

---

# 计算机科学与技术学院

## 课程报告

课 程 名 称	Linux 应用实践
学 生 姓 名	
学 号	
专 业 班 级	
设 计 时 间	
地 点	

xxx 大学计算机学院

《Linux 应用实践》评分表

专业班级：\_\_\_\_\_ 学号：\_\_\_\_\_ 姓名：\_\_\_\_\_

支撑指标	评定项目	课程目标	评定标准	考核等级	分值范围(分)	得分
能够将数学、自然科学、工程基础和专业知用于解决软件工程应用领域复杂工程问题。	工程知识	课程目标 1	能将数学、自然科学、工程基础及软件工程专业知识与技术应用于解决软件工程应用领域复杂工程问题，包括判定系统的复杂性（工程解决能力）	优/ 良/ 合格/ 不合格	20	
能够应用数学、自然科学和工程科学的基本原理，识别、表达、并通过文献研究分析软件工程应用领域复杂工程问题，以获得有效结论。	问题分析	课程目标 2	能够通过文献调研、方案推理等方法给出软件工程应用领域复杂工程问题的多种解决方案并能确定适合具体问题的解决方案（方案分析）	优/ 良/ 合格/ 不合格	40	
能够设计针对软件工程应用领域的复杂工程问题的解决方案，设计满足特定需求的软硬件系统，并能够在设计环节中体现创新意识，考虑社会、健康、安全、法律、文化以及环境等因素。	设计/开发解决方案	课程目标 3	能够根据目标和解决方案，设计或开发满足特定需求的软件工程软硬件系统、组件及产品（设计开发）	优/ 良/ 合格/ 不合格	40	
课程设计评分老师签名：				总分		
问题及回答：						

---

## 目 录

实验一 Linux 基本命令的使用 .....	4
1、实验目的 .....	4
2、实验内容和步骤 .....	4
3、实验结论 .....	10
实验二 简单 Shell 程序设计 .....	20
1、实验目的 .....	20
2、实验内容和步骤 .....	20
3、实验结论 .....	26

---

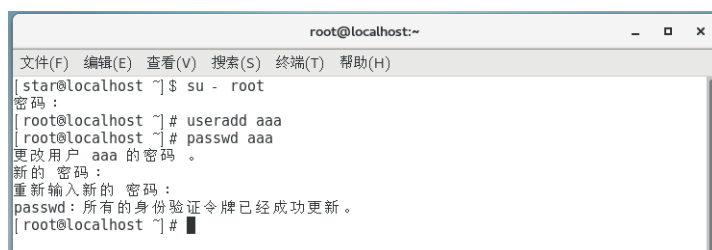
## 实验一 Linux 基本命令的使用

### 1、实验目的

学习和掌握 Linux 的基本命令。

### 2、实验内容和步骤（写出命令、结果或抓图）

**步骤 1：**以 root 登录 Linux，创建一个普通账户和口令。（useradd...）



```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[star@localhost ~]$ su - root  
密码：  
[root@localhost ~]# useradd aaa  
[root@localhost ~]# passwd aaa  
更改用户 aaa 的密码。  
新的 密码：  
重新输入新的 密码：  
passwd：所有的身份验证令牌已经成功更新。  
[root@localhost ~]#
```

**步骤 2：**使用新创建的用户账户和口令登录 Linux 系统，察看登录后的界面。



```
aaa@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[star@localhost ~]$ su - root  
密码：  
[root@localhost ~]# useradd aaa  
[root@localhost ~]# passwd aaa  
更改用户 aaa 的密码。  
新的 密码：  
重新输入新的 密码：  
passwd：所有的身份验证令牌已经成功更新。  
[root@localhost ~]# su - aaa  
[aaa@localhost ~]$
```

**步骤 3：**使用 pwd 命令察看当前的工作目录，然后用 ls 命令查看当前目录下的内容，尝试使用 -a, -l, -F, -A, -lF 等不同选项并比较不同之处。

```
star@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[star@localhost ~]$ pwd  
/home/star  
[star@localhost ~]$ ls  
公共 模板 视频 图片 文档 下载 音乐 桌面  
[star@localhost ~]$ ls -l  
总用量 0  
drwxr-xr-x. 2 star star 6 6月 23 16:36 公共  
drwxr-xr-x. 2 star star 6 6月 23 16:36 模板  
drwxr-xr-x. 2 star star 6 6月 23 16:36 视频  
drwxr-xr-x. 2 star star 53 6月 23 16:41 图片  
drwxr-xr-x. 2 star star 6 6月 23 16:36 文档  
drwxr-xr-x. 2 star star 6 6月 23 16:36 下载  
drwxr-xr-x. 2 star star 6 6月 23 16:36 音乐  
drwxr-xr-x. 2 star star 6 6月 23 16:36 桌面  
[star@localhost ~]$ ls -F  
公共/ 模板/ 视频/ 图片/ 文档/ 下载/ 音乐/ 桌面/  
[star@localhost ~]$ ls -a  
.  
..  
.bash_history .bashrc .esd_auth .mozilla  
[star@localhost ~]$ ls -A  
.bash_history .bashrc .esd_auth .mozilla 视频 下载  
.bash_logout .cache .ICEauthority 公共 图片 音乐  
.bash_profile .config .local 模板 文档 桌面  
[star@localhost ~]$ ls -lF  
公共/  
模板/  
视频/  
图片/  
文档/  
下载/  
音乐/  
桌面/  
[star@localhost ~]$
```

