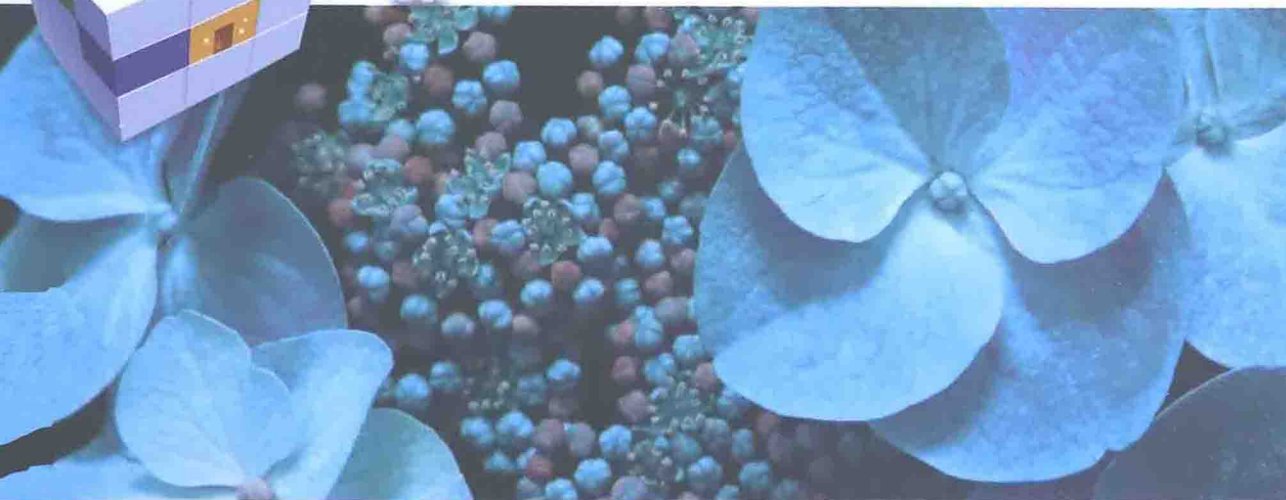


C++

从零开始学

(视频教学版)



