# Linux基础

## bash概览

### shell分类

**GUI**: 图形用户界面

Gnome ，KDE , Xface

**CLI**: 命令行接口

sh csh (加州伯克利bill joy研发，与c接近)，ksh

bash (Linux系统兼具以上特性，功能更强大，Linux 发行版默认使用) ，

zsh 不是特别流行 ,tcsh

### shell和进程

sh 也是一个软件，sh什么时候启动呢？

在系统启动的时候启动，提供给用户使用输入登陆命令。

sh程序只有一个，有几个用户登陆就有几个sh进程。

Linux中还允许一个用户在当前系统启动多个虚拟终端进行登陆，但是每个sh进程都是独立的。互不干涉，互不影响。名称一样也没关系，进程 pid 不一样即可。

**进程**： **在每个进程看来，当前主机之存在内核和自己**。

**进程是程序副本，进程是程序执行的实例。所以进程是有生命周期的，进程的生命周期是由内核来管理的。bash也不例外。**

**type bash**

**bash is /bin/bash 是一个外部命令，启动的时候还有自由命令，像cd之类。**



### 用户的工作环境

每个用户的工作环境是可以自定义的，但是默认是一样的，除了管理员。

bash :

**#** 管理员

**$** 普通用户

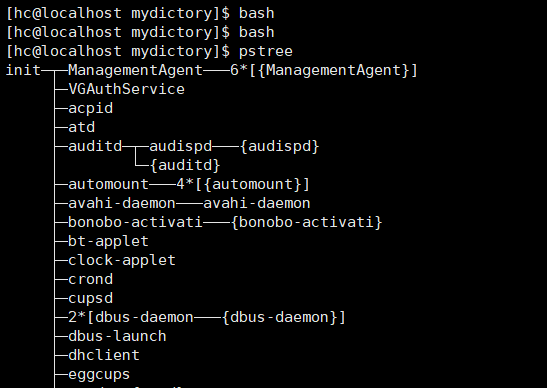
bash支持用户自己定义环境。

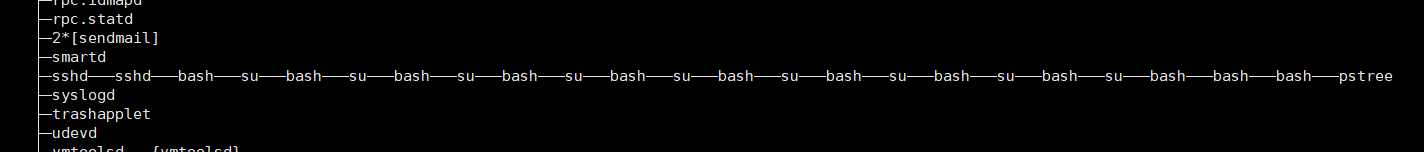
### sh 和子sh

bash

bash

pstree (查看进程树)





对父sh的设定，对子sh是无效的，反过来也一样，因为他们互不影响。互不干涉。

## bash特性

1. 命令历史
2. 命令补全
3. 管道，重定向
4. 命令别名
5. 命令行编辑
6. 命令行展开
7. **文件名通配**
8. **支持使用变量**
9. **支持编程 -- shell脚本**

