

你必须知道的213个 C语言问题

范立锋 李世欣 编著

213个编程新手最常遇到的C语言问题

菜鸟想问不敢开口？

扫除入门者的障碍，开辟成长捷径



人民邮电出版社
POSTS & TELECOM PRESS

你必须知道的213个 C语言问题

范立饬 李世欣 编著

人民邮电出版社

北京

图书在版编目 (C I P) 数据

你必须知道的213个C语言问题 / 范立锋, 李世欣编

著. — 北京: 人民邮电出版社, 2010.6

ISBN 978-7-115-22460-6

I. ①你… II. ①范… ②李… III. ①C语言—程序设计—问答 IV. ①TP312-44

中国版本图书馆CIP数据核字(2010)第038630号

内 容 提 要

本书精选了 213 个在 C 语言程序设计中经常遇到的问题, 目的是帮助读者解决在 C 语言学习和开发中遇到的实际困难, 提高读者学习和开发的效率。这些问题涵盖了 C 语言与软件开发、C 语言基础、编译预处理、字符串、函数、键盘操作、文件、目录和磁盘、数组、指针和结构、DOS 服务和 BIOS 服务、日期和时间、重定向 I/O 和进程命令、C 语言开发常见错误及程序调试等内容, 均是作者经过充分的调研, 从实际项目开发中总结出来的典型问题, 浓缩了作者多年从事开发工作的心得体会和经验教训, 对初学者具有重要的参考价值。书中每节都提供了程序设计的示例代码。

本书适合已经初步掌握 C 语言编程概念和用法的读者阅读。

你必须知道的 213 个 C 语言问题

◆ 编 著 范立锋 李世欣

责任编辑 蒋 佳

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

北京艺辉印刷有限公司印刷

◆ 开本: 800×1000 1/16

印张: 22

字数: 492 千字

2010 年 6 月第 1 版

印数: 1—4 000 册

2010 年 6 月北京第 1 次印刷

ISBN 978-7-115-22460-6

定价: 45.00 元

读者服务热线: (010)67132692 印装质量热线: (010)67129223

反盗版热线: (010)67171154

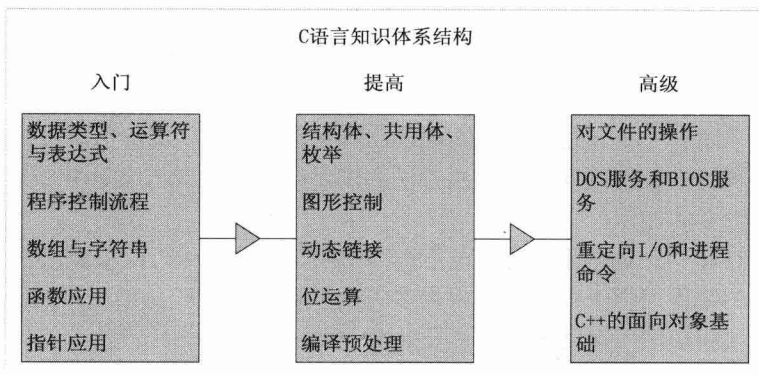
前言

C语言是目前国内外使用最广泛的程序设计语言之一。该语言具有简洁、表达能力强、使用灵活、程序执行效率高和数据结构丰富等特点,既有高级语言的特征,又涵盖了汇编语言的功能,有较强的系统处理能力。通过使用C语言,开发人员可以直接实现对操作系统硬件和外部接口的控制。

FAQ是英文“Frequently Asked Questions”的缩写,意思是“经常问到的问题”或“常见问题解答”。本书以C语言为背景,精编了213个FAQ,为读者解决学习与使用C语言时经常遇到的各种疑难,并结合实际开发给出解决这些问题的建议。

知识体系结构

C语言知识体系结构如下图所示。



本书结构

全书共分为12章,主要包含内容如下表所示。

章节名	章节内容
第1章 C语言与软件开发	针对C语言开发环境与开发流程所产生的疑惑与解答
第2章 C语言基础	针对C语言数据类型、运算符及代码方式等基础知识产生的疑惑与解答
第3章 编译预处理	针对C语言的宏定义、文件包含及条件编译等预处理命令所产生的疑惑与解答
第4章 字符串	针对C语言中的字符串长度、字符大小写等知识所产生的疑惑与解答