面向C++编程的初学者，详细介绍C++编程方法和技巧
提供了大量C++编程的示例，方便读者快速掌握C++编程
配套代码、课件、教学视频，方便读者学习

刘增杰 编著



代码、课件、教学视频

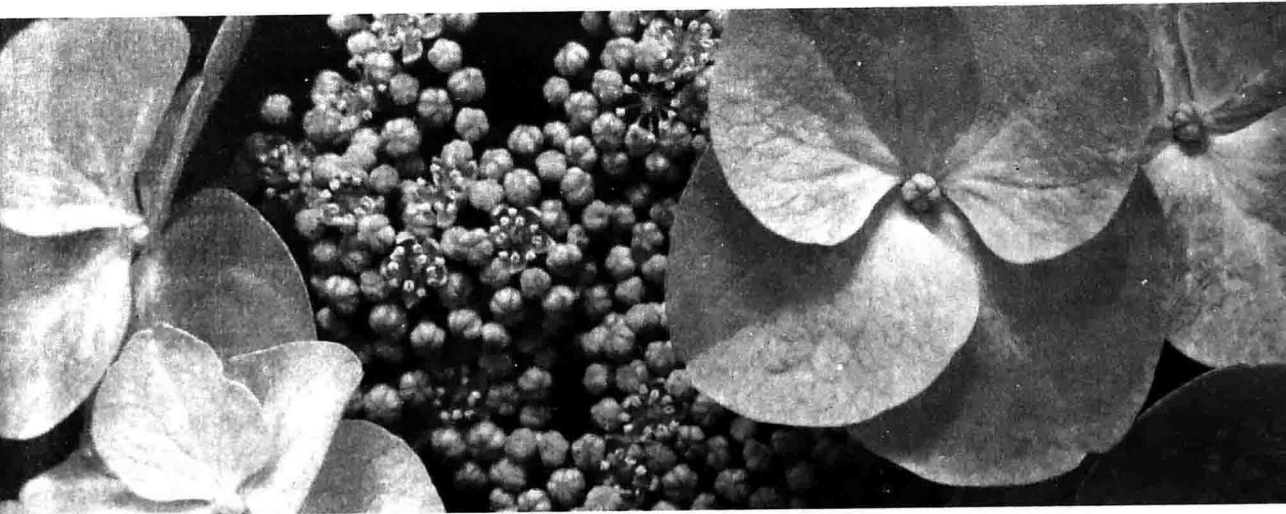


清华大学出版社

C++

从零开始学

刘增杰 编著 （视频教学版）



清华大学出版社
北京

内 容 简 介

本书面向 C++编程的初学者,以及广大 C++编程爱好者,循序渐进地介绍了 C++应用与开发的相关基础知识,提供了大量具体操作 C++编程的示例,供读者实践。每节都清晰阐述了代码如何工作及作用,使读者能在最短的时间内有效地掌握 C++编程。

全书共分为 17 个章节,分别介绍了为什么要学习 C++、C++程序结构、基本数据类型和数据处理、运算符与表达式、程序流程控制、函数、数组与字符串、指针、struct 与其他复合数据类型、类的使用方法、对象的初始化和清除、运算符的重载、类的继承、虚函数和抽象类、C++中的文件处理、异常处理和 C++的高级概念等知识,并在每章的后面提供自我评估的习题,供读者操作练习,从而加深理解。

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++从零开始学:视频教学版/刘增杰编著. -- 北京:清华大学出版社,2015
ISBN 978-7-302-39872-1

I. ①C… II. ①刘… III. ①C 语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2015)第 080636 号

责任编辑:夏非彼

封面设计:王 翔

责任校对:闫秀华

责任印制:何 芊

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

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者:北京密云胶印厂

经 销:全国新华书店

开 本:190mm×260mm 印 张:22 字 数:563 千字
(附光盘 1 张)

版 次:2015 年 5 月第 1 版 印 次:2015 年 5 月第 1 次印刷

印 数:1~3000

定 价:49.00 元

前言

本书是一本面向 C++ 初学者的高质量书籍，通过简明而不失详细的语法讲解和实例分析，读者可以快速入门，为数据结构学习、数据分析处理、应用编程开发等打下基础。

本书内容丰富全面，图文并茂，步骤清晰，通俗易懂，使读者能理解 C++ 的技术构成，并能解决实际生活或工作中的问题，真正做到知其然更知其所以然。本书注重实用，操作性强，详细讲解书中每一个知识点和每一个 C++ 操作的方法和技巧，使零基础的读者能循序渐进地掌握 C++ 语言编程基础知识和技巧，真正体现本书“从零开始学”的含义，是一本物超所值的好书。

本书特色

- **内容全面：**知识点讲解安排由浅入深，涵盖了所有 C++ 的基础知识点，由浅入深地掌握 C++ 开发技术。
- **图文并茂：**注重操作，图文并茂，在介绍案例的过程中，每一个操作均有对应步骤和过程说明。这种结合的方式使读者在学习过程中能够直观、清晰地看到操作的过程以及效果，便于更快地理解和掌握。
- **易学易用：**颠覆传统“看”书的观念，变成一本能“操作”的图书。
- **案例丰富：**把知识点融汇于系统的案例实训当中，并且结合综合案例进行讲解和拓展，从而达到“知其然，并知其所以然”的效果。
- **提示技巧：**本书对读者在学习过程中可能会遇到的疑难问题以“提示”和“技巧”的形式进行了说明，以免读者在学习的过程中走弯路。
- **超值光盘：**本书提供了视频教程和经典习题，让您在编程应用中掌握 C++ 的每一项技能。

读者对象

本书是一本完整介绍 C++ 的教程，内容丰富，条理清晰，实用性强，适合如下读者学习使用：

- 对 C++ 完全不了解或者有一定了解的初学者。
- 对编程有兴趣，希望快速、全面掌握 C++ 的读者。
- 对没有任何 C++ 经验，想学习 C++，并进行应用开发的读者。