### 命令行编辑

#### 光标跳转

**ctrl+a** 跳到行首



**ctr+e 跳到行末**



#### 命令删除

**ctr+u**

从光标处删除命令内容至行首

**ctr+k**

从光标处删除命令内容至行尾

**ctr+->** ctr+ **<-** 跳一个单词，只有主机上的虚拟终端才可以

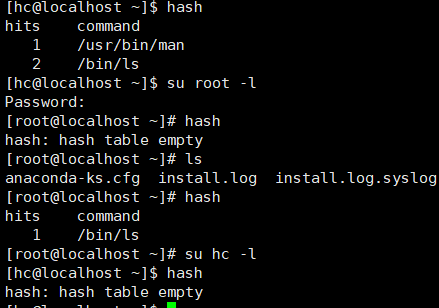
ctr+l = clear 清屏

### 命令历史

曾经执行过的命令会保存在缓存中，只需要按上下箭头就可以载入。

#### hash

hash 用于保存曾经执行过的命令（一次登陆）

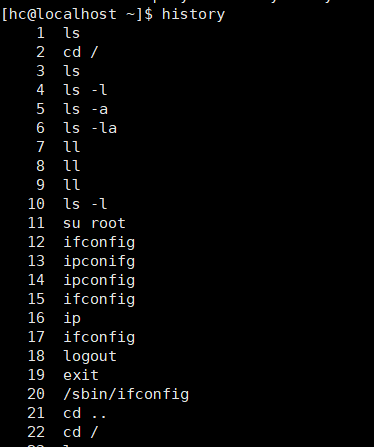


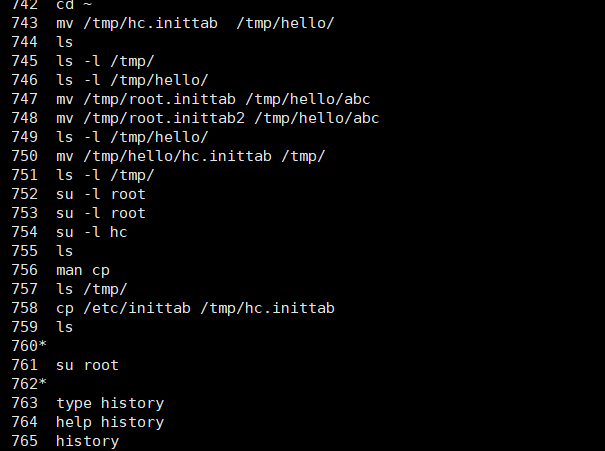
#### history

[hc@localhost ~]$ type history

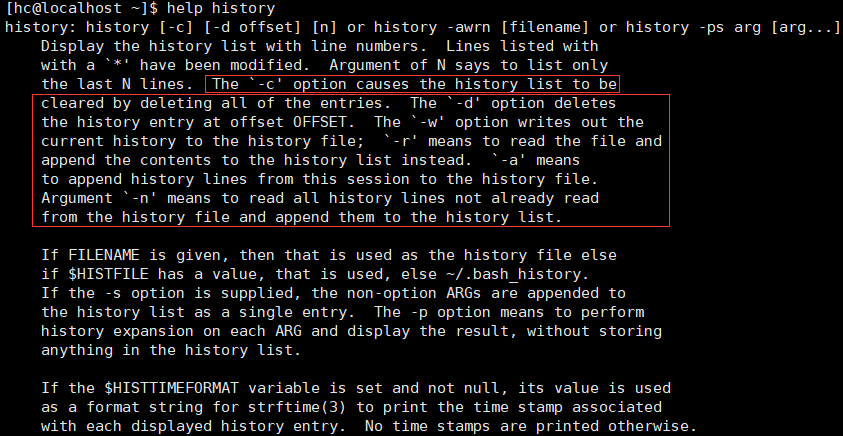
**history is a shell builtin**

**history是shell内部命令**





这些执行命令的历史保存在缓存中，如果不想保留历史，可以清除



**-c**

The ‘-c’ option causes the history list to be ceared by deletring all of the entries.

清空命令历史

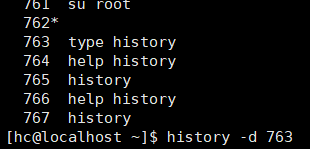
**-d**

The ‘-d’ optiondeletes the history entry at offset OFFSET.