续表

章节名	章节内容
第5章 函数	针对 C 语言中的库函数、自定义函数应用及变量的定义等知识所产生的疑惑与解答
第6章 键盘操作	针对 C 语言中从键盘读写字符的操作、文本控制所产生的疑惑与解答
第7章 文件、目录和磁盘	针对 C 语言中对文件、目录及磁盘操作所产生的疑惑与解答
第8章 数组、指针和结构	针对 C 语言中对数组、指针及结构具体应用所产生的疑惑与解答
第9章 DOS 服务和 BIOS 服务	针对 C 语言中所属的 DOS 和 BIOS 各种服务信息操作、动态分配内存所产生的疑惑与解答
第10章 日期与时间	针对 C 语言中获取当前时间、设置 DOS 系统日期与时间及 BIOS 计时器所产生的疑惑与解答
第11章 重定向 I/O 与进程命令	针对 C 语言中重定向 I/O、进程命令及文件句柄所产生的疑惑与解答
第12章 C 语言开发常见错误及程序调试	针对 C 语言中在实际开发中遇到的常见错误、程序调试步骤的疑惑与解答

本书特点

目前市场上 C 语言相关图书很多,但以问答形式介绍 C 语言的特点与关键技术的书籍确实很少。笔者设计每个问答时,根据技术难度的不同加以标识,并给出在实际开发中的处理意见。

FAQ 的核心内容如下。

- 问题详述——将问题所涉及的背景、情况、需求及状况等信息进行详细的描述。
- 核心解答——给出问题的解决办法和满足需求的解决方案,并做适当延伸,使读者获得更多的知识。
- 疑难点评——对问题的特点进行详细说明,并给出处理此问题的注意事项。
- 良好的编程习惯——是笔者根据多年的软件研发经验总结出来的。开发时需要注意的事项和小窍门有助于提高开发效率。
- 知识链接——指明了当前 FAQ 与相关知识点的 FAQ 的链接。

FAQ 的特点如下。

- 常见问题一网打尽。
- 示例详解,深入浅出。
- 与实际开发紧密结合。

阅读方法

- 本书可以作为学习使用 C 语言开发应用程序的工具书。读者可以将开发过程中遇到的疑惑在本书中进行查询,解开疑惑的答案,达到速查速用的目的。
- 对于编码性的技术问题,本书已将实例的关键代码全部给出,读者可以通过书中的

代码了解并掌握解决问题的关键技术，最终得到书中所涉及的实例的运行效果。

读者对象

- C 语言的初学者
- C 语言编程爱好者
- 大、中专院校的老师和学生
- 对 C 语言知识点产生疑问的人群

技术支持

本书由范立锋、李世欣组织编写，参加编写的有范立锋、于琦、张明、胡晶、唐瑶、吴新伟、吕正超、程峰、黄立东、霍晶馨、曹可欣、陈艳、王毅等。在本书编写过程中我们精益求精，由于笔者能力有限，疏漏之处在所难免，敬请读者批评指正。感谢您购买本书，希望本书能够成为您的良师益友。

范立锋
2010 年 3 月

目 录

第 1 章 C 语言与软件开发	1
FAQ1.01 C 语言有哪些特点?	1
FAQ1.02 C 语言与 C++ 语言及 VC++ 比较有什么优势?	2
FAQ1.03 如何安装 Turbo C++ 3.0?	4
FAQ1.04 C 语言的编译环境有哪些?	6
FAQ1.05 如何使用 Turbo C++ 3.0 开发 C 语言程序?	7
第 2 章 C 语言基础	9
FAQ2.01 C 语言的开发流程是怎样的?	9
FAQ2.02 典型的 C 程序是怎样构成的?	11
FAQ2.03 如何在新的一行输出结果?	12
FAQ2.04 如何应对开发过程中遇到的语法错误?	14
FAQ2.05 如何理解 C 语言中的变量?	17
FAQ2.06 一个变量可以既被声明为变量又被声明为常量吗?	18
FAQ2.07 C 语言中的变量包含哪些类型? 这些类型是如何表示的?	20
FAQ2.08 如何自定义类型?	22
FAQ2.09 如何理解数据溢出?	23
FAQ2.10 什么时候可以应用类型转换? 什么时候不能应用?	25
FAQ2.11 不同类型的数据进行运算时会出现什么问题?	26
FAQ2.12 C 语言提供了哪些运算符? 运算符的优先级和结合性是怎样的?	27
FAQ2.13 如何理解 C 语言中的头文件?	30
FAQ2.14 为什么需要加入程序注释?	31
FAQ2.15 声明的变量和定义的变量有什么不同之处?	32
FAQ2.16 什么情况下要用到 switch 语句? 如何使用 switch 语句?	34
FAQ2.17 在一个 switch 语句中, default 语句是否必须存在呢?	36
FAQ2.18 for 语句的 3 个子语句是否都是必须存在的?	37
FAQ2.19 如何区分 break 和 continue?	39
FAQ2.20 如何使用 goto 语句提高程序灵活性?	41

