Linux修炼计划一

一.【命令格式】：命令 [选项] [参数]
注意个别命令的使用不遵循此类格式，当有多个选项时，可以写在一起，简化选项与完整选项：-a 等于 --all
二.【常用命令】
1.【文件处理命令】
基础命令：
ls命令
-a 显示所有文件，包括隐藏文件（.开头的文件）
-l 显示详细信息（ls -l == ll）
-d 查看目录属性
-h 人性化显示文件大小
-i 显示inode
目录相关：
创建目录：mkdir -p [目录名] （-p：递归创建）
创建空文件夹：touch [文件夹名]
切换目录：cd [目录名] （/家目录，-上次访问的目录，..上一级目录）
【注意】相对路径 绝对路径
相对路径：cd ../user/local/src/
绝对路径：cd /etc/user/local/src/
删除目录：rmdir [目录名]（删除空白目录，很少使用）
rm -rf [文件夹]（rm本身是用来删除文件的，-r删除目录，-f强制）
拷贝命令：cp [选项] [源文件或目录] [目标目录]
-r ：复制目录
-p：连带文件属性复制（相当于连同时间等属性信息一起复制）
-d：若源文件是链接文件，则复制连接属性
-a：相当于pdr
剪切命令：mv [源文件或目录] [目标目录]
无参数选项（有别于拷贝命令的特点）
重命名文件：mv [旧文件名] [新文件名]

Linux常用一级目录：cd /
bin： 保存普通用户使用命令的目录（普通用户就可以读取的命令）
sbin： 保存超级用户使用命令的目录
boot： 保存启动目录，启动相关文件
dev： 设备文件保存目录
etc： 配置文件保存目录
home：普通用户的家目录
root： 超级用户的家目录
lib： 系统函数库保存目录
mnt： 系统挂载目录
media：挂载目录
proc和sys：保存的是内存的挂载点，不能直接操作
tmp： 临时目录
usr： 系统软件资源目录（/bin：普通用户。/sbin：超级用户）
var： 保存系统相关可见文档的内容
链接命令：ln -s [源文件] [目标文件] 【备注】有待深究
作用：把源文件生成链接文件，-s 创建软连接
硬链接特征：
1. 拥有相同的i节点和存储block块，可以看做是同一个文件
2. 可通过i节点识别
3. 不能跨分区
4. 不能针对目录使用
软链接特征：
1. 类似windows的快捷方式
2. 软链接拥有自己的i节点和Block块，但是数据块中只保存源文件的文件名和i节点号，并没有实际的文件数据
3. lrwxrwxrwx l 软链接（软链接文件权限都为rwxrwxrwx）
4. 修改任意文件，另一个都改变
5. 删除原文件，软链接不能使用

2.【文件搜索命令】
文件搜索命令【locate】 ————>（搜索文件名）（速度快，功能单一只能搜索文件名）
locate [文件名]，在后台数据库中按照文件名搜索，搜索速度更快
/var/lib/mlocate：#locate命令所搜索的后台数据库（不是实时更新的）
updatedb：强制更新locate数据库

/etc/updatedb.conf配置文件　【locate搜索的配置文件】
开启搜索限制：.PRUNE_BIND_MOUNTS="yes"
搜索时，不搜索的文件系统：PRUNEFS=
搜索时，不搜索的文件类型：PRUNENAMES=
搜索时，不搜索的路径：PRUNEPATHS=

命令搜索命令【whereis与which】————>（搜索命令）
whereis：搜索命令所在路径及帮助文档所在位置，只能搜索系统命令。-b：只查找可执行的文件 -m：只查找帮助文件
which：　和whereis命令相似，除了搜索系统命令的所在位置，如果命令有别名，还可以搜索出其别名。
备注：但并不是所有的命令都会同伙whereis或which找到所在路径和文档位置。
　　比如cd，这不是系统级的命令，而是shell自带的。它自身就具有相应的功能，不需要可执行文件来运行。
PATH环境变量：（echo $PATH）定义的是系统搜索命令的路径，所有的命令其实都是通过命令所在的绝对路径来执行的。
比如：执行(ls)，就相当于执行(/bin/ls)，而系统变量（echo $PATH）则是搜索目录顺序，系统级命令之所以能够直接使用，则是按照该顺序来查找并使用的，如果有自定义命令，必须使用绝对路径来使用，或是放在PATH路径下。

