**实验一 Linux命令使用**

班级： 姓名： 学号： 上机时间：

任课教师： 黄连丽 实验教师： 黄连丽 实验成绩：

**一、实验目的**

1. 掌握Linux的一般命令格式。
2. 掌握有关文件和目录操作的常用命令。
3. 掌握有关进程操作的常用命令。
4. 熟练使用man命令。

**二、实验注意事项**

实验环境与系统是共用设施，请不要在系统内做对系统或对其他用户不安全的事情。要求每个同学登录后系统后，要在主目录（/home/student）下创建一个属于自己的子目录（以自己(拼音)名字或学号）。以后所有工作都要在自己的目录内进行。建议以后的实验都在同台计算机上做，这样可以保持连续性。

**三、实验内容及步骤：**

1．登录进入系统。

开机，打开powershell 输入wsl 进入linux子系统

2．使用简单命令：date，cal，who，echo，clear等，了解Linux命令格式。

要启动一个终端，可以选择  点击屏幕上的终端图标或者通过快捷键Ctrl+Alt+T 。

在终端提示符下输入：date ↲ 显示： 当前日期和间 。

在终端提示符下输入：who ↲ 显示： 当前的登录用户列表 。

在终端提示符下输入：who am I 显示： 当前使用的用户名 。

在终端下显示Hello! 命令： echo "Hello!" 。

在终端提示符下输入：clear ↲ 显示： 清空终端屏幕 。

在终端提示符下输入：cal ↲ 显示： 当前的月份的日历 。

文本

描述已自动生成

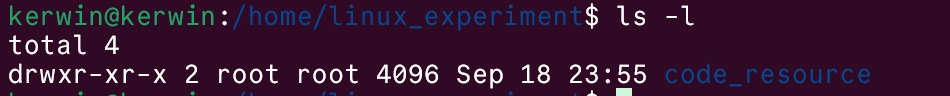
3．浏览文件系统：

（1）运行pwd 命令，确定当前工作目录，记录之： /home/linux\_experiment 。

（2）运行ls命令，查看当前目录下有哪些文件和子目录，简略记录。

图形用户界面, 文本

描述已自动生成

（3）运行ls -l命令，理解文件各字段的含义，找一个文件标注。

d：这是文件类型的标志。d 表示这是一个目录（directory）。如果是 - 则代表这是一个普通文件（regular file），其他字母如 l 可能代表符号链接（symbolic link）等。

rwxr-xr-x：这是文件的权限，分为三个部分，每三个字符一组，分别对应文件所有者（owner）、所属组（group）和其他用户（others）的权限：

第一组 rwx：文件所有者的权限。这里的 rwx 表示所有者可以读取（read）、写入（write）和执行（execute）该目录。

第二组 r-x：所属组成员的权限。这里的 r-x 表示组内的成员可以读取（read）和执行（execute）该目录，但不能写入（write）。

第三组 r-x：其他用户的权限。这里的 r-x 表示任何不属于文件所有者或所属组的用户可以读取（read）和执行（execute）该目录，但不能写入（write）。

2：这是硬链接的数量（hard link count）。对于目录来说，这个数字至少是2，因为每个目录都有两个默认的硬链接：“.”指向自身，“..”指向其父目录。

root：这是文件所有者的用户名。在这个例子中，文件的所有者是 root 用户。

root：这是文件所属组的名字。在这个例子中，文件所属组也是 root 组。

4096：这是文件或目录占用的空间大小，以字节（bytes）为单位。对于目录而言，这通常是一个固定的大小，除非目录中有非常大的数量的文件。

Sep 18 23:55：这是文件或目录最近一次修改的时间。在这个例子中，code\_resource 目录最后修改的时间是 2024 年 9 月 18 日 23:55。

code\_resource：这是文件或目录的名字，在这个例子中是一个名为 code\_resource 的目录。

（4）运行ls -ai命令，理解各字段的含义。

2581：这是当前目录（.）的 inode 号。inode 是文件系统中的一个数据结构，用于存储文件的元数据（如权限、所有者、时间戳等），但不包含文件名或文件内容。每个文件或目录都有一个唯一的 inode 号。

