

<<你必须知道的213个C语言问题>>

图书基本信息

书名：<<你必须知道的213个C语言问题>>

13位ISBN编号：9787115224606

10位ISBN编号：7115224609

出版时间：2010-6

出版时间：人民邮电出版社

作者：范立锋，李世欣 编著

页数：340

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<你必须知道的213个C语言问题>>

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。

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。

<<你必须知道的213个C语言问题>>

内容概要

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>>

书籍目录

- 第1章 C语言与软件开发 FAQ1.01 C语言有哪些特点？
 - FAQ1.02 C语言与C++语言及VC++比较有什么优势？
 - FAQ1.03 如何安装Turbo C++ 3.0？
 - FAQ1.04 C语言的编译环境有哪些？
 - FAQ1.05 如何使用Turbo C++ 3.0开发C语言程序？
- 第2章 C语言基础 FAQ2.01 C语言的开发流程是怎样的？
 - FAQ2.02 典型的C程序是怎样构成的？
 - FAQ2.03 如何在新的一行输出结果？
 - FAQ2.04 如何应对开发过程中遇到的语法错误？
 - FAQ2.05 如何理解C语言中的变量？
 - FAQ2.06 一个变量可以既被声明为变量又被声明为常量吗？
 - 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？
 - FAQ2.08 如何自定义类型？
 - FAQ2.09 如何理解数据溢出？
 - FAQ2.10 什么时候可以应用类型转换？
- 什么时候不能应用？
 - FAQ2.11 不同类型的数据进行运算时会出现什么问题？
 - 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？
 - FAQ2.13 如何理解C语言中的头文件？
 - FAQ2.14 为什么需要加入程序注释？
 - FAQ2.15 声明的变量和定义的变量有什么不同之处？
 - FAQ2.16 什么情况下要用到switch语句？
- 如何使用switch语句？
 - FAQ2.17 在一个switch语句中，default语句是否必须存在呢？
 - FAQ2.18 for语句的3个子语句是否都是必须存在的？
 - FAQ2.19 如何区分break和continue？
 - FAQ2.20 如何使用goto语句提高程序灵活性？
 - FAQ2.21 “&”与“&&”，“|”与“||”有什么区别？
 - FAQ2.22 已经有了for循环，为什么还要用while循环？
 - FAQ2.23 如何强制操作符的运算顺序？
- 第3章 编译预处理 FAQ3.01 如何理解C语言中的宏？
- 如何使用宏？
 - FAQ3.02 标准的预定义宏包括哪些？
 - FAQ3.03 如何改变预处理器的行计数？
 - FAQ3.04 宏与函数有什么区别？
 - FAQ3.05 如何自定义头文件？
 - FAQ3.06 头文件都包含哪些信息？
 - FAQ3.07 文件包含命令可以嵌套吗？
 - FAQ3.08 如何避免多次包含同一个文件？
 - FAQ3.09 除了.h文件以外其他文件能被#include命令所包含吗？
 - FAQ3.10 #include文件名和#include“文件名”有何不同？
 - FAQ3.11 如何进行条件编译预处理？
 - FAQ3.12 如何创建自定义宏？

