C++ 程序设计教程

C++ 程序设计系列教材

钱 能 主编

清 华 大 学 出 版 社
http://www.tup.tsinghua.edu.cn

C++ 程序设计教程

钱 能 主编

清华大学出版社

(京) 新登字 158 号

内 容 简 介

C++ 是一种高效实用的程序设计语言，它既可进行过程化程序设计，也可进行面向对象程序设计，因而成为了编程人员最广泛使用的工具。学好 C++，很容易触类旁通其他软件，C++ 架起了通向强大、易用、真正的软件开发应用的桥梁。许多高等院校已经开设了 C++ 程序设计语言课，急需一本实用的教材。本书是作者总结两年教学实践的经验写成的，适合用作大学计算机专业和非计算机专业的程序设计基础课程教材，也可供自学的读者使用。

本书共分两大部分。第一部分，第 1 章至第 10 章是基础部分，主要介绍 C++ 程序设计语言、程序结构和过程化基础。第二部分，第 11 章至第 21 章，是面向对象程序设计部分，它建立在 C++ 程序设计基础之上，讲述了面向对象程序设计方法。

版权所有，翻印必究。
本书封面贴有清华大学出版社激光防伪标签，无标签者不得销售。

图书在版编目(CIP)数据

C++ 程序设计教程/钱能主编；董灵平，张敏霞编著 . - 北京：清华大学出版社，1999.4
ISBN 7-302-03421-4

Ⅰ.C… Ⅱ. ①钱… ②董… ③ 张… Ⅲ. C 语言-程序设计-教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(1999)第 07793 号

出版者：清华大学出版社（北京清华大学[illegible]邮编：100084）
http：//www.tup.tsinghua.edu.cn
印刷者：北京市清华园胶印厂
发行者：新华书店总店北京科技发行所
开 本：787×1092 1/16 印张：30.25 字数：624 千字
版 次：1999 年 4 月第 1 版 1999 年 9 月第 4 次印刷
书 号：ISBN 7-302-03421-4/TP·1858
印 数：18001～26000
定 价：39.50 元

前　言

C++是一门高效实用的程序设计语言，它既可进行过程化程序设计，也可进行面向对象程序设计。C++语言强调对高级抽象的支持。C++实现了类的封装、数据隐藏、继承及多态，使得其代码容易维护及高度可重用。随着C++渐渐成为ANSI标准，这种新的面向对象程序设计语言迅速成为了程序员最广泛使用的工具。几乎在所有计算机研究和应用领域，都能看到C++的影子。

C++从C进化而来，是C语言的超集。C++在程序结构的本质上与C是一致的，都是用函数驱动机制实现。学过C语言，再来看C++，就会感到C++更简单和容易理解。我们说，过程化程序设计与面向对象程序设计之间并无水火不容的矛盾，面向对象程序设计是过程化程序设计的自然升华。

本书对于学过或没有学过C语言的读者都是适用的。如果学过ANSI C，则可以跳过第一部分的程序设计基础，直接阅读第二部分。在学习第二部分时，遇到某些概念不清之处，可以根据章节目录查阅第一部分的有关内容。本书配备有《C++实验指导书》和《C++习题解答》，这对自学者尤为方便。

本书适合做大学计算机专业和非计算机专业的程序设计基础课程教材。通过本书可达到以下三个培养目标：

1. 程序设计入门，领略什么是面向对象程序设计；
2. 掌握程序设计方法，领会面向对象程序设计；
3. 把握C++程序设计的灵魂，掌握面向对象程序设计的方法。

纵观当前，C++的发展领导了程序设计语言的潮流，大有取代其他程序语言之趋势。在教学上，它以其面向对象的特征和严密的类型系统而正在悄无声息地取代PASCAL；在科学计算功能上，它比Fortran更为可靠和方便；在系统软件的研究开发上，它又是上选的语种；在小规模控制应用上，C++的效率比之C毫不逊色；在大规模应用软件开发上，以Windows环境为代表的C++类库以及组件（组合类库）在迅速发展，它的触角触及各个领域。

