# 4.关键技术

## 4.1成组链接法

在UNIX系统中，将空闲块分成若干组，每100个空闲块为一组。其中仅有一组会被调入内存，被调入内存的块称为超级块。每组的第一空闲块登记了下一组空闲块的物理盘块号和空闲块总数。如果一个组的第二个空闲块号等于0，则有特殊的含义，意味着该组是最后一组，即无下一个空闲块。成组链接法的基本架构如图4.1所示（为简化说明未完全画完盘块）：

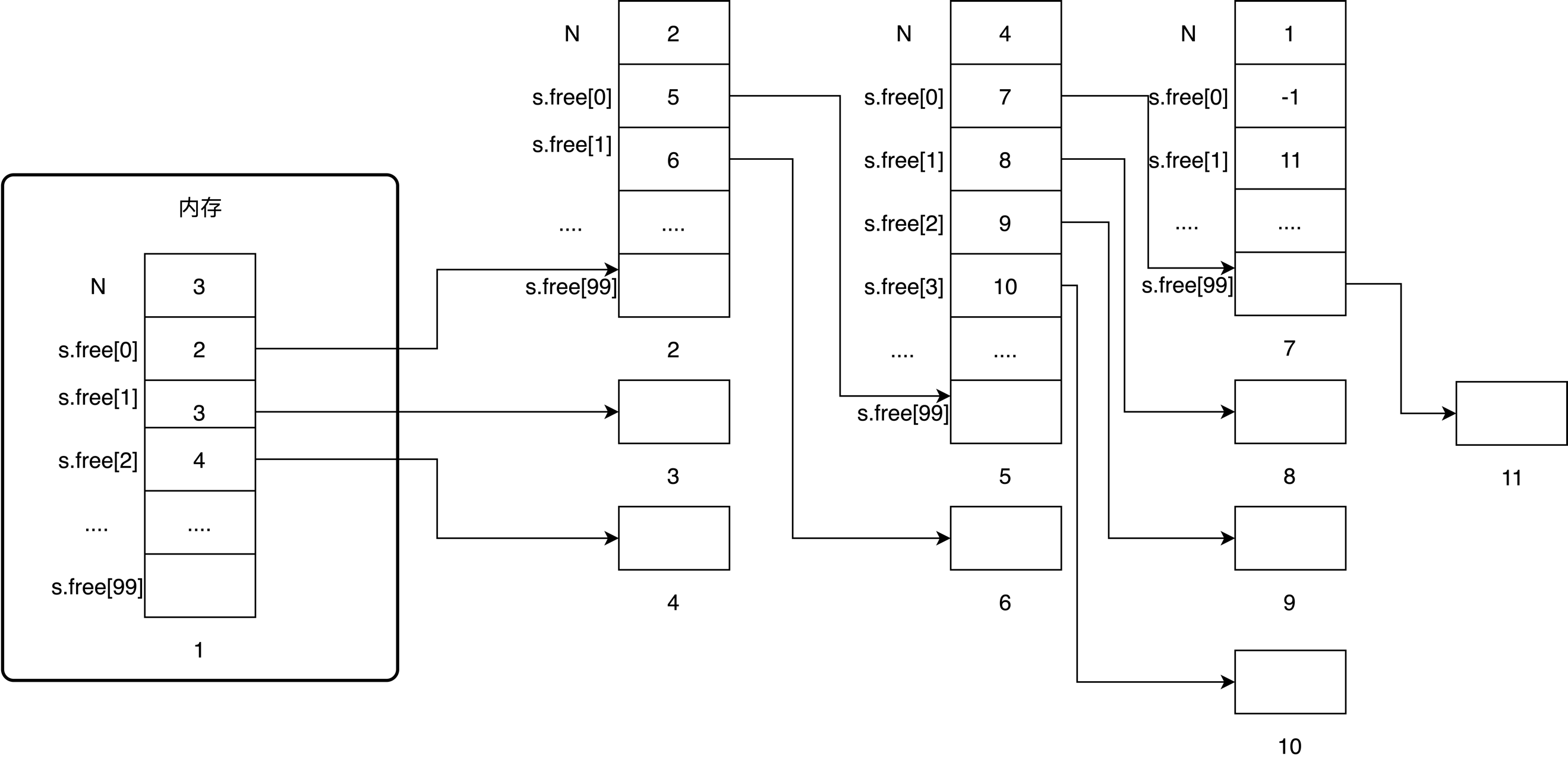


图4.1 成组链接法结构

在需要空间分配时，假如要为一个文件分配一个盘块，系统将会从超级块中出栈一个盘块，然后把这个编号对应的盘块分配给文件。以图4.1为例，此时若需要分配一个盘块，则4号盘块分配给文件，4号出栈，并且超级块中的N就会减1变成2，如果还不够继续分配，那么就继续出栈，下一个出栈的编号是3，同样的N接着减1变成1，也就是说此时超级块只剩下一个盘块，这个盘块含有一个栈，它保存了后续盘块的信息。当还需要分配盘块时，需要将它里面的栈和N保存到超级块中。假如这时仍需要给文件分配盘块，那么就将2的信息先保存到超级块中，然后把2分配给文件，然后重新更新一下指针。超级块中的内容被原来盘块2中的内容所覆盖，也就是原来盘块2中保存的后续节点的信息被转移到了超级块中，结果如图4.2所示。