**步骤 4:** 在当前目录下建立一个名为 test 的新目录，然后将工作目录切换到 test 下，尝试将/etc 目录下的文件 passwd 拷贝到该目录下（cp 源文件 目的目录）。察看当前目录下的 passwd 文件的属主和文件权限。

```
[star@localhost ~]$ su - root  
密码：  
上一次登录：二 6月 23 16:39:54 CST 2020pts/0 上  
[root@localhost ~]# ^C  
[root@localhost ~]# cp /etc/passwd /root/test  
[root@localhost ~]# ls  
anaconda-ks.cfg initial-setup-ks.cfg test  
[root@localhost ~]# ls -l  
总用量 12  
-rw-r--r--. 1 root root 1523 6月 23 16:28 anaconda-ks.cfg  
-rw-r--r--. 1 root root 1571 6月 23 16:35 initial-setup-ks.cfg  
-rw-r--r--. 1 root root 2096 6月 23 17:05 test  
[root@localhost ~]#
```

**步骤 5:** 尝试向当前目录下的 passwd 文件和/etc/passwd 文件分别写入一些新内容（可使用 echo “字符串” >>文件的命令），看看操作能否成功，如果不能成功，请说明原因。用 cat 命令浏览文件 password 的内容，用 more 命令进行浏览翻页操作，再用 less 命令浏览文件的内容。比较这几个命令的不同之处

```
[root@localhost ~]# echo "123456">>passwd  
[root@localhost ~]# cat passwd  
123456  
[root@localhost ~]# more passwd  
123456  
[root@localhost ~]#
```

原因：对当前目录中成功，因为该目录下的 passwd 文件对当前用户具有写的权限。对 /etc/passwd 文件的操作被拒绝因为对当前用户不具有写的权限。

cat 命令将文件的内容全部显示，more 命令将文件内容分屏显示，less 命令进行刷新的全部显示

**步骤 6：**用 ls 命令查看 test 下文件的权限，用 mv 命令更改文件 password 的文件名为 test.txt, 尝试用 chown 和 chgrp 更改文件的属主为 root、组为 root，看看能否成功，不成功，请说明原因。尝试用 chmod 将文件权限为 “-rw-” 。看看能否成功，不成功，请说明原因。

```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[root@localhost ~]# cd  
[root@localhost ~]# cd..  
bash: cd.: 未找到命令...  
[root@localhost ~]# ls  
anaconda-ks.cfg  initial-setup-ks.cfg  passwd  test  test.txt  
[root@localhost ~]# mv passwd test  
[root@localhost ~]# ls -l  
总用量 16  
-rw- 1 root root 1523 6月 23 16:28 anaconda-ks.cfg  
-rw-r--r-- 1 root root 1571 6月 23 16:35 initial-setup-ks.cfg  
-rw-r--r-- 1 root root 2096 6月 23 17:31 test  
-rw-r--r-- 1 root root 0 6月 23 17:09 test.txt  
-rw-r--r-- 1 root root 7 6月 23 17:32 test  
[root@localhost ~]# chown root test  
[root@localhost ~]# chgrp root test  
[root@localhost ~]# ls -l  
总用量 16  
-rw- 1 root root 1523 6月 23 16:28 anaconda-ks.cfg  
-rw-r--r-- 1 root root 1571 6月 23 16:35 initial-setup-ks.cfg  
-rw-r--r-- 1 root root 2096 6月 23 17:31 test  
-rw-r--r-- 1 root root 0 6月 23 17:09 test.txt  
-rw-r--r-- 1 root root 7 6月 23 17:32 test  
[root@localhost ~]# chmod 600 test  
bash: chmod: 未找到命令...  
相似命令是: 'chmod'  
[root@localhost ~]# chmod 600 test  
[root@localhost ~]# ls -l  
总用量 16  
-rw- 1 root root 1523 6月 23 16:28 anaconda-ks.cfg  
-rw-r--r-- 1 root root 1571 6月 23 16:35 initial-setup-ks.cfg  
-rw-r--r-- 1 root root 2096 6月 23 17:31 test  
-rw-r--r-- 1 root root 0 6月 23 17:09 test.txt  
-rw- 1 root root 7 6月 23 17:32 test  
[root@localhost ~]#
```