<<你必须知道的213个C语言问题>>

FAQ3.13 宏有类型吗？

FAQ3.14 如何重写一个定义好的宏？

FAQ3.15 使用枚举和使用#define定义常量有什么不同？

第4章 字符串 FAQ4.01 C语言是如何存储字符串的？

FAQ4.02 如何判断字符串的长度？

FAQ4.03 如何判断两个字符串是否相同？

FAQ4.04 如何将一个字符串的内容追加到另一个字符串中？

FAQ4.05 如何为字符串追加N个字符？

FAQ4.06 如何将一个字符串复制到另一个字符串中？

FAQ4.07 如何在比较字符串时忽略字符大小写？

FAQ4.08 如何转换字符串中字符的大小写？

FAQ4.09 如何获取字符串中首次与末次出现某个字符的位置？

FAQ4.10 如何计算一个字符在字符串中出现的次数？

FAQ4.11 如何将字符串转换为数字？

FAQ4.12 如何将数字转换为字符串？

FAQ4.13 如何判断字符是何种类型？

第5章 函数 FAQ5.01 如何理解C语言中的函数？

FAQ5.02 如何理解函数原型？

FAQ5.03 形参和实参分别是什么？

如何使用它们？

FAQ5.04 如何解决自定义函数与库函数命名冲突问题？

FAQ5.05 如何理解函数的开销问题？

FAQ5.06 主调函数如何调用被调函数？

FAQ5.07 return和exit有什么不同之处？

FAQ5.08 局部变量和全局变量有何区别？

FAQ5.09 当局部变量与全局变量发生名称冲突时如何解决？

FAQ5.10 如何更好地定义全局变量的有效范围？

FAQ5.11 如何理解传值调用？

FAQ5.12 C语言支持传址调用吗？

FAQ5.13 为什么要用到静态变量？

静态变量何时被初始化？

FAQ5.14 如何理解递归函数？

什么情况下要用到递归？

FAQ5.15 使用递归函数时对程序的执行效率有何影响？

FAQ5.16 如何使用其他方法代替递归？

FAQ5.17 函数如何对字符串进行堆栈处理？

FAQ5.18 如何使用外部变量及外部静态变量？

FAQ5.19 如何调用结构和基指针？

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？

FAQ5.21 如何创建支持多参数多类型的函数？

FAQ5.22 内部函数和外部函数有什么不同？

第6章 键盘操作 FAQ6.01 如何从键盘读入字符？

FAQ6.02 如何使用缓冲输入？

FAQ6.03 如何使用直接I/O读入字符？

FAQ6.04 如何实现不显示字符的键盘输入？

FAQ6.05 如何实现直接输出？

FAQ6.06 如何将按键放回键盘缓存？

<<你必须知道的213个C语言问题>>

FAQ6.07 为什么直接I/O能够更快地输出字符串？

FAQ6.08 如何更快地从键盘输入字符串？

FAQ6.09 如何实现定位光标的屏幕输出？

FAQ6.10 如何在屏幕中插入空行？

FAQ6.11 如何将屏幕上的文本复制到缓冲区？

FAQ6.12 如何判断文本模式设置？

FAQ6.13 如何控制文本颜色？

FAQ6.14 如何指定背景颜色？

FAQ6.15 如何控制文本的显示亮度？

FAQ6.16 如何在屏幕上移动文本？

第7章 文件、目录和磁盘 FAQ7.01 如何理解FILE结构？

FAQ7.02 如何打开文件？

如何关闭文件？

FAQ7.03 如何实现每次读/写文件信息的一个字符？

FAQ7.04 如何判断当前文件位置？

FAQ7.05 文本模式和二进制模式有什么区别？

FAQ7.06 如何使用低级和高级文件的I/O？

FAQ7.07 如何理解文件句柄？

FAQ7.08 进程文件表有什么作用？

FAQ7.09 如何获取进程文件表的入口？

FAQ7.10 如何获取并显示系统文件表的信息？

FAQ7.11 如何从流指针中导出文件句柄？

FAQ7.12 如何重命名文件？

FAQ7.13 如何删除文件？

FAQ7.14 如何判断程序访问文件？

FAQ7.15 如何设置文件访问模式？

FAQ7.16 如何检测文件流错误？

FAQ7.17 如何判断文件长度？

FAQ7.18 如何使用临时文件？

FAQ7.19 如何搜索环境入口的子目录？

FAQ7.20 为什么要尽量减少文件的I/O操作？

FAQ7.21 对目录的操作有哪些？

如何实现？

FAQ7.22 如何删除目录树？

FAQ7.23 如何列出一个目录中的所有文件？

FAQ7.24 如何建立完全路径名？

FAQ7.25 如何分解目录路径？

FAQ7.26 如何使用低级函数打开和关闭文件？

FAQ7.27 如何打开20个以上的文件？

FAQ7.28 如何改变文件长度？

FAQ7.29 如何控制文件打开操作的读写模式？

FAQ7.30 如何将缓冲区赋给文件？

FAQ7.31 如何分配文件缓冲区？

FAQ7.32 如何创建唯一文件名？

FAQ7.33 如何从文件流中读取结构数据？

FAQ7.34 如何复制文件句柄？

FAQ7.35 如何强制文件句柄设置？

<<你必须知道的213个C语言问题>>

- FAQ7.36 如何实现文件共享？
- FAQ7.37 如何锁定文件内容？
- FAQ7.38 textcopy是否能够复制二进制文件？
- FAQ7.39 如何读取格式化的文件数据？
- FAQ7.40 如何重新打开文件流？

第8章 数组、指针和结构 FAQ8.01 数组的下标总是从零开始吗？

- FAQ8.02 越界的数组元素是否依然有效？
- 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？
- FAQ8.04 为什么不能将数组大小初始化为一个常量？
- FAQ8.05 数组与动态存储孰优孰劣？
- FAQ8.06 如何理解多维数组？
- FAQ8.07 C语言是如何存放多维数组的？
- FAQ8.08 可以在程序运行时才去声明数组的长度吗？
- FAQ8.09 如何使用结构数组？
- FAQ8.10 如何理解联合？
- FAQ8.11 使用联合是否能够节省内存？
- FAQ8.12 如何使用位字段结构？
- FAQ8.13 是否可以对指针进行类型转换？
- FAQ8.14 两次释放同一指针会产生什么结果？
- FAQ8.15 指针占用的内存空间是否与基类型有关？
- FAQ8.16 什么是空指针？

