

知识点 1 【linux 命令的概述】

以根目录/开始的。

```
edu@edu:~$ cd /
edu@edu:/$ ls
bin    dev    initrd.img  lost+found  opt    run    srv    usr
boot   etc    lib         media       proc   sbin   sys    var
cdrom  home  lib64      mnt         root   snap   tmp    vmlinuz
edu@edu:/$
```

命令的格式：命令 选项 参数

命令：就是命令名

选项：一般以-开头 比如：-a -p -r 等 扩展命令的功能。

参数：描述命令的目标

知识点 2 【linux 常用命令】

1、--help 命令的帮助信息

需要查看的命令 --help

2、man 查看命令、库函数、系统调用的帮助信息

man 是以章节的方式 管理命令、库函数、系统调用

第 1 章节 默认是命令

第 2 章节 是系统调用

第 3 章节 是库函数

man 1 ls 从第 1 章节中查找 ls 命令

man 2 open 从第 2 章节查找系统调用函数 open

man 3 strcpy 从第 3 章节中查找库函数 strcpy

如果省略章节 默认为第 1 章节：比如 man ls

3、tab 自动补全的命令

4、history 查看历史命令

```
edu@edu:~/work$ history
 1  ls
 2  cd vmware-tools-distrib/
 3  ls
 4  sudo ./vmware-install.pl
 5  clear
 6  ls
 7  clear
```

5、重定向命令 >

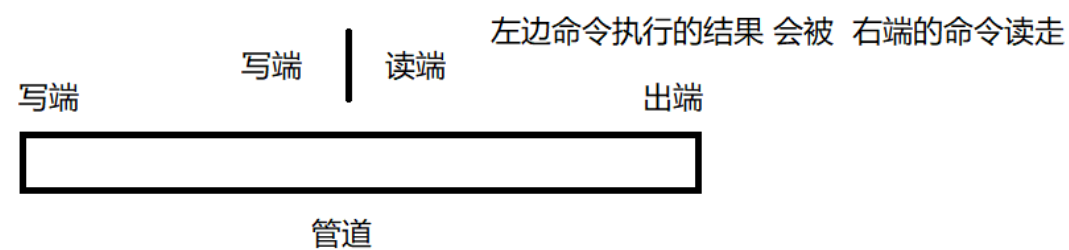
echo "hello world" > a.txt 将本应该显示到终端上的字符串 输入到 a.txt 文件中

>是覆盖源文件

echo "hello world" >> a.txt 将本应该显示到终端上的字符串 输入到 a.txt 文件中

>> 在文件末尾追加一行输入

6、管道命令 |



ps -A 将所有的进程信息 显示到终端上

grep 查找命令，比如：grep hehe 从终端的输入中查找 hehe

```
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work$ ps -A | grep ssh
5218 ?      00:00:00 sshd
8535 ?      00:00:00 sshd
8553 ?      00:00:00 sshd
8565 ?      00:00:00 sshd
8606 ?      00:00:00 sshd
edu@edu:~/work$ \
```

grep 从 ps -A 的结果找查找 ssh

7、ls 命令 默认查看当前目录下的文件信息

```
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work$ ls
a.txt  test
edu@edu:~/work$
```

```
edu@edu:~/work$ ls -a
.  ..  a.txt  test
edu@edu:~/work$
```

显示所有文件 包含隐藏文件

上一级目录

代表当前目录

```
edu@edu:~/work$ ls
a.txt  test
edu@edu:~/work$ touch .c.txt
edu@edu:~/work$ ls
a.txt  test
edu@edu:~/work$ ls -a
.  ..  a.txt  .c.txt  test
edu@edu:~/work$
```

```
edu@edu:~/work$ ls
```

```
a.txt test
```

```
edu@edu:~/work$ ls -a -l
```

1

以列表的形式显示

```
总用量 16
```

```
drwxrwxr-x 3 edu edu 4096 7月 18 10:10 .
drwxr-xr-x 20 edu edu 4096 7月 18 09:06 ..
-rw-rw-r-- 1 edu edu 22 7月 18 09:42 a.txt
-rw-rw-r-- 1 edu edu 0 7月 18 10:10 .c.txt
drwxrwxr-x 2 edu edu 4096 7月 18 10:08 test
```

```
edu@edu:~/work$
```

```
edu@edu:~/work$ ls -a -l
```

```
总用量 224
```

```
drwxrwxr-x 3 edu edu 4096 7月 18 10:13 .
drwxr-xr-x 20 edu edu 4096 7月 18 09:06 ..
-rwxrw-r-- 1 edu edu 209463 4月 11 14:44 001.jpg
-rw-rw-r-- 1 edu edu 22 7月 18 09:42 a.txt
-rw-rw-r-- 1 edu edu 0 7月 18 10:10 .c.txt
drwxrwxr-x 2 edu edu 4096 7月 18 10:08 test
```

```
edu@edu:~/work$ ls -a -l -h
```