**步骤 7：**用 rm 命令删除 test 目录下的所有文件，再用 rmdir 命令删除 test 目录。（想一想有没有一条命令将目录及目录下的所有文件删除，写出这条命令）

```
root@localhost: /
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[root@localhost ~]# chmod 600 test
[root@localhost ~]# ls -l
总用量 16
-rw-----. 1 root root 1523 6月 23 16:28 anaconda-ks.cfg
-rw-r--r--. 1 root root 1571 6月 23 16:35 initial-setup-ks.cfg
-rw-r--r--. 1 root root 2096 6月 23 17:31 test
-rw-r--r--. 1 root root 0 6月 23 17:09 test.txt
-rw-----. 1 root root 7 6月 23 17:32 text
[root@localhost ~]# rm *
rm: 是否删除普通文件 "anaconda-ks.cfg"? y
rm: 是否删除普通文件 "initial-setup-ks.cfg"? y
rm: 是否删除普通文件 "test"? y
rm: 是否删除普通文件 "test.txt"? y
rm: 是否删除普通文件 "text"? y
[root@localhost ~]# ls -l
总用量 0
[root@localhost ~]# cd ..
[root@localhost /]# ls -l
总用量 20
lrwxrwxrwx. 1 root root 7 6月 23 16:22 bin -> usr/bin
dr-xr-xr-x. 5 root root 4096 6月 23 16:35 boot
drwxr-xr-x. 19 root root 3240 6月 23 16:34 dev
drwxr-xr-x. 132 root root 8192 6月 23 16:41 etc
drwxr-xr-x. 4 root root 29 6月 23 16:40 home
lrwxrwxrwx. 1 root root 7 6月 23 16:22 lib -> usr/lib
lrwxrwxrwx. 1 root root 9 6月 23 16:22 lib64 -> usr/lib64
drwxr-xr-x. 2 root root 6 11月 5 2016 media
drwxr-xr-x. 2 root root 6 11月 5 2016 mnt
drwxr-xr-x. 3 root root 16 6月 23 16:25 opt
dr-xr-xr-x. 196 root root 0 6月 23 16:34 proc
dr-xr-xr-x. 5 root root 174 6月 23 17:37 root
drwxr-xr-x. 41 root root 1240 6月 23 16:37 run
lrwxrwxrwx. 1 root root 8 6月 23 16:22 sbin -> usr/sbin
drwxr-xr-x. 2 root root 6 11月 5 2016 srv
dr-xr-xr-x. 13 root root 0 6月 23 16:34 sys
drwxrwxrwt. 18 root root 4096 6月 23 17:31 tmp
drwxr-xr-x. 13 root root 155 6月 23 16:22 usr
drwxr-xr-x. 20 root root 282 6月 23 16:34 var
[root@localhost /]# rmdir aaa
```

命令: `rm -rf test`

**步骤 8:** 使用 `ps` 命令查看当前系统内的进程, 并利用 `man` 命令获取 `ps` 命令的参数, 写出获取当前终端进程执行情况的 `ps` 命令。

```
[root@localhost /]# ps
  PID TTY          TIME CMD
 49074 pts/0        00:00:00 su
 49082 pts/0        00:00:00 bash
 49509 pts/0        00:00:00 ps
[root@localhost /]# ps -man
  PID TTY          MAJFLT MINFLT   TRS    DRS  SIZE  SWAP   RSS   SHRD   LIB   DT COMMAND
 11525 tty1            1   26697   2240  298383 75156    -  43704    -    -    - /usr/bi
 49031 pts/0            0    2013    882  115417 29075    -   2804    -    -    - bash
 49074 pts/0            0    2306    24  220835 55215    -   4248    -    -    - su - ro
 49082 pts/0            0   4344    882  115269 29038    -   2784    -    -    - bash
 49525 pts/0            0     528    89  148850 37235    -   1452    -    -    - ps -man
[root@localhost /]#
```

**步骤 9:** 使用 `df` 命令查看当前系统已安装的文件系统的空间使用情况, 记录结果。

```

[root@localhost ~]# df
文件系统              1K-块      已用      可用  已用% 挂载点
/dev/mapper/centos-root 40137576 3209492 36928084    8% /
devtmpfs                917592      0   917592    0% /dev
tmpfs                   933512      0   933512    0% /dev/shm
tmpfs                   933512    9760   923752    2% /run
tmpfs                   933512      0   933512    0% /sys/fs/cgroup
/dev/sda1              1038336 182340 855996   18% /boot
/dev/mapper/centos-home 19593216 37424 19555792    1% /home
tmpfs                  186704    36 186668    1% /run/user/1000
tmpfs                  186704      0 186704    0% /run/user/0
[root@localhost ~]#