.：这是当前目录的符号表示。. 总是指向当前目录本身。

24577：这是父目录（..）的 inode 号。每个目录都有一个指向其父目录的链接。

..：这是父目录的符号表示。.. 总是指向当前目录的上级目录。

5589：这是 code\_resource 目录的 inode 号。

code\_resource：这是 code\_resource 目录的名称

（5）使用cd /命令，将当前目录改变为根目录（/）。运行ls -l命令，结合书中给出的Linux树状目录结构，了解各目录的作用。

命令： cd / ↲ 回到：根目录（/）。这是文件系统的最顶级目录，所有的其他目录和文件都从这里开始。

命令： pwd ↲ 显示： /。这表示当前目录是根目录。

命令： ls -l ↲ 显示：（简略记之）

图形用户界面, 文本, 应用程序

描述已自动生成

（6）直接使用cd，回到了那里？用pwd验证之。

命令： cd ↲ 回到： 当前用户的主目录

命令： pwd ↲ 显示： /home/kerwin

（7）用 mkdir建立一个子目录subdir。

命令： mkdir subdir ↲

（8）将当前目录改到subdir。

命令： cd subdir

4．文件操作：

（1）验证当前目录在subdir。

命令： pwd ↲ 显示： /home/linux\_experiment/subdir

（2）运行date>file1，然后运行cat file1，看到什么信息？

命令： date > file1 ↲ ，命令： cat file1 ↲

显示：



“ >”是什么符号？ 这是重定向符号。它将一个命令的输出重定向到一个文件中，而不是显示在终端上。如果文件不存在，它会被创建；如果文件已经存在，它会被覆盖。

解释“ date>file1 ”的含义： 这个命令将 date 命令的输出（即当前的日期和时间）重定向到 file1 文件中。也就是说，file1 文件的内容将会是 date 命令的输出。

（3）运行cat subdir，会有什么结果？为什么？

结果：



为什么？ cat 命令用于显示文件的内容。但它不能用于显示目录的内容。当你试图使用 cat 命令显示一个目录时，系统会报错，提示你 subdir 是一个目录，而不是一个文件。 。

（4）分别用cat和more查看文件内容（提示：/etc/manpath.config）。比较cat和more这两个命令的异同。

文本

描述已自动生成

**比较 cat 和 more 命令的异同**

**相同点**

* **目的**：两者都是用于查看文件内容的命令。
* **基本用法**：都可以通过简单的命令语法 cat filename 或 more filename 来使用。

**不同点**

* **显示方式**：
  + **cat**：将文件内容一次性全部输出到终端，适合查看较短的文件或需要将文件内容重定向到其他地方（如另一个文件或管道）。
  + **more**：分页显示文件内容，适合查看较长的文件，用户可以逐屏查看内容，便于阅读和导航。
* **交互性**：
  + **cat**：非交互式命令，没有用户输入的选项。
  + **more**：交互式命令，用户可以使用空格键、Enter 键和 q 键进行导航和退出。
* **适用场景**：
  + **cat**：适用于需要将文件内容快速输出到终端或与其他命令结合使用（如 cat file | grep pattern）。
  + **more**：适用于需要逐屏查看文件内容，特别是文件内容较长时。

（5）利用man 命令显示date命令的使用说明。

命令： man date ↲

显示：(简略记之)

文本

描述已自动生成

如何从man命令状态退回到终端窗口？ **按 q 键** 。

（6）运行man date>>file1，再运行cat file1 ，看到什么？

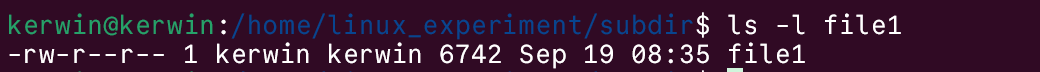
命令： **man date >> file1** 命令： cat file1

文本

描述已自动生成 显示：

man date>>file1含义： 这个命令将 man date 的输出追加到 file1 文件中。如果 file1 文件不存在，它会被创建；如果文件已经存在，新的内容会被追加到文件末尾。。

（7）利用ls -l file1，了解链接计数是多少？运行 ln file1 ../fa，再运行ls -l file1,看链接计数有无变化？用cat命令显示fa文件内容。**（注意：做完后要删除所做硬链接，恢复原样以便后面同学再做此实验）**