C++编译器以Borland C++和Visual C++为典型，其版本在激烈的竞争中快速更新。那些老版本，如Microsoft C和Tubor C已经很少为人所用。即使学习C语言的人们目前也多在C++的编译环境中上机操作与调试。计算机等级考试的程序设计要求也在补充和更新。这就是C++不久即将取代C的无可抗拒的趋势与事实。C++作为程序设计基础教学也将以强有力的态势迅速取代C语言。

C++程序的集成开发器如Borland C++和Visual C++（2.0以下版），作为软件开发工具，已经不再时髦。Borland C++将不再有新版本，Borland公司的C++ Builder将取代现在的Borland C++，就像当初Borland C++取代Tubor C一样。软件开发进入了可视化程序开发环境的时代。

一方面是陈旧的软件开发工具在淘汰另一方面是新的软件开发工具在崛起。作者竭力推崇新的软件开发工具BC++ Builder 4.0和Visual C++6.0，因为它们不但是优秀的可视化

开发工具，而且由于它们是基于 C++ 的，可以充分发挥 C++ 语言强大灵活的特点，用面向对象程序设计方法去阅读、理解、思考和分析、设计以及编程。

学好 C++，很容易触类旁通其他软件，如基于 Java 语言的 Visual J++ 和 J++ builder，而且像 ActiveX SDK 和 DirectX SDK 等都是 C++ 形式的。C++ 架起了通向强大、易用、真正的软件开发应用的桥梁。计算机发展到今天，培养一个具有相当的应用软件开发能力的人才，已经不需要很长的周期了。这给我们每个人都带来了最好的机遇和最大的挑战。我们的战略是，大学一年级，学习 C++ 课程，大学二年级，在适当的指导下就可学习如 C++ Builder 和 Visual C++ 这些快速应用程序开发工具了。这意味着计算机专业的学生，大学二年级，就可以冲向计算机应用的前沿，以此与计算机其他课程的学习相辅相成。非计算机专业的学生，同样也可以在应用开发的实践中找到自己的位置。

学程序设计，不仅要在运用程序设计的思想上获得一个大的突破，也应在运用程序设计的开发工具上，走向真正的实用。这才是真正的理论联系实际，而且是事半功倍的学习方法。

本书分为两大部分。第一部分，第 1 章至第 10 章，是基础部分，主要介绍 C++ 程序设计语言，程序结构和过程化基础。第二部分，第 11 章至第 21 章，是面向对象程序设计部分，它建立在 C++ 程序设计基础之上，讲述了面向对象程序设计的方法。

本书内容特点

1. 强化重要概念

函数：函数是理解 C/C++ 语言的重要基础。在书中作了较多的描述。

程序结构：程序结构的理解与实践是从 C++ 语言的学习跨向实际应用的关键。作者在这方面也注入了较多的笔墨。

指针：指针是理解各种 C++ 语言现象的关键，透过指针能够更好地理解语言的表达和程序的工作。这本书对指针重点展开介绍。

链表：链表是数据结构的基础，也是软件设计的重要技术。作者强调链表实现技术，是想从根本上让读者领会程序设计的技术性和艺术性。

引用：引用是 C++ 特有的，C++ 面向对象的程序能够被人阅读理解，引用起了很大的作用。本书专辟一章讲述。

类：类是面向对象程序设计的首要概念。书中从第 11 章起，不停地描述类类型机制和面向对象程序设计的各项内容。

继承与虚函数：继承与虚函数是 C++ 实现类的多态性的机制。它因而也是面向对象程序设计的关键之一。理解继承与虚函数能使读者从整体上把握面向对象程序设计的方法，从而走向真正的实用。

2. 舍弃次要内容

联合与位域：本书只介绍结构而不介绍联合和位域，是考虑到这些内容相对整个程序设计的功能来说已经退化。

试读结束：需要全本请在线购买： www.ertongbook.com

多级指针：书中只介绍多级指针的概念（多维数组也如此），而不具体展开。

考虑到篇幅不讲成员指针：在《C++习题解答》中，有对成员指针的介绍，作为本书的补充。

3. 不求面面俱到，但求通俗易懂

本书不是语法参考书，是自成体系的C++教科书。它的目的是让读者通过自学或在老师讲授下，能够运用C++语言的基本要素，进行基本的结构化程序设计和面向对象的程序设计。因此全书紧紧围绕着如何进行程序设计而展开。前半部分，讲述C++程序设计基础，在此基础之上，后半部分，介绍面向对象程序设计的具体方法。