FAQ2.21	“&”与“&&”，“ ”与“ ”有什么区别？	42
FAQ2.22	已经有了for循环，为什么还要用while循环？	43
FAQ2.23	如何强制操作符的运算顺序？	45
第3章	编译预处理	46
FAQ3.01	如何理解C语言中的宏？如何使用宏？	46
FAQ3.02	标准的预定义宏包括哪些？	50
FAQ3.03	如何改变预处理器的行计数？	51
FAQ3.04	宏与函数有什么区别？	53
FAQ3.05	如何自定义头文件？	54
FAQ3.06	头文件都包含哪些信息？	55
FAQ3.07	文件包含命令可以嵌套吗？	56
FAQ3.08	如何避免多次包含同一个文件？	57
FAQ3.09	除了.h文件以外其他文件能被#include命令所包含吗？	58
FAQ3.10	#include<文件名>和#include“文件名”有何不同？	59
FAQ3.11	如何进行条件编译预处理？	60
FAQ3.12	如何创建自定义宏？	63
FAQ3.13	宏有类型吗？	64
FAQ3.14	如何重写一个定义好的宏？	65
FAQ3.15	使用枚举和使用#define定义常量有什么不同？	66
第4章	字符串	69
FAQ4.01	C语言是如何存储字符串的？	69
FAQ4.02	如何判断字符串的长度？	70
FAQ4.03	如何判断两个字符串是否相同？	72
FAQ4.04	如何将一个字符串的内容追加到另一个字符串中？	73
FAQ4.05	如何为字符串追加N个字符？	75
FAQ4.06	如何将一个字符串复制到另一个字符串中？	76
FAQ4.07	如何在比较字符串时忽略字符大小写？	77
FAQ4.08	如何转换字符串中字符的大小写？	79
FAQ4.09	如何获取字符串中首次与末次出现某个字符的位置？	81
FAQ4.10	如何计算一个字符在字符串中出现的次数？	83
FAQ4.11	如何将字符串转换为数字？	84
FAQ4.12	如何将数字转换为字符串？	86
FAQ4.13	如何判断字符是何种类型？	89

第 5 章 函数	91
FAQ5.01 如何理解 C 语言中的函数?	91
FAQ5.02 如何理解函数原型?	92
FAQ5.03 形参和实参分别是什么? 如何使用它们?	94
FAQ5.04 如何解决自定义函数与库函数命名冲突问题?	96
FAQ5.05 如何理解函数的开销问题?	97
FAQ5.06 主调函数如何调用被调函数?	99
FAQ5.07 return 和 exit 有什么不同之处?	101
FAQ5.08 局部变量和全局变量有何区别?	102
FAQ5.09 当局部变量与全局变量发生名称冲突时如何解决?	103
FAQ5.10 如何更好地定义全局变量的有效范围?	105
FAQ5.11 如何理解传值调用?	106
FAQ5.12 C 语言支持传址调用吗?	107
FAQ5.13 为什么要用到静态变量? 静态变量何时被初始化?	108
FAQ5.14 如何理解递归函数? 什么情况下要用到递归?	110
FAQ5.15 使用递归函数时对程序的执行效率有何影响?	112
FAQ5.16 如何使用其他方法代替递归?	113
FAQ5.17 函数如何对字符串进行堆栈处理?	115
FAQ5.18 如何使用外部变量及外部静态变量?	116
FAQ5.19 如何调用结构和基指针?	118
FAQ5.20 如何在 C 程序中调用汇编语言函数并获得汇编语言函数返回值?	119
FAQ5.21 如何创建支持多参数多类型的函数?	121
FAQ5.22 内部函数和外部函数有什么不同?	124
第 6 章 键盘操作	126
FAQ6.01 如何从键盘读入字符?	126
FAQ6.02 如何使用缓冲输入?	127
FAQ6.03 如何使用直接 I/O 读入字符?	128
FAQ6.04 如何实现不显示字符的键盘输入?	130
FAQ6.05 如何实现直接输出?	131
FAQ6.06 如何将按键放回键盘缓存?	132
FAQ6.07 为什么直接 I/O 能够更快地输出字符串?	134
FAQ6.08 如何更快地从键盘输入字符串?	135
FAQ6.09 如何实现定位光标的屏幕输出?	137

