

国外计算机科学经典教材

Mc
Graw
Hill Education

C++: A Beginner's Guide

Second Edition

C++ 基础教程

(第2版)

(美) Herbert Schildt 著
王 军 译



清华大学出版社

国外计算机科学经典教材

C++基础教程 (第2版)

(美) Herbert Schildt 著

王 军 译

清华大学出版社

北 京

Herbert Schildt C++: A Beginner's Guide, Second Edition

EISBN: 0-07-223215-3

Copyright© 2004 by The McGraw-Hill Companies, Inc.

Original language published by The McGraw-Hill Companies, Inc. All Rights reserved. No part of this publication may be reproduced or distributed by any means, or stored in a database or retrieval system, without the prior written permission of the publisher.

Simplified Chinese translation edition is published and distributed exclusively by Tsinghua University Press under the authorization by McGraw-Hill Education(Asia) Co., within the territory of the People's Republic of China only (excluding Hong Kong, Macao SAR and Taiwan). Unauthorized export of this edition is a violation of the Copyright Act. Violation of this Law is subject to Civil and Criminal Penalties.

本书中文简体字翻译版由美国麦格劳-希尔教育出版(亚洲)公司授权清华大学出版社在中华人民共和国境内(不包括中国香港、澳门特别行政区和中国台湾地区)独家出版发行。未经许可之出口视为违反著作权法,将受法律之制裁。未经出版者预先书面许可,不得以任何方式复制或抄袭本书的任何部分。

北京市版权局著作权合同登记号 图字: 01-2003-8454

本书封面贴有 McGraw-Hill 公司防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

C++基础教程:第2版/(美)斯柴德(Schildt, H.)著;王军译.—北京:清华大学出版社,2004

书名原文: C++: A Beginner's Guide, Second Edition

(国外计算机科学经典教材)

ISBN 7-302-08264-2

I. C… II. ①斯…②王… III. C 语言—程序设计—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2004)第 018509 号

出 版 者: 清华大学出版社 地 址: 北京清华大学学研大厦

<http://www.tup.com.cn> 邮 编: 100084

社 总 机: 010-62770175 客户服务: 010-62776969

组稿编辑: 曹 康

文稿编辑: 崔 伟

封面设计: 康 博

版式设计: 康 博

印 刷 者: 北京市清华园胶印厂

装 订 者: 三河市新茂装订有限公司

发 行 者: 新华书店总店北京发行所

开 本: 185×260 印张: 32 字数: 806 千字

版 次: 2004 年 5 月第 1 版 2004 年 5 月第 1 次印刷

书 号: ISBN 7-302-08264-2/TP·5960

印 数: 1~5000

定 价: 59.00 元

本书如存在文字不清、漏印以及缺页、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)62770175-3103 或 (010)62795704

国外计算机科学经典教材

编审委员会

主任委员:

孙家广

清华大学教授

副主任委员:

周立柱

清华大学教授

委员（按姓氏笔画排序）:

王成山

天津大学教授

王 珊

中国人民大学教授

冯少荣

厦门大学教授

冯全源

西南交通大学教授

刘乐善

华中科技大学教授

刘腾红

中南财经政法大学教授

吉根林

南京师范大学教授

孙吉贵

吉林大学教授

阮秋琦

北京交通大学教授

何 晨

上海交通大学教授

吴百锋

复旦大学教授

李 彤

云南大学教授

沈钧毅

西安交通大学教授

邵志清

华东理工大学教授

陈 纯

浙江大学教授

陈 钟

北京大学教授

陈道蓄

南京大学教授

周伯生

北京航空航天大学教授

孟祥旭

山东大学教授

姚淑珍

北京航空航天大学教授

徐佩霞

中国科学技术大学教授

徐晓飞

哈尔滨工业大学教授

秦小麟

南京航空航天大学教授

钱培德

苏州大学教授

曹元大

北京理工大学教授

龚声蓉

苏州大学教授

谢希仁

中国人民解放军理工大学教授

前言

C++是一门非常优秀的编程语言，掌握它，可以开发高性能的软件。它所定义的语法和风格影响了随后出现的所有编程语言。例如，Java 和 C# 都源自 C++。而且 C++是目前普遍使用的编程语言，几乎所有的专业程序员都可以使用这门语言编程。掌握 C++将为您完成编程任务打下坚实的基础。