4. 以C作为背景

本书讲述的内容以ANSI C++为标准。书中出现的C内容是C++的参照。如关于C库函数printf和scanf，出现在与C++流作比较的场合，库函数malloc和free出现在与C++的new和delete作比较的场合。另外，还有C中的结构与C++中的结构之差异，C中的函数名与C++中的函数名之差异等。

考虑到准备参加C语言等级考试的读者，我们在书中对C的数学和字符串操作库函数作了适当的介绍。

本书编排特点

- 每章首均引出本章将要讲述的内容和学习要求。
- 在每章结束时，都做了小结，给出本章内容的概括性描述，指出本章在全书中的重要程度，并且指导学生进一步学习有关内容。
- 每章末安排的习题都是操作性习题，能通过上机加以验证。
- 书中遇有在理解上特别重要的内容用**黑体标记**。
- 与本书章节内容紧密联系的一些概念和说明用➡引导，另起编段。
- 本书的各章与《C++实验指导书》中的实验内容相配合，相辅相成。
- 本书的各章与《C++习题解答》中的各章一一对应。教程中没有写进去的内容有些在习题解答中得到了补充。教程中的习题有些在习题解答中有类似的题解。
- 本书强调程序的可读性。使用统一的程序设计风格，并希望读者自始至终地模仿。
- 本书强调程序的可移植性。不以某个C++编译器为标准，而以最新的ANSI C++标准为编书标准。程序中的编译错误，一般用Borland C++（简称BC）中的出错信息给出。有些BC中的警告在Visual C++（简称VC）中是编译错误或反之，则同时也给出VC的出错信息。
- 本书中包含了大量的程序例子，并附有运行结果。凡在程序开头带有程序名编号的，都是完整的程序，它们经过了上机调试，可以直接在计算机上编译运行。运行结果中，如果是手工键入的字符，则用下划线表示。

本书中所有编号的程序，读者可到Internet网清华大学出版社站点 *ftp://ftp.tsinghua.edu.cn* 上查找。

本书第2,3,4,5章由董灵平老师写稿,第6,9,10章由张敏霞老师写稿,其余由钱能老师编写,最后由钱能老师统稿。

有几十年教学经验的彭铭南老师,以其浑厚的数学功底,广博的计算机知识,始终给予作者以深切的关心和培养,书中也不时地表述着他的观点和看法。

衷心感谢C++教学研究组的胡同森老师无偿提供给作者珍贵的C++讲稿,该讲稿使作者获得了许多写作的动力和创作的灵感。胡老师还给该书内容提出了许多宝贵的建议。

宋亮和胡坚是作者的两位忠实的学生,他们阅读了全书,从写书风格、可理解性、详略程度等方面都提了许多非常宝贵的意见,这些建议大多数被作者采纳。

许多同事对编著此书的关心给了作者巨大的动力。

大量的学生频繁地问的各种各样的问题,给了作者以最大的帮助和启示,在这里深表谢意。

大学C++程序设计基础教学尚在起步,教学方法,手段和形式还在摸索中。本教材编写中难免有不足之处,我们诚恳期待读者的批评指正和建议,以供再版时参考,使本书日臻完善。

作者的电子邮件地址是:

qianneng@mail.hz.zj.cn

通信地址是:

杭州市德胜新村12幢1单元101室　钱　能　收　　(邮编310014)

目　录

第一部分　C++过程化语言基础

第1章　C++入门 …… 1
1.1　从C到C++ …… 1
1.2　程序与语言 …… 2
1.3　结构化程序设计 …… 4
1.4　面向对象程序设计 …… 5
1.5　程序开发过程 …… 6
1.6　最简单的程序 …… 7
1.7　函数 …… 8
小结 …… 11