致谢

本书作者长期从事 C++ 实训的培训工作。参与本书编写的人员除了封面署名人员以外，还有

刘玉萍、胡同夫、王英英、肖品、孙若淞、王攀登、王维维、梁云亮、刘海松、陈伟光、包惠利等人参与编写工作。虽然倾注了编者的努力,但由于水平有限、时间仓促,书中难免有错漏之处,请读者谅解。如果遇到问题或有意见,敬请与我们联系,我们将全力提供帮助,我们的技术支持 QQ: 2606372761。

编 者
2015 年 01 月

目 录

第 1 章 为什么要学习 C++——认识 C++	1
1.1 C/C++的起源	1
1.2 C++的特色	2
1.3 关于 ANSI/ISO C++标准	3
1.4 语言的翻译过程	4
1.5 编写代码前的准备——安装开发环境	5
1.6 小试身手——新建一个 C++项目	8
1.7 疑难解惑	11
疑问 1 C++与 C 的区别?	11
疑问 2 C++编译过程如何?	11
疑问 3 C++都有什么版本?	11
1.8 自我评估	12
第 2 章 C++程序结构	13
2.1 简单程序	13
2.2 C++程序分析	14
2.2.1 #include 指令及头文件	14
2.2.2 main 函数	15
2.2.3 变量声明和定义	16
2.2.4 函数的声明	18
2.2.5 关于注释	19
2.3 输入输出对象	21
2.3.1 cout 输出数据	21
2.3.2 cin 读取输入数据	24
2.4 标识符	25
2.4.1 保留字	26
2.4.2 标识符命名	26
2.5 预处理	26
2.6 命名空间	29
2.6.1 命名空间的定义	29
2.6.2 using 关键字	30

2.6.3 命名空间 std.....	32
2.7 小试身手——入门经典程序	32
2.8 疑难解惑	34
疑难 1 下列标识符哪些是合法的?.....	34
疑难 2 下面一段程序的含义是什么?	34
疑难 3 注释有什么作用? C++中有哪几种注释的方法? 它们之间有什么区别?.....	35
2.9 自我评估	35
第 3 章 基本数据类型	36
3.1 变量与常量	36
3.1.1 变量	36
3.1.2 常量	39
3.2 基本变量类型	42
3.2.1 整数类型	42
3.2.2 字符类型	43
3.2.3 浮点数类型	45
3.2.4 布尔类型	47
3.3 typedef	48
3.4 小试身手——测试基本数据类型的字节长度	51
3.5 疑难解惑	52
疑问 1 C++在代码移植中, 使用整型时注意什么?.....	52
疑问 2 C++中, 0 所扮演的不同角色是什么?	53
疑问 3 typedef 和 define 的区别是什么?.....	53
3.6 自我评估	53
第 4 章 运算符和表达式	54
4.1 运算符概述	54
4.1.1 赋值运算符	54
4.1.2 算术运算符	55
4.1.3 关系运算符	57
4.1.4 逻辑运算符	58
4.1.5 自增和自减运算符	60
4.1.6 位逻辑运算符	61
4.1.7 移位运算符	63
4.1.8 三元运算符	64
4.1.9 逗号运算符	65
4.1.10 类型转换运算符	66
4.2 运算符优先级和结合性	67
4.2.1 运算符优先级	67
4.2.2 运算符结合性	68
4.3 小试身手——综合运用运算符	70

4.4 疑难解惑	71
疑问 1 C++位逻辑运算符的作用?	71
疑问 2 加、减、乘、除结果的数据类型和什么有关系?	72
疑问 3 使用条件运算符需要注意什么?	72
4.5 自我评估	72
第 5 章 程序流程控制	73
5.1 顺序语句	73
5.2 条件判断语句	74
5.2.1 if 条件	74
5.2.2 if-else 条件	76
5.2.3 条件运算符	77
5.3 循环语句	78
5.3.1 for 循环	78
5.3.2 while 循环	79
5.3.3 do-while 循环	81
5.4 跳出循环	82
5.4.1 continue	82
5.4.2 break	83
5.5 多重选择语句	84
5.6 小试身手——流程控制综合案例	86
5.7 疑难解惑	89
疑问 1 do-while 和 while 有什么区别?	89
疑问 2 条件语句如何嵌套? 如何匹配 else 子句?	89
疑问 3 switch 语句的执行顺序是什么?	90
5.8 自我评估	90
第 6 章 函数	91
6.1 函数的基本结构	91
6.1.1 函数的声明、定义和调用	91
6.1.2 参数的传递方式	93
6.1.3 函数的默认参数	95
6.1.4 函数的返回值	97
6.2 变量的作用域	97
6.2.1 局部变量	97
6.2.2 静态局部变量	98
6.2.3 外部变量	99
6.2.4 寄存器变量	100
6.3 特殊函数调用方式——递归调用	101
6.4 内联函数	102
6.5 预处理器	103