本书讲授 C++编程的基础知识。它采用循序渐进的方式组织内容，并配以许多示例、自测题目和项目实例。学习本书不要求您具备编程经验。本书将从基础知识讲起，如怎样编译和运行 C++程序。然后讨论组成 C++语言的关键字、特征和结构。通过学习本书，您将牢牢掌握 C++编程的基础知识。

在学习本书之前，首先对读者声明：本书介绍的只是利用 C++编程的入门知识。C++是一门庞大、复杂的语言，C++编程不仅仅涉及关键字、运算符和定义语言的语法，它还涉及如何使用类和函数库来帮助开发程序。虽然本书讨论了库的几个元素，但限于篇幅，大多数都没有讨论。要成为一流的 C++程序员，还需掌握 C++库。在学完本书后，您将具备探索 C++库和其他领域的必要知识。

本书组织结构

本书的结构严谨，并且循序渐进地安排内容，每一部分的内容都以前面部分的内容为基础。全书共有 12 章，每章讨论 C++语言的一个方面。本书的独特之处在于，它使用一些特色段落来强化您所学习的知识和技能。

本章学习目标

每章的开头都列出本章要完成的学习目标。

本章自测

每章章末都有一个自测部分，检验您对知识的掌握程度。全部答案附在附录 C 中。

一分钟练习

在讲完每个主要部分时都附有一个小练习，测试您对前面所学关键点的理解。答案就在问题的下面。

专家答疑

贯穿全书的“专家答疑”，是书中各相应主题的补充信息和重要评论。

项目

每章都包含一个或多个项目实例，演示如何应用所学的知识。您也可以将其用作自己程序的起点。

不要求编程经验

本书不要求读者有编程经验。即使以前从来没有编写过程序，也可以学习本书。不过大多数读者多少都有一点编程的经验。其中很多人都是从 Java 或 C#获得的。您将了解到，C++就是这两种语言的起源。因此，如果您对 Java 或 C#有所了解，就很容易学习 C++了。

需要的软件

要编译和运行本书的程序，需要安装一个新版的 C++编译器，如 Microsoft Visual C++或 Borland C++ Builder 的最新版本。

免费代码下载

最后，告诉大家一个好消息：本书所有示例和项目的源代码都可以从 www.osborne.com 免费下载。

目 录

第 1 章 C++基础	1
1.1 C++简史	1
1.1.1 C 语言：现代编程的开端	2
1.1.2 对 C++的需求	2
1.1.3 C++的诞生	3
1.2 C++的演化	3
1.3 C++同 Java 与 C# 的关系	4
1.4 面向对象编程	5
1.4.1 封装性	6
1.4.2 多态性	6
1.4.3 继承性	7
1.5 第一个简单的程序	8
1.5.1 键入程序	8
1.5.2 编译程序	8
1.5.3 运行程序	9
1.5.4 逐行讲解第一个示例程序	10
1.6 处理语法错误	11
1.7 另一个简单的程序	12
1.8 使用运算符	13
1.9 读取键盘输入	15
1.10 一些输出选项	16
1.11 另一种数据类型	18
1.12 两条控制语句	21
1.12.1 if 语句	21
1.12.2 for 循环	23
1.13 使用代码块	24
1.14 分号与定位	25
1.15 函数	28
1.16 C++的库	30
1.17 C++的关键字	31
1.18 标识符	31
1.19 本章自测	32

第 2 章 数据类型和运算符	33
2.1 数据类型的重要意义	33
2.2 C++的数据类型	33
2.2.1 整型数据	36
2.2.2 字符型数据	37
2.2.3 浮点型数据	38
2.2.4 布尔型数据	39
2.2.5 Void 型数据	40
2.3 字面值	42
2.3.1 十六进制和八进制的字面值	43
2.3.2 字符串字面值	44
2.3.3 字符转义序列	44
2.4 变量详解	45
2.4.1 初始化变量	46
2.4.2 动态初始化	46
2.5 运算符	47
2.6 算术运算符	47
2.7 关系运算符和逻辑运算符	50
2.8 赋值运算符	54
2.8.1 复合赋值运算符	54
2.8.2 赋值运算中的类型转换	55
2.9 表达式	56
2.10 表达式中的类型转换	56
2.10.1 布尔型的转换	56
2.10.2 类型强制转换	56
2.11 空格和圆括号	57
2.12 本章自测	60
第 3 章 程序控制语句	62
3.1 if 语句	62
3.1.1 条件表达式	64
3.1.2 嵌套的 if 语句	65
3.1.3 if-else-if 阶梯语句	66
3.2 switch 语句	67
3.3 for 循环语句	74
3.3.1 for 循环语句的一些变化	75
3.3.2 可以缺少的部分	77
3.3.3 无限 for 循环	78