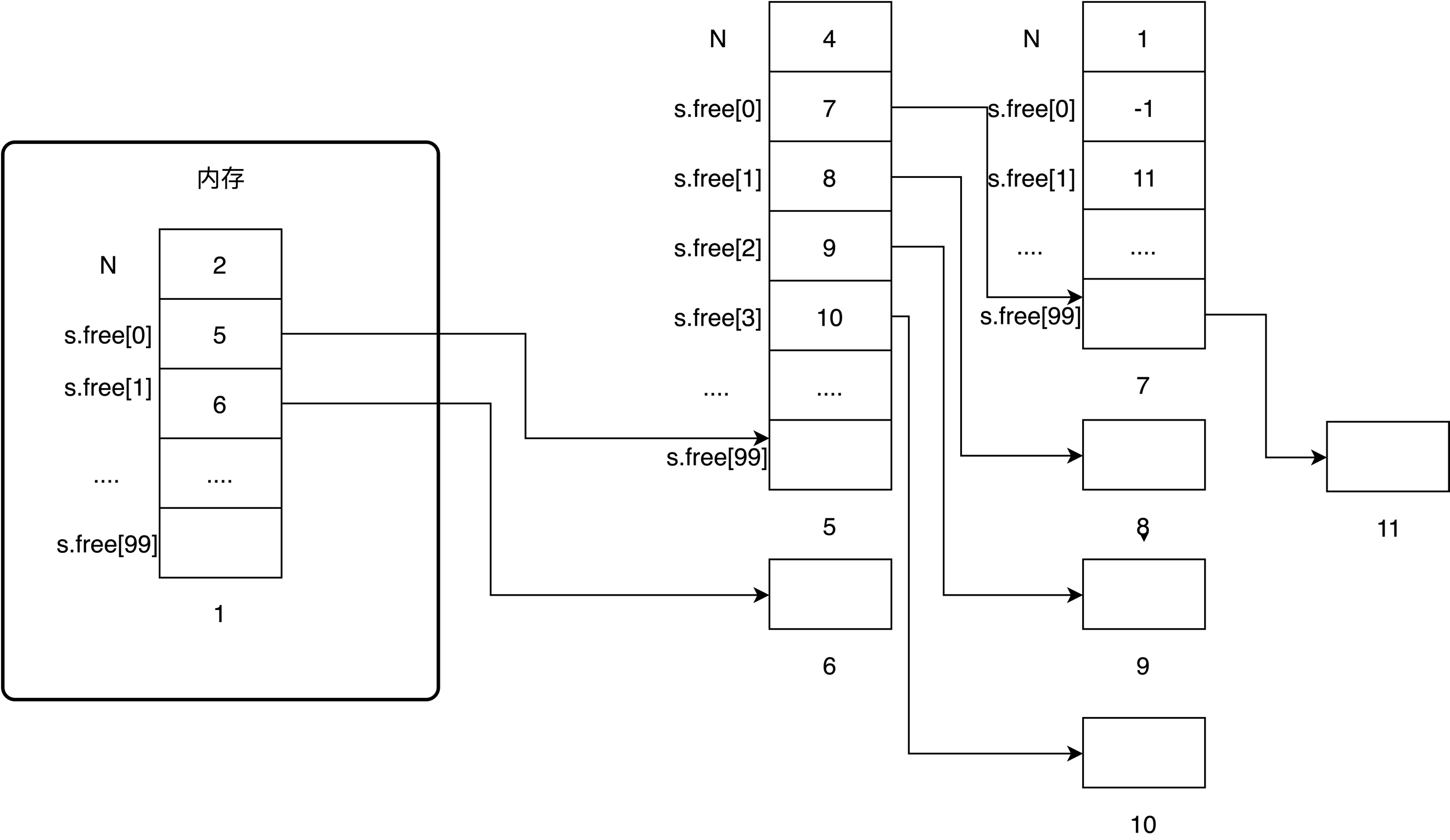


图4.2 成组链接法查找空闲块

在需要空间回收的时候，直接让其入栈超级块，并更新指针。假如此时来了一个磁盘块4，则直接将盘块4压入栈中，之后更新N。就这样来一个盘块就压入一个。当栈满了之后（为了方便演示，假设下一步就满了），就会把超级块的内容复制到新回收的盘块中，假如新回收的盘块是3，就把当前超级块中的内容复制到盘块3中，然后更新一下指针，盘块3与之后的盘块建立了联系，然后根据盘块3更新超级块的栈和N，显然N为1，因为超级块所指的下一组只有这个盘块3，并且把盘块3压入栈中。这就是成组链接法的空间回收，如图4.3所示:

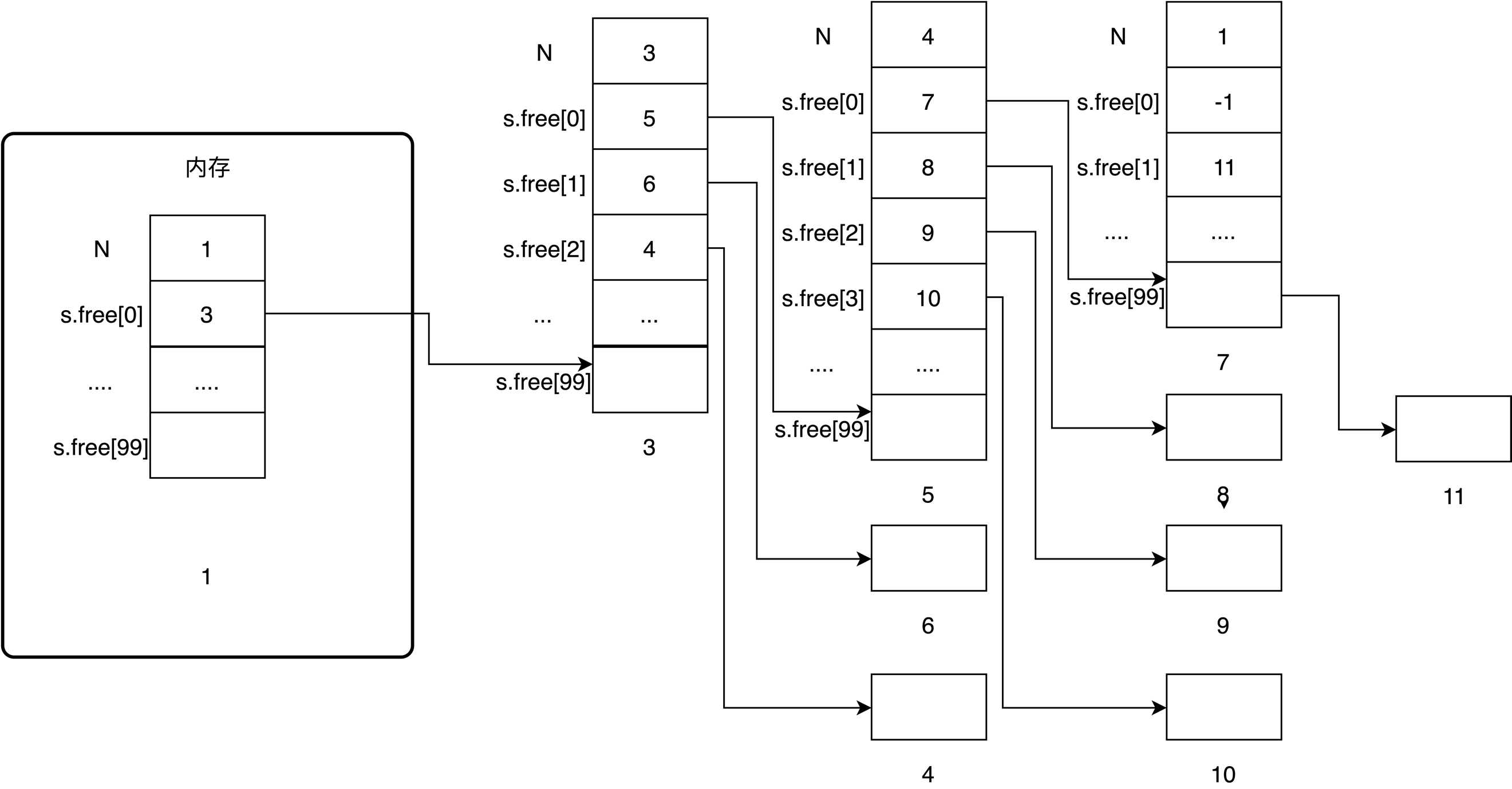


图4.3 成组链接法空间回收

# 5.运行结果

## 5.1 运行环境

CPU: 2.5 GHz/2.7 GHz

内存：2G内存

操作系统：Ubuntu 18.04

## 5.2 运行结果

5.2.1 登陆界面