```

**步骤 10:** 使用 `du` 命令查看用户的工作目录占用了多少空间，记录结果。

```

root@localhost:~#
root@localhost:~# find /usr -type d | sort
0  /usr/local/lib64
0  /usr/local/libexec
0  /usr/local/sbin
4  /usr/local/share/applications
0  /usr/local/share/info
0  /usr/local/share/man/man1
0  /usr/local/share/man/man1x
0  /usr/local/share/man/man2
0  /usr/local/share/man/man2x
0  /usr/local/share/man/man3
0  /usr/local/share/man/man3x
0  /usr/local/share/man/man4
0  /usr/local/share/man/man4x
0  /usr/local/share/man/man5
0  /usr/local/share/man/man5x
0  /usr/local/share/man/man6
0  /usr/local/share/man/man6x
0  /usr/local/share/man/man7
0  /usr/local/share/man/man7x
0  /usr/local/share/man/man8
0  /usr/local/share/man/man8x
0  /usr/local/share/man/man9
0  /usr/local/share/man/man9x
0  /usr/local/share/man/man
0  /usr/local/share/man
4  /usr/local/share
0  /usr/local/src
4  /usr/local
0  /usr/src/debug
0  /usr/src/kernels
0  /usr/src
2980776 /usr
0  /media
0  /mnt
0  /opt/rh
0  /opt
0  /srv
3281108 .
|root@localhost ~/#

```

**步骤 11:** 使用 `free` 命令查看内存资源的使用情况，记录结果。

```
|root@localhost /| # free
```

	total	used	free	shared	buff/cache	available
Mem:	1867024	816272	77600	10872	973152	771164
Swap:	2097148	0	2097148			

```
|root@localhost /| #
```

**步骤 12:** 使用 `man` 获取 `tar` 和 `gzip` 的帮助信息，尝试将 `test` 目录下的文件打包并压缩，然后到另外一目录 `tmp` 下解包，写出这几条命令。

man tar

man gzip

```
tar -zcvf tar.gz -C /home/root/ahut1/tmp/
```

注意不要雷同 banban  
<https://github.com/dream4789/Computer-learning-resources.git>



**步骤 13:** 尝试执行 “ls -l > tmp”，看看这条命令的执行会出现什么结果，解释一下这条命令。

```
[root@localhost ~]# ls -l > tmp
[root@localhost ~]# ls
bin  dev  home  lib64  mnt  proc  run  srv  tem  usr
boot  etc  lib  media  opt  root  sbin  sys  tmp  var
[root@localhost ~]# cat tmp
总用量 20
lrwxrwxrwx. 1 root root 7 6月 23 16:22 bin -> usr/bin
dr-xr-xr-x. 5 root root 4096 6月 23 16:35 boot
drwxr-xr-x. 19 root root 3240 6月 23 16:34 dev
drwxr-xr-x. 132 root root 8192 6月 23 16:41 etc
drwxr-xr-x. 4 root root 29 6月 23 16:40 home
lrwxrwxrwx. 1 root root 7 6月 23 16:22 lib -> usr/lib
lrwxrwxrwx. 1 root root 9 6月 23 16:22 lib64 -> usr/lib64
drwxr-xr-x. 2 root root 6 11月 5 2016 media
drwxr-xr-x. 2 root root 6 11月 5 2016 mnt
drwxr-xr-x. 3 root root 16 6月 23 16:25 opt
dr-xr-xr-x. 196 root root 0 6月 23 16:34 proc
dr-xr-xr-x. 5 root root 174 6月 23 17:37 root
drwxr-xr-x. 41 root root 1240 6月 23 16:37 run
lrwxrwxrwx. 1 root root 8 6月 23 16:22 sbin -> usr/sbin
drwxr-xr-x. 2 root root 6 11月 5 2016 srv
dr-xr-xr-x. 13 root root 0 6月 23 16:34 sys
-rw-r--r--. 1 root root 0 6月 23 17:53 tem
drwxrwxrwt. 18 root root 4096 6月 23 17:51 tmp
drwxr-xr-x. 13 root root 155 6月 23 16:22 usr
drwxr-xr-x. 20 root root 282 6月 23 16:34 var
[root@localhost ~]#
```

**步骤 14:** 尝试执行 find /usr/src -name \*.c -print | xargs grep “#include”，看看这条命令的执行会出现什么结果，解释一下这条命令。

```
[root@localhost ~]# find /usr/src -name *.c -print | xargs grep "#include"
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/kallsyms.c:#include <stdio.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/kallsyms.c:#include <stdlib.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/kallsyms.c:#include <string.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/kallsyms.c:#include <ctype.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/kallsyms.c:    printf("#include
<asm/types.h>\n");
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:#include <ctype.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:#include <errno.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:#include <stdarg.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:#include <stdio.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:#include <stdlib.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:#include <string.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:#include <unistd.h>
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/pnmtologo.c:    fputs("#include
<linux/linux_logo.h>\n\n", out);
/usr/src/kernels/3.10.0-957.el7.x86_64/scripts/recordmcount.c:#include <sys/ty
```