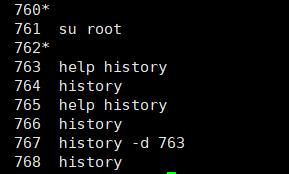
删除第几个历史命令

history -d 763

删除第763个命令type history

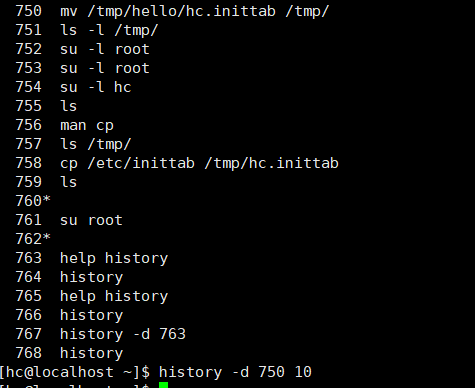


第763个命令被删除

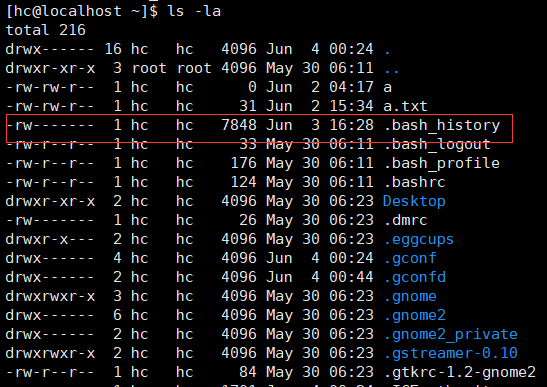


向后删除10个历史命令

history -d 750 10



如果用户正常退出，执行过的历史命令会被保存到用于相应的文件中，/home/.bash history (这是一个隐藏文件)