在使用系统之前首先需要登陆到系统，当用户输入的账号或密码出错时，会提醒用户输入错误，要求重新输入。如图5.1所示

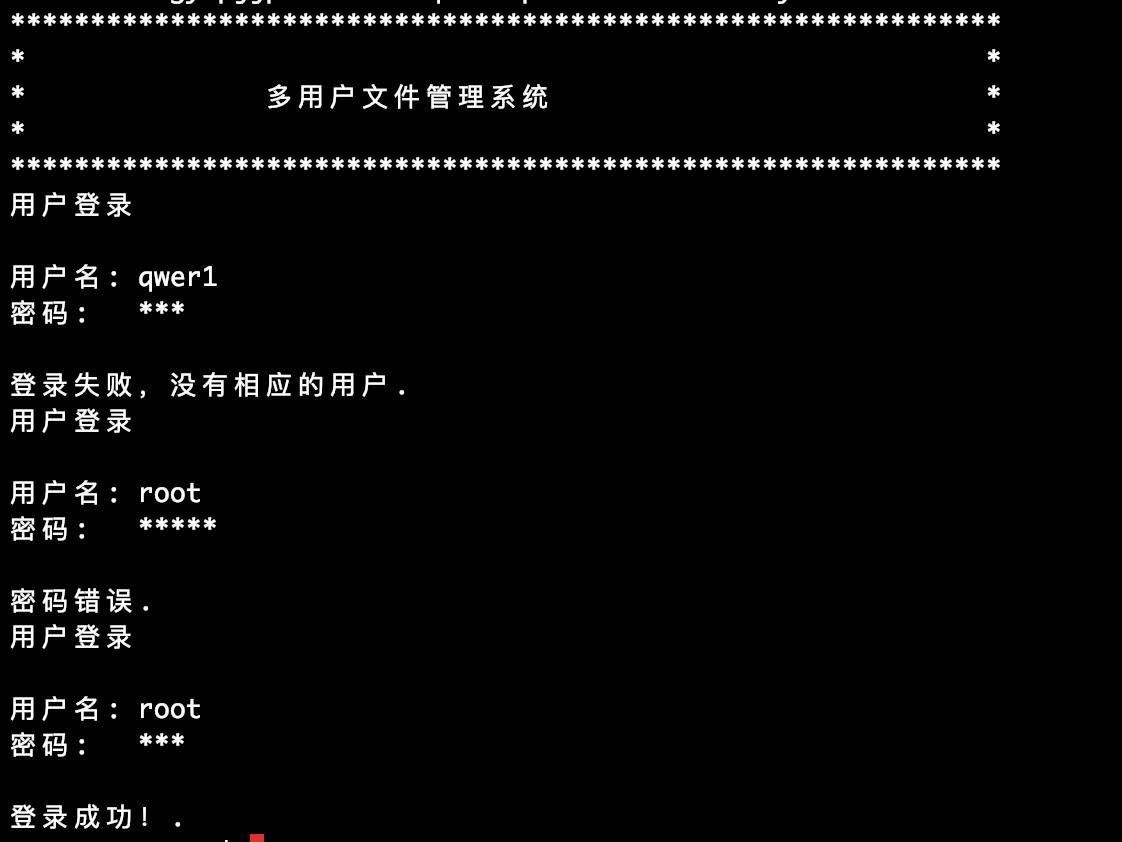


图5.1 登陆界面

5.2.2 help指令

在系统中输入help指令时，会显示系统支持的所有指令名称及其对应的功能，当需要查找对应指令使用方法时也可使用help指令，效果如图5.2所示:

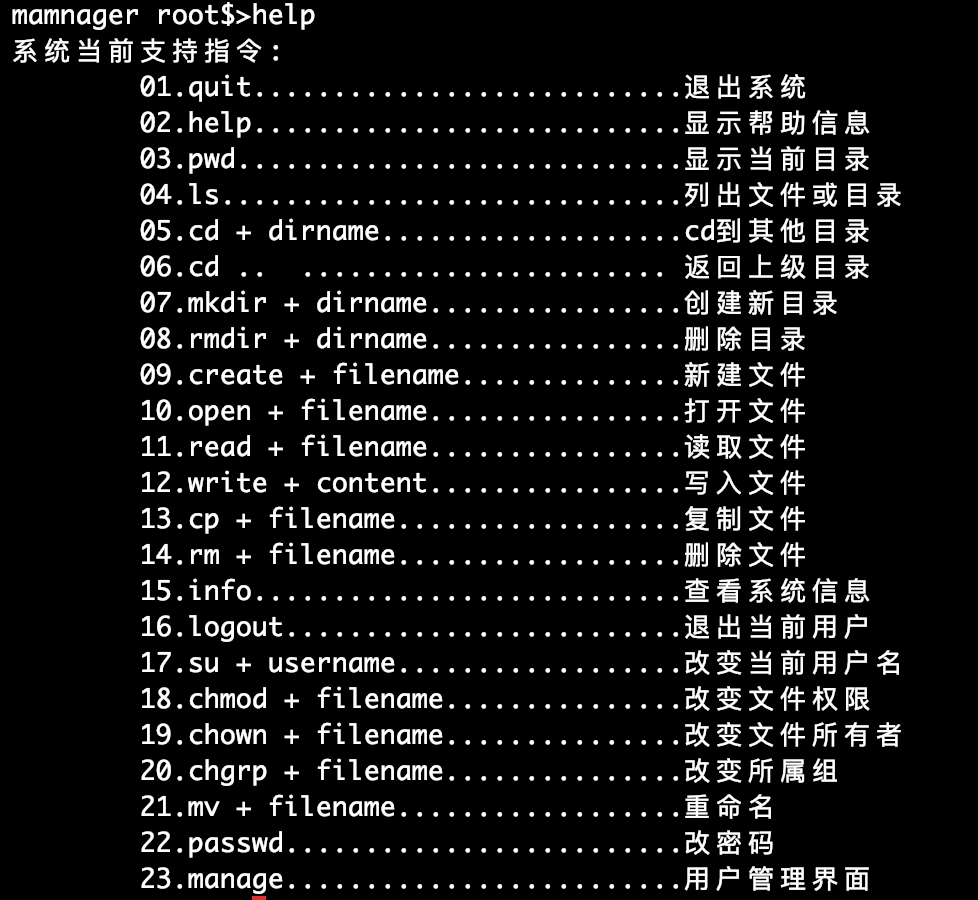


图5.2 help指令界面

5.2.3 pwd指令

在系统中输入pwd指令时，系统会显示当前的绝对路径，如图5.3所示



图5.3 pwd指令界面

5.2.4 ls指令

在系统中输入ls指令时，系统会显示当前文件夹中所有文件的详细信息，如图5.4所示

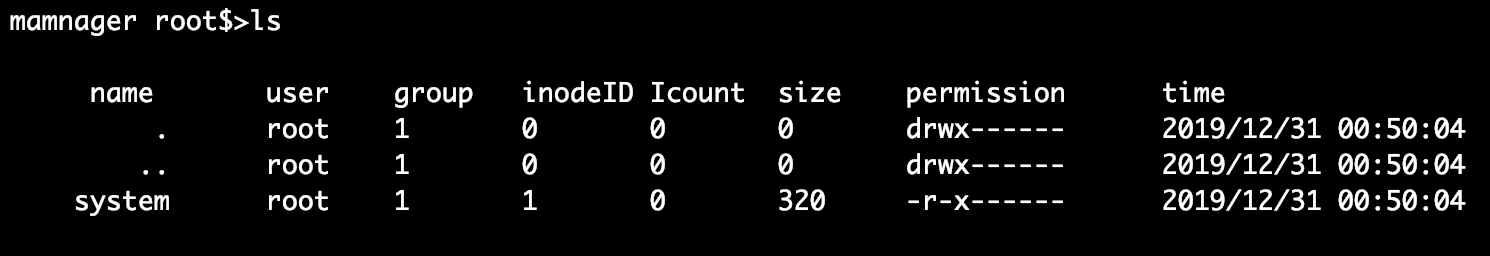


图5.4 ls指令界面

5.2.5 mkdir指令

当在系统中输入mkdir + dirname指令时，系统会在当前目录创建一个命名为dirname的次级目录，如图5.5所示