第2章　基本数据类型与输入输出 …… 12
2.1　字符集与保留字 …… 12
2.2　基本数据类型 …… 13
2.3　变量定义 …… 15
2.4　常量 …… 17
2.5　常量定义 …… 20
2.6　I/O流控制 …… 22
2.7　printf与scanf …… 28
小结 …… 32
练习 …… 32

第3章　表达式和语句 …… 34
3.1　表达式 …… 34
3.2　算术运算和赋值 …… 36
3.3　算术类型转换 …… 38
3.4　增量和减量 …… 39
3.5　关系与逻辑运算 …… 41
3.6　if语句 …… 44
3.7　条件运算符 …… 47
3.8　逗号表达式 …… 48
3.9　求值次序与副作用 …… 49

小结 …… 50
练习 …… 51

第 4 章　过程化语句 …… 54
4.1　while 语句 …… 54
4.2　do...while 语句 …… 55
4.3　for 语句 …… 58
4.4　switch 语句 …… 60
4.5　转向语句 …… 63
4.6　过程应用：求 π …… 66
4.7　过程应用：判明素数 …… 68
4.8　过程应用：求积分 …… 71
小结 …… 73
练习 …… 74

第 5 章　函数 …… 77
5.1　函数概述 …… 77
5.2　函数原型 …… 78
5.3　全局变量与局部变量 …… 81
5.4　函数调用机制 …… 83
5.5　静态局部变量 …… 86
5.6　递归函数 …… 87
5.7　内联函数 …… 90
5.8　重载函数 …… 93
5.9　默认参数的函数 …… 95
小结 …… 97
练习 …… 98

第 6 章　程序结构 …… 100
6.1　外部存储类型 …… 100
6.2　静态存储类型 …… 102
6.3　作用域 …… 106
6.4　可见性 …… 110
6.5　生命期 …… 112
6.6　头文件 …… 113
6.7　多文件结构 …… 115
6.8　编译预处理 …… 116
小结 …… 117
练习 …… 118

第 7 章　数组 …… 120
7.1　数组定义 …… 120
7.2　访问数组元素 …… 122
7.3　初始化数组 …… 124
7.4　向函数传递数组 …… 128
7.5　二维数组 …… 130
7.6　数组应用：排序 …… 133
7.7　数组应用：Josephus 问题 …… 140
7.8　数组应用：矩阵乘法 …… 142
小结 …… 143
练习 …… 144

第 8 章　指针 …… 145
8.1　指针概念 …… 145
8.2　指针运算 …… 150
8.3　指针与数组 …… 153
8.4　堆内存分配 …… 155
8.5　const 指针 …… 159
8.6　指针与函数 …… 161
8.7　字符指针 …… 167
8.8　指针数组 …… 171
8.9　命令行参数 …… 174
8.10　函数指针 …… 177
小结 …… 182
练习 …… 182

第 9 章　引用 …… 185
9.1　引用的概念 …… 185
9.2　引用的操作 …… 186
9.3　什么能被引用 …… 188
9.4　用引用传递函数参数 …… 189
9.5　返回多个值 …… 191
9.6　用引用返回值 …… 192
9.7　函数调用作为左值 …… 196
9.8　用 const 限定引用 …… 198
9.9　返回堆中变量的引用 …… 200
小结 …… 201
练习 …… 202

第 10 章　结构 …… 204
10.1　结构 …… 204
10.2　结构与指针 …… 208
10.3　结构与数组 …… 209
10.4　传递结构参数 …… 212
10.5　返回结构 …… 214
10.6　链表结构 …… 217
10.7　创建与遍历链表 …… 219
10.8　删除链表结点 …… 222
10.9　插入链表结点 …… 224
10.10　结构应用：Josephus 问题 …… 226
小结 …… 228
练习 …… 229

第二部分　面向对象程序设计

第 11 章　类 …… 232
11.1　从结构到类 …… 232
11.2　软件方法的发展必然 …… 234
11.3　定义成员函数 …… 236
11.4　调用成员函数 …… 240
11.5　保护成员 …… 244
11.6　屏蔽类的内部实现 …… 247
11.7　再论程序结构 …… 252
小结 …… 256
练习 …… 257