3.3.4 无循环体的循环	78
3.3.5 在 for 语句循环体内声明循环控制变量	79
3.4 while 循环语句	80
3.5 do-while 循环	82
3.6 使用 break 语句退出循环	87
3.7 使用 continue 语句	89
3.8 嵌套的循环	93
3.9 使用 goto 语句	95
3.10 本章自测	95
第 4 章 数组、字符串和指针	97
4.1 一维数组	97
4.2 二维数组	101
4.3 多维数组	102
4.4 字符串	105
4.4.1 字符串基本知识	105
4.4.2 从键盘中读取字符串	106
4.5 一些字符串库函数	108
4.5.1 strcpy 函数	108
4.5.2 strcat 函数	108
4.5.3 strcmp 函数	108
4.5.4 strlen 函数	109
4.5.5 字符串函数实例	109
4.5.6 使用 null 终结符	110
4.6 数组的初始化	111
4.7 字符串数组	115
4.8 指针	116
4.9 指针的概念	117
4.10 指针运算符	117
4.10.1 指针的基本类型很重要	118
4.10.2 通过指针赋值	120
4.11 指针表达式	121
4.11.1 指针运算	121
4.11.2 指针的比较	122
4.12 指针和数组	123
4.13 字符串常量	127
4.14 指针数组	130
4.15 null 指针的约定	131

4.16	多重间接访问	132
4.17	本章自测	134
第 5 章	函数简介	135
5.1	函数的基础知识	135
5.1.1	函数的通式	135
5.1.2	创建函数	136
5.1.3	使用实际参数	137
5.1.4	使用 return 语句	139
5.1.5	返回值	141
5.1.6	在表达式中使用函数	143
5.2	作用域法则	144
5.2.1	局部作用域	144
5.2.2	全局作用域	149
5.3	将指针和数组传递给函数	151
5.3.1	传递指针	151
5.3.2	传递数组	152
5.3.3	传递字符串	155
5.4	返回指针	156
5.5	main() 函数	157
5.5.1	给 main() 函数传送命令行参数	157
5.5.2	传递数字命令行参数	159
5.6	函数原型	160
5.7	递归	162
5.8	本章自测	168
第 6 章	函数详解	170
6.1	传递参数的方法	170
6.1.1	C++ 如何传递参数	170
6.1.2	使用指针创建按引用调用	171
6.2	引用参数	173
6.2.1	返回引用	177
6.2.2	独立引用	179
6.2.3	使用引用时的几个限制	180
6.3	函数重载	181
6.4	默认的函数参数	193
6.4.1	默认参数与重载	194
6.4.2	正确使用默认参数	196
6.5	函数重载和多义性	197

6.6 本章自测	199
第 7 章 更多数据类型和运算符	201
7.1 const 和 volatile 限定符	201
7.1.1 const 限定符	201
7.1.2 volatile 限定符	203
7.2 存储类说明符	204
7.2.1 auto 说明符	204
7.2.2 extern 说明符	204
7.2.3 static 变量	205
7.2.4 register 变量	209
7.3 枚举	210
7.4 typedef 关键字	213
7.5 按位运算符	214
7.5.1 AND、OR、XOR 和 NOT 运算符	215
7.5.2 移位运算符	219
7.6 ?运算符	226
7.7 逗号运算符	227
7.8 多重赋值	229
7.9 复合赋值	229
7.10 使用 sizeof 运算符	229
7.11 优先级小结	231
7.12 本章自测	231
第 8 章 类和对象	233
8.1 类的基础知识	233
8.1.1 类的通式	233
8.1.2 定义类并创建对象	234
8.1.3 向类中添加函数	238
8.2 构造函数和析构函数	246
8.2.1 带参数的构造函数	248
8.2.2 将构造函数添加到 Vehicle 类中	250
8.2.3 另一种初始化方法	251
8.3 内联函数	253
8.4 对象数组	261
8.5 指向对象的指针	264
8.6 对象引用	266
8.7 本章自测	266