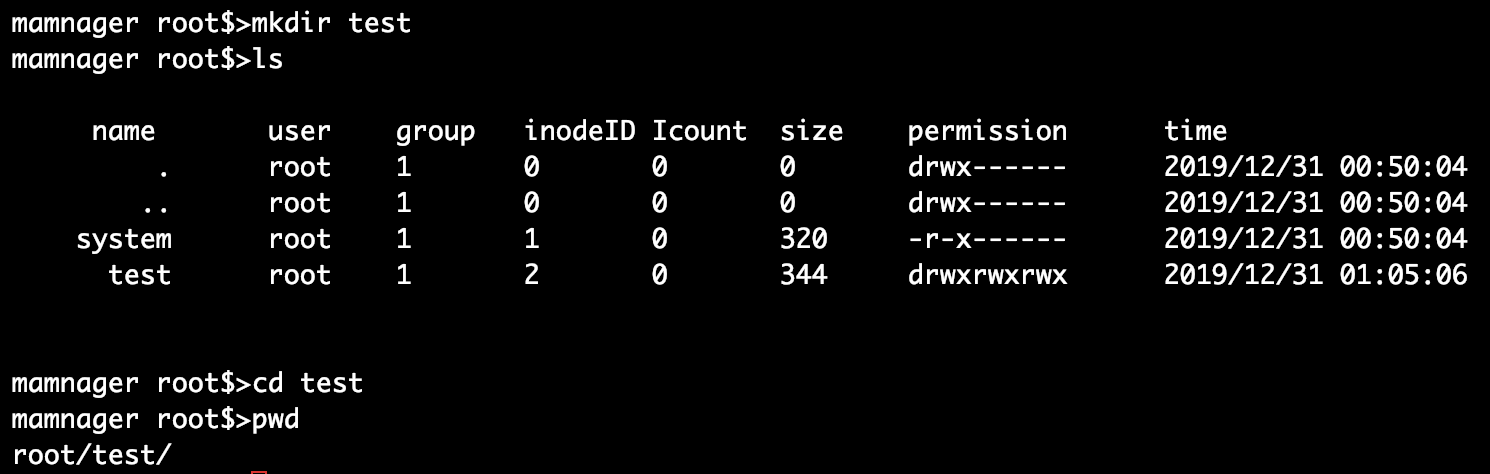


图5.5 mkdir指令界面

5.2.6 create指令

当在系统中输入create+filename指令时，系统会在当前目录创建一个命名为filename的文件，如图5.6所示

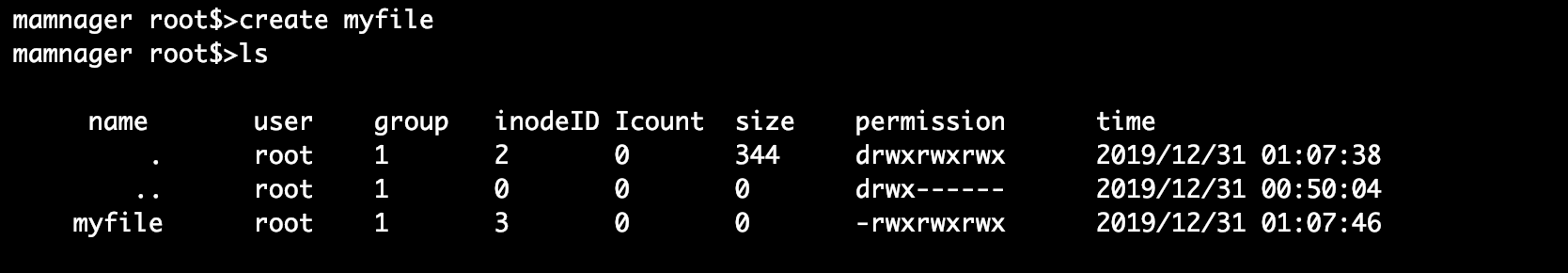


图5.6 create指令界面

5.2.7 open, write, read指令

在系统中输入open+filename指令时，系统会打开命名为filename的文件，此时可以输入write+content指令进行写入文件操作，之后可以输入read+filename指令读出文件内容，如图5.7所示