-h人性化的方式显示文件大小

```
总用量 224K
```

```
drwxrwxr-x 3 edu edu 4.0K 7月 18 10:13 . 一般配合-l使用
drwxr-xr-x 20 edu edu 4.0K 7月 18 09:06 ..
-rwxrw-r-- 1 edu edu 205K 4月 11 14:44 001.jpg
-rw-rw-r-- 1 edu edu 22 7月 18 09:42 a.txt
-rw-rw-r-- 1 edu edu 0 7月 18 10:10 .c.txt
drwxrwxr-x 2 edu edu 4.0K 7月 18 10:08 test
```

```
edu@edu:~/work$
```

-a -l -h 没有顺序

```

edu@edu:~/work$ ls -a -h -l
总用量 224K
drwxrwxr-x  3 edu  edu  4.0K 7月  18 10:13 .
drwxr-xr-x 20 edu  edu  4.0K 7月  18 09:06 ..
-rwxrw-r--  1 edu  edu 205K 4月  11 14:44 001.jpg
-rw-rw-r--  1 edu  edu   22 7月  18 09:42 a.txt
-rw-rw-r--  1 edu  edu    0 7月  18 10:10 .c.txt
drwxrwxr-x  2 edu  edu  4.0K 7月  18 10:08 test
edu@edu:~/work$ ls -l -a -h
总用量 224K
drwxrwxr-x  3 edu  edu  4.0K 7月  18 10:13 .
drwxr-xr-x 20 edu  edu  4.0K 7月  18 09:06 ..
-rwxrw-r--  1 edu  edu 205K 4月  11 14:44 001.jpg
-rw-rw-r--  1 edu  edu   22 7月  18 09:42 a.txt
-rw-rw-r--  1 edu  edu    0 7月  18 10:10 .c.txt
drwxrwxr-x  2 edu  edu  4.0K 7月  18 10:08 test
edu@edu:~/work$ ls -alh
总用量 224K
drwxrwxr-x  3 edu  edu  4.0K 7月  18 10:13 .
drwxr-xr-x 20 edu  edu  4.0K 7月  18 09:06 ..
-rwxrw-r--  1 edu  edu 205K 4月  11 14:44 001.jpg
-rw-rw-r--  1 edu  edu   22 7月  18 09:42 a.txt
-rw-rw-r--  1 edu  edu    0 7月  18 10:10 .c.txt
drwxrwxr-x  2 edu  edu  4.0K 7月  18 10:08 test
edu@edu:~/work$

```

8、tree 树状显示目录文件信息

默认 ubuntu 不支持 tree 需要安装

sudo apt-get update

sudo apt-get install tree

```

edu@edu:~/work$ tree
.
├── 001.jpg
├── a.txt
└── test
    └── b.txt

1 directory, 3 files
edu@edu:~/work$ ls
001.jpg  a.txt  test
edu@edu:~/work$