文件搜索命令【find】————>（搜索文件名）
格式：find [搜索范围] [搜索条件]
实例：find / -name install.log（在整个根下，搜索名称为install.log的文件）
备注：避免大范围的搜索，会非常耗费系统资源。
　　find是在系统当中搜索符合条件的文件名，如果需要匹配，使用通配符匹配，通配符是完全匹配。
Linux中的通配符：
　　*　匹配任意内容；？　匹配任意一个字符；[]　匹配任意一个括号内的字符
　　实例：find /root -name "ab[cd]"搜索/root下名称为abc或abd的任意文件
常用选项参数：
　　1. -name：名称；-iname：不区分大小写；-user：按照所有者搜索；-nouser：查找没有所有者的文件
　　　eg：find /root -name data.log
　　2. -mtime　：修改文件内容；-atime：文件访问时间；-ctime：改变文件属性
　　　　+10：10天前；-10：10天内；10：10天当天；
　　　eg：find /var/log/ -mtime +10（查找10天前修改的文件）
　　3. -size：文件大小（-25:小于25KB的文件，25：等于25KB的文件，+25大于25KB的文件）
　　　-inum：i节点（节点数）
　　　eg：find . -size 25k(M)　　find . -inum 26732（搜索当前目录.）
　　4. -a (and) 逻辑与，两个条件都满足
　　　-o (or)　逻辑或，两个条件满足一个即可
　　　eg：find /etc -size +20k -a -size -50k（查找/etc/目录下，大于25KB并且小于50KB的文件）
　　5. -exec/ ok 命令 {} \;（对搜索结果执行操作）
　　　eg：find /etc/ -size +20k -a -size -50k -exec ls -lh {} \;
　　　　（查找/etc/目录下，大于25KB并且小于50KB的文件，并且显示详细信息）

字符串搜索命令grep
grep [选项] 字符串 文件名（在指定文件当中匹配符合条件的字符串，选项：-i 忽略大小写；-v 排除指定字符串）

find命令与grep命令的区别
find命令：在系统当中搜索符合条件的文件名，如果需要匹配，使用通配符匹配，通配符是完全匹配。
grep命令：在指定文件中搜索符合条件的字符串，如果需要匹配，使用正则表达式匹配，正则表达式是包含匹配。

3.【帮助命令】

man命令：
获取指定的帮助，eg：man ls（获取ls命令的帮助）
man级别：（man man）
　1. 查看命令的帮助 2. 查看可被内核调用的函数的帮助 3. 查看函数和函数库的帮助
　4. 查看特殊文件的帮助（/dev目录下） 5. 查看配置文件的帮助 6. 查看游戏的帮助
　7. 查看其他杂项的帮助 8. 查看系统管理员可用命令的帮助 9. 查看和内和相关文件的帮助
man -f 命令：相当于whatis 命令，eg：man 5 passwd；man 4 null；man 8 ifconfig
man -k 命令：相当于apropos命令，eg：apropos passwd（查看该命令所有相关的帮助）

help命令：
获取选项帮助，eg：ls --help（查看ls命令相关选项的帮助说明）

shell内部命令：
获取shell内部命令的帮助（shell相当于Linux的外壳，即用户操作的界面，Linux命令解释器）
whereis cd（cd是否是shell内部的命令），找不到命令执行文件路径，即为shell内部命令
help cd（获取内部命令的帮助）

详细命令帮助info：
info命令：
　　-回车：进入子帮助页面（带有*号标记）
　　-u：　进入上层页面
　　-n：　进入下一个帮助小节
　　-p：　进入上一个帮助小节
　　-q：　退出