第 12 章　构造函数 …… 260
12.1　类与对象 …… 260
12.2　构造函数的需要性 …… 261
12.3　构造函数的使用 …… 263
12.4　析构函数 …… 268
12.5　带参数的构造函数 …… 271
12.6　重载构造函数 …… 273
12.7　默认构造函数 …… 276
12.8　类成员初始化的困惑 …… 278
12.9　构造类成员 …… 281
12.10　构造对象的顺序 …… 285
小结 …… 288

试读结束：需要全本请在线购买：　www.ertongbook.com

练习 …… 288

第 13 章　面向对象程序设计 …… 290
13.1　抽象 …… 290
13.2　分类 …… 291
13.3　设计和效率 …… 292
13.4　讨论 Josephus 问题 …… 293
13.5　结构化方法 …… 294
13.6　结构化方法的实现 …… 296
13.7　面向对象方法 …… 298
13.8　面向对象方法的实现 …… 301
13.9　程序维护 …… 306
小结 …… 310
练习 …… 311

第 14 章　堆与拷贝构造函数 …… 312
14.1　关于堆 …… 312
14.2　需要 new 和 delete 的原因 …… 312
14.3　分配堆对象 …… 314
14.4　拷贝构造函数 …… 315
14.5　默认拷贝构造函数 …… 318
14.6　浅拷贝与深拷贝 …… 319
14.7　临时对象 …… 323
14.8　无名对象 …… 324
14.9　构造函数用于类型转换 …… 325
小结 …… 326
练习 …… 326

第 15 章　静态成员与友元 …… 330
15.1　静态成员的需要性 …… 330
15.2　静态成员的使用 …… 331
15.3　静态数据成员 …… 335
15.4　静态成员函数 …… 338
15.5　需要友元的原因 …… 341
15.6　友元的使用 …… 344
小结 …… 347
练习 …… 347

第 16 章　继承 …… 349
16.1　继承的概念 …… 349
16.2　继承的工作方式 …… 350
16.3　派生类的构造 …… 352
16.4　继承与组合 …… 354
16.5　多态性 …… 355
16.6　多态的思考方式 …… 356
16.7　多态性如何工作 …… 358
16.8　不恰当的虚函数 …… 360
16.9　虚函数的限制 …… 363
16.10　类的冗余 …… 363
16.11　克服冗余带来的问题 …… 368
16.12　类的分解 …… 371
16.13　抽象类 …… 375
16.14　由抽象类派生具体类 …… 377
16.15　纯虚函数的需要性 …… 378
小结 …… 379
练习 …… 380

第 17 章　多重继承 …… 381
17.1　多继承如何工作 …… 381
17.2　继承的模糊性 …… 382
17.3　虚拟继承 …… 383
17.4　多继承的构造顺序 …… 387
17.5　继承的访问控制 …… 389
17.6　保护继承与私有继承 …… 391
小结 …… 394
练习 …… 394

第 18 章　运算符重载 …… 396
18.1　运算符重载的需要性 …… 396
18.2　如何重载运算符 …… 397
18.3　值返回与引用返回 …… 400
18.4　运算符作成员函数 …… 402
18.5　重载增量运算符 …… 405
18.6　转换运算符 …… 408
18.7　赋值运算符 …… 410
小结 …… 413
练习 …… 414

第 19 章　I/O 流 …… 415
19.1　printf 和 scanf 的缺陷 …… 415
19.2　I/O 标准流类 …… 416
19.3　文件流类 …… 418
19.4　串流类 …… 420
19.5　控制符 …… 421
19.6　使用 I/O 成员函数 …… 424
19.7　重载插入运算符 …… 428
19.8　插入运算符与虚函数 …… 431
19.9　文件操作 …… 433
小结 …… 437
练习 …… 437