6.5.1	#define 预处理器	103
6.5.2	#define 的作用	104
6.5.3	const 修饰符	105
6.6	函数的重载	106
6.7	小试身手——汉诺塔问题函数	108
6.8	疑难解惑	109
	疑问 1 const 和#define 的区别?	109
	疑问 2 使用内联函数, 应该注意什么问题?	109
	疑问 3 C++中, 形参与实参有什么区别?	109
6.9	自我评估	110
第 7 章	数组与字符串	111
7.1	一维数组	111
7.1.1	一维数组的声明	111
7.1.2	数组初始化	112
7.1.3	数组的操作	113
7.2	二维数组和多维数组	115
7.2.1	二维数组的声明	115
7.2.2	二维数组的使用和存取	115
7.2.3	多维数组	117
7.3	数组与函数	119
7.3.1	一维数组作为函数的参数	119
7.3.2	传送多维数组到函数	120
7.4	字符串类	121
7.4.1	字符串的声明	121
7.4.2	字符串的输入和输出	122
7.4.3	字符串处理	123
7.5	小试身手——判断字符串回文	126
7.6	疑难解惑	128
	疑问 1 使用数组时, 如何清 0 数组?	128
	疑问 2 如何将 int 类型转化为字符串?	128
	疑问 3 C++中, 两个字符串怎么连接?	129
7.7	自我评估	129
第 8 章	指针	130
8.1	指针概述	130
8.1.1	什么是指针	130
8.1.2	为什么要用指针	131
8.1.3	指针的地址	131
8.2	指针变量	132
8.2.1	指针变量的声明	132

8.2.2 指针变量的使用	133
8.3 指针与函数	134
8.3.1 指针传送到函数中	134
8.3.2 返回值为指针的函数	135
8.3.3 函数指针	137
8.4 指针与数组	138
8.4.1 指针的算术运算	138
8.4.2 利用指针存储一维数组的元素	139
8.4.3 利用指针传输一维数组到函数中	140
8.5 指针与字符串	141
8.6 void 指针	142
8.7 指向指针的指针	142
8.8 动态内存配置	143
8.8.1 使用基本数据类型做动态配置	144
8.8.2 使用数组做动态配置	145
8.9 小试身手——判断字符串中有多少个整数	146
8.10 疑难解惑	148
疑问 1 数组指针与指针数组的区别?	148
疑问 2 指针函数和函数指针的区别?	148
疑问 3 C++中, 动态内存分配应注意什么问题?	148
8.11 自我评估	149
第 9 章 struct 和其他复合类型	150
9.1 struct	150
9.1.1 struct 的声明	150
9.1.2 struct 变量的初始化与使用	152
9.1.3 struct 数组初始化	153
9.2 将结构体变量作为函数参数	155
9.2.1 将整个结构体传送到函数	155
9.2.2 传送结构体的地址到函数	156
9.3 union	157
9.3.1 union 的定义和声明	157
9.3.2 union 类型的初始化和使用	158
9.3.3 struct 和 union 的差异	159
9.4 enum	160
9.4.1 enum 的定义和声明	160
9.4.2 enum 的初始化和使用	161
9.5 小试身手——判断字符串中有多少个整数	163
9.6 疑难解惑	165
疑问 1 C 和 C++中的 struct 有什么不同?	165