4.【压缩和解压缩命令】

常用压缩格式：.zip　.gz　.bz2　.tar.gz　.tar.bz2

.zip格式解压缩：（可压缩文件和目录）

压缩文件——zip 压缩文件名(不区分后缀) 源文件　eg：zip test.zip test

压缩目录——zip -r 压缩文件名 源目录　eg：zip -r test.zip test

解压缩　——unzip 压缩文件名　eg：unzip test.zip

.gz格式解压缩：（压缩目录下的所有字文件）

gzip 源文件　（压缩为.gz格式的压缩文件，源文件会消失）

gzip -c 源文件 > 压缩文件　（压缩为.gz格式，源文件会保留）eg：gzip -c test > test.gz

gzip -r 目录　（压缩目录下所有的子文件，但是不能压缩目录）

解压缩：（gzip -d 压缩文件）或（gunzip 压缩文件）

.bz2格式解压缩：（不支持压缩目录）

bzip2 源文件　（压缩为.bz2格式，不保留源文件）

bzip2 -k　（压缩为.bz2格式，保留源文件）

注意：bzip2命令不能压缩目录

解压缩：（bzip2 -d 压缩文件）或（bunzip2 压缩文件），注意 -k保留压缩文件

.tar.gz格式解压缩：（扩展.gz格式）————解决目录压缩

意为：将.gz格式打包为.tar的文件包（其实.tar.gz格式是先打包为.tar格式，再压缩为.gz格式）

tar -cvf 打包文件名 源文件（eg：tar -cvf test.tar test）

选项：-c：打包　-v：显示过程　-f：指定打包后的文件名

tar -zcvf 压缩包名.tar.gz 源文件

选项：-z：压缩为.tar.gz格式

tar -zxvf 压缩包名.tar.gz

选项：-x：解压缩.tar.gz格式

.tar.bz2格式解压缩：（扩展.bz2格式）————解决目录压缩

意为：将.bz2格式打包为.tar的文件包（其实.tar.bz2格式是先打包为.tar格式，再压缩为.bz2格式）

tar -jcvf 压缩包名.tar.bz2 源文件

选项：-j：压缩为.tar.bz2格式

tar -jxvf 压缩包名.tar.bz2

选项：-x：解压缩.tar.bz2格式

5.【关机和重启命令】

1. shutdown [选项] [时间]　（shutdown操作安全）

选项：

-c 取消前一个关机命令：shutdown -c

-h 关机：　shutdown -h 05:30（凌晨5点关机）

-r 重启：　shutdown -r now

2. 其余关机命令（操作不安全）：halt，poweroff，init 0
3. 其余重启命令：reboot（相对安全），init 6（不安全）
4. 系统运行级别：init 调用级别（eg：init 0）

0 关机；1 单用户；2 不完全多用户，不含NFS服务；3 完全多用户（字符级别）；4 未分配；5 图形界面； 6 重启

5. 查询当前系统运行级别：runlevel（eg：N 3）N代表进入该级别之前的系统级别，3代表当前系统运行的级别
6. 修改系统默认运行级别：cat /etc/inittab（id:3:initdefault:）即：进入系统后默认进入字符级别（可修改，不能为0，6）
7. 退出登录：logout

6.【其他常用命令】

挂载命令：

挂载 == 分配盘符 == 挂载点

用户登录查看和用户交互命令：

四.【其余相关】

1. 文件说明

eg：【-rw-r--r-- 1 liyao-sz liyao-sz 29380240 Jan 29 18:20 ddlicai-manage.war】

第一部分：文件的权限信息（默认是10位）：（-rw-r--r--）

第一位（1位）：代表文件类型　-（-文件 d普通目录 l软连接文件）

第一组（3位）：代表所有者u　rw-

第二组（3位）：代表所属组g　r--

第三组（3位）：代表其他人o　r--

第二部分：1 引用计数，代表该文件被引用的次数

第三部分：所有者

第四部分：所属组

第五部分：文件大小，以字节单位，可使用（ls -lh）命令，将其大小格式化表示

第六部分：最后一次修改时间

第七部分：文件名

五.【Shell相关】

1. Shell概述

Shell介绍：

a. Shell是一个命令行解释器，它为用户提供了一个向Linux内核发送请求，以便于运行程序的界面系统级程序，用户可以用Shell来启动、挂起、停止甚至是编写一些程序。

