《程序员的自我修养》勘误

1.第 1 章，P4，+8

原文：`int main()`

修改：`int main(void)`

2.第 1 章，P5，-13

原文：不过相信 90%以上的读者也是，

修改：不过相信 90%以上的读者也是这样，

3.第 1 章，P6，-2

原文：包括左边的内存和 PCI 总线

修改：包括右边的内存和 PCI 总线

4.第 1 章，P7，-12

原文：CPU 的频率从几十 KHz 到现在的 4GHz，

修改：CPU 的频率从几十 kHz 到现在的 4GHz，

5.第 1 章，P16，+2

原文：那么我们假设有一个地址从 0x00000000 到 0x00A00000 的 10MB 大小的一个假象的空间，

修改：那么我们假设有一个地址从 0x00000000 到 0x00A00000 的 10MB 大小的一个假想的空间，

6.第 1 章，P26，+5

原文：所谓同步，既是指在一个线程访问数据未结束的时候，其他线程不得对同一个数据进行访问。

修改：所谓同步，即指在一个线程访问数据未结束的时候，其他线程不得对同一个数据进行访问。

7.第 1 章，P33，图 1-13

修改：删掉 Kernel Thread 间多出的一根横线

8. 第 2 章，P40，+4

原文：编译过程就是把预处理完的文件进行一系列词法分析 、语法分析、语义分析及优化后生产相应的汇编代码文件，

修改：编译过程就是把预处理完的文件进行一系列词法分析 、语法分析、语义分析及优化后生成相应的汇编代码文件，

9.第 2 章，P41，+8

原文：具体地说分析静态链接的章节。

修改：具体地说是分析静态链接的章节。

10. 第 2 章，P42，-6

原文： `CompilerExpression.c`

修改：删除该行

11.第 2 章，P46，-5，-6

原文：在三地址码的基础上进行优化时，优化程序会将 2+6 的结果计算出来，得到 t1 = 6。然后将后面代码中的 t1 替换成数字 6。

修改：在三地址码的基础上进行优化时，优化程序会将 2+6 的结果计算出来，得到 t1 = 8。然后将后面代码中的 t1 替换成数字 8。

12.第 2 章，P47，+5

原文：源代码优化器产生中间代码标志着下面的过程都属于编辑器后端。

修改：源代码优化器产生中间代码标志着下面的过程都属于编译器后端。

13.第 3 章，P60，+14

原文：如果系统中运行了数百个进程，可以想象共享的方法来节省大量空间。

修改：如果系统中运行了数百个进程，可以想象共享的方法将节省大量空间。

14.第 3 章，P76，+19

原文：因为 Intel x86 系统要求浮点数的存储地址必须是本身的整数倍

修改：因为 Intel x86 系统要求浮点数的存储地址必须是本身所占存储空间的整数倍

15.第 3 章，P95，+8

原文：由 DWARF 标准委员会由 2006 年颁布。

修改：由 DWARF 标准委员会 2006 年颁布。

16.第 3 章，P60，+7

原文：所以内存中只须要保存一份改程序

修改：所以内存中只须要保存一份该程序

17.第 3 章，P81，+1

原文：只有分析 ELF 文件头，就可以得到段表和段表字符串表的位置，

修改：只要分析 ELF 文件头，就可以得到段表和段表字符串表的位置，

18. 第 3 章，P84，-6

原文：即第四列和第五列分别为符号类型和绑定信息,即对应 st_info 的低 4 位和高 28 位

修改：即第四列和第五列分别为符号类型和绑定信息,即对应 st_info 的低 4 位和高 4 位

19.第 3 章，P88，-8

原文：幸好这种名称修饰方法我们平时程序开发中也很少手工分析名称修饰问题，

修改：幸好我们平时程序开发中也很少手工分析名称修饰问题，

20.第 4 章，P105

原文：图 4-4“c4 44 24 04”

修改：图 4-4“c7 44 24 04”