第 20 章　模板 …… 438
20.1　模板的概念 …… 438
20.2　为什么要用模板 …… 439
20.3　函数模板 …… 441
20.4　重载模板函数 …… 442
20.5　类模板的定义 …… 443
20.6　使用类模板 …… 446
20.7　使用标准模板类库：Josephus 问题 …… 447
小结 …… 449
练习 …… 450

第 21 章　异常处理 …… 451
21.1　异常的概念 …… 451
21.2　异常的基本思想 …… 452
21.3　异常的实现 …… 453
21.4　异常的规则 …… 455
21.5　多路捕捉 …… 458
21.6　异常处理机制 …… 461
21.7　使用异常的方法 …… 464
小结 …… 465
练习 …… 465

参考文献 …… 467

第一部分　C++过程化语言基础

第1章　C++入门

C++是一门优秀的程序设计语言。C++比C更容易为人们所学习和掌握，并且以其独特的语言机制在计算机科学领域中得到广泛的应用。学习本章后，要求了解C++语言的概念，了解C与C++之间的关系，了解C++语言对程序设计方法的支持，了解C++程序开发的过程，了解简单的C++程序结构，学会最简单的C++程序开发。

1.1　从C到C++

C语言是贝尔实验室的Dennis Ritchie在B语言的基础上开发出来的，1972年在一台DEC PDP—11计算机上实现了最初的C语言。C是作为UNIX操作系统的开发语言而开始广为人们所认识。实际上，当今许多新的重要的操作系统都是用C或C++编写的。在过去20年内，C语言已经能够用在绝大多数计算机上了。C语言是与硬件无关的。由于C语言的严谨设计，使得把用C语言编写的程序移植到大多数计算机上成为可能。到70年代末，C已经演化为现在所说的“传统的C语言”。Kernighan和Ritchie在1978年出版的《The C Programming Language》一书中全面地介绍了传统的C语言，这本书已经成为最成功的计算机学术著作之一。

C语言在各种计算机上的快速推广导致了许多C语言版本。这些版本虽然是类似的，但通常是不兼容的。对希望开发出的代码能够在多种平台上运行的程序开发者来说，这是他们面临的一个严重的问题。显然，人们需要一种标准的C语言版本。为了明确地定义与机器无关的C语言，1989年美国国家标准协会制定了C语言的标准(ANSI C)。Kernighan和Ritchie编著的第二版《The C programming Language》(1988年版)介绍了ANSI C的全部内容。

至此，C语言以其如下独有的特点风靡了全世界：

(1) 语言简洁、紧凑，使用方便、灵活。C语言只有32个关键字，程序书写形式自由。

(2) 丰富的运算符和数据类型。

(3) C语言可以直接访问内存地址，能进行位操作，使其能够胜任开发操作系统的工作。

(4) 生成的目标代码质量高，程序运行效率高。

(5) 可移植性好。

C 语言盛行的同时，也暴露出它的局限：

(1) C 类型检查机制相对较弱，这使得程序中的一些错误不能在编译时发现。

(2) C 本身几乎没有支持代码重用的语言结构，因此一个程序员精心设计的程序，很难为其他程序所用。

(3) 当程序的规模达到一定的程度时，程序员很难控制程序的复杂性。

为了满足管理程序的复杂性需要，1980 年，贝尔实验室的 Bjarne Stroustrup 开始对 C 进行改进和扩充。最初的成果称为“带类的 C”，1983 年正式取名为 C++，在经历了 3 次C++ 修订后，于 1994 年制定了 ANSI C++ 标准的草案。以后又经过不断完善，成为目前的 C++。C++ 仍在不断发展中。

C++ 包含了整个 C，C 是建立 C++ 的基础。C++ 包括 C 的全部特征、属性和优点，同时添加了对面向对象编程(OOP)的完全支持。

1.2 程序与语言

1. 程序

程序是以某种语言为工具编制出来的动作序列，它表达了人的思想。计算机程序是用计算机程序设计语言所要求的规范书写出来的一系列动作，它表达了程序员要求计算机执行的操作。