**-w**

The `-w' option writes out the current history to the history file

保存命令到.bash hsitory文件中去

那么能保存多少个命令呢？

这个用户可以自己配置。在环境变量中可自行配置大小

任何变量都可以 这样查看 echo $变量名

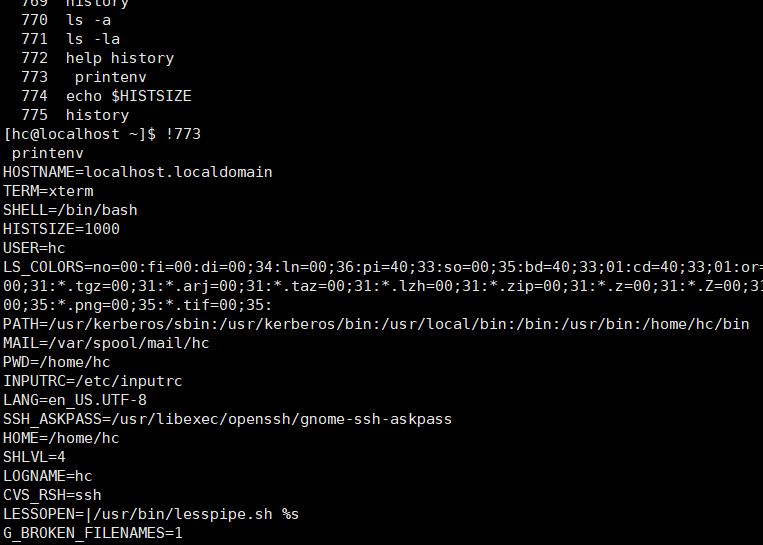
echo $HISTSIZE



### 命令历史的使用技巧

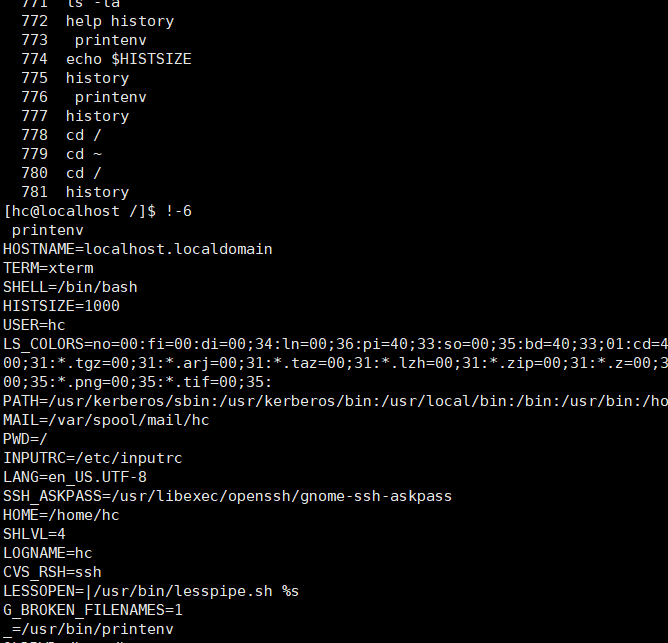
#### ！n

执行命令历史中的第n个命令



#### ！-n

执行历史命令中倒数第n个命令

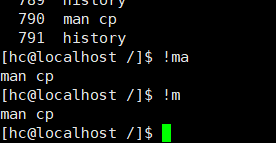


#### ！！

执行上一条命令

#### ！字符串

执行命令历史中最近一个以指定字符串开头的命令

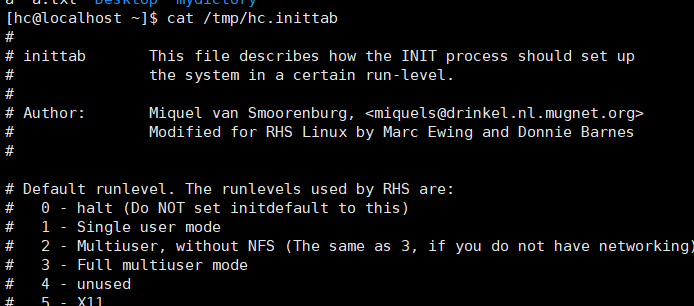


#### ！$

引用前一个命令的最后一个参数

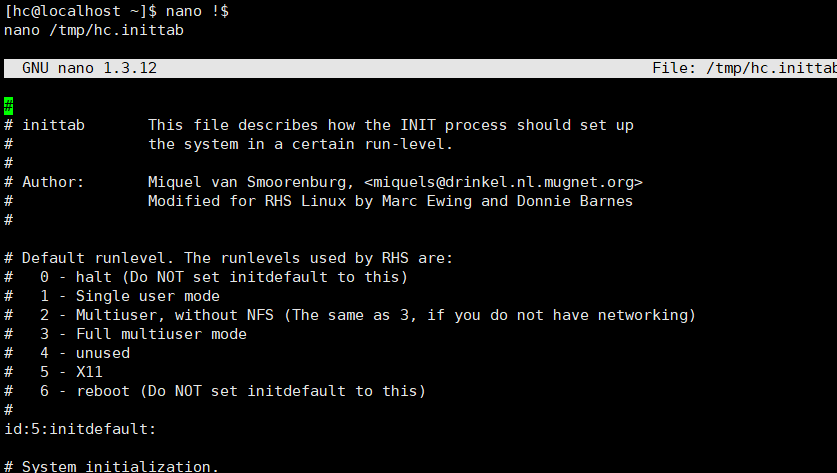
nano !$

cat /tmp/hc.inittab



这时我想编辑这个hc.inittab就用

nano !$ 就打开了



当然还有更简单的办法 esc .

#### Esc .

先按下esc 松开，才按 . 就可以得到前一个命令的参数

#### alt+.

按着alt在按点，远程终端不支持。

### 命令补全

前提，path环境变量正常，可以去搜索提供的字符串开头的命令

tab键 补全

给的开头字符足够标识这个命令，那么敲一下tab,就可以补全

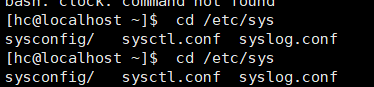
如果不能够标识这个命令 ，**连敲两下tab**，提示出这个字符开头的所有命令



### 路径补全

路径补全是你提供打头路径补全，

cd /etc/sys 连敲两下tab 会提示路径和文件



路径补全和命令补全用法都一样，但是补全机制不相同。

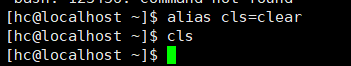
### 命令别名

还可以给命令起一个别名，如 clear -- cls

#### alias

alias CMDALIAS = ‘COMMAND [options] [arguments]’