21.第 4 章，P125, -5

原文：WRITE 调用的调用号为 4，则 eax=0

修改：WRITE 调用的调用号为 4，则 eax=4

22.第 4 章，P126，+13

原文： 关于系统库已经系统调用的细节我们在这里不详细展开

修改：关于系统库及系统调用的细节我们在这里不详细展开

23.第 4 章，P128，-11

原文：但是段名字符串表用户保存段名，所以它是必不可少的。

修改：但是段名字符串表为用户保存段名，所以它是必不可少的。

24.第 4 章，P128，-11

原文：ld 链接器的链接脚本语法继承与 AT&T 链接器命令语言的语法，

修改：ld 链接器的链接脚本语法继承于 AT&T 链接器命令语言的语法，

25.第 6 章，P151，+5

原文：我们在下文中以 32 位的地址空间为主，64 位的与 32 位类似。

修改：我们在下文的讨论中以 32 位的地址空间为主，64 位的与 32 位类似。

26.第 6 章，P159，-13

原文：第三步其实也是最简单的一部，

修改：第三步其实也是最简单的一步，

27.第 6 章，P175，-8

原文：装载一个 PE 可执行文件并且装载它，是个比 ELF 文件相对简单的过程：

修改：装载一个 PE 可执行文件是个比 ELF 文件相对简单的过程：

28.第 7 章，P181，+1

原文：并且它们还共用 Lib.o 这两模块。

修改：并且它们还共用 Lib.o 这个模块。

29.第 7 章，P192，-8

原文：数据的相对寻址往往没有相对与当前指令地址（PC）的寻址方式，

修改：数据的相对寻址往往没有相对于当前指令地址（PC）的寻址方式，

30.第 7 章，P194，-2

原文：我们来看看 GOT 如何做到指令的地址无关性。从第二中类型的数据访问我们了解到，

修改：我们来看看 GOT 如何做到与指令的地址无关。从第二种类型的数据访问我们了解到，

31.第 7 章，P195，+12

原文：我们再来看看 pic.so 的需要在动态链接时重定位项：

修改：我们再来看看 pic.so 的需要在动态链接时的重定位项：

32.第 7 章，P197，-4

原文：它无法根据这个上下文判断 global 是定义在同一个模块的的其他目标文件还是定义在另外一个共享对象之中，

修改：它无法根据这个上下文判断 global 是定义在同一个模块的其他目标文件还是定义在另外一个共享对象之中，

33.第 7 章，P223，-7

原文：定义非常简洁，两个参数，

修改：定义非常简洁，有两个参数，

34.第 7 章，P224，-15

原文：那么由于全局符号表使用的装载序列，

修改：那么由于全局符号表使用的是装载序列，

35.第 8 章，P232，+16

原文：它规定共享库的文件名规则必须如下：

修改：它规定共享库的文件命名规则必须如下：

36.第 8 章，P234，-7

原文：否则这个这个程序 A 就无法正常运行。

修改：否则这个程序 A 就无法正常运行。

37.第 8 章，P236，-3

原文：Linux 下的 Glibc 从版本 2.1 之后开始支持一种叫做基于符合的版本机制（Symbol Versioning）的方案。

修改：Linux 下的 Glibc 从版本 2.1 之后开始支持一种叫做基于符号的版本机制（Symbol Versioning）的方案。

38.第 9 章，P254，-7

原文：实际上.def 文件在 MSVC 链接过程中的作用与链接脚本文件（Link Script）文件在 ld 链接过程中的作用类似，

修改：实际上.def 文件在 MSVC 链接过程中的作用与链接脚本（Link Script）文件在 ld 链接过程中的作用类似，

39.第 9 章，P273，-1

原文：使得原先要占有数十 M 内存降低到只占用数 M 内存。

修改：使得原先要占有数十 MB 内存降低到只占用数 MB 内存。

40.第 10 章，P299，-1

原文：因此无论如何也不可能直接用过 eax 传递。

修改：因此无论如何也不可能直接用 eax 传递。

41.第 10 章，P314，+1

原文：

Q&A

修改：删除此行。

42.第 11 章，P350，-9

原文：寄存器是执行流的基本数据，

修改：寄存器存放的数据是执行流的基本数据，

43.第 12 章，P401，-15

原文：dd if=/proc/self/mem of=linux-gate.dso bs=4096 skip=1048574 count=1 这个命令是如何得到 vdso 的印像文件的？

修改：dd if=/proc/self/mem of=linux-gate.dso bs=4096 skip=1048574 count=1 这个命令是如何得到 vdso 的映像文件的？

44.第 13 章，P417，-5

原文：我们在第 9 章时已经介绍过堆分配算法的原理，在实现上也基本一致。整个堆空间按照是否被占用而被分割成了若干个空闲（Free）块和占用（Used）块，

修改：我们在第 9 章时已经介绍过堆分配算法的原理，在实现上是将整个堆空间按照是否被占用而被分割成了若干个空闲（Free）块和占用（Used）块，