```

9、clear 清屏

10、cd 切换目录的命令

cd 具体的目录 进入目录

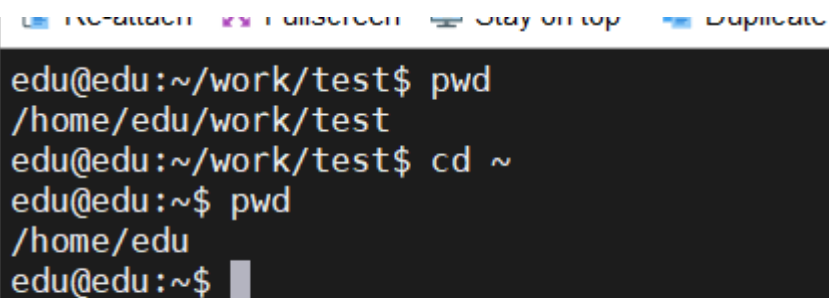
cd ..返回上一级

cd ~ 进入家目录

cd 进入家目录

cd -进入上一次目录

11、pwd 显示你的位置

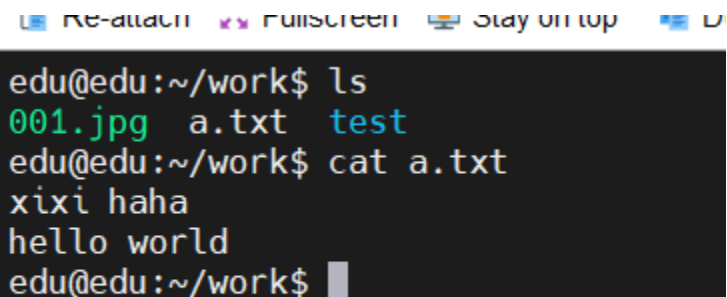


A terminal window with a dark background and light-colored text. At the top, there are window control buttons: 'Re-attach', 'Fullscreen', 'Stay on top', and 'Duplicate'. The terminal shows the following commands and output:

```
edu@edu:~/work/test$ pwd
/home/edu/work/test
edu@edu:~/work/test$ cd ~
edu@edu:~$ pwd
/home/edu
edu@edu:~$
```

12、cat 命令 查看文件的内容

cat 将文件的内容 直接显示的终端上。



A terminal window with a dark background and light-colored text. At the top, there are window control buttons: 'Re-attach', 'Fullscreen', 'Stay on top', and 'Duplicate'. The terminal shows the following commands and output:

```
edu@edu:~/work$ ls
001.jpg  a.txt  test
edu@edu:~/work$ cat a.txt
xixi haha
hello world
edu@edu:~/work$
```

13、rm 删除文件

1、删除文件 rm 文件名

```
edu@edu:~/work$ ls
001.jpg a.txt test
edu@edu:~/work$ rm 001.jpg
edu@edu:~/work$ ls
a.txt test
edu@edu:~/work$
```

```
edu@edu:~/work$ ls
a.txt test
edu@edu:~/work$ touch b.txt c.txt
edu@edu:~/work$ ls
a.txt b.txt c.txt test
edu@edu:~/work$ rm *.txt
edu@edu:~/work$ ls
test
edu@edu:~/work$
```


删除.txt结尾的所有文件

2、如果 rm 删除文件夹 必须夹-r

rm test -r 删除文件夹 test

3、如果想要强制删除 加-f


```
edu@edu:~/work$ ls
edu@edu:~/work$ touch a.txt
edu@edu:~/work$ ls -l
总用量 0
-rw-rw-r-- 1 edu edu 0 7月 18 10:48 a.txt
edu@edu:~/work$ chmod 0444 a.txt
edu@edu:~/work$ ls
a.txt
edu@edu:~/work$ ls -l
总用量 0
-r--r--r-- 1 edu edu 0 7月 18 10:48 a.txt
edu@edu:~/work$ rm a.txt
rm: 是否删除有写保护的普通空文件 'a.txt'? n
edu@edu:~/work$ ls
a.txt
edu@edu:~/work$ rm a.txt -f
edu@edu:~/work$ ls
edu@edu:~/work$
```



强制删除文件

Re-attach Fullscreen Stay on top Duplicate

```
edu@edu:~/work$ ls
edu@edu:~/work$ sudo rm /* -rf
```



别试

14、cp 文件或目录的拷贝

1、拷贝文件

cp 源文件 目的目录。


```

edu@edu:~/work$ tree
.
├── a.txt
└── test

1 directory, 1 file
edu@edu:~/work$ cp a.txt test
edu@edu:~/work$ tree
.
├── a.txt
└── test
    └── a.txt

1 directory, 2 files
edu@edu:~/work$

```

2、拷贝文件夹 -r

cp 源目录 目的目录 -r

```

edu@edu:~/work$ tree
.
├── a
├── a.txt
└── test
    └── a.txt

2 directories, 2 files
edu@edu:~/work$ cp test a -r
edu@edu:~/work$ tree
.
├── a
│   └── test
│       └── a.txt
├── a.txt
└── test
    └── a.txt

3 directories, 3 files
edu@edu:~/work$

```

拷贝文件夹

3、备份文件

cp 源文件 目的文件

```
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work$ ls
a a.txt test
edu@edu:~/work$ cp a.txt b.txt
edu@edu:~/work$ ls
a a.txt b.txt test
edu@edu:~/work$ cat b.txt
"hellworld"
edu@edu:~/work$
```

