C语言最大难点揭秘[1] PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/246/2021_2022_C_E8_AF_AD_E8_A8_80_E6_9C_80_c97_246242.htm 正确的内存管理的重要性 存在内存错误的 C 和 C 程序会导致各种问题。如果它们泄漏内存，则运行速度会逐渐变慢，并最终停止运行；如果覆盖内存，则会变得非常脆弱，很容易受到恶意用户的攻击。从 1988 年著名的莫里斯蠕虫 攻击到有关 Flash Player 和其他关键的零售级程序的最新安全警报都与缓冲区溢出有关：“大多数计算机安全漏洞都是缓冲区溢出”， Rodney Bates 在 2004 年写道。 在可以使用 C 或 C 的地方，也广泛支持使用其他许多通用语言（如 Java&#8482.、Ruby、Haskell、C#、Perl、Smalltalk 等），每种语言都有众多的爱好者和各自的优点。但是，从计算角度来看，每种编程语言优于 C 或 C 的主要优点都与便于内存管理密切相关。与内存相关的编程是如此重要，而在实践中正确应用又是如此困难，以致于它支配着面向对象编程语言、功能性编程语言、高级编程语言、声明性编程语言和另外一些编程语言的所有其他变量或理论。 与少数其他类型的常见错误一样，内存错误还是一种隐性危害：它们很难再现，症状通常不能在相应的源代码中找到。例如，无论何时何地发生内存泄漏，都可能表现为应用程序完全无法接受，同时内存泄漏不是显而易见。 因此，出于所有这些原因，需要特别关注 C 和 C 编程的内存问题。让我们看一看如何解决这些问题，先不谈是哪种语言。 内存错误的类别 首先，不要失去信心。有很多办法可以对付内存问题。我们先列出所有可能存在的实际问题： 内存泄漏 错误分配，包

括大量增加 free() 释放的内存和未初始化的引用 悬空指针 数组边界违规 这是所有类型。即使迁移到 C 面向对象的语言，这些类型也不会有明显变化；无论数据是简单类型还是 C 语言的 struct 或 C 的类，C 和 C 中内存管理和引用的模型在原理上都是相同的。以下内容绝大部分是“ 纯 C ”语言，对于扩展到 C 主要留作练习使用。 内存泄漏 在分配资源时会发生内存泄漏，但是它从不回收。下面是一个可能出错的模型（请参见清单 1）： 清单 1. 简单的潜在堆内存丢失和缓冲区覆盖 void f1(char *explanation) { char p1. p1 = malloc(100). (void) sprintf(p1, "The f1 error occurred because of %s.", explanation). local_log(p1). } 您看到问题了吗？除非 local_log() 对 free() 释放的内存具有不寻常的响应能力，否则每次对 f1 的调用都会泄漏 100 字节。在记忆棒增量分发数兆字节内存时，一次泄漏是微不足道的，但是连续操作数小时后，即使如此小的泄漏也会削弱应用程序。 在实际的 C 和 C 编程中，这不足以影响您对 malloc() 或 new 的使用，本部分开头的句子提到了“ 资源 ”不是仅指“ 内存 ”，因为还有类似以下内容的示例（请参见清单 2）。FILE 句柄可能与内存块不同，但是必须对它们给予同等关注： 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com