alias cls=clear

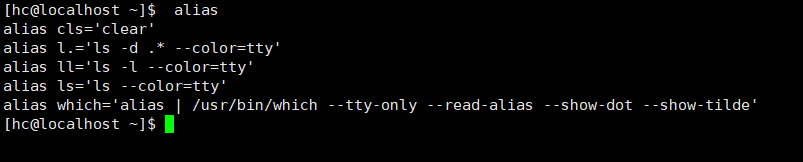


这些别名都是变量，当用户系统重启了，这些变量就没有了。我们定义的别名都在当前sh进程的生命周期内有效。哪怕是同一个用户，打开了两个sh进程。

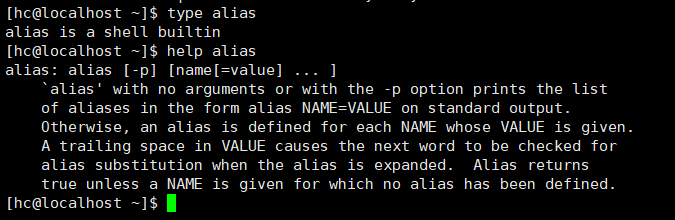
我下次登陆系统想用别名都的重新定义，很麻烦，所有要想永久保存自定义，需要通过程序的配置文件来定义，命令定义只管当前用户的shell进程。

所以要想永久使用这个别名，只需要把alias cls=clear 配置到bash的配置文件中即可。作用范围对多有用户都有效。

当前系统定义的别名和用户自己定义的命令



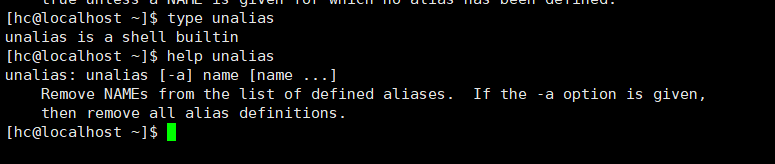
alias 是shell 内置命令



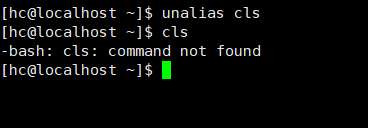
如果我不想使用这个别名

#### unalias

撤销命令别名



unalias cls



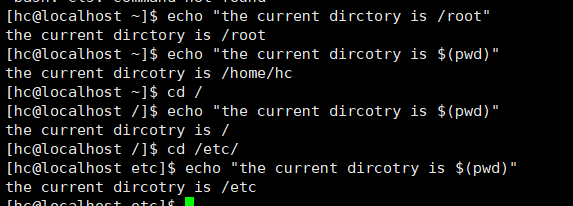
系统中有很多的别名就是原来命令本身，不过加了很多参数和选项，这是要想使用原来的命令怎么半？ \command即可

### 命令替换

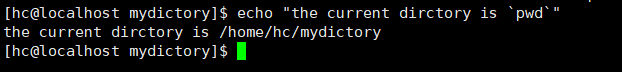
命令替换，就是要把命令的某个子命令替换其执行结果的过程就叫命令替换

$(子命令) == ``(反引号)

echo “the current dirctory is $(pwd)”



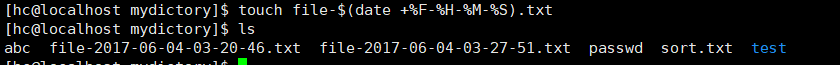
echo “the current dirctory is `pwd`”