疑问 2 定义结构体类型变量要注意什么问题?	165
疑问 3 C++中共用体有什么特点?	165
9.7 自我评估	166
第 10 章 类	167
10.1 认识类	167
10.1.1 类的基本概念	167
10.1.2 类的定义	167
10.1.3 类对象的生成	169
10.1.4 类对象指针	170
10.2 成员函数	172
10.3 嵌套类	174
10.4 const 成员函数	174
10.5 类成员的访问控制	176
10.5.1 私有成员	176
10.5.2 公有成员	177
10.5.3 保护成员	179
10.6 静态成员	179
10.6.1 静态数据成员	179
10.6.2 静态成员函数	181
10.7 友元	182
10.8 小试身手——栈类的实现	184
10.9 疑难解惑	186
疑问 1 定义类要注意哪些事项?	186
疑问 2 如何选择使用类和结构?	186
疑问 3 C++中, const 成员和 const 对象的区别?	186
10.10 自我评估	186
第 11 章 构造函数和析构函数	188
11.1 构造函数初始化类对象	188
11.1.1 什么是构造函数	188
11.1.2 使用构造函数	189
11.2 析构函数清除类对象	190
11.2.1 析构函数的概念	191
11.2.2 析构函数的调用	191
11.3 默认构造函数	193
11.4 重载构造函数	194
11.4.1 重载构造函数的作用	194
11.4.2 重载构造函数的调用	194
11.5 类对象数组的初始化	195
11.5.1 类对象数组调用	196

11.5.2	类对象数组和默认构造函数.....	197
11.5.3	类对象数组和析构函数.....	199
11.6	拷贝构造函数.....	200
11.6.1	拷贝构造函数的概念.....	200
11.6.2	深拷贝和浅拷贝.....	201
11.7	小试身手——构造函数和析构函数的应用.....	203
11.8	疑难解惑.....	205
疑问 1	派生类如何初始化基类继承的成员?	205
疑问 2	基类和派生类构造函数的执行顺序是什么?	205
疑问 3	基类和派生类的析构函数的执行顺序是什么?	206
11.9	自我评估.....	206
第 12 章	运算符的重载.....	207
12.1	什么是运算符重载.....	207
12.1.1	运算符重载的形式.....	208
12.1.2	可重载的运算符.....	210
12.2	重载前置运算符和后置运算符.....	210
12.2.1	重载前置运算符.....	211
12.2.2	重载后置运算符.....	212
12.3	插入运算符和析取运算符的重载.....	214
12.3.1	插入运算符的重载.....	214
12.3.2	析取运算符的重载.....	215
12.4	常用运算符的重载.....	217
12.4.1	“<”运算符的重载.....	217
12.4.2	“+”运算符的重载.....	218
12.4.3	“=”赋值运算符的重载.....	220
12.5	小试身手——运算符重载实例.....	222
12.6	疑难解惑.....	224
疑问 1	在什么情况下使用运算符重载?	224
疑问 2	重载一元运算符时, 应该用友元函数重载吗?	224
疑问 3	是否可以用一个重载的运算符, 重载另一个运算符?	224
12.7	自我评估.....	224
第 13 章	类的继承.....	225
13.1	面向对象编程概述.....	225
13.1.1	面向对象编程的几个概念.....	225
13.1.2	面向对象编程与面向过程编程的区别.....	226
13.2	继承的基本概念.....	226
13.2.1	基类和继承类.....	227
13.2.2	简单的基础实例.....	228
13.2.3	调用父类中的构造函数.....	233

13.3 子类存取父类成员	236
13.3.1 私有成员的存取	236
13.3.2 继承与静态成员	238
13.3.3 多继承	239
13.4 小试身手——继承的应用	242
13.5 疑难解惑	243
疑问 1 在类继承中, 构造函数的执行顺序是什么?	243
疑问 2 在多继承中, 如果两个基类有同名的变量, 如何消除二义性?	243
疑问 3 类不能继承基类的哪些特征?	243
13.6 自我评估	244
第 14 章 虚函数和抽象类	245
14.1 什么是虚函数	245
14.1.1 虚函数的作用	245
14.1.2 动态绑定和静态绑定	247
14.2 抽象类与纯虚函数	249
14.2.1 定义纯虚函数	249
14.2.2 抽象类的作用	250
14.2.3 虚析构函数	252
14.3 抽象类的多重继承	253
14.4 虚函数表	254
14.4.1 什么是虚函数表	255
14.4.2 继承关系的虚函数表	256
14.5 小试身手——抽象类的应用	258
14.6 疑难解惑	260
疑问 1 虚函数在编程过程中的使用技巧?	260
疑问 2 含有纯虚函数的类是否可以被实例化?	261
疑问 3 为什么在虚函数和纯函数中不能有 static 标识符?	261
14.7 自我评估	261
第 15 章 C++中文件处理	262
15.1 文件的基本概念	262
15.1.1 文件 I/O	262
15.1.2 文件顺序读写	266
15.1.3 随机文件读写	266
15.2 文件的打开与关闭	268
15.2.1 文件的打开	268
15.2.2 文件的关闭	270
15.3 文本文件的处理	271
15.3.1 将变量写入到文件	271
15.3.2 将变量写入文件尾部	272