b. Shell还是一个功能相当强大的编程语言，易编写，易调试，灵活性较强。Shell是解释执行的脚本语言，在Shell中可以直接调用Linux系统命令。

Shell分类：

Bourne Shell：从1979年起Unix就开始使用Bourne Shell，Bourne Shell的主文件名为sh。（基本淘汰）

C Shell：C Shell主要在BSD版的Unix系统中使用，其语法和C语言相类似而得名。

Shell语法类型：两种主要语法类型：Bourne 和 C，这两种语法彼此不兼容。

Bourne 家族主要包括：sh、ksh、Bash（Linux中的标准shell）、psh、zsh；

C 家族只要包括：csh、tcsh。（主要用于Unix中）

查看当前计算机运行的shell：echo $SHELL

Bash：Bash与sh兼容，现在使用的Linux就是使用Bash作为用户的基本Shell。（vi /etc/shells 可查看）

2. 脚本执行方式

1. echo输出命令

echo [选项] [输出内容]（选项：--e 支持反斜线控制的字符切换）

eg：echo -e "h\te\tl\nl\to\t"

\a：输出警告音

\b：退格键，也就是向左删除键

\n：换行符

\r： 回车键

\t： 制表符，也就是Tab键

\v： 垂直制表符

\0nnn：按照八进制ASCLL码表输出字符，其中0位数字0，nnn是三位八进制数

\xhh：按照十六进制ASCLL码表输出字符，其中hh是两位十六进制数

eg：echo -e "\e[1;31m我是一名程序员\e[0m"

说明：设置输出文字的字体颜色

#30m=黑色 31m=红色 32m=绿色 33m=黄色

#34m=蓝色 35m=洋红 36m=青色 37m=白色

2. 第一个脚本

注意：（#开头的均为注释，除了#!/bin/bash，它是标识以下程序是Linux的标准脚本，不能省略）

```
#!/bin/bash
#The first program
    echo -e "\e[1;37m 我是一名程序员 \e[0m"
```

脚本执行（两种方式）：

a. 赋予执行权限，然后通过路径运行

```
chmod 755 Myfirst.sh
./Myfirst.sh
```

b. 通过Bash调用执行脚本

```
bash Myfirst.sh
```

3. Bash的基本功能

1. 命令别名与快捷键

命令别名

查看别名：alias（查看系统当中所有的命令别名）

设置别名：alias 别名='原命令'

注意：这是临时存放，系统重启后会自动消失。如要永久使用，需要写到环境变量配置文件中：【vi ~/.bashrc】

写入bashrc文件后，需要重新登录设置的别名才会生效，执行【source .bashrc】即可使文件立即生效

删除别名：unalias 别名（同样也是临时删除）

命令生效顺序：

第一顺位执行用绝对路径或相对路径执行的命令

第二顺位执行别名

第三顺位执行Bash的内部命令

第四顺位执行按照$PATH环境变量定义的目录查找顺序找到的第一个命令

快捷键

ctrl + c：强制终止当前命令

ctrl + l：清屏

ctrl + a：光标移动到命令行首

ctrl + e：光标移动到命令行尾

ctrl + u：从光标所在位置删除到行首

ctrl + z：把命令放入后台

ctrl + r：在历史命令中搜索

2. 历史命令

history [选项] [历史命令保存文件]
选项：
-c：清空历史命令
-w：把缓存中的历史命令写入历史命令保存文件~/.bash_history
注意：历史命令默认会保存1000条，可以在环境变量配置文件/etc/profile中进行修改 HISTSIZE值。

历史命令的调用：
使用上下箭头调用以前的历史命令
使用"!n"重复执行第n条历史命令
使用"!!"重复执行上一条历史命令
使用"!字串"重复执行最后一条以该字串开头的命令