图5.7 open, write, read指令界面

5.2.8 chmod指令

在系统中输入chmod+filename or dirname指令，可以为系统文件或文件夹设定访问权限，如图5.8所示

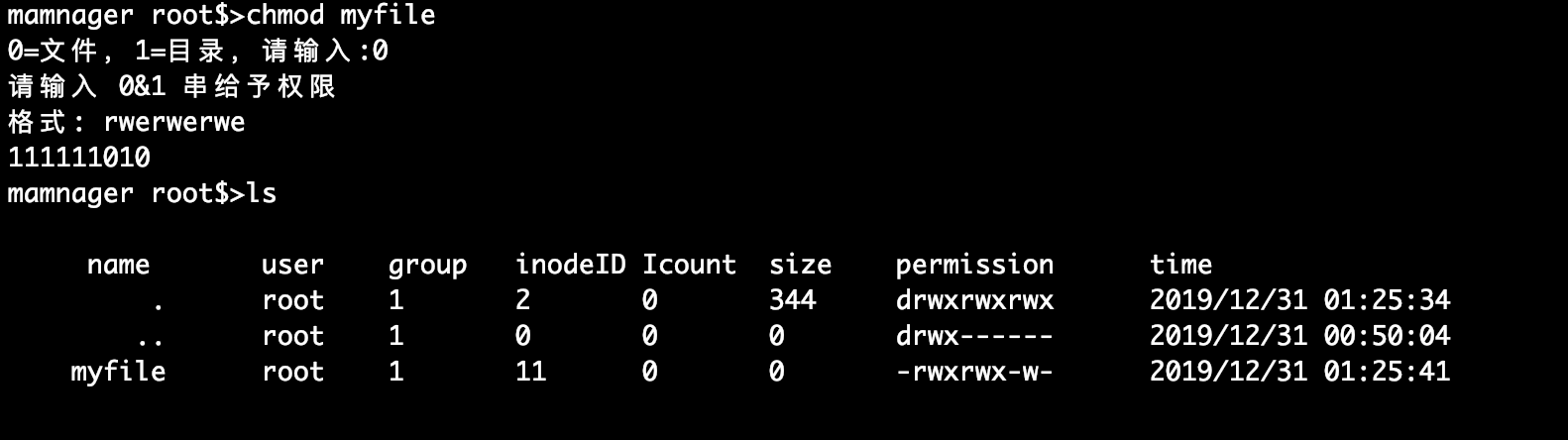


图5.8 open, write, read指令界面

5.2.9 chgrp指令

在系统中输入chgrp+filename or dirname指令，可以修改文件或文件夹所属组，如图5.9所示

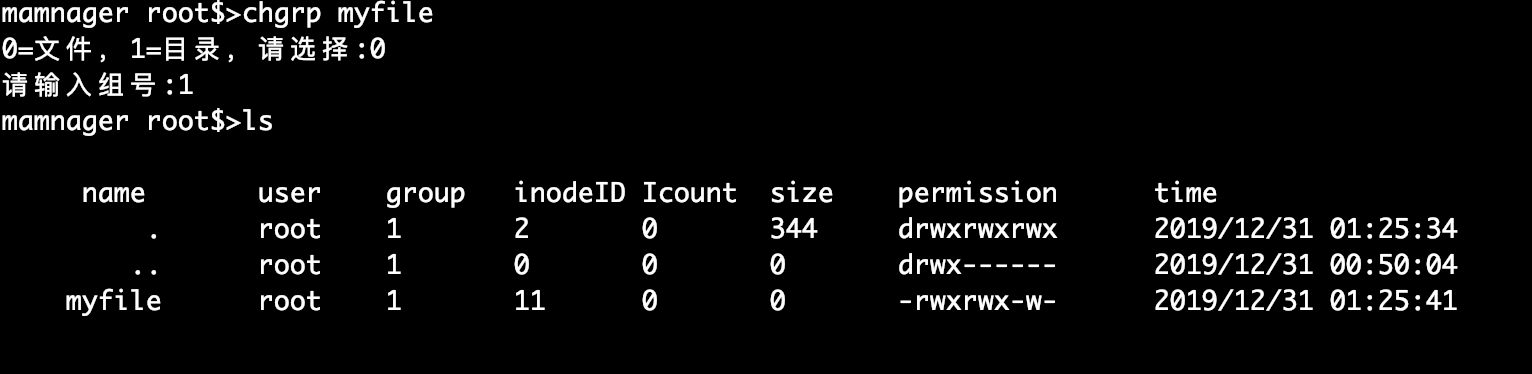


图5.9 chgrp指令界面

5.2.10 chown指令

在系统中输入chown+filename or dirname指令，可以修改文件或者文件夹所属用户，如图5.10所示

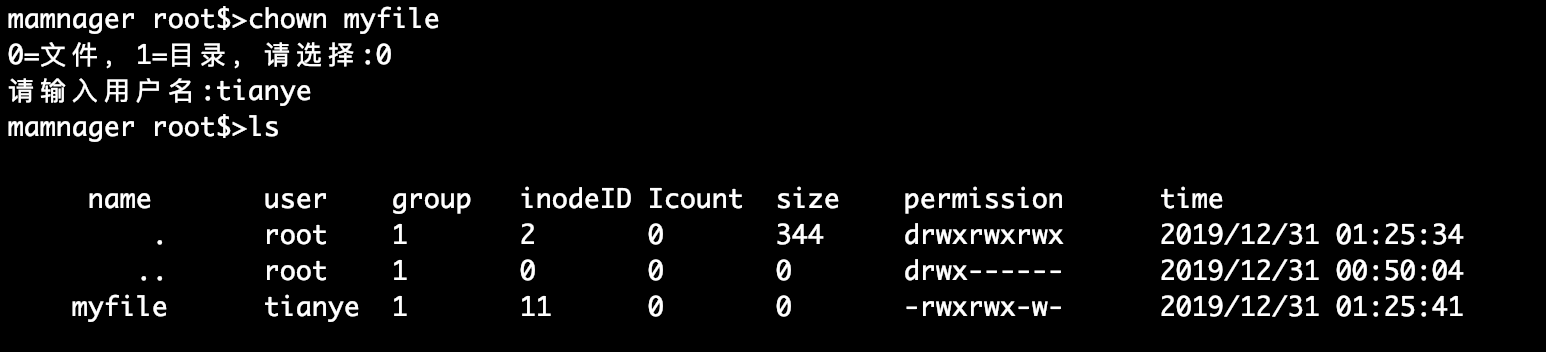


图5.10 chown指令界面

5.2.11 manage指令

管理员在系统中输入manage指令，可以进入用户管理界面，在此界面可以查看用户信息，创建以及删除用户，如图5.11所示

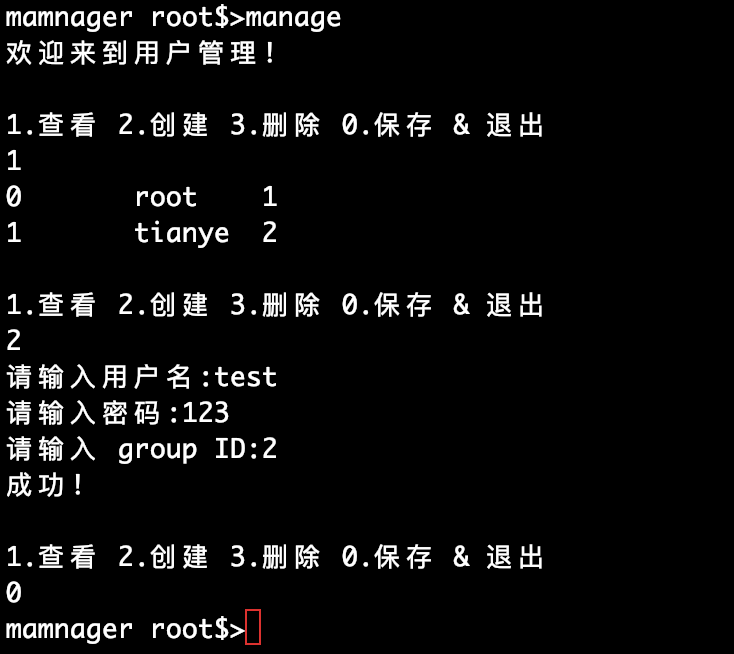


图5.11 manage指令界面

# 6.调试和改进

在编写本系统的时候，其中最困难的部分就是成组链接算法的实现，因为要实现这一算法，会涉及到很多指针操作，只是在平时使用诸如java, python之类的高级语言时不会面临的情况，所以在调试指针时花费了大量的时间。