1 给文件取别名

cp 源目录 目的目录 （如果目的目录不存在 就是给源目录备份）

```
edu@edu:~/work$ cp a hehe -r
edu@edu:~/work$ ls
a a.txt b.txt hehe test
edu@edu:~/work$ tree
.
├── a
│   └── test
│       └── a.txt
├── a.txt
├── b.txt
├── hehe
│   └── test
│       └── a.txt
└── test
    └── a.txt

5 directories, 5 files
edu@edu:~/work$
```

cp 源目录 目的目录 （如果目的目录存在 就是将源目录拷贝到目的目录下）

15、mv 剪切文件或目录

1、剪切文件

mv 源文件 目的目录

```
edu@edu:~/work$ ls
a.txt  test
edu@edu:~/work$ tree
.
├── a.txt
└── test

1 directory, 1 file
edu@edu:~/work$ mv a.txt test
edu@edu:~/work$ tree
.
├── test
└── a.txt

1 directory, 1 file
edu@edu:~/work$
```

2、剪切文件夹

```
Re-attach  Fullscreen  Stay on top  Duplicate  [icons]
edu@edu:~/work$ ls
hehe  test
edu@edu:~/work$ mv test hehe
edu@edu:~/work$ tree
.
├── hehe
│   └── test
│       └── a.txt
└── test

2 directories, 1 file
edu@edu:~/work$
```

不用加-r

3、给文件重命名

mv 源文件 目的文件

```
edu@edu:~/work$ ls
a.txt  hehe
edu@edu:~/work$ mv a.txt b.txt
edu@edu:~/work$ ls
b.txt  hehe
edu@edu:~/work$
```

4、给文件夹重命名

```
edu@edu:~/work$ ls
b.txt  hehe
edu@edu:~/work$ mv hehe test
edu@edu:~/work$ ls
b.txt  test
edu@edu:~/work$
```

如果当前不存在test
那就是将hehe重命名为test

16、mkdir 创建文件夹

默认是在当前目录下 创建文件夹

```
edu@edu:~/work$ ls
b.txt  test
edu@edu:~/work$ mkdir xixi
edu@edu:~/work$ ls
b.txt  test  xixi
edu@edu:~/work$ mkdir haha lala heihei
edu@edu:~/work$ ls
b.txt  haha  heihei  lala  test  xixi
edu@edu:~/work$
```

```
edu@edu:~/work$ mkdir a/b/c -p
edu@edu:~/work$ tree
.
├── a
│   └── b
│       └── c
└── 
```

3 directories, 0 files
edu@edu:~/work\$

如果路径中某个目录不存在 加-p可以创建该目录
然后继续创建剩下的目录

17、touch 只能创建文件 不能编辑文件

```
edu@edu:~/work$ ls
a
edu@edu:~/work$ touch a.txt
edu@edu:~/work$ ls
a a.txt
edu@edu:~/work$ touch b.txt c.txt
edu@edu:~/work$ ls
a a.txt b.txt c.txt
edu@edu:~/work$
```

18、find 查找文件的命令

用法：find 路径 -name 文件名

```
edu@edu:~/work$ ls
a a.txt b.txt c.txt
edu@edu:~/work$ sudo find /* -name sources.list
[sudo] edu 的密码:
/etc/apt/sources.list
find: `/run/user/1000/gvfs': 权限不够
/usr/share/doc/apt/examples/sources.list
edu@edu:~/work$ sudo find /* -name smb.conf
/etc/samba/smb.conf
/run/samba/upgrades/smb.conf
find: `/run/user/1000/gvfs': 权限不够
/usr/share/samba/smb.conf
/usr/share/doc/nautilus-share/examples/smb.conf
edu@edu:~/work$
```

```
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work$ tree
.
├── a
│   └── b
│       └── c
├── a.txt
├── b.txt
└── c.txt

3 directories, 3 files
edu@edu:~/work$ find ./ -name a.txt
./a.txt
edu@edu:~/work$ sudo find /* -name a.txt
/home/edu/work/a.txt
find: `/run/user/1000/gvfs': 权限不够
edu@edu:~/work$
```

19、grep 查找内容的命令

1、grep 默认是从 终端输入的内容中查找指定的内容

```
edu@edu:~/work$ grep hehe
ni hao xixi haha
heihei lala wuwu hehe
heihei lala wuwu hehe
hehexixi
hehexixi
```