FAQ6.10	如何在屏幕中插入空行？	138
FAQ6.11	如何将屏幕上的文本复制到缓冲区？	140
FAQ6.12	如何判断文本模式设置？	142
FAQ6.13	如何控制文本颜色？	144
FAQ6.14	如何指定背景颜色？	145
FAQ6.15	如何控制文本的显示亮度？	147
FAQ6.16	如何在屏幕上移动文本？	148
第 7 章	文件、目录和磁盘	150
FAQ7.01	如何理解 FILE 结构？	150
FAQ7.02	如何打开文件？如何关闭文件？	151
FAQ7.03	如何实现每次读/写文件信息的一个字符？	153
FAQ7.04	如何判断当前文件位置？	155
FAQ7.05	文本模式和二进制模式有什么区别？	156
FAQ7.06	如何使用低级和高级文件的 I/O？	158
FAQ7.07	如何理解文件句柄？	159
FAQ7.08	进程文件表有什么作用？	160
FAQ7.09	如何获取进程文件表的入口？	160
FAQ7.10	如何获取并显示系统文件表的信息？	163
FAQ7.11	如何从流指针中导出文件句柄？	163
FAQ7.12	如何重命名文件？	165
FAQ7.13	如何删除文件？	166
FAQ7.14	如何判断程序访问文件？	167
FAQ7.15	如何设置文件访问模式？	169
FAQ7.16	如何检测文件流错误？	171
FAQ7.17	如何判断文件长度？	171
FAQ7.18	如何使用临时文件？	173
FAQ7.19	如何搜索环境入口的子目录？	174
FAQ7.20	为什么要尽量减少文件的 I/O 操作？	175
FAQ7.21	对目录的操作有哪些？如何实现？	176
FAQ7.22	如何删除目录树？	177
FAQ7.23	如何列出一个目录中的所有文件？	178
FAQ7.24	如何建立完全路径名？	181
FAQ7.25	如何分解目录路径？	182
FAQ7.26	如何使用低级函数打开和关闭文件？	184

FAQ7.27	如何打开 20 个以上的文件？	186
FAQ7.28	如何改变文件长度？	187
FAQ7.29	如何控制文件打开操作的读写模式？	188
FAQ7.30	如何将缓冲区赋给文件？	190
FAQ7.31	如何分配文件缓冲区？	191
FAQ7.32	如何创建唯一文件名？	193
FAQ7.33	如何从文件流中读取结构数据？	194
FAQ7.34	如何复制文件句柄？	195
FAQ7.35	如何强制文件句柄设置？	197
FAQ7.36	如何实现文件共享？	198
FAQ7.37	如何锁定文件内容？	199
FAQ7.38	textcopy 是否能够复制二进制文件？	201
FAQ7.39	如何读取格式化的文件数据？	202
FAQ7.40	如何重新打开文件流？	203
第 8 章	数组、指针和结构	205
FAQ8.01	数组的下标总是从零开始吗？	205
FAQ8.02	越界的数组元素是否依然有效？	207
FAQ8.03	浏览数组元素时，使用指针和使用数组下标有什么区别？	209
FAQ8.04	为什么不能将数组大小初始化为一个常量？	210
FAQ8.05	数组与动态存储孰优孰劣？	212
FAQ8.06	如何理解多维数组？	214
FAQ8.07	C 语言是如何存放多维数组的？	216
FAQ8.08	可以在程序运行时才去声明数组的长度吗？	218
FAQ8.09	如何使用结构数组？	219
FAQ8.10	如何理解联合？	221
FAQ8.11	使用联合是否能够节省内存？	222
FAQ8.12	如何使用位字段结构？	224
FAQ8.13	是否可以对指针进行类型转换？	226
FAQ8.14	两次释放同一指针会产生什么结果？	227
FAQ8.15	指针占用的内存空间是否与基类型有关？	229
FAQ8.16	什么是空指针？哪些情况会用到空指针？	229
FAQ8.17	使用指针变量操作字符串和使用字符数组操作字符串有什么不同？	231
FAQ8.18	如何将指针操作作为函数参数？	233
FAQ8.19	指针函数和函数指针分别是什么？	234