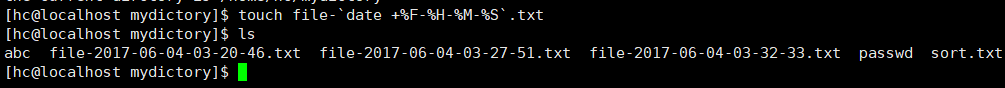
创建一个文件 file+当前时间

格式： file-2017-06-04 10:00:00.txt 使用命令替换

[hc@localhost mydictory]$ touch file-$(date +%F-%H-%M-%S).txt



[hc@localhost mydictory]$ touch file-`date +%F-%H-%M-%S`.txt



## bash所支持的引号

` ` ：反引号 命令替换

“ ” ：双引号 弱引用，可以实现变量替换

‘ ’ ：单引号 强引用，不完成变量替换

## 文件名通配

{ }展开

~ 展开

名称展开 - 文件名通配 (globbing)

\* 匹配任意长度的任意字符

? 匹配任意单个字符

[ ] 匹配指定范围内的单个字符

[abc] [a-z] [A-Z] [0-9] [a-zA-Z] [0-9a-zA-Z]

[:space:] 表示空白字符

[[:space:]] 匹配有空白字符的文件

[:punct:] 所有的标点符号

[:lower:] 所有的小写字母

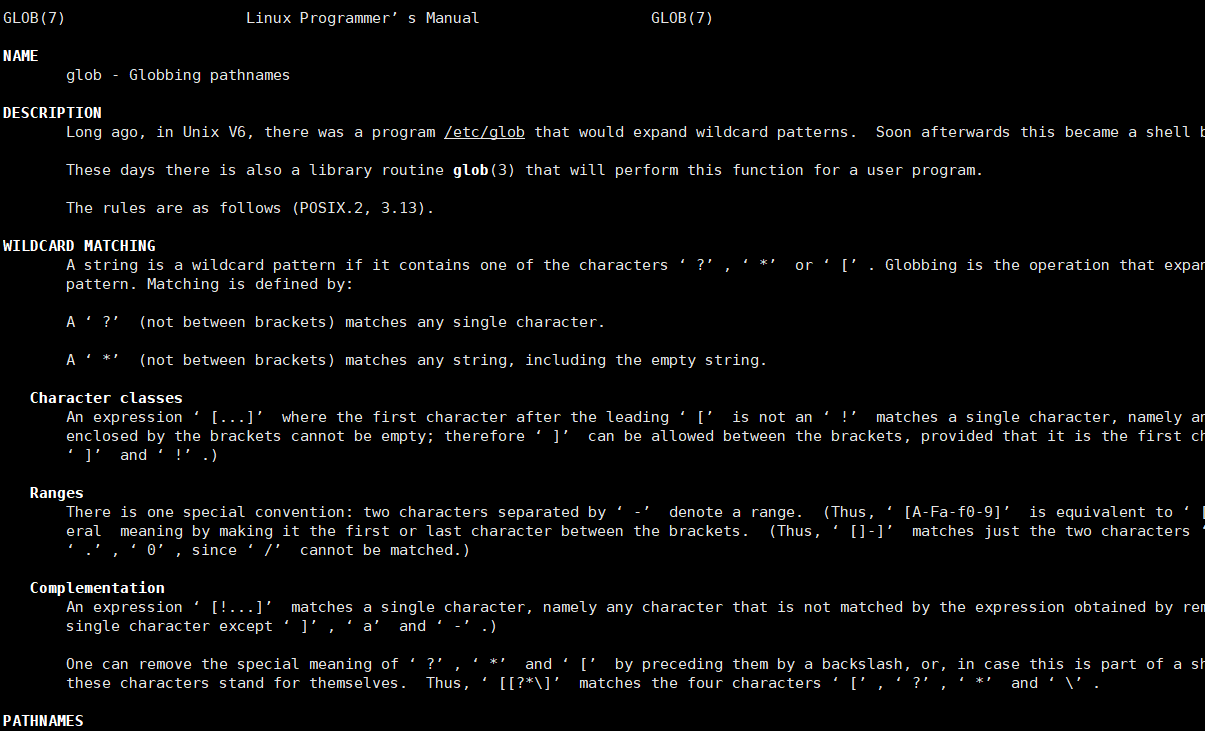
[:upper:] 所有的大写字母

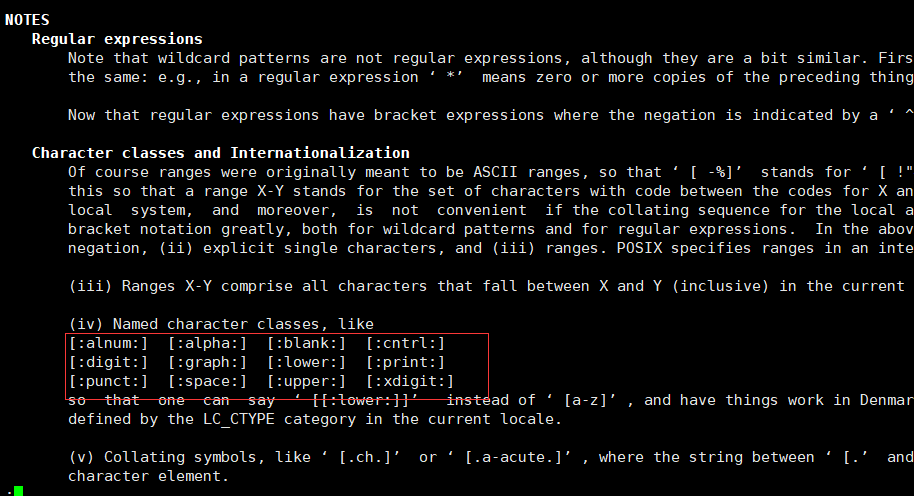
[:alpha:] 所有的大小写字母

[:digit:] 数字

[:alnum:] 大小写字母和数字

man 7 glob 就可以获得这些内容，

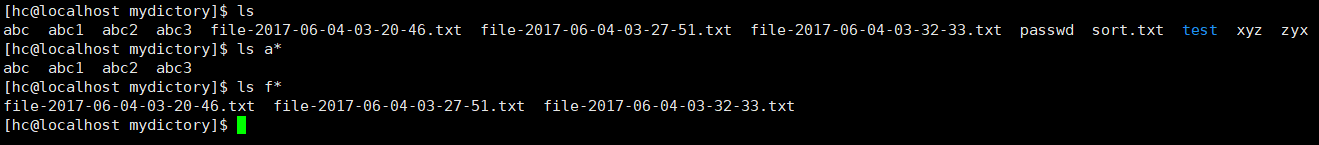




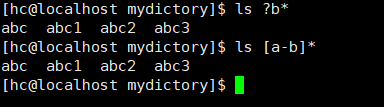
[^ ] 匹配指定范围外的单个字符

ls a\*

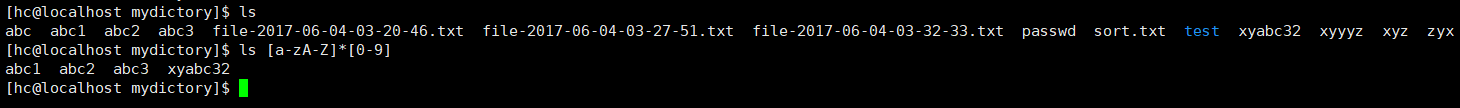
ls f\*



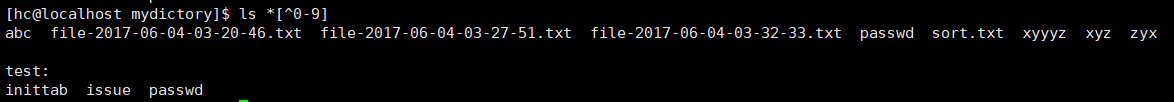
ls a\*3 以啊开头，3结尾的文件



ls [a-zA-Z]\*[0-9] 以小写a-z 大写A-Z 中间任意字符，数字结尾的