对于计算机来说，一组机器指令就是程序。当我们说机器代码或者机器指令时，都是指的程序，它是按计算机硬件设计规范的要求，编制出来的动作序列。

对于使用计算机的人来说，程序员用某高级语言编写的语句序列也是程序。程序通常以文件的形式保存起来。所以，源文件、源程序和源代码都是程序。

程序是任何有目的的、预想好的动作序列。它是一种软件。

计算机要运转起来，需要一整套可运行软件，即计算机程序。

学术界对程序的定义是比较严格的，这里不作详述。

2. 程序语言的发展

最早，程序员使用最原始的计算机指令，即机器语言程序。只有机器语言才能为机器所识别和运行。这些指令由一串二进制的数表示。不久，发明了汇编语言，它可以将机器指令映射为一些能被人读懂的助记符，如 ADD，SUB。程序员运行汇编程序将用助记符写成的源程序转换成机器指令，然后再运行机器指令程序，得到所要的结果。那时，编写程序的都是计算机专业人员，编写程序的语言都是低级的或较低级的。

以后，随着硬件的发展，Fortran，BASIC，Pascal、C 等几十种甚至几百种高级语言应运而生，中间经历了严酷的优胜劣汰过程，最后剩下的是一些比较优秀的高级语言。C++ 首当其冲。

多年来，计算机程序的主要目标是力求编写出短小的代码以使运行速度更快。因为硬件成本和上机运行费很高。当计算机变得更小、更廉价、速度更快时，程序员开发程序、维护

程序的费用急剧上升，而计算机硬件和运行的成本快速下降，程序设计的目标也就发生了变化。

在程序正确的前提下，可读性，易维护，可移植是程序设计首要的目标。所谓可读，就是使用良好的书写风格和易懂的语句编写程序。所谓易维护，是指当业务需求发生变化时，不需要太多的开销就可以扩展和增强程序的功能。所谓可移植，是指编写的程序在各种计算机和操作系统上都能运行，并且运行结果一样。

3. 高级语言和低级语言

C++语言是高级语言，机器语言是低级语言，汇编语言基本上是低级语言。例如：对于C++语言的语句：

```
a = 3 * a - 2 * b + 1; //3a - 2b + 1 的值赋给 a
```

写成汇编语言和对应的机器语言为：

```
mov eax, DWORD PTR a_$[ebp]          8b 45 fc
lea eax, DWORD PTR [eax + eax * 2]   8d 04 40
mov ecx, DWORD PTR b_$[ebp]          8b 4d f8
add ecx, ecx                         03 c9
sub eax, ecx                         2b c1
inc eax                              40
mov DWORD PTR a_$[ebp], eax          89 45 fc
```

第一条命令是将a放入寄存器eax中(ebp是数据段的指针，a_$是变量a的偏移位置)。

第二条命令是将eax的内容加上2倍的eax内容放到eax中，即eax中值为3 * a。

第三条命令是将b放入寄存器ecx中。

第四条命令是将ecx的内容加上ecx，即ecx中的值为2 * b。

第五条命令是将eax减去ecx的值(3 * a - 2 * b)放入eax。

第六条命令是eax的值加1。此时，eax中的值为3 * a - 2 * b + 1。

最后一条命令是将寄存器eax的值放入a变量中。即实现a = 3 * a - 2 * b + 1。

可以看出，程序语言越低级，描写程序越复杂，指令越难懂。语言越低级，就越靠近机器，越高级，就越靠近人的表达与理解。

程序语言的发展，总是从低级到高级，直到可以用人的自然语言来描述。

程序语言的发展，也是从具体到抽象的发展过程。编制一个表达式，无须将表达式的具体操作过程描述出来，否则人会感到太累，大量的精力会被无谓地浪费，无法进行更大规模的设计与思考。而低级语言则必须详尽地描述任何操作。所以抽象表达能力越强，语言越高级。

4. C与C++

C++语言包括过程性语言部分和类部分。过程性语言部分与C并无本质的差别，无非版本提高了，功能增强了。类部分是C中所没有的，它是面向对象程序设计的主体。要学习面向对象程序设计，首先必须具有过程性语言的基础。所以**学习C++，必先学习其过程**

试读结束：需要全本请在线购买： www.ertongbook.com