查询 /usr/src 下所有.c 文件并筛选出带#include 的行并打印

**步骤 15:** 执行 cal 和 date 命令，说说这两条指令的用途。

```
[root@localhost ~]# cal
      六月 2020
日 一 二 三 四 五 六
 1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30

[root@localhost ~]# date
2020年 06月 23日 星期二 18:09:54 CST
[root@localhost ~]#
```

cal 显示当前月份的日历，date 显示当前所在的日期和时间。

**步骤 16:** 执行命令 clear 和 logout，退出系统。（想一想有没有其他的方法，写出这种方法）

```
star@localhost:~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[root@localhost ~]# logout
[star@localhost ~]$
```

exit

**步骤 17:** 执行命令 shutdown，关闭系统。（想一想有没有更简单的命令，写出这条命令）

```
[root@localhost ~]# shutdown
Shutdown scheduled for 二 2020-06-23 18:17:31 CST, use 'shutdown -c' to cancel.
[root@localhost ~]#
Broadcast message from root@localhost.localdomain (Tue 2020-06-23 18:16:31 CST):
The system is going down for power-off at Tue 2020-06-23 18:17:31 CST!
```

- 1、halt 立刻关机
- 2、poweroff 立刻关机
- 3、shutdown-h now 立刻关机(root 用户使用)
- 4、shutdown-h 10 10 分钟后自动关机

### 3、实验结论

掌握 Linux 的基本命令很重要，对于基本命令知道越多使用 Linux 就越方便越顺手。

1. ls 命令：

格式：：ls [选项] [目录或文件]

功能：对于目录，列出该目录下的所有子目录与文件；对于文件，列出文件名以及其他信息。

常用选项：

注意不要雷同  
banban  
<https://github.com/dream4789/Computer-learning-resources.git>

- 
- a : 列出目录下的所有文件, 包括以 . 开头的隐含文件。
  - d : 将目录像文件一样显示, 而不是显示其他文件。
  - i : 输出文件的 i 节点的索引信息。
  - k : 以 k 字节的形式表示文件的大小。
  - l : 列出文件的详细信息。
  - n : 用数字的 UID, GID 代替名称。
  - F : 在每个文件名后面附上一个字符以说明该文件的类型, “\*”表示可执行的普通文件; “/”表示目录; “@”表示符号链接; “l”表示 FIFOs; “=”表示套接字。

## 2. cd 命令

格式: cd [目录名称]

常用选项:

cd .. 返回上一级目录。

cd ../.. 将当前目录向上移动两级。

cd - 返回最近访问目录。

## 3. pwd 命令

格式: pwd

功能: 显示出当前工作目录的绝对路径。

## 4. touch 命令

格式: touch[选项] 文件名...

功能: touch 命令参数可以更改文档或目录的日期时间, 包括存取时间和更改时间, 或者新建一个不存在的文件。

常用选项:

-a 仅改变指定文件的存取时间。

-c 或 -no-creat 不创建任何文件。

-m 仅改变指定文件的修改时间。

-d 使用指定的日期时间, 而非现在的时间。

-f 此参数将忽略不予处理, 仅负责解决 BSD 版本 touch 指令的兼容性问题。

---

## 5. mkdir 命令

格式: mkdir [选项] dirname...

功能: mkdir 命令用来创建目录。

常用选项:

-p -parents 可以是一个路径名称。此时若路径竞争的某些目录尚不存在, 加上此选项后, 系统将自动建立好那些尚不存在的目录, 即一次可以建立多个目录。

-m -mode=MODE 将新建目录的存取权限设置为 MODE, 存取权限用给定的八进制数字表示。

## 6. rm 命令

格式: rm [选项] 文件列表

功能: rm 命令删除文件或目录。

常用选项:

-f -force 忽略不存在的文件, 并且不给出提示信息。

-r -R, -recursive 递归地删除指定目录及其下属的各级子目录和相应的文件。

-i 交互式删除文件。

说明: rm 命令删除指定的文件, 默认情况下, 它不能删除目录。如果文件不可写, 则标准输入是 tty (终端设备)。如果没有给出选项 -f 或者 -force, rm 命令删除之前会提示用户是否删除该文件; 如果用户没有回答 y 或者 Y, 则不删除该文件。

## 7. rmdir 命令

格式: rmdir [选项] dirname

功能: 删除目录。

常用选项:

-p -parents 递归删除目录 dirname, 当子目录删除后其父目录为空时, 也一同被删除。如果有非空的目录, 则该目录保留下来。

## 8. man 命令

格式: man [选项] 命令

功能: man 命令格式化并显示某一命令的联机帮助手册页。

常用选项:

---

-k 根据关键字搜索联机帮助。

num 只在第 num 章节找。

-a 将所有章节的都显示出来。

说明：面手册分为 8 章：

1. 一般用户的命令；
2. 系统调用；
3. C 语言函数库；
4. 有关驱动程序和系统设备的解释；
5. 配置文件的解释；
6. 游戏程序的命令；
7. 有用的杂类命令，如宏命令包等；
8. 有关系统维护 and 管理的命令。

#### 9. cp 命令

格式：cp [选项] 源文件或目录 目标文件或目录

功能：复制文件或目录。

常用选项：

-f -force 强行复制文件或目录，不论文件或目录是否存在。

-d 复制时保留文件链接。

-i -interactive 覆盖文件之前先询问用户。

-r 递归处理，将指定目录下的文件与子目录一并处理。若源文件或目录的形态，不属于目录或符号链接，则一律视为普通文件处理。

-R 或 -recursive 递归处理，将指定目录下的文件及子目录一并处理。

#### 10. mv 命令

格式：mv [选项] 源文件或目录 目标文件或目录

功能：mv 命令对文件或目录重新命名，或者将文件从一个目录移到另一个目录中。

常用选项：

-f force 强制的意思，如果目标文件已经存在，不会询问而直接覆盖。

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>

---

-i 若目标文件（destination）已经存在时，就会询问是否覆盖。

## 11. cat/tac 命令

格式：cat [选项] [文件]

功能：查看目标文件的内容。

常用选项：

-b 对非空输出行编号。

-n 对输出的所有行编号。

-s 不输出多行空行。

## 12. more 命令

格式：more [选项] [文件]

功能：more 命令显示文件内容，每次显示一屏。

常用选项：

-n 对输出的所有行编号。

-s 将文件中连续的空白行压缩成一个空白行显示。

-num 这个选项制定一个整数，表示一屏显示多少行。

q 退出 more。

## 13. less 命令

格式：less [参数] 文件

功能：less 命令与 more 命令类似，但二者存在差别，less 命令允许用户向前或向后浏览文件，而 more 命令只能向前浏览。

常用选项：

-i 忽略搜索时的大小写。

-N 显示每行的行号。

## 14. head 命令

格式：head [选项] [文件]

功能：head 命令在屏幕上显示指定文件的开头若干行。

常用选项：

---

`-c -bytes=[-]N` 显示每个文件前面 N 字节。

`-n -lines=[-]N` 显示指定文件的前面 N 行。

## 15. tail 命令

格式: `tail [选项] [文件]`

功能: 用于显示指定文件的末尾, 不指定文件时, 作为输入信息进行处理。常用查看日志文件。

说明: `tail` 命令从指定点开始将文件写到标注输出。使用 `tail` 命令的 `-f` 选项可以方便的查阅正在改变的日志文件, `tail -f filename` 会把 `filename` 里最尾部的内容显示在屏幕上, 并且不断刷新, 使你看到最新的文件内容。

常用选项:

`-f` 循环读取。

`-n <行数>` 显示行数。

## 16. 时间相关的命令

`date` 显示

`date` 指定格式显示时间: `date+%Y: %m: %d`

`date` 用法: `date: date[OPTION]... [+FORMAT]`

`%H` 小时 (00..23)

`%M` 分钟 (00..59)

`%S` 秒 (00..61)

`%X` 相当于 `%H: %M: %S`

`%d` 日 (01..31)

`%m` 月份 (01..12)

`%Y` 完整年份 (0000..9999)

`%F` 相当于 `%Y-%m-%d`

时间戳

时间->时间戳: `date+%s`

时间戳->时间: Unix 时间戳是从 1970 年 1 月 1 日开始所经过的秒数, 不考虑闰秒。

## 17. cal 命令

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>

---

格式: `cal [参数] [月份] [年份]`

功能: 用于查询日历等时间信息, 如果只有一个参数, 则表示年份 (1-9999), 如果有两个参数, 则表示月份和年份。

常用选项:

`-3` 显示系统前一个月, 当前月, 下一个月的月历。

`-j` 显示在当年中的第几天 (一年日期按天算, 从 1 月 1 日算起, 默认显示当前月在一年中的天数)。

`-y` 显示当前年份的日历。

#### 18. find 命令

格式: `find pathname -options`

功能: 用于在文件树中查找文件, 并作出相应的处理 (可能访问磁盘)。

常用选项:

`-name` 按照文件名查找文件。

#### 19. grep 命令

格式: `grep [选项] 搜寻字符串 文件`

功能: 在文件中搜索字符串, 将找到的行打印出来。

常用选项:

`-i` 忽略大小写的不同, 所以大小写视为相同。

`-n` 顺便输出行号。

`-v` 反向选择, 亦即显出没有 ‘搜寻字符串’ 内容的那一行。

#### 20. zip/unzip 命令

格式: `zip` 压缩文件 `zip` 目录或文件

功能: 将目录或文件压缩成 `zip` 格式。

常用选项:

`-r` 递归处理, 将指定目录下的所有文件和子目录一并处理。

#### 21. tar 命令

格式: `tar [-cxtzjvf] 文件与目录... 参数`

功能: 打包/解包, 不打开它, 直接看内容。

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>



---

-c 建立一个压缩文件的参数指令（create 的意思）。

-x 解开一个压缩文件的参数指令。

-t 查看 tarfile 里面的文件。

-z 是否同时具有 gzip 的属性？亦即是否需要用 gzip 压缩？

-j 是否同时具有 bzip 的属性？亦即是否需要用 bzip 压缩？

-v 压缩的过程中显示文件。这个常用，不建议用在背景执行过程。

-f 使用档名，请注意，在 f 之后要立即接档名，不要再加参数。

-C 解压到指定目录。

## 22. bc 命令

bc 命令可以很方便的进行浮点运算。

## 23. uname -r 命令

格式：uname [选项]

功能：uname 用来获取电脑和操作系统的相关信息。

常用选项：

-a 或 -all 详细输出所有信息，依次为内核名称，主机名，内核版本号，内核版本，硬件名，处理器类型，硬件平台类型，操作系统名称。

## 三、Linux 系统根目录下各个目录的作用

/bin 二进制可执行命令。该目录下存放着普通用户的命令

/dev 系统的设备文件，即设备的驱动程序

/home 存放用户文件的主目录，用户数据

/lib 存放着和系统运行相关的库文件

/mnt 存放临时的映射文件，通常是一些用来安装其他设备的子目录

---

/boot 存放启动 linux 的核心文件

/media 存放着可移除的设备，比如软盘，光盘

/misc 储存着一些特殊的字符的定义

/net 存放着和网络相关的一些文件

/proc 存放着用户与内核的交互信息

/sbin 系统的管理命令，这里存放的是系统管理员使用的程序

/srv 系统启动服务时可以访问的数据库目录

/tmp 临时文件，重启后自动清空

/var 存放系统产生的经常变化的文件

/etc 系统所有的配置文件都在这个目录中

/opt (option : 自由选择)主要给源码安装软件时选择的安装目录位置

/root 超级用户的目录

/selinux 主要用来加固操作系统，提高系统的安全性

/sys 管理设备文件

---

/usr 最大的目录，存放着应用程序和文件

/lost-found 这个目录平时是空的，当系统非正常关机而留下的“无家可归”的文件便会储存在这里

## 实验二 简单 Shell 程序设计

### 1、实验目的

- (1) 学习和掌握 vi 编辑器的基本使用方法。
- (2) 学习和掌握编写简单的 shell 程序。

### 2、实验内容和步骤

#### 2.1 vi 的使用

**步骤 1:** 以 root 用户身份登录进入 Linux 系统，要求给出抓图。



**步骤 2:** 启动 Vi，切换到输入模式，输入一段英文：，要求给出抓图。

it is an exercise!

we use Vi to edit it.

Left, down, up, right,

安徽工业大学 计算机学院 姓名 XXXX 学号 XXXX 籍贯 XXXX



**步骤 3:** 尝试匹配其中的一串字符，如 it，写出这命令及执行结果或抓图。

**1: 键入 Esc 进入命令模式;**

**结果:**



注意不要雷同  
<https://github.com/dream4789/Computer-learning-resources.git>

**步骤 4:** 尝试替换其中的一串字符，写出这命令及执行结果或抓图。

1. 命令模式下：9s

Rabcd：从当前光标开始的字符依次替换为 abcd

2. 结果



```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
it is an exercise!  
we use Vi to edit it.  
abcd,down,up,right,  
安徽工业大学 计算机学院 姓名张文迪 学号199074368 籍贯安徽省阜阳市
```

**步骤 5:** 尝试复制/删除其中的一行或几行文本，写出命令及执行结果或抓图。

1. 命令模式：yy p

2. 结果:



```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
it is an exercise!  
we use Vi to edit it.  
abcd,down,up,right,  
abcd,down,up,right,  
安徽工业大学 计算机学院 姓名张文迪 学号199074368 籍贯安徽省阜阳市
```