 命令：ls -l file1↲ 显示的链接计数：

命令：ln file1 ../fa↲

命令：ls -l file1↲ 显示的链接计数： 2 ,有无变化？ 连接数加一 。

命令：cat fa↲ 显示：

文本

描述已自动生成

（8）使用head和tail命令分别显示file1文件的前十行和后十行的内容。

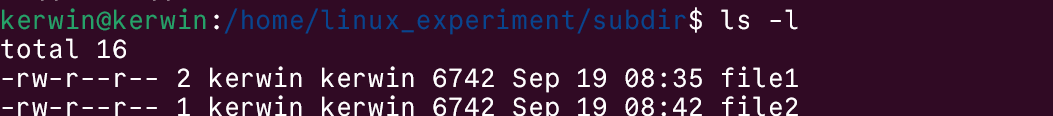
命令： head file1 ↲

命令： tail file1 ↲

（9）运行cp file1 file2，然后ls -l，看到什么？

命令：cp file1 file2 ↲

ls -l↲

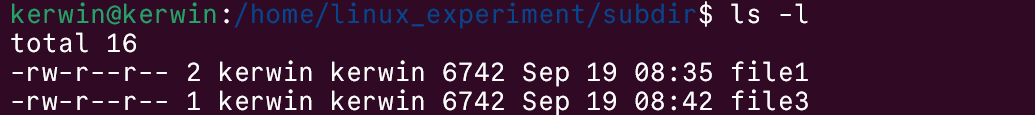
 解释显示结果：

* file1 和 file2 都存在，且它们的大小和权限相同。
* 链接计数仍然是 1，因为 file2 是 file1 的一个独立副本，而不是硬链接。

运行mv file2 file3, 然后ls -l，看到什么？

命令：mv file2 file3↲

ls –l ↲

 解释显示结果：

* file2 已经被重命名为 file3。
* file1 和 file3 都存在，且它们的大小和权限相同。
* 链接计数仍然是 1，因为 file3 仍然是 file1 的一个独立副本。

运行cat f\*，结果怎样？

命令：cat f\* ↲

解释显示结果：

文本

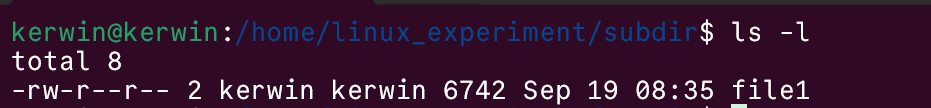
描述已自动生成

命令将显示所有以 f 开头的文件内容。在这个例子中，包括 file1 和 file3

（10）运行rm file3，然后ls -l，看到什么？

命令：rm file3 ↲

命令：ls -l ↲

 解释显示结果：

* 这个命令将删除 file3 文件。

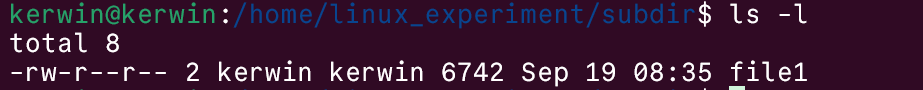
（11）在/etc/passwd文件中查找适合你的注册名的行。

命令： grep kerwin /etc/passwd

文本

描述已自动生成

（12）运行ls -l理解各文件的权限是什么？(选择两个不同文件标注)



**total 8**

* **total 8**：表示当前目录下所有文件和目录的总大小，单位是千字节（KB）。在这个例子中，总大小是 8 KB。

**-rw-r--r-- 2 kerwin kerwin 6742 Sep 19 08:35 file1**

* **-rw-r--r--**：权限部分
  + -：表示这是一个普通文件（Regular File）。
  + rw-：文件所有者的权限（kerwin 用户）：读（r）、写（w）、不执行（-）。
  + r--：文件所属组的权限（kerwin 组）：读（r）、不写（-）、不执行（-）。
  + r--：其他用户的权限：读（r）、不写（-）、不执行（-）。
* **2**：链接数。对于普通文件，这个数字通常为1，除非文件有硬链接。这里链接数为2，表示 file1 有2个链接。
* **kerwin**：文件所有者（Owner）。
* **kerwin**：文件所属组（Group）。
* **6742**：文件大小，单位是字节。
* **Sep 19 08:35**：文件的最后修改时间。
* **file1**：文件名。

**解释各参数的含义**

* **-rw-r--r--**：
  + **-**：表示这是一个普通文件。
  + **rw-**：文件所有者（kerwin）的权限：读（r）、写（w）、不执行（-）。
  + **r--**：文件所属组（kerwin）的权限：读（r）、不写（-）、不执行（-）。
  + **r--**：其他用户的权限：读（r）、不写（-）、不执行（-）。
* **2**：链接数。表示 file1 有2个链接。
* **kerwin**：文件所有者。表示 file1 文件的所有者是 kerwin 用户。
* **kerwin**：文件所属组。表示 file1 文件所属的组是 kerwin 组。
* **6742**：文件大小。表示 file1 文件的大小是 6742 字节。
* **Sep 19 08:35**：文件的最后修改时间。表示 file1 文件最后一次修改的时间是 2023年9月19日 08:35。
* **file1**：文件名。表示文件的名称是 file1。

**总结**

* **-rw-r--r--**：文件所有者有读写权限，组和其他用户只有读权限。
* **2**：文件有2个链接。
* **kerwin**：文件所有者和所属组都是 kerwin。
* **6742**：文件大小为 6742 字节。
* **Sep 19 08:35**：文件的最后修改时间为 2023年9月19日 08:35。
* **file1**：文件名为 file1。