15.3.3 从文本文件中读入变量	272
15.3.4 使用 <code>get()</code> 、 <code>getline()</code> 和 <code>put()</code> 函数	273
15.4 二进制文件的处理	276
15.5 小试身手——文件操作	278
15.6 疑难解惑	280
疑问 1 <code>get()</code> 和 <code>getline()</code> 的区别?	280
疑问 2 缓存同步如何实现?	280
疑问 3 文件中,插入器和析取器如何定义使用?	280
15.7 自我评估	281
第 16 章 异常处理	282
16.1 异常的基本概念	282
16.2 异常处理机制	282
16.3 抛出异常	284
16.4 重新抛出异常	286
16.5 捕获所有异常	287
16.6 不是错误的异常	288
16.7 未捕捉到的异常	289
16.8 标准异常	289
16.9 异常规范	290
16.10 异常与继承	290
16.11 异常处理的应用	292
16.11.1 自定义异常类	292
16.11.2 捕获多个异常	293
16.12 小试身手——异常处理	295
16.13 疑难解惑	297
疑问 1 抛出异常而没有捕获会如何?	297
疑问 2 异常处理通过什么来匹配?	297
疑问 3 异常抛出后资源如何释放?	297
16.14 自我评估	298
第 17 章 模板与类型转换	299
17.1 模板	299
17.1.1 函数模板	300
17.1.2 类模板	301
17.1.3 模板参数	303
17.1.4 模板的特殊化	304
17.1.5 重载和函数模板	305
17.2 类型识别和强制转换运算符	307
17.2.1 运行时类型识别	307
17.2.2 强制类型转换运算符	311

17.3 小试身手——模板应用	313
17.4 疑难解惑	315
疑问 1 模板类实现有什么方法?	315
疑问 2 模板类可以继承吗?	315
疑问 3 4 种强制类型转换有什么异同?	316
17.5 自我评估	316
第 18 章 容器和迭代器	317
18.1 STL	317
18.2 迭代器	317
18.3 顺序容器	319
18.3.1 向量	319
18.3.2 双端队列	320
18.3.3 列表	322
18.4 关联容器	323
18.4.1 集合和多集	323
18.4.2 映射 (map) 和多重映射 (multimap)	325
18.5 容器适配器	326
18.5.1 栈	326
18.5.2 队列	327
18.5.3 优先级队列	329
18.6 小试身手——容器操作实例	330
18.7 疑难解惑	335
疑问 1 顺序容器和关联容器有什么区别?	335
疑问 2 什么是迭代器的范围?	335
疑问 3 STL 有哪 7 种主要容器?	335
疑问 4 deque 和 vector 的不同之处?	335
18.8 自我评估	335

第 1 章 为什么要学习 C++——认识 C++



学习目标 Objective

本章将带领读者步入 C++ 的世界，教会您用自己的双手开启 C++ 之门——创建一个应用程序，了解 C++ 程序的起源和特色，剖析 C++ 语言的编译过程，掌握 C++ 开发环境的安装以及在开发环境中如何创建一个应用程序。



内容导航 Navigation

- C++ 特点
- 语言翻译过程
- 熟悉 C++ 环境

1.1 C/C++ 的起源

要想学好 C++ 编程，了解 C/C++ 的历史演变过程是一个必需的前提，C++ 是从 C 语言发展来的，所以首先从 C 语言的历史讲起。

C 语言是由计算机科学家丹尼斯·里奇（Dennis Ritchie）创造的。在 1967 年，丹尼斯·里奇进入著名的贝尔实验室工作（C 语言、C++ 语言和 Unix 操作系统都在此诞生）。在贝尔实验室工作的过程中，里奇为了解决在工作中遇到的问题，创造了 C 语言。