FAQ8.20	指针如何进行运算?	237
FAQ8.21	如何将指针作为函数返回值?	238
FAQ8.22	如何使用指向字符串指针的指针?	240
FAQ8.23	最多可以使用多少级指针?	242
FAQ8.24	为什么使用结构? 如何声明结构?	243
FAQ8.25	C 语言如何为结构分配内存空间?	246
FAQ8.26	free()函数如何决定到底释放多大的内存空间?	247
FAQ8.27	如何使用结构作为函数参数?	249
FAQ8.28	如何使用指向结构体的指针?	251
FAQ8.29	结构体和共用体有哪些异同点?	253
第 9 章	DOS 服务和 BIOS 服务	256
FAQ9.01	如何理解 DOS 服务和 BIOS 服务?	256
FAQ9.02	如何理解寄存器?	258
FAQ9.03	如何理解软件中断?	260
FAQ9.04	如何使用 BIOS 访问指针?	261
FAQ9.05	如何暂时挂起程序?	263
FAQ9.06	如何控制声音?	264
FAQ9.07	如何应用 BIOS 键盘服务?	265
FAQ9.08	如何获取 BIOS 设备列表?	268
FAQ9.09	如何控制串行接口的 I/O?	270
FAQ9.10	如何判断 BIOS 常规内存数量?	272
FAQ9.11	如何分配动态内存?	273
FAQ9.12	动态分配的内存空间会被自动释放吗?	274
FAQ9.13	malloc()与 calloc()函数的区别?	276
FAQ9.14	如何解决 64KB 堆的限制?	277
FAQ9.15	如何从堆栈中分配内存?	278
FAQ9.16	如何改变被分配内存区域的大小?	280
第 10 章	日期与时间	282
FAQ10.01	如何使用单个数字存储日期信息? 必须遵循什么标准?	282
FAQ10.02	如何获取当前的日期与时间?	284
FAQ10.03	如何判断程序的耗时?	286
FAQ10.04	如何设置 DOS 系统时间与系统日期?	289
FAQ10.05	如何读取 BIOS 计时器?	290

FAQ10.06	如何获取与设置系统日期以及系统时间?	292
FAQ10.07	如何处理日期与字符串之间的转换?	293
FAQ10.08	如何创建格式化日期与时间串?	294
第 11 章	重定向 I/O 与进程命令行	297
FAQ11.01	如何编写密码函数?	297
FAQ11.02	如何使用输入/输出重定向?	299
FAQ11.03	如何使用管道运算符?	300
FAQ11.04	如何自定义 more 命令?	301
FAQ11.05	如何防止 I/O 重定向?	303
FAQ11.06	如何应用 STDPRN 文件句柄?	304
FAQ11.07	如何将重定向输出分割到一个文件中?	305
FAQ11.08	如何应用 STDAUX 文件句柄?	306
FAQ11.09	如何使用命令行变元?	307
FAQ11.10	如何从命令行中显示文件内容?	309
FAQ11.11	如何创建定时的 more 命令?	310
FAQ11.12	如何在重定向输入内寻找字符串?	312
FAQ11.13	如何指定重定向输入显示行数?	313
FAQ11.14	如何定义在程序结束时执行的函数?	314
第 12 章	C 语言开发常见错误及程序调试	316
FAQ12.01	使用 C 语言开发会遇到哪些常见错误?	316
FAQ12.02	程序调试包括哪几步?	320
FAQ12.03	如何使用编译工具找出错误信息对应代码位置?	321
FAQ12.04	如何检测内存漏洞?	323
FAQ12.05	如何让程序发送失败报告?	324
FAQ12.06	哪些原因会导致运行的程序挂起?	326
FAQ12.07	没有声明函数原型会造成怎样的结果?	329
FAQ12.08	函数参数的个数有限制吗?	332
FAQ12.09	exit()函数与 return 语句有什么不同吗?	334
FAQ12.10	return 语句是必须存在的吗?	336
FAQ12.11	退出 main()函数就意味着程序运行的结束吗?	339