哪些情况会用到空指针？

- 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？
- FAQ8.18 如何将指针操作作为函数参数？
- FAQ8.19 指针函数和函数指针分别是什么？
- FAQ8.20 指针如何进行运算？
- FAQ8.21 如何将指针作为函数返回值？
- FAQ8.22 如何使用指向字符串指针的指针？
- FAQ8.23 最多可以使用多少级指针？
- FAQ8.24 为什么使用结构？

如何声明结构？

- FAQ8.25 C语言如何为结构分配内存空间？
- FAQ8.26 free()函数如何决定到底释放多大的内存空间？
- FAQ8.27 如何使用结构作为函数参数？
- FAQ8.28 如何使用指向结构体的指针？
- FAQ8.29 结构体和共用体有哪些异同点？

第9章 DOS服务和BIOS服务 FAQ9.01 如何理解DOS服务和BIOS服务？

- FAQ9.02 如何理解寄存器？
- FAQ9.03 如何理解软件中断？
- FAQ9.04 如何使用BIOS访问指针？
- FAQ9.05 如何暂时挂起程序？
- FAQ9.06 如何控制声音？
- FAQ9.07 如何应用BIOS键盘服务？
- FAQ9.08 如何获取BIOS设备列表？
- FAQ9.09 如何控制串行接口的I/O？
- FAQ9.10 如何判断BIOS常规内存数量？
- FAQ9.11 如何分配动态内存？

<<你必须知道的213个C语言问题>>

FAQ9.12 动态分配的内存空间会被自动释放吗？

FAQ9.13 malloc()与calloc()函数的区别？

FAQ9.14 如何解决64KB堆的限制？

FAQ9.15 如何从堆栈中分配内存？

FAQ9.16 如何改变被分配内存区域的大小？

第10章 日期与时间 FAQ10.01 如何使用单个数字存储日期信息？
必须遵循什么标准？

FAQ10.02 如何获取当前的日期与时间？

FAQ10.03 如何判断程序的耗时？

FAQ10.04 如何设置DOS系统时间与系统日期？

FAQ10.05 如何读取BIOS计时器？

FAQ10.06 如何获取与设置系统日期以及系统时间？

FAQ10.07 如何处理日期与字符串之间的转换？

FAQ10.08 如何创建格式化日期与时间串？

第11章 重定向I/O与进程命令行 FAQ11.01 如何编写密码函数？

FAQ11.02 如何使用输入/输出重定向？

FAQ11.03 如何使用管道运算符？

FAQ11.04 如何自定义more命令？

FAQ11.05 如何防止I/O重定向？

FAQ11.06 如何应用STDPRN文件句柄？

FAQ11.07 如何将重定向输出分割到一个文件中？

FAQ11.08 如何应用STDAUX文件句柄？

FAQ11.09 如何使用命令行变元？

FAQ11.10 如何从命令行中显示文件内容？

FAQ11.11 如何创建定时的more命令？

FAQ11.12 如何在重定向输入内寻找字符串？

FAQ11.13 如何指定重定向输入显示行数？

FAQ11.14 如何定义在程序结束时执行的函数？

第12章 C语言开发常见错误及程序调试 FAQ12.01 使用C语言开发会遇到哪些常见错误？

FAQ12.02 程序调试包括哪几步？

FAQ12.03 如何使用编译工具找出错误信息对应代码位置？

FAQ12.04 如何检测内存漏洞？

FAQ12.05 如何让程序发送失败报告？

FAQ12.06 哪些原因会导致运行的程序挂起？

FAQ12.07 没有声明函数原型会造成怎样的结果？

FAQ12.08 函数参数的个数有限制吗？

FAQ12.09 exit()函数与return语句有什么不同吗？

FAQ12.10 return语句是必须存在的吗？

FAQ12.11 退出main()函数就意味着程序运行的结束吗？

<<你必须知道的213个C语言问题>>

章节摘录

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。

<<你必须知道的213个C语言问题>>

编辑推荐

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
请相信，你并不是第一个遇到问题的人。
发现问题，思考问题，寻找答案，解决问题。
《你必须知道的213个C语言问题》内容涵盖： 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 日期与时
间 重定向I/O与进程命令 开发常见错误及程序调试

<<你必须知道的213个C语言问题>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>