第 9 章 类的详解	267
9.1 重载构造函数	267
9.2 对象赋值	268
9.3 将对象传递给函数	270
9.3.1 构造函数、析构函数和对象传递	271
9.3.2 通过引用传递对象	273
9.3.3 传递对象时潜在的问题	274
9.4 返回对象	275
9.5 创建和使用拷贝构造函数	277
9.6 友元函数	280
9.7 结构体和共用体	285
9.7.1 结构体	285
9.7.2 共用体	286
9.7.3 匿名共用体	288
9.8 This 关键字	289
9.9 运算符重载	291
9.10 使用成员函数进行运算符重载	291
9.10.1 顺序的重要性	294
9.10.2 使用成员函数重载一元运算符	295
9.11 非成员运算符函数	299
9.11.1 使用友元重载一元运算符	303
9.11.2 运算符重载的技巧和限制	305
9.12 本章自测	315
第 10 章 继承、虚函数和多态性	316
10.1 继承	316
10.2 基类访问控制	322
10.3 使用受保护的成员	324
10.4 构造函数和继承	326
10.5 创建多层结构	336
10.6 继承多个基类	339
10.7 构造函数和析构函数执行的顺序	340
10.8 指向派生类型的指针	341
10.9 虚函数和多态性	342
10.9.1 虚函数的基础知识	342
10.9.2 继承虚函数	344
10.9.3 使用虚函数的原因	346
10.10 应用虚函数	346

10.11 纯虚函数和抽象类	350
10.12 本章自测	354
第 11 章 C++ I/O 系统	355
11.1 对早期和现代的 C++ I/O 进行比较	355
11.2 C++流	356
11.3 C++流类	356
11.4 重载 I/O 运算符	358
11.4.1 创建插入函数	358
11.4.2 使用友元函数重载插入函数	360
11.4.3 重载提取函数	361
11.5 格式化 I/O	362
11.5.1 使用 ios 成员函数进行格式化	363
11.5.2 使用 I/O 操控符	368
11.5.3 创建自己的操控符函数	370
11.6 文件 I/O	372
11.6.1 打开和关闭文件	372
11.6.2 读写文本文件	374
11.6.3 非格式化和二进制 I/O	376
11.6.4 读写数据块	378
11.7 更多的 I/O 函数	380
11.7.1 更多的 get() 版本	380
11.7.2 getline()	381
11.7.3 检测 EOF	381
11.7.4 peek() 和 putback()	381
11.7.5 flush()	381
11.8 随机存取	385
11.9 检查 I/O 状态	388
11.10 本章自测	389
第 12 章 异常、模板和其他高级主题	390
12.1 异常处理	390
12.1.1 异常处理的基础知识	390
12.1.2 使用多个 catch 语句	395
12.1.3 捕获所有异常	397
12.1.4 指定由函数抛出的异常	398
12.1.5 再次抛出异常	400
12.2 模板	401
12.2.1 通用函数	402

12.2.2	具有两个通用类型的函数	403
12.2.3	显式重载通用函数	404
12.2.4	通用类	406
12.2.5	显式的类具体化	408
12.3	动态分配内存	413
12.3.1	初始化分配的内存	415
12.3.2	给数组分配内存	416
12.3.3	给对象分配内存	417
12.4	命名空间	420
12.4.1	命名空间的基础知识	421
12.4.2	using 语句	424
12.4.3	匿名命名空间	426
12.4.4	std 命名空间	426
12.5	静态类成员	427
12.5.1	静态成员变量	427
12.5.2	静态成员函数	429
12.6	运行时类型标识(RTTI)	431
12.7	强制类型转换运算符	434
12.7.1	dynamic_cast	435
12.7.2	const_cast	435
12.7.3	static_cast	436
12.7.4	reinterpret_cast	436
12.8	接下来做什么	436
12.9	本章自测	437
附录 A	预处理器	438
附录 B	使用旧版本 C++编译器	448
附录 C	测验答案	450

第1章 C++ 基础

本章学习目标

- 了解 C++的发展史
- 学习 C++与 C、C#及 Java 的关系
- 学习面向对象程序设计的 3 条准则
- 创建、编译及运行 C++程序
- 使用变量
- 使用运算符
- 从键盘上读取输入
- 处理 if 和 for 语句
- 应用代码块
- 使用函数
- 了解 C++关键字
- 理解 C++标识符的规则