第 1 章

C 语言与软件开发

C 语言作为一种基本的程序设计语言，是开发人员必须掌握的基本功。掌握 C 语言，不仅可以对结构化的编程有全面的了解，而且能够深入理解操作系统的运作方式及硬件编程控制方式等。本章解答与 C 语言的特点和优势相关的问题。

FAQ1.01 C 语言有哪些特点？

📖 难度系数：★★

📖 问题几率：90%

问题详述

C 语言之所以能存在与发展，是因为其具有不同于其他语言的特点。那么 C 语言具有哪些特点呢？

核心解答

C 语言既具有高级语言的特点，又具有汇编语言的特点，因此该语言是一种应用非常广泛的程序设计语言。下面将详细介绍 C 语言的特点。

1. 灵活方便、简洁、紧凑

C 语言有 32 个关键字和 9 种控制语句，程序书写自由。关键字主要使用小写字母表示。它能够将高级语言的基本结构和语句与低级语言的实用性结合起来。

C 语言可以像汇编语言一样对位、字节和地址进行操作，这三者都是计算机最基本的工作单元。

2. 运算符丰富

C 语言有 34 个运算符。C 语言将括号、赋值和强制类型转换等都作为运算符来处理，这样使得运算符类型极其丰富和表达式类型多样化。灵活使用 C 语言的各种运算符可以实现其他高级语言中难以实现的运算。

3. C语言是结构化语言

结构化语言一个显著的特点就是代码及数据的分隔化,即程序的各个部分除了必要的信息结合外彼此之间是独立的。这种结构化的形式使程序层次清晰,便于维护和调试。C语言是以函数为单位的,这些函数可以使用多种控制语句,从而使程序完全面向结构化。

4. 数据结构丰富

C语言的数据类型有整型、实型、字符型、数组类型、指针类型、结构体类型和共用体类型等,能用来实现各种复杂的数据类型的运算,同时引入指针的概念,使程序运行的效率更高。

另外,C语言具有强大的图形功能,支持多种显示器和驱动器,且计算功能非常强大。

5. C语言的语法限制不是很严格,程序设计的自由度大

一般的高级语言的语法检查比较严格,能够检查出几乎所有的语法错误,而C语言与其相反,允许程序有较大的自由度。

6. C语言允许直接访问物理地址,可以直接对硬件进行操作

C语言能实现汇编语言的大部分功能,可以用来编写系统软件。

7. C语言具有可移植性

使用C语言编写的程序可移植性好(与汇编语言相比),基本上不用修改就能应用于各种型号的计算机和各种操作系统。

疑难点评

C语言应用非常广泛,许多高级软件都是使用C语言来编写的,这是因为C语言的可移植性好和硬件控制能力非常高,同时表达和运算能力非常强。以前只能用汇编语言处理的问题,现在都可以使用C语言解决。

FAQ1.02 C语言与C++语言及VC++比较有什么优势?

📖 难度系数: ★★

📖 问题几率: 80%

问题详述

C语言和C++语言可以使用同一种集成开发环境(例如VC++),而VC++是使用C++语言的集成开发环境。C++语言是对C语言的扩展,并且C++语言能够使用C语言的所有函数。那么C语言与C++语言及VC++比较有什么优势呢?