默认从终端上查找hehe

2、grep 如果想从指定的文件中 查找内容

grep 查找信息 文件名 参数-n 显示行号

```
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work$ ls
a a.txt b.txt c.txt
edu@edu:~/work$ grep hehe a.txt -n
2:xixi lala hehe
3:world hehe
edu@edu:~/work$
```

在a.txt中查找hehe -n会显示行号

```
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work$ grep hehe *.txt -n
a.txt:2:xixi lala hehe
a.txt:3:world hehe
b.txt:2:hehe
b.txt:3:heihei hehe
edu@edu:~/work$ grep hehe ./ * -n
grep: ./a: 是一个目录
./a.txt:2:xixi lala hehe
./a.txt:3:world hehe
./b.txt:2:hehe
./b.txt:3:heihei hehe
edu@edu:~/work$
```

20、ln 链接文件

1、软链接

ln 源文件名 链接文件名 -s

软连接：链接文件 是源文件的快捷方式 二者内容时刻同步 如果删除源文件 快捷方式 无

法使用 (背)

```

edu@edu:~/work$ ls
a.c
edu@edu:~/work$ ln a.c a_s -s
edu@edu:~/work$ ls -l
总用量 4
-rw-rw-r-- 1 edu edu 12 7月 18 14:00 a.c
lrwxrwxrwx 1 edu edu 3 7月 18 14:00 a_s -> a.c
edu@edu:~/work$ cat a_s
hello world
edu@edu:~/work$ echo "xixi haha" >> a_s
edu@edu:~/work$ cat a.c
hello world
xixi haha
edu@edu:~/work$ ls -l
总用量 4
-rw-rw-r-- 1 edu edu 22 7月 18 14:01 a.c
lrwxrwxrwx 1 edu edu 3 7月 18 14:00 a_s -> a.c
edu@edu:~/work$

```

2、硬链接

ln 源文件名 链接问名

```

edu@edu:~/work$ ls
a.c
edu@edu:~/work$ ln a.c a_h
edu@edu:~/work$ ls -l
总用量 8
-rw-rw-r-- 2 edu edu 12 7月 18 14:04 a.c
-rw-rw-r-- 2 edu edu 12 7月 18 14:04 a_h
edu@edu:~/work$ cat a_h
hello world
edu@edu:~/work$ echo "xixi haha" >> a_h
edu@edu:~/work$ cat a.c
hello world
xixi haha
edu@edu:~/work$