**步骤 6:** 尝试复制/删除其中的一个单词或几个字符，写出命令及执行结果或抓图。

1. 命令模式：5yw，移动光标至下一行

2. 结果



```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
it is an exercise!  
we use Vi to edit it.  
abcd,down,up,right,  
abcd,down,up,right,  
安徽工业大学 计算机学院 姓名张文迪 学号199074368 籍贯安徽省阜阳市
```

**步骤 7:** 尝试使用鼠标复制文本，写出命令及执行结果或抓图。



```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
it is an exercise!  
we use Vi to edit it.  
abcd,down,up,right,  
abcd,down,up,right,  
安徽工业大学 计算机学院 姓名张文迪 学号199074368 籍贯安徽省阜阳市  
t is an exercise!  
█
```

注意不要雷同

banban

<https://github.com/dream4789/Computer-learning-resources.git>

---

**步骤 8:** 尝试存盘退出操作，写出命令，抓图。

### 1. 命令模式: wq



---

## 2.2 简单的 shell 程序编写，要求给出程序和运行结果

**第一题:** 请详细查看如下几个数字的规律，并使用 shell 脚本输出后面的 20 个数字。

10 31 53 77 105 141 .....

程序:

```
#!/bin/bash
```

```
##
```

```
echo "====后面的十个数字为: "
```

```
i=10
```

```
for n in `seq 0 15`
```

```
do
```

.....  
<https://github.com/dream4789/Computer-learning-resources.git>

---

```
a=$((2**$n))

b=$((20+$a))

let "i+= $b"

if [ $n -lt 15 ] && [ $n -ge 5 ]

then

    echo $i

fi

done
```

结果:



```
root@localhost:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[ root@localhost ~] # vi abc.sh  
[ root@localhost ~] # source abc.sh  
10  
31  
53  
77  
105  
141  
193  
277  
425  
701  
1233  
2277  
4345  
8461  
16673  
33077  
65865  
131421  
262513  
524677  
1048985  
2097581  
4194753  
8389077  
16777705  
33554941  
67109393  
[ root@localhost ~] #
```

**第二题:** 打印 99 乘法表, 要求对齐。

程序:

---

```
#!/bin/bash

for ((i=1; i<=9; i++)); do

    for ((j=1; j<=i; j++))

        do

            printf "$i*$j=$((i*$j))\t"

        done

    echo ""

done
```

结果:

```
| root@localhost ~] # vi bcd.sh
| root@localhost ~] # source bcd.sh
1*1=1
2*1=2  2*2=4
3*1=3  3*2=6  3*3=9
4*1=4  4*2=8  4*3=12  4*4=16
5*1=5  5*2=10  5*3=15  5*4=20  5*5=25
6*1=6  6*2=12  6*3=18  6*4=24  6*5=30  6*6=36
7*1=7  7*2=14  7*3=21  7*4=28  7*5=35  7*6=42  7*7=49
8*1=8  8*2=16  8*3=24  8*4=32  8*5=40  8*6=48  8*7=56  8*8=64
9*1=9  9*2=18  9*3=27  9*4=36  9*5=45  9*6=54  9*7=63  9*8=72  9*9=81
| root@localhost ~] #
```

**第三题:** 编写 calCircleArea, calRectArea, calCircleCylinderVolume (计算圆柱体体积)、calRectCylinderVolume (计算矩形柱体体积), 体积函数要求调用面积函数。

程序:

```
#!/bin/bash

echo "输入半径"

read r

echo "输入圆柱体高"

read h

echo "输入矩形长、宽"
```



---

```
read a

read b

echo "输入矩形柱体高"

read c

calCircleArea()

{

    area1=`echo 3.14*$1*$1 | bc`

}
```

```
calCircleCylinderVolume()

{

    v1=`echo $1*$2 | bc`

}
```

```
calRectArea()

{

    area2=`echo $1*$2 | bc`

}
```

```
calRectCylinderVolume()

{

    v2=`echo $1*$2 | bc`

}
```

---

```
calCircleArea $r

calCircleCylinderVolume $area1 $h

calRectArea $a $b

calRectCylinderVolume $area2 $c


echo "圆面积 = $area1"

echo "圆柱体体积 = $v1"

echo "矩形面积 = $area2"

echo "矩形柱体体积 = $v2"
```

结果：



```
[root@localhost ~]# vi a.sh
[root@localhost ~]# source a.sh
输入半径
2
输入圆柱体高
3
输入矩形长、宽
4
5
输入矩形柱体高
6
圆面积 = 12.56
圆柱体体积 = 37.68
矩形面积 = 20
矩形柱体体积 = 120
[root@localhost ~]#
```

### 3、实验结论

**vi** 编辑器使用很方便,掌握 **vi** 的基本命令是灵活使用 **vi** 的基础,**shell** 可以作为 **Linux** 的高级语言来编写我们需要的代码,是我们进一步了解使用 **Linux** 的有力工具。