当前，如果说有哪种语言能够表达编程本质的话，那么它就是 C++。C++是一种优秀的、可以开发高性能软件的编程语言。它的语法已经成为专业编程语言的标准，它的设计原理贯穿于整个计算领域。C++也是 Java 和 C#的始祖。总之，要想成为一名专业的编程人员，就必须精通 C++。它是通向现代编程领域的大门。

本章简要介绍 C++，包括它的发展历史，设计原理和一些最重要的特性。到目前为止，学习编程语言最困难的是：所有的组成部分都不是孤立存在的。相反，语言的所有组成部分都是共同发挥作用的。这种相关性使得很难单独介绍 C++的一个方面。为了克服这个困难，本章将简要介绍 C++的几个特性，包括 C++程序的一般格式，一些基本的控制语句和运算符。本章不深入介绍细节内容，而主要阐述所有 C++程序共有的一般性概念。

1.1 C++简史

C++的发展源于 C 语言，原因非常容易理解：C++建立在 C 的基础之上。因此，C++是 C 的一个超集。C++扩展并增强了 C 语言的功能，支持面向对象编程(将在本章后面讲述)。同时 C++还对 C 语言的其他方面作了多处改进，包括对库例程的扩展。然而，C++的思想和风格还是直接继承于 C。因此，要想完全理解和掌握 C++，就需要对 C 语言有较为深入的理解。

1.1.1 C 语言：现代编程的开端

C 语言的诞生开辟了一个崭新的编程时代。它的影响绝对不可低估，因为它从根本上改变了程序设计和思考问题的方式。它的设计原理和语法对以后出现的所有主流编程语言都产生了影响。在计算机领域，C 曾是主要的、具有革命性的语言。

C 是由 Dennis Ritchie 在一台使用 UNIX 操作系统的 DEC PDP-11 计算机上创造并首次实现的。C 是由一门更“古老”的语言 BCPL 发展演化而来的。BCPL 语言是由 Martin Richards 开发的。BCPL 语言对 B 语言产生了重要影响，B 语言由 Ken Thompson 开发，它推动了 20 世纪 70 年代 C 语言的发展。

在 C 语言诞生之前，计算机编程语言的设计通常被视为一个学术活动，或者说它是由委员会的专家们开发的。而 C 不同，它是由真正的编程人员设计、实现和开发的，反映了他们处理编程任务的方式。C 的各种特性都由那些真正使用的人琢磨、测试和再三思考过。因此，C 吸引了大量支持者，并迅速成为软件界人员首选的编程语言。

C 语言诞生的源动力是 20 世纪 60 年代的结构化编程革命。在结构化编程出现之前，大型程序很难编写，因为程序逻辑总会退化为“意大利面条式的代码”，杂乱无序的跳转、调用和返回使人们很难理清其中的头绪。结构化编程语言通过添加定义清晰的控制语句、带有局部变量的子例程和其他改进措施解决了这个问题。使用结构化的编程语言，才可能编写比较大型的程序。

尽管当时还出现了其他的结构化语言，例如 Pascal，但是 C 首先成功实现了集功能强大、简洁易用和表达丰富等众多特征于一身。它的简洁、易于使用的语法与其基本设计理念直接相关——“程序是由程序员来管理的，而不是由语言控制的”。这让很多人迅速转向使用 C 语言。这种轰动效应在今天看来有些难于理解，但是 C 带来了程序员们期待已久的一股清新气息。因此，C 迅速成为 20 世纪 80 年代使用最为广泛的结构化编程语言。

1.1.2 对 C++的需求

看了前面的叙述之后，您也许会不解，为什么要发明 C++。既然 C 是一种成功的计算机编程语言，那为什么还需要其他语言？答案在于复杂性。纵览编程技术的历史，程序的日益复杂要求人们用更好的方法来解决这些复杂性，于是，C++就应运而生了。为了更好地理解程序的日益复杂和计算机语言发展之间的相关性，可以考虑下列问题。