```

硬链接：源文件和链接文件 都是独立的文件 二者内容时刻同步 删除源文件 不影响链接

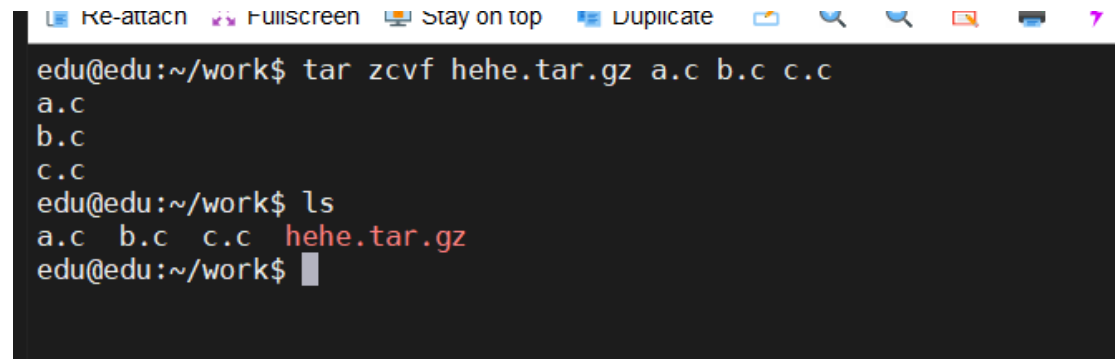
文件。（背）

21、tar 压缩和解压

tar 是一个打包的命令 需要加上选项 才能完成压缩或解压的功能。

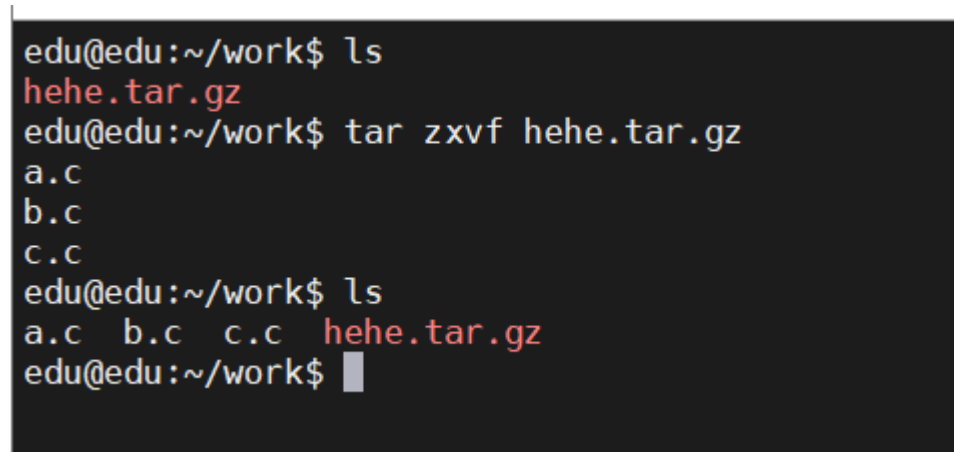
1、gzip 格式

压缩命令: tar zcvf 压缩包包名 文件 1 文件 2 ...

A terminal window with a dark background and light text. At the top, there are window control buttons: 'Re-attach', 'Fullscreen', 'Stay on top', 'Duplicate', and several icons. The terminal shows the following commands and output:

```
edu@edu:~/work$ tar zcvf hehe.tar.gz a.c b.c c.c
a.c
b.c
c.c
edu@edu:~/work$ ls
a.c b.c c.c hehe.tar.gz
edu@edu:~/work$
```

解压命令: tar zxvf 压缩包的名字

A terminal window with a dark background and light text. The terminal shows the following commands and output:

```
edu@edu:~/work$ ls
hehe.tar.gz
edu@edu:~/work$ tar zxvf hehe.tar.gz
a.c
b.c
c.c
edu@edu:~/work$ ls
a.c b.c c.c hehe.tar.gz
edu@edu:~/work$
```

解压命令: tar zxvf 压缩包的名字 -C 指定的目录

```

edu@edu:~/work$ tree
.
├── hehe.tar.gz
└── test

1 directory, 1 file
edu@edu:~/work$ tar zxvf hehe.tar.gz -C ./test
a.c
b.c
c.c
edu@edu:~/work$ tree
.
├── hehe.tar.gz
└── test
    ├── a.c
    ├── b.c
    └── c.c

1 directory, 4 files
edu@edu:~/work$ █

```

2、bz2 格式

压缩命令: tar jcvf 压缩包包名 文件 1 文件 2 ...

```

edu@edu:~/work$ ls
a.c  b.c  c.c  test
edu@edu:~/work$ tar jcvf hehe.tar.bz2 a.c b.c c.c
a.c
b.c
c.c
edu@edu:~/work$ ls
a.c  b.c  c.c  hehe.tar.bz2  test
edu@edu:~/work$ █

```

解压命令: tar jxvf 压缩包包名

```
edu@edu:~/work$ ls
hehe.tar.bz2  test
edu@edu:~/work$ tar jxvf hehe.tar.bz2
a.c
b.c
c.c
edu@edu:~/work$ ls
a.c  b.c  c.c  hehe.tar.bz2  test
edu@edu:~/work$
```

解压命令:tar jxvf 压缩包包名 -C 指定目录

```
edu@edu:~/work$ tree
.
├── hehe.tar.bz2
└── test

1 directory, 1 file
edu@edu:~/work$ tar jxvf hehe.tar.bz2 -C test
a.c
b.c
c.c
edu@edu:~/work$ tree
.
├── hehe.tar.bz2
└── test
    ├── a.c
    ├── b.c
    └── c.c

1 directory, 4 files
edu@edu:~/work$
```

3、总结

gzip 格式:

压缩:tar zcvf 压缩包名 需要压缩的各个文件

解压:tar zxvf 压缩包名

bz2 格式:

压缩:tar jcvf 压缩包名 需要压缩的各个文件

解压:tar jxvf 压缩包名

暴力解压:

tar xvf 压缩包名

知识点 3【编辑器】

1、编辑器之 gedit

gedit 是 linux 下文本编辑器。



2、编辑器之 vim

1、安装 vim 软件

```
sudo apt-get update
```

```
sudo apt-get install vim
```

```
sudo apt-get install ctags
```