一开始在编写这个算法的时候纯粹就是想到什么就写了什么，所以导致在后面整个代码显得特别杂乱，当需要改动一个小地方的时候，可能关联到这里一点，那里一点，修改起来十分麻烦，甚至可能后面忘记了之间的关联性，要调适很久才能找到修改的地方。后来索性重新写了这部分代码，这次注重的功能之间的分离，让每个模块只负责专门的功能，尽量少在某个模块内想其他的模块传递参数。减少模块之间的耦合性，增加模块的内聚性。

# 7.心得和结论

## 7.1结论和体会

本系统完成了一个多用户的文件管理系统，具有基本的目录内容显示、创建/删除目录、更改当前目录、创建/删除文件、读/写文件、打开/关闭文件、及用户权限管理、组管理功能。但目前系统还不够完善，鲁棒性仍然很弱，在输入非指定命令格式时可能会出现意想不到的错误。

经过这次的设计，让我明白了在写代码之前动手分析、设计结构等步骤等重要性，不能低估这些没有写代码的时间，相反充分利用好这些设计的时间，设计出一个高内聚低耦合的系统，在编写代码、后期调试时会达到事半功倍的效果。

## 7.2 进一步改进方向

相较于常用的文件管理系统而言，目前设计出的文件管理系统仍然具有很多方面的不足，例如：在输入时不允许按下删除键，不支持指令可选功能，支持的指令还太少等，如果能再有一些时间，那么接下来的改进方向就是增加系统的鲁棒性，让系统在输入非指定命令时仍能正常工作，增加系统支持的指令，扩展系统的功能，让其达到做到一个真正的文件管理系统能做的所有事情。

# 主要参考文献

[1] Operating\_Systems\_Concepts[7th] Abraham Silbershatz