命令与文件的补全：
在Bash中，命令与文件补全是非常方便与常用的功能，我们只要在输入命令或文件时，按下Tab键就会自动补全。
命令的自动补全最终还是依赖于$PATH的路径搜索，而文件补全是依赖于目录。

3. 输出重定向（常用）
定义：标准输出方向是输出到显示器上，输出重定向就是改变输出的方向，不再输出到显示器上，而是输出到文件当中。
注意：错误输出的2左右不能有空格
eg：ifconfig > test.log（即将命令写到了文件当中，test.log不存在则创建，存在则覆盖）
标准正确输出（只能将正确的命令写到文件）：>（覆盖） >>（命令并存追加）
标准错误输出（可以将错误的命令写到文件）：2>（覆盖） 2>>（命令并存追加）
正确输出和错误输出同时保存：
覆盖： 命令 > 文件 2>&1
追加： 命令 >> 文件 2>&1 （如果命令正确直接写入文件；如果错误，把错误写入正确，再将两者同时写入）
覆盖： 命令 &>文件
追加： 命令 &>>文件 （如果命令正确直接写入文件；如果错误，把错误写入正确，再将两者同时写入）
追加： 命令>>文件1 2>>文件2（正确输出保存在文件1，错误输出保存在文件2）
4. 输入重定向（不常用）
命令格式：wc [选项] [文件名]
选项：-c（统计字节数） -w（统计单词数） -l（统计行数）
命令 < 文件把文件作为命令的输入 eg：wc < test.log（统计test.log文件的内容）==wc test.log
5. 多命令顺序执行

多命令执行符	格式	作用
;	命令1 ; 命令2	多个命令顺序执行，命令之间没有任何逻辑关系
&&	命令1 && 命令2	逻辑与 当命令1正确执行，则命令2才会执行 当命令1 执行不正确，则命令2不会执行
||	命令1 || 命令2	逻辑或 当命令1执行不正确，则命令2才会执行 当命令1 正确执行，则命令2不会执行

eg：【ls && echo yes || echo no】根据输出可用于判断命令是否正确执行，yes正确执行；no错误执行。
6. 管道符
格式：【命令1 | 命令2】（命令1的正确输出作为命令2的操作对象，若命令1错误，命令2不会执行）
注意：命令2 一定要可以操作命令1 的结果
eg：
单管道符： ls -l | /etc/ more
双管道符：netstat -an | grep ESTABLISHED | wc -l（统计当前网络连接中正在连接网络的总人数(行数)）
7. 通配符
主要是用来匹配文件名或者目录名称，如果想要匹配文件当中的数据，需要使用正则表达式来匹配。

通配符	作用
?	匹配一个任意字符
*	匹配0个或任意多个任意字符，也就是可以匹配任何内容
[]	匹配中括号中任意一个字符。例如：[abc]代表一定匹配一个字符，或者是a或者是b或者是c
[-]	匹配中括号中任意一个字符，-代表一个范围。例如：[a-z]代表匹配任意一个小写字母
[^]	逻辑非 表示匹配不是中括号内的一个字符。例如： [^0-9]：代表匹配一个不是数字的字符

6. Shell中特殊符号

符号	作用
‘’	单引号，在单引号中所有的特殊符号，如“$”和“`”[反引号]都没有特殊含义
“”	双引号，在双引号中的特殊符号都没有特殊含义，但是"$"、"`"、"\"是例外，拥有"调用变量的值"，“引用系统命令”、"转义符"的特殊含义
`	反引号，反引号括起来的内容是系统命令，在Bash中会限制性它，和$()作用一样，不过推荐使用$()，因为反引号非常容易看错
$()	和反引号作用一样，用来引用系统命令
#	在Shell脚本中，#开头的行代表注释
$	用于调用变量的值，如需要调用变量name的值时，需要使用$name的方式得到变量值
\	转义符，跟在\之后的特殊符号将失去特殊意义，变为普通字符，如 \$ 将输出"$"的符号，而不当做是变量引用