2、配置 vim

← → ▾ ▴ 03-第3天 (linux常用命令、vi编辑器、gcc编译) > 03-相关资料 > vi参考资料 ▾			
名称	修改日期	类型	大小
vim_configure	2023/7/16 17:37	文件夹	
参考资料	2023/7/16 17:37	文件夹	
Linux常用命令全集.CHM	2023/1/31 15:08	编译的 HTML 帮...	364 KB
vim_configure.zip	2023/1/31 15:08	ZIP 压缩文件	554 KB
vim键盘.bmp	2023/1/31 15:08	BMP 文件	2,149 KB
vim快捷键.bmp	2023/1/31 15:08	BMP 文件	2,600 KB
vim快捷键.pdf	2023/1/31 15:08	Microsoft Edge ...	79 KB

放入ubuntu的tools目录

← → ▾ ▴ > 网络 > 10.9.42.114 > edu > tools >			
名称	修改日期	类型	大小
vim_configure	2023/7/18 14:43	文件夹	

```
cd ~/tools/  
cd vim_configure  
sudo ./copy_con.sh
```

```
edu@edu:~/tools/vim_configure$ sudo ./copy_con.sh  
start copy  
/home/edu  
copy successful  
edu@edu:~/tools/vim_configure$
```

成功

测试成功：

```
cd ~/work/  
vim a.c
```

3、vim 的三种模式

编辑模式：修改代码 复制、粘贴、剪贴、查找



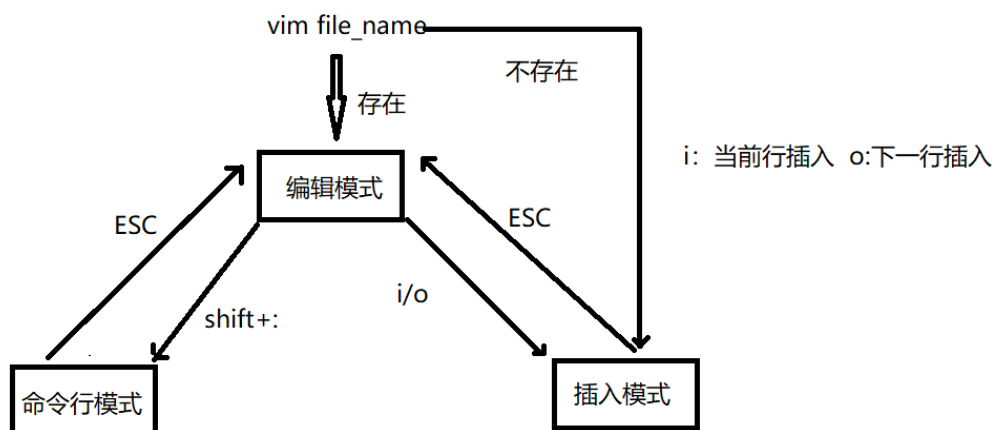
插入模式：用户写代码



命令函数：保存 退出



4、三种模式的切换



5、插入模式就是写代码

6、编辑模式下的命令

u 撤消前面多次修改。ctl r 反撤销

[n]x 删除光标后 n 个字符。3x 表示删除光标后 3 个字符

[n]X 删除光标前 n 个字符。3X 表示删除光标前 3 个字符

[n]dd 删除从当前行开始的 n 行

[n]yy 复制从当前行开始的 n 行。

p 粘贴

dw 删除一个单词

yw 复制一个单词

. 执行上一次操作

shift +zz(按住 shift 按两下 z 键) 保存退出当前文件

[n]G: 将光标定位到第 n 行开始处

5G 将光标移动第 5 行开始处

G: 将光标定位到文件结束处

gg:将光标定位到文件开始处

/字符串: 从光标开始处向文件尾查找字符串。

n: 同一方向重复上一次查找命令。

N: 反方向重复上一次查找命令

:nohl 取消高亮

:set hl 设置高亮 vi 配置文件的作用

7、命令行模式（最后一行模式）

q 退出 w 保存 ! 强制执行

wq! 强制保存并退出

w!强制保存不退出

w 文件名 另存

打开多文件：

vim a.c b.c c.c

切换某个文件:open c.c 切换到 c.c 中

!linux 命令 等价在终端上运行

8、编辑模式的扩展命令

VIM-PLUGIN c-support.vim VERSION 5.5 HOT KEYS Key mappings for Vim with and without GUI. Plugin: http://vim.sourceforge.net	
(i) insert mode, (n) normal mode, (v) visual mode	
Help	
\hp	help (plugin) (n,i)
Comments	
\cl	end-of-line comment (n,v,i)
\cj	adjust end-of-line comment (n,v,i)
\cs	set end-of-line comment column (n)
\c*	code ⇒ comment /* */ (n,v)
\cc	code ⇒ comment // (n,v)
\co	comment ⇒ code (n,v)
\cfr	frame comment (n,i)
\cfu	function comment (n,i)
\cme	method description (n,i)
\ccl	class description (n,i)
\cd	date (n,v,i)
\ct	date & time (n,v,i)
Statements	
\sd	do { } while (n,v,i)
\sf	for (n,i)
\sfo	for { } (n,v,i)
\si	if (n,i)
\sif	if { } (n,v,i)
\sie	if else (n,v,i)
\sife	if { } else { } (n,v,i)
\sw	while (n,i)
\swh	while { } (n,v,i)
\ss	switch (n,v,i)
\sc	case (n,i)