为了使 C 语言更好地被应用，里奇用 C 语言将 Unix 操作系统重新写了一遍，同时发表了《可移植的 C 语言编译程序》，使 C 语言知名度大幅提高，因此各种型号的计算机都开始支持 C 语言。

在 1978 年，里奇和布朗出版了《C 语言》。该书是 C 语言的鼻祖，产生了广泛的影响，使 C 语言成为当时世界上最受欢迎的高级语言。由于里奇对计算机语言发展的卓越贡献，在 1983 年，里奇获得了计算机科学的最高荣誉——图灵奖。

人们对计算机技术追求的脚步并没有停止，C++ 随着 C 语言的发展而来。1979 年，Bjarne 博士为了分析 Unix 的内核，苦于当时没有合适的工具将 Unix 的内核模块化，于是他为 C 加上了一个类似 Simula 的机制，而贝尔实验室对 Bjarne 博士的这种创新非常感兴趣，专门为此成立了一个开发小组。

当时，这个语言并不是叫做 C++，而是叫做 C with class，它仅仅被当作 C 语言的一种补充。

下面一起来回顾一下 C++ 历史上的主要事件，如表 1-1 所示。

表1-1 C++历史上的主要事件

时间	事件
1983 年 8 月	第一个 C++实现投入使用
1983 年 12 月	Rick Mascitti 建议命名为 C lus Plus，即 C++
1985 年 2 月	第一个 C++ Release E 发布
1985 年 10 月	CFront 的第一个商业发布，CFront Release 1.0
1986 年 11 月	C++第一个商业移植 CFront 1.1,Glockenspiel
1987 年 2 月	CFront Release 1.2 发布
1987 年 11 月	第一个 USENIX C++会议在新墨西哥州举行
1988 年 10 月	第一次 USENIX C++实现者工作会议在科罗拉多州举行
1989 年 12 月	ANSI X3J16 在华盛顿组织会议
1990 年 3 月	第一次 ANSI X3J16 技术会议在新泽西州召开
1990 年 5 月	C++的又一个传世经典 ARM 诞生
1990 年 7 月	模板被加入
1990 年 11 月	异常被加入
1991 年 6 月	The C++ Programming Language 第二版完成
1991 年 6 月	第一次 ISO WG21 会议在瑞典召开
1991 年 10 月	CFront Release 3.0 发布
1993 年 3 月	运行时类型识别在俄勒冈州被加入
1993 年 7 月	名字空间在德国慕尼黑被加入
1994 年 8 月	ANSI/ISO 委员会草案登记
1997 年 7 月	The C++ Programming Language 第三版完成
1997 年 10 月	ISO 标准通过表决被接受
1998 年 11 月	ISO 标准被批准，ISO 颁布了 ISO/IEC 1488-1998
2003 年 10 月	ISO 颁布了 ISO/IEC 14882:2003
2011 年 9 月	ISO 颁布了 ISO/IEC 14882:2011

1.2 C++的特色

C++由 C 语言发展而来，继承了 C 语言的优点，同时对其也进行了大量的改进。

C++语言是一种支持面向对象的高级程序设计语言。面向对象的设计与面向过程的设计有很大区别。因此，它的一些特点也主要体现在其对面向对象编程的支持上。

(1) C++支持数据封装，支持数据封装就是支持数据抽象。在 C++中，类是支持数据封装的工具，对象则是数据封装的实现。在 C++中，将数据和对该数据进行合法操作的函数封装在一起作为一个类的定义，数据将被隐藏在封装体中，该封装体通过操作接口与外界交换信息。对象被说明具有一个给定类的变量。在 C++中，结构可作为一种特殊的类，它虽然可以包含函数，但是它没有私有或保护的成员。

(2) C++类中包含私有、公有和保护成员。C++类中可定义三种不同访问控制权限的成员。私有(Private)成员，只有在类中说明的函数才能访问该类的私有成员，而在该类外的函数不可以访问私有成员；另一种是公有 (Public) 成员，类外面也可访问公有成员，成为该类的接口；还有一