核心解答

C 语言与 C++语言及 VC++的主要区别如下。

(1) C 语言和 C++语言是两种不同的、互相独立的语言, C++语言是 C 语言的扩展, 但 C++语言并不依赖 C 语言。C 语言既是高级语言, 也可以看作是低级语言; C++语言是具有国际标准的面向对象编程语言; 而 VC++ (Visual C++) 是一种用 C++语言作为编程语言的可视化软件开发工具, 它不仅是一个 C++编译器, 而且是一个基于 Windows 操作系统的可视化集成开发环境 IDE。

(2) 在 C 语言的函数库中, 不同功能的函数被放置在不同的头文件中。但是 C++语言增加了很多 C 语言中没有的函数, 例如 Windows 系统的大多数 API 与 C++语言中 API 的组合。VC++包含了 MFC (微软基础类, Microsoft Foundation Classes), MFC 将传统的 API 进行分类封装, 并创建了程序的一般框架。

(3) VC++对 C++标准还不能够完全兼容, 因为 VC++并不严格遵循 C++语言标准, 但 VC++语言对 C 语言是完全兼容的, 并且 VC++对 C 语言标准有很多扩展。

(4) C 语言一般应用于操作系统、硬件驱动; C++语言多数应用于游戏软件的开发; VC++专门用来开发 Windows 程序。

C 语言与 C++及 VC++比较有以下几个方面的优势。

1. 代码复用

C 语言使用函数库或者 DLL 方式实现代码复用, 在接口稳定的前提下实现内部修改和数据及其实现的封装。C++提供了类库和多种继承机制, 从而实现了具有层次的代码复用。它同时通过重载等各种机制将进一步实现复用功能, 使得类库和代码更加容易维护, 虽然建立类库在人员、组织等各个方面还是比较麻烦的。

2. 效率

C 语言在本质上擅长底层接口的编写并且非常注重效率问题; 但是事物总是具备矛盾的两面, 过于偏重效率和软件危机的出现, 反而增加了程序设计的难度。从现实世界角度考虑, OOA 更加贴近实际, 使得代码或程序更具备稳定性、可扩展性和可维护性。

3. 简单性

通常情况下, 在应用程序中 C 语言更多的要求是设计简单化。例如, C 语言不支持类似于继承的重用, 因此为了达到同样的目的不得不更多地使用组合, 即使使用函数指针实现多态, 也不可能像 C++那样把类逐层地扩展。

4. 在操作系统底层的应用

在操作系统底层一般使用 C 语言而不会使用 C++语言。例如, 一个内核程序在得到一个自旋锁后调用 C++语言的一些特征语句, 由于封装, 其内部存在内存分配 (假设已经将 new/delete 重载到内核内存分配函数上), 因此可能导致进入睡眠状态, 这在操作系统中是绝对不允许的。

5. 移植性

C语言适用于各种操作系统,例如DOS、UNIX操作系统,也适用于多种机型,因此C语言的移植性高。

疑难点评

使用C语言、C++语言及VC++开发实际项目时,通常使用的是Visual C++ 6.0和C++ Builder 6.0两种比较经典的开发工具。程序员应该根据实际需要选择不同的开发工具。下面分别介绍这两种开发工具。

- Visual C++ 6.0: 具有稳定而且强大的IDE开发环境,具有丰富的调试功能,定制宏的功能也是该工具的一个显著优势,但Visual C++ 6.0对C++标准支持的程度不够好。
- C++ Builder 6.0: 也是一个功能强大的IDE开发环境;与Visual C++ 6.0相比,速度稍慢,稳定性稍弱,但对C++标准支持的程度较好。

良好的编程习惯

建议读者在学习C语言时应该做到以下几点。

- (1) 学习C语言的编程思想。
- (2) 能够将逻辑推理转换为代码。例如“判断”在C语言中对应为代码“if”语句。
- (3) 掌握C语言基础(关键字、函数……)。

FAQ1.03 如何安装 Turbo C++ 3.0?

📖 难度系数: ★★

📖 问题几率: 90%

问题详述

在C语言的发展过程中,其开发工具也发展迅速,到目前为止Turbo C++ 3.0是C语言常用的编译工具,那么如何安装Turbo C++3.0呢?

核心解答

对于Turbo C++ 3.0来说,只需要修改一个设置选项,就能够在同一个IDE集成开发环境下设计和编译以标准C语言设计的程序。具体安装步骤如下。

- (1) 在Turbo C++ 3.0安装文件夹中,双击“INSTALL.EXE”可执行文件,系统将弹出Turbo C++ 3.0安装界面1。该界面主要显示安装Turbo C++ 3.0的欢迎信息,如图1-1所示。
- (2) 按下【Enter】键,将进入Turbo C++ 3.0安装界面2。该界面主要显示选择安装