\s{	{ } (n,v,i)

Preprocessor	
\p<	#include<...> (n,i)
\p"	#include"... " (n,i)
\pd	#define (n,i)
\pu	#undef (n,i)
\pie	#if #else #endif (n,v,i)
\pid	#ifdef #else #endif (n,v,i)
\pin	#ifndef #else #endif (n,v,i)
\pind	#ifndef #def #endif (n,v,i)
\pi0	#if 0 #endif (n,v,i)
\pr0	remove #if 0 #endif (n,i)
\pe	#error (n,i)
\pl	#line (n,i)
\pp	#pragma (n,i)
Snippet	
\nr	read code snippet (n)
\nw	write code snippet (n,v)
\ne	edit code snippet (n)
\np	pick up prototype (n,v)
\ni	insert prototype(s) (n)
\nc	clear prototype(s) (n)
\ns	show prototype(s) (n)
\ntl	edit local templates (n)
\ntg	edit global templates (n)
\ntr	reread the templates (n)
Idioms	
\if	function (n,v,i)
\isf	static function (n,v,i)
\im	main() (n,v,i)
\i0	for(x=0; x<n; x+=1) (n,v,i)
\in	for(x=n-1; x>=0; x-=1) (n,v,i)
\ie	enum + typedef (n,v,i)
\is	struct + typedef (n,v,i)
\iu	union + typedef (n,v,i)
\ip	printf() (n,i)
\isc	scanf() (n,i)
\ica	p=calloc() (n,i)
\ima	p=malloc() (n,i)
\isi	sizeof() (n,v,i)
\ias	assert() (n,v,i)
\ii	open input file (n,v,i)
\io	open output file (n,v,i)

C++	
\+co	cout << endl; (n,i)
\+c	class (n,i)
\+cn	class (using new) (n,i)
\+ci	class implementation (n,i)
\+cni	class (using new) implementation (n,i)
\+mi	method implementation (n,i)
\+ai	accessor implementation (n,i)
\+tc	template class (n,i)
\+tcn	template class (using new) (n,i)
\+tci	template class implementation (n,i)
\+tcni	template class (using new) impl. (n,i)
\+tmi	template method implementation (n,i)
\+tai	template accessor implementation (n,i)
\+tf	template function (n,i)
\+ec	error class (n,i)
\+tr	try ... catch (n,v,i)
\+ca	catch (n,v,i)
\+c.	catch(...) (n,v,i)
Run	
\rc	save and compile (n,i)
\rl	link (n,i)
\rr	run (n,i)
\ra	set comand line arguments (n,i)
\rm	run make (n,i)
\rg	cmd. line arg. for make (n,i)
\rp	run splint ¹ (n,i)
\ri	cmd. line arg. for splint (n,i)
\rk	run CodeCheck ² (n,i)
\re	cmd. line arg. for CodeCheck (n,i)
\rd	run indent (n,i,v)
\rh	hardcopy buffer (n,i,v)
\rs	show plugin settings (n,i)
\rx	set xterm size (n,i. only Unix & GUI)
\ro	change output destination (n,i)
Menu(s)	
\lcs	Load Menus (n & GUI only)
\ucs	Unload Menus (n & GUI only)

¹ www.splint.org

² CodeCheckTM is a product of Abraxas Software, Inc.