在计算机发明之后，编程方法发生了巨大的变化。例如，当计算机刚发明时，编程是通过在计算机的前面板上手动操作二进制机器指令来进行的。当时程序只有几百个指令，这种方法还可以奏效。随着程序的增长，出现了汇编语言，程序员使用机器指令的符号表示法，就可以处理更大型的、更复杂的程序。当程序进一步增长时，高级语言就出现了，它们给程序员提供了更多的工具来处理这些复杂问题。

实际上，第一种广泛使用的计算机语言是 FORTRAN。虽然 FORTRAN 给人留下了深刻的印象，但它并不能编写出清晰、易于理解的程序。在 20 世纪 60 年代诞生了结构化的编程方法，它是类 C 语言所提倡的编程方法。结构化编程语言的出现，第一次实现了轻松编写中等复杂程度的程序。然而，即使是使用结构化的编程方法，一旦项目达到一定的规模，其复杂性仍然超

出了程序员的管理能力。到 20 世纪 70 年代末期,许多项目已经接近或者达到了这种规模。

为了解决这个问题,一种新的编程方法出现了,这就是面向对象编程(object-oriented programming, OOP)。使用 OOP,程序员可以处理更大型、更为复杂的程序。而问题是 C 语言并不支持面向对象编程。对 C 语言支持面向对象编程版本的期待最终导致了 C++ 的诞生。

归根到底,虽然 C 语言是业界最受欢迎、使用最广泛的专业编程语言,但是现在它却面临着其处理复杂性的能力小于实际需要的严重挑战。一旦程序达到了一定的规模,它就非常复杂,因而难以作为一个整体来管理。C++ 扫清了这种障碍,帮助程序员理解并管理更大型、更复杂的程序。

1.1.3 C++ 的诞生

1979 年, Bjarne Stroustrup 在新泽西州 Murray Hill 的贝尔实验室中发明了 C++。最初他将其称为“带类的 C”。但在 1983 年,这种语言的名称改为 C++。

Stroustrup 在 C 的基础上创建了 C++ 语言, C++ 包括 C 的所有功能、属性和优点。他仍然坚持“程序是由程序员来管理的,而不是由语言控制的”这个 C 的基本设计理念。这对理解 Stroustrup 并没有创建一门全新的编程语言,而仅仅是改进了一门已经非常成功的语言,是至关重要的。

Stroustrup 向 C 中添加的绝大多数功能都用于支持面向对象编程的。从本质上讲, C++ 是 C 的面向对象版本。Stroustrup 将 C++ 建立在 C 的基础之上,使得 C 向 OOP 的过渡变得非常平滑。无需学习全新的语言, C 程序员只需要再掌握少量的功能,即可享受面向对象编程带来的好处。

在最初创建 C++ 时, Stroustrup 就知道保持 C 的原始风格是很重要的,包括它的高效、灵活性及其设计理念,与此同时再添加对面向对象的支持。令人高兴的是,他的目标完全实现了。C++ 除了支持面向对象的功能外,依然向程序员提供了 C 的灵活性与控制功能。

尽管 C++ 最初的设计目标是帮助管理超大型的程序,但是它的作用并不局限于此。事实上, C++ 的面向对象功能可以高效地运用在所有编程任务上。C++ 在诸如编辑器、数据库、个人文件系统、网络工具和通信程序等项目中的应用很广泛。因为 C++ 共享了 C 的高效性,所以许多大型高性能系统软件都使用 C++ 创建。同时, C++ 也是编写 Windows 程序最常用的语言。

1.2 C++ 的演化

自从 C++ 发明之后,它经历了 3 次主要的修订,每一次修订都会对这种语言进行一些添加和修改。第一次修订在 1985 年,第二次在 1990 年。第三次修订发生在 C++ 的标准化过程中。几年前,人们开始对 C++ 进行标准化工作。为了实现这一目标,成立了由 ANSI(American National Standards Institute, 美国国家标准化协会)和 ISO(International Standards Organization, 国际标准化组织)参加的联合标准化委员会。推荐的第一个标准草案产生于 1994 年 1 月 25 日。在这个草案中, ANSI/ISO C++ 委员会(作者也曾是其委员)保持了 Stroustrup 最初定义的功能,并添加了一些新的内容。但是,总的来说,这个最初的草案反映了当时 C++ 的状况。

在 C++ 第一个标准草案完成不久,发生了一件事情,它导致了 C++ 标准的大大扩展:这就是 Alexander Stepanov 创建了标准模板库(Standard Template Library, STL)。STL 是一组通用的