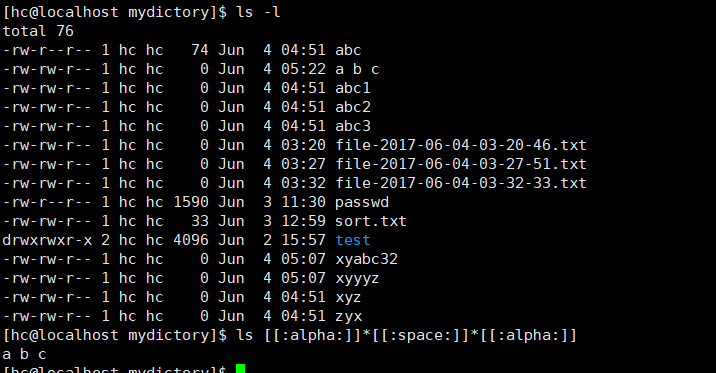


ls \*[^0-9] 以非数字结尾的文件



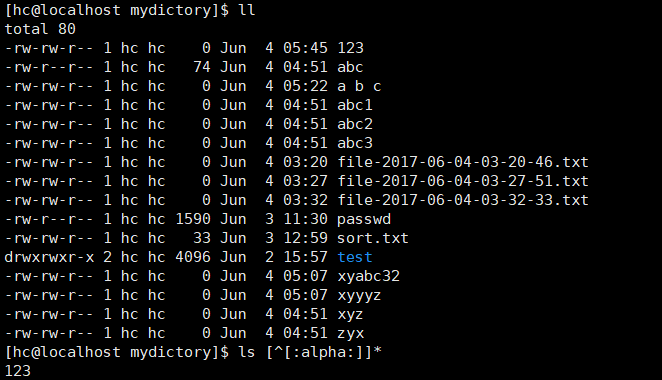
[hc@localhost mydictory]$ ls [[:alpha:]]\*[[:space:]]\*[[:alpha:]]

以字母开头中间有任意字符还有空白任意字母结尾的



[hc@localhost mydictory]$ ls [^[:alpha:]]\*

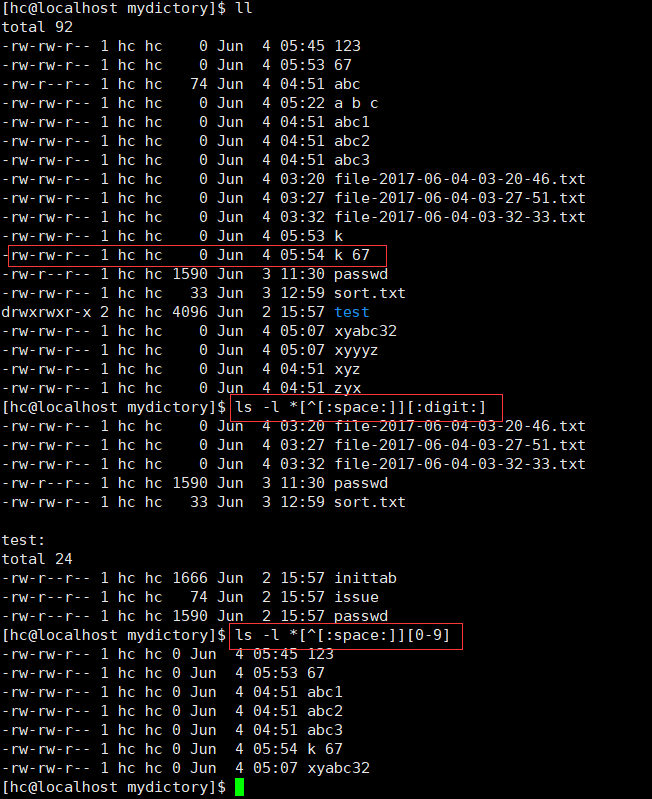
不是以字母开头的所有文件



匹配以任意字符开头中间没有空格还以数字结尾的，结果就有问题了 ？？？？

[hc@localhost mydictory]$ ls -l \*[^[:space:]][:digit:]

[hc@localhost mydictory]$ ls -l \*[^[:space:]][0-9]



通配符也有解决不了的问题，那就用正则表达式