（13）用两种方式改变file1的权限（实例）。

①以符号模式改变权限： chmod a+x file1

②以绝对方式改变权限： chmod 777 file1

（14）统计file1文件的行数、字数、字节数。

命令： wc file1 ↲

显示： 208 855 6742 file1

解释各参数的含义：

* **208**：表示文件的行数（lines）。
* **855**：表示文件的字数（words）。
* **6742**：表示文件的字节数（bytes）。
* **file1**：表示文件名。

（15）运行命令man ls |more

* “man ls |more”的含义： 运行 man ls | more 后，终端会分页显示 ls 命令的手册页内容。你可以按回车键逐行滚动，按空格键逐屏滚动，按 q 键退出 more。

运行cat file1|head -20|tee file5，结果如何？

显示结果：

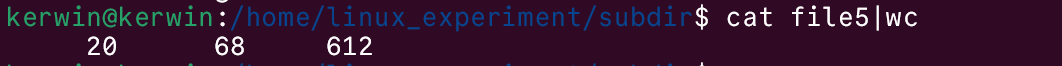
文本

描述已自动生成

* “cat file1|head -20|tee file5”的含义： 终端会显示 file1 文件的前 20 行内容。
* 同时，这 20 行内容会被写入 file5 文件。

运行cat file5|wc，结果如何？

显示结果：



* “cat file5|wc”的含义： 终端会显示 file5 文件的行数、字数和字节数。

（16）运行cd，回到主目录，运行mkdir mydir，建立mydir 子目录，运行ls -l查看目录，再运行rmdir mydir，看到什么结果？如果要删除mydir目录，该如何操作？

运行命令：cd <回车键> 回到： 当前用户的主目录

运行命令：mkdir mydir 建立： mydir 子目录

运行命令： ls -l 查看所建目录内容

运行命令：cp /etc/profile /home/student/mydir

解释该命令：

* cp 命令用于复制文件。
* /etc/profile 是要复制的源文件。
* /home/student/mydir 是目标目录。

查看mydir内容命令： ls -l mydir

运行命令： rmdir mydir ↲

看到什么结果： rmdir: failed to remove 'mydir': Directory not empty

如何处理？ 如果要删除包含文件的目录，可以使用 rm -r 命令（递归删除）：

***注意***：在这个实验中没有特别的难点，但是要注意的是，rm命令删除文件的时候，不给任何提示，因此一定要谨慎，不然很容易删除不应该删除的文件。

**四、实验报告要求**

1．按实验步骤写出实验命令及运行结果，回答所提问题。

2．总结上机过程中所遇到的问题和解决方法及感想。

在上机实验过程中，我遇到了几个问题并找到了相应的解决方法。首先，当我尝试查找 /etc/passwd 文件时，发现该文件不存在。通过使用 ls -l /etc/passwd 命令确认文件确实不存在后，我改用了 getent passwd 命令从系统用户数据库中获取用户信息，成功解决了这个问题。接着，在尝试删除 mydir 目录时，发现 rmdir 命令无法删除非空目录。通过查阅文档，我了解到可以使用 rm -r 命令递归删除目录及其内容，从而顺利删除了 mydir 目录。

在理解 ls -l 命令的输出时，我详细学习了各个字段的含义，包括权限部分、链接数、文件所有者、文件所属组、文件大小、修改时间和文件名。这不仅帮助我更好地理解了文件的详细信息，还为后续的操作提供了基础。此外，我还学习了如何使用 chmod 命令以符号模式和绝对模式改变文件权限。通过具体的实例操作，我掌握了这两种模式的使用方法，能够在不同场景下灵活应用。

在统计文件的行数、字数和字节数时，我使用了 wc 命令，并通过不同的选项（如 -l, -w, -c）单独统计行数、字数和字节数。这让我明白了如何根据具体需求选择合适的命令选项。另外，通过使用管道和重定向，我学会了如何将多个命令组合起来完成复杂的任务。例如，使用 man ls | more 将手册页分页显示，使用 cat file1 | head -20 | tee file5 截取文件的前20行内容并保存到新文件中，这些操作大大提高了我的文件管理能力。

通过这次上机实验，我不仅熟悉了常用的 Linux 命令，还提升了自主学习和解决问题的能力。遇到问题时，通过查阅文档和使用 man 命令，我学会了如何自己寻找解决方案。这种自主学习的能力在未来的学习和工作中将非常有用。此外，实践操作让我对 Linux 系统有了更深入的理解，发现了理论与实践之间的差距。未来，我将继续通过实践来巩固和提升自己的技能，不断提高自己的技术水平。