
```

```

    ASS,          // =
    READ,
    WRITE,
    SEMI,         //;
    OPEN,         //(
    CLOSE,        //)
    CONST,
    END,          //{
};

```

9. 语义信息

在进行语法分析时,要用到 Token 的编码,但在进行语义分析时还需要其他信息,例如常数的值和标识符的名字本身等,这些信息称为 Token 的语义信息(简称为词义)。对于其他 Token 来说,即使只有 Token 编码也是可以的。总的来说,Token 由两部分组成:其一是单词编码,其二是语义信息。下面将用类 C 语言形式 Token.class 和 Token.seman 分别表示 Token 的词码和词义部分。Token 数据结构的类 C 语言描述如下:

```

typedef struct {
    int class;
    char seman[];
    position_in_file pos;
} Token_type;

```

不难看出,Token.class 的值是一个正整数,而 Token.seman 则是字符串数组(作为字符串文件使用)。其中 position_in_file 的定义如下,它表示当前单词在程序文件中的位置,其内容包括指向单词位置的文件指针,单词在程序文本中的行号,以及单词在行中的序号。当然,这里给出的只是一种类 C 语言描述的方案,但对任何解释程序来说,单词的类别和语义信息都是不可缺少的。

```

typedef struct {
    char * file_name;
    int line_number;
    int token_number;
} position_in_file;

```

10. 读进字符

源程序文件是字符序列文件,读源程序的最小动作是读进该文件的下一字符。读出来的实际上是该字符的 ASCII 码(8 位 0/1 串)。C 语言提供了很多有关字符和字符串的标准函数,例如语句 `ch = fgetc(fp)`,将从当前文件读进头字符,并把相应的 ASCII 码值赋给 `ch`。

11. 识别字符

有时需要判断字符属于哪类,为此可定义一些判断函数,其返回值是 1 或 0。它们都是很短的程序段,因此 C 语言中可采用宏定义的方法。下面给出一些例子: