

操作系统与Linux程序设计

第一章 Linux基本使用方法

彭成磊

南京大学电子科学与工程学院

2022年2月

目录 I

- 1 Linux是什么?
- 2 GPL等软件许可协议
- 3 Linux系统的使用环境
- 4 掌握一款高效的编辑器
- 5 熟练使用Linux命令行
- 6 Shell编程初步

目录

- 1 Linux是什么？
- 2 GPL等软件许可协议
- 3 Linux系统的使用环境
- 4 掌握一款高效的编辑器
- 5 熟练使用Linux命令行
- 6 Shell编程初步

Linux是什么？



图：Linux官方吉祥物¹

- Linux是一套类UNIX的操作系统，来源于UNIX
- Linux最早由Linus Torvalds于1991年开始开发
- Linux is OpenSource and Free Software.

¹图片来源于wikipedia

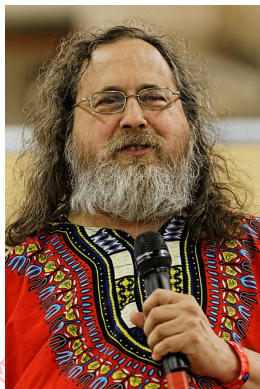
UNIX是什么？

- UNIX是一套多用户、多任务操作系统
- UNIX最早由Ken Thompson、Dennis Ritchie和Douglas McIlroy于1969年在AT&T的贝尔实验室开发。
- UNIX不仅是一个OS，更具有独特的设计哲学和美学。
 - ▶ 简洁至上(KISS原则, Keep It Simple, Stupid)
 - ▶ 提供机制而非策略

Linux发展史

- UNIX商业授权昂贵
- 1983年，美国Richard M. Stallman发起GNU Project。
- 1985年，自由软件基金会(FSF)成立，发布GPL。
 - ▶ 逐渐开发了Emacs、GCC、GDB等大部分UNIX系统中的程序，唯独缺少内核。
- 1987年，荷兰Andrew S. Tanenbaum发布Minix，用于教学。
- 1991年，Linus Torvalds以Minix为样本开发Linux内核。
- 1992年，Linux内核+GNU软件=GNU/Linux，基于GPL发布。
- 主要的发行版本：Ubuntu，Fedora，Arch Linux

三个重要人物



图：Dennis Ritchie, Richard M. Stallman and Linus Torvalds²

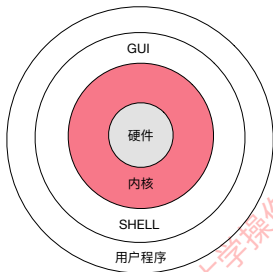
■ 1971, UNIX/C; 1983, GNU; 1991, Linux

²图片来源于wikipedia

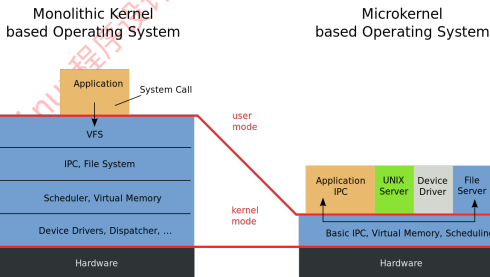
Linux系统架构1

■ Linux采用宏内核(monolithic kernel)

- ▶ 微内核(micro kernel), Minix, Mach, QNX
- ▶ 混合内核(hybrid kernel), Windows NT, Mac OS X



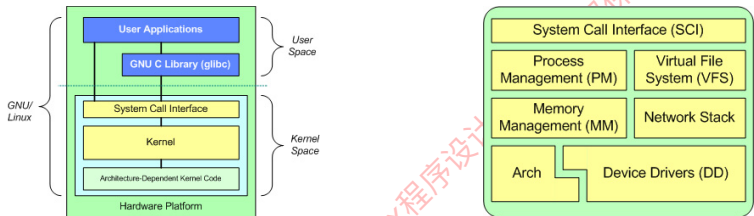
图：Linux系统架构1



图：Monolithic vs Micro³

³图片来源于Wikipedia

Linux系统架构2



图：Linux系统架构²⁴

Linux Kernel 内部

- ▶ 用户空间，内核空间（不同的地址空间）
- ▶ 理解GNU C Library (glibc)的作用
- ▶ System Call Interface提供用户空间访问内核空间的函数接口
- ▶ VFS将不同的文件系统类型进行抽象，统一表示
- ▶ 进程管理、内存管理、网络协议、设备驱动

²⁴图片来源于IBM developerWorks

Linux功能

- The Bootloader (启动引导程序)
- The kernel (内核)
- Daemons (后台服务)
- The Shell (外壳)
- The Libraries (各种运行库, 如glibc, 或开发环境)
- The GUI environment (GNOME, KDE桌面环境)
- Applications (应用程序)

为什么要学习Linux

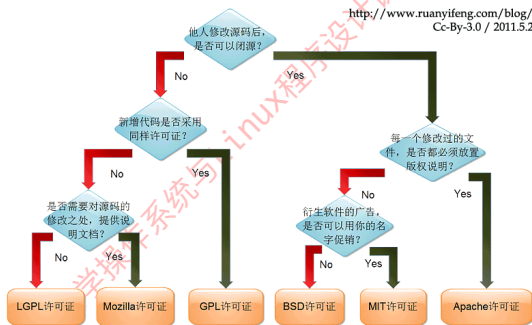
- 开放源代码，很好的学习操作系统原理的平台
- 帮助你更深入理解C语言从源代码到二进制的本质
- 支持Linux的开源社区项目众多
- 完善编程语言环境，c/c++，python，及各种开发库
- 科学计算平台，gnuplot，octave，numpy，scipy，tensorflow
- Linux是主流的嵌入式软件开发平台，也是主流嵌入式产品内核

目录

- 1 Linux是什么?
- 2 GPL等软件许可协议
- 3 Linux系统的使用环境
- 4 掌握一款高效的编辑器
- 5 熟练使用Linux命令行
- 6 Shell编程初步

软件许可协议

- Linux内核遵循GPL v2，自由软件
- 其他的许可协议有：BSD，Apache，LGPL，MIT等
- 下图由乌克兰程序员Paul Bagwell绘制，阮一峰做了中文翻译



图：几种license比较⁵

⁵图片来源于阮一峰的网络日志

软件许可协议

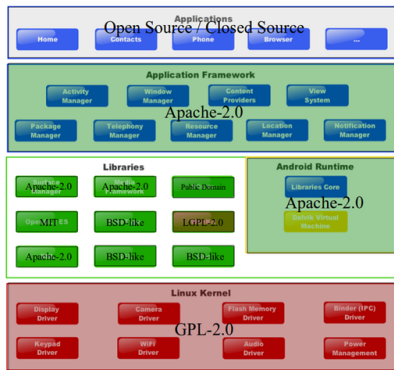
许可证	版本	包含许可证	包含源代码	链接	状态变化	商业使用	散布	修改	专利授权	私人使用	授权转售	无担保责任	没有商标
Apache许可证	2.0	是			是	是	是	是	是	是	是	是	是
3句版BSD许可证		是				是	是	是		是	是	是	是
2句版BSD许可证		是				是	是	是		是	是	是	
GNU通用公共许可证	2.0	是	是		是	是	是	是	是	是	否	是	
GNU通用公共许可证	3.0	是	是		是	是	是	是	是	是	是	是	
GNU宽通用公共许可证	2.1	是	是	是		是	是	是	是	是	是	是	
GNU宽通用公共许可证	3.0	是	是	是		是	是	是	是	是	是	是	
MIT许可证		是				是	是	是		是	是	是	
Mozilla公共许可证	2.0	是	是			是	是	是	是	是	是	是	是
Eclipse公共许可证	1.0	是	是			是	是	是	是	是	是	是	
Affero通用公共许可证		是	是		是	是	是	是		是	是	是	
一般的著作权 ^[note 1]		是				是	否	否		是	否		

图：几种license比较⁶

⁶图片来源于oschina.net

Android的软件许可协议

- Android软件许可协议以Apache License 2.0为主



2009 © Alvaro Fuentes Vasquez (Kronos), released under GFDL-1.2+, with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

图：Android license⁷

⁷图片来源于Shallon blog

目录

- 1 Linux是什么?
- 2 GPL等软件许可协议
- 3 Linux系统的使用环境**
- 4 掌握一款高效的编辑器
- 5 熟练使用Linux命令行
- 6 Shell编程初步

南京大学操作系统与Linux程序设计课程所有权保留

文件系统目录

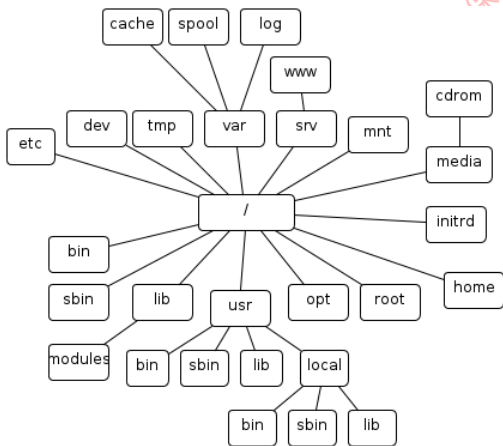
■ Windows 文件系统目录

- ▶ 与物理设备或分区紧密相关，如C:盘，D:盘。
- ▶ 各盘符下面按照目录树结构存储。
- ▶ FAT，NTFS等文件系统类型。
- ▶ 硬盘→分区→分区上文件系统

■ Linux 文件系统目录

- ▶ 物理设备或分区挂载到不同目录上。
- ▶ 整个文件系统处于根目录/之下。
- ▶ Ext2/3/4等文件系统类型。
- ▶ 虚拟文件系统→硬盘→分区→分区上文件系统

Linux目录树结构



图：Linux目录树结构⁸

⁸FHS标准

主要目录定义

bin	基本命令二进制文件
boot	引导加载器的静态文件
dev	设备文件
etc	特定于主机的系统配置
lib	基本共享库和内核模块
media	可插拔介质的挂载点
mnt	临时挂载文件系统的挂载点
opt	附加应用程序包
sbin	基本系统二进制文件
srv	该系统提供的服务的数据
tmp	临时文件
usr	次要层次结构
var	变量数据
root	超级用户主目录
home	普通用户主目录

用户类型

- 超级用户root；普通用户通过su或sudo提升权限
- 用户(Users)和组(Groups)
 - ▶ /etc/passwd, /etc/shadow, /etc/group, /etc/sudoers, /home/*
 - ▶ whoami, id, \$ PS1环境变量, who, su, sudo, passwd
- 文件类型(UNIX一切皆文件)
 - ▶ 普通文件
 - ▶ 目录
 - ▶ 其他文件(设备、管道、网络等)
- 文件属主(ownership)和权限(permissions)


文件属主和权限⁹

```
$ ls -l
drwxrwxr-x 2 ubuntu Users 4096 Oct 21 04:36 docs
-rwxrwxr-x 1 ubuntu Users 8519 Oct 21 04:37 hello
-rw-rw-r-- 1 ubuntu Users 74 Oct 21 04:37 hello.c

permissions    user group size  date  time  filename
```

■ permissions有10个字节

- ▶ 第一个字节表示文件类型(-,d,p,l,c,b,s)
- ▶ 2,3,4位表示user的权限(读r写w执行x)(u)
- ▶ 5,6,7位表示group的权限(g)
- ▶ 最后3位表示others的权限(o)
- ▶ 最后1位用于访问控制安全(.或+)

⁹本节参考资料：IBM developerWorks: Learn Linux 

更改文件属主和权限

```
# chown tester hello.c
# ls -l hello.c
-rw-rw-r-- 1 tester Users 74 Oct 21 04:37 hello.c
# chgrp Testers hello.c
# ls -l hello.c
-rw-rw-r-- 1 tester Testers 74 Oct 21 04:37 hello.c
# chmod g-rw hello.c
-rw----r-- 1 tester Testers 74 Oct 21 04:37 hello.c
# chmod 644 hello.c
-rw-r--r-- 1 tester Testers 74 Oct 21 04:37 hello.c
```

登录注销和手册页

■ Linux 登录和注销

- ▶ 开启一个终端，进入shell(bash)
- ▶ 注销，logout, Ctrl+d, exit
- ▶ 关机，shutdown, reboot, poweroff

■ Linux 手册页

- ▶ man
- ▶ help
- ▶ info

目录

- 1 Linux是什么?
- 2 GPL等软件许可协议
- 3 Linux系统的使用环境
- 4 掌握一款高效的编辑器**
- 5 熟练使用Linux命令行
- 6 Shell编程初步

编辑器

■ 图形化编辑器

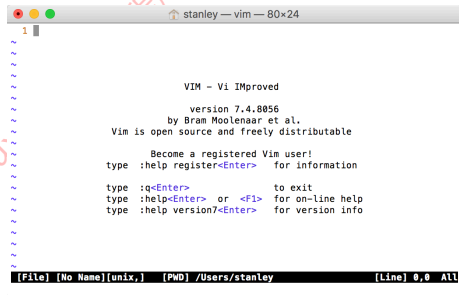
- ▶ Gedit
- ▶ Kate
- ▶ Sublime Text

■ 命令行编辑器

- ▶ Emacs
- ▶ Vim

■ VIM参考资料

- ▶ Practical Vim参考书
- ▶ Vim插件介绍参考



图：VIM窗口

VIM编辑器

- 运行VIM, 在终端输入vim后敲回车
- VIM模式
 - ▶ Command mode (命令模式), 移动、编辑操作
 - 敲入i, o, s, a等字符即可进入编辑模式
 - ▶ Insert mode (编辑模式), 输入操作
 - 敲入esc, Ctrl+c 等字符退出编辑模式

VIM移动、搜索操作

- 上h下j左k右l移动光标，以字符或行为单位
- 前b后w，前B后W(跳过标点)，e/E到末尾，以word为单位
- 0到当前行头，^到当前行首个非空字符，\$ 跳到行尾
- 上一段，下一段，以空白行做分割
- gg跳到顶部，G跳到底部，:x跳到x行
- ctrl+d, ctrl+f 向下翻页，ctrl+u, ctrl+b向上翻页
- /xxx 搜索xxx，n下一个，N上一个
- # 向前搜索光标所在的word，*向后搜索
- fx 在当前行移动到光标之后第一个字符x的位置 f(ind)x

VIM编辑操作

- i(nsert)在光标当前位置插入，I在行首插入
- a(fter)在当前光标位置后插入，A在行尾插入
- o向下O向上插入一空行
- x删除当前字符，Ndd删除N行，dw删除当前word
- Nyy复制N行，p(aste)粘贴
- :w保存，:q退出，:wq或ZZ保存退出
- 大部分命令前都可以加上数字，如6j(下移6行)

目录

- 1 Linux是什么?
- 2 GPL等软件许可协议
- 3 Linux系统的使用环境
- 4 掌握一款高效的编辑器
- 5 熟练使用Linux命令行**
- 6 Shell编程初步

终端(Shell)

- 相对于图形用户接口(Graphics-User-Interface, GUI)而言, 命令行接口 (Command-Line-Interface, CLI)是嵌入式Linux开发的主要工作方式
- 提供命令行工作环境的设备即为终端(Shell)
- Shell接收用户的键盘输入, 分析和执行输入字符串中的命令, 并返回结果。
 - ▶ 内部命令、外部命令
- 命令行格式
 - ▶ `command [options] [parameters]`
 - ▶ 例如: `ls -l hello.c`
 - ▶ `./hello`
 - ▶ 注意环境变量PATH

文件目录操作

- 常用命令查询: <http://man.linuxde.net/>
- 列出目录和文件命令: `ls`
 - ▶ `-a` 列出隐藏文件
 - ▶ `-l` 列出详细信息
 - ▶ `-t` 根据修改时间排序
- 移动和重命名命令: `mv`
 - ▶ `mv hello.c src/`
 - ▶ `mv hello.c hello2.c`

文件目录操作

■ 拷贝命令： cp

- ▶ -a 拷贝所有属性
- ▶ -r 递归拷贝(含子目录)
- ▶ -f 强制拷贝

■ 远程拷贝： scp

- ▶ `scp /home/abc/hello.tar.gz root@192.168.200.36:/home/root`
- ▶ `scp root@192.168.200.36:/home/root/hello.tar.gz /home/abc/`

文件目录操作

- 判断文件类型: `file`
- 删除文件: `rm`
 - ▶ `rm -rf hello` 强制递归删除hello目录
- 创建空文件或更新文件时间: `touch`
 - ▶ `touch hello.c`
- 改变目录: `cd`
 - ▶ `cd /home/ab`
 - ▶ `cd ~/abc`
 - ▶ `cd ../hello`

文件目录操作

■ 创建目录: `mkdir`

- ▶ `mkdir -p abc/def/ghi/test` 递归创建
- ▶ `mkdir -m 666 src` 自定义目录权限

■ 删除空目录: `rmdir`

■ 显示内容: `echo`

- ▶ `echo "hello\nworld"`
- ▶ `echo -e "hello\nworld"`

■ 显示或合并: `cat`

- ▶ `-n` 显示行号
- ▶ `cat hello.c`
- ▶ `cat hello1.c hello2.c > hello.c`

查看文件

- 查看整个文件
 - ▶ `more hello.c` 空格翻页, q退出
 - ▶ `less hello.c` 可逐行上下查看
- 加行号显示: `nl hello.c`
- 显示前几行: `head -n 2 hello.c`
- 显示后几行: `tail -n 2 hello.c`
- 查看命令位置: `which ls`
- 定位命令、源代码、帮助文件位置: `whereis ls`

查找文件

- 在文件树中查找文件，并做出处理： find
 - ▶ `find ./ -mmin 2` 找出当前目录下2分钟之前修改过的文件
 - ▶ `find ./` 查找当前目录下所有文件
 - ▶ `find ./ -iname "hello.c"`
 - ▶ `find . -name "*.txt" -o -name "*.pdf"`
 - ▶ `find /home ! -name "*.txt"`
 - ▶ `find . -maxdepth 3 -type f`
 - ▶ `find . -type f -size +10k`
 - ▶ `find . -empty`
- `find . -type f -name "*.txt" -delete`
- `find . -type f -user root -exec chown tom {} \;`
- `find . -type f -name "*.txt" -exec cat {} \;> all.txt`

搜索内容

■ 用正则表达式搜索文本：grep

- ▶ `grep "a dog" book.txt`
- ▶ `grep d.g book.txt`
- ▶ `grep -v "a dog" book.txt`
- ▶ `echo "this is a test line." | grep -o -E "[a-z]+\."`
- ▶ `grep -c "text" filename` 统计包含text的行数
- ▶ `grep "text" -n filename` 显示包含text的行号

■ 序列产生命令：seq, printf

- ▶ `seq -f "%03g" 1 100`
- ▶ `seq -w 1 100`
- ▶ `printf "%03d\n" {1..100}`

■ 统计命令：wc 四种选项 -l -c -m -w

打包、压缩和解压

- 压缩相关: gzip, bzip2, xz, tar
 - ▶ `tar -cvf abc.tar abc`
 - ▶ `gzip abc.tar`, `bzip2 abc.tar`, `xz abc.tar`
 - ▶ `tar -zcvf abc.tar.gz abc` 或 `-jcvf` 或 `-Jcvf`
 - ▶ `tar -jxvf abc.tar.bz2 -C abc2`
- 历史记录: `last`, `lastlog`, `history`
- 显示文本文件指定列的内容: `cut`
 - ▶ `cut -d " " -f 1 my.txt`
- 排序文本文件: `sort`, `uniq`
 - ▶ `sort -u my.txt`
 - ▶ `uniq my.txt`
 - ▶ `sort -k 2 -t " " test.txt`

管道与重定向

- 管道：|
- 重定向：>, >>, <, <<, <<<
 - ▶ <, 将后面文件作为前面命令的输入
 - ▶ <<, 带命令作用全文匹配某个字符串后结束
 - ▶ <<<, 部分匹配某个字符串
- 浏览file文件的200-225行
 - ▶ head -n 225 file | tail -n 26
 - ▶ head -n 225 file | tail -n 26 > newfile

文件比较与打补丁

比较文件和打补丁: `diff`, `patch` `diff -u hello1.c hello2.c`

```
#include <stdio.h>

int main()
{
    print("Hello World\n");
}
```

```
#include <stdio.h>

int main()
{
    print("Hello World\n");
    return 0;
}
```

`diff -u hello1.c hello2.c > hello.diff`

```
--- hello1.c 2017-11-03 21:19:04.000000000 -0700
+++ hello2.c 2017-11-03 21:19:29.000000000 -0700
@@ -3,4 +3,5 @@
 int main()
 {
     print("Hello World\n");
+    return 0;
 }
```

`patch < hello.diff`

进程控制

- 环境变量: `export`, `env`, `alias`, `unalias`
 - ▶ `echo $PATH; export PATH=$PATH: /bin`
 - ▶ `env | grep PATH`
 - ▶ `alias ll='ls -l'; unalias ll`
- 磁盘相关: `du`, `df`, `free`, `mount`
 - ▶ `mount /dev/sda1 /data`
- 进程管理: `ps`, `pstree`, `top`, `kill`, `killall`
 - ▶ `ps -aux ; pstree`
 - ▶ `top`
 - ▶ `kill -9 23232; killall -9 hello`

网络操作

- 网络操作: ifconfig, route, ping, netstat, telnet, ssh, wget
- 网络文件系统: NFS, /etc/exports, mount

```
ifconfig eth0 192.168.10.100
ifconfig eth0 up/down
route add -net 224.0.0.0 netmask 240.0.0.0 dev eth0
route add/del default gw 192.168.120.240
ping 192.168.10.1
netstat -a
telnet -l username bbs.nju.edu.cn
ssh -l username 192.168.20.100
wget http://192.168.10.100/test.zip
mount 192.168.10.200:/nfsroot /mnt
```

目录

- 1 Linux是什么?
- 2 GPL等软件许可协议
- 3 Linux系统的使用环境
- 4 掌握一款高效的编辑器
- 5 熟练使用Linux命令行
- 6 Shell编程初步**

什么是Shell

- 类似于GUI(图形用户接口), Shell为用户程序提供了一种命令行式的接口。
- Shell接收用户的键盘输入, 分析和执行输入字符串中的命令, 并返回结果。
 - ▶ 内部命令、外部命令
- 命令行格式
 - ▶ `command [options]`
`[arguments]`
 - ▶ 例如: `ls -l hello.c`

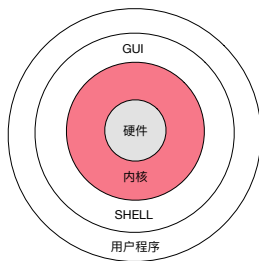


图: Linux系统架构

Shell脚本语言简介

■ 解释型语言 vs. 编译型语言

- ▶ sh, bash, csh, ksh, sed, awk
- ▶ Perl, Tcl, Python, PHP, Ruby, Javascript
- ▶ c/c++, Java

bash例子

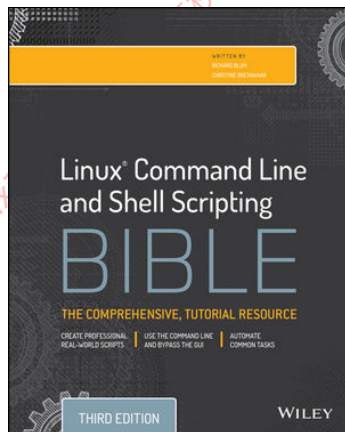
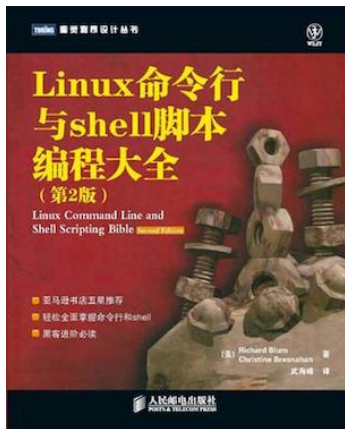
```
#!/bin/bash  
  
#adsfafdfasdfsdf  
echo "Hello World"  
printf "hello World\n"
```

python例子

```
print("Hello World")
```

`source` hello.sh; 或者 `chmod +x hello.sh; ./hello.sh`
`python` hello.py

shell编程参考书



Shell变量

■ 定义变量

- ▶ `var1="nanjing"` 等号前后不能有空格
- ▶ `for file in `ls /home``

■ 使用变量

- ▶ `echo $var1` 使用美元符号进行引用
- ▶ `echo ${var1}` 建议加上花括号

■ 注释使用 # 符号

- ▶ # 这一行是注释

Shell字符串

■ 单引号‘’

- ▶ `str='This is a string'`
- ▶ 单引号内字符原样输出

■ 双引号“”

```
your_name='nju'
str="Hello, I know you are \"$your_name\"! \n"
```

- 双引号内可以有变量和转义字符
- `echo ${#str}` 获取字符串长度

Shell数值操作

■ 求模运算

- ▶ `expr 5 % 2`
- ▶ `let i=5%2; echo $i`
- ▶ `((i=5%2)); echo $i`
- ▶ `echo "5 % 2" | bc`

■ 求幂运算

- ▶ `let i=5**2; echo $i`
- ▶ `((i=5**2)); echo $i`
- ▶ `echo "5^2" | bc`

■ 获取ASCII编码

- ▶ `echo -n '01234' | od -h`

Shell浮点运算

- `echo "scale=3; 1/13" | bc`
- `echo "1 13" | awk '{printf("%0.3f\n", $1/$2)}'`
- 浮点运算取整
 - ▶ `echo 5 3 | awk '{ printf "%d\n", $1/$2}'`
 - ▶ `echo 5 3 | awk '{ printf "%d\n", $1/$2+0.5}'`
 - ▶ `echo 4 3 | awk '{ printf "%d\n", $1/$2+0.5}'`

Shell流程控制if else

if else流程控制

例1:

```
if condition  
then
```

```
    command1
```

```
    command2
```

```
    ...
```

```
    commandN
```

```
else
```

```
    command
```

```
fi
```

例2:

```
if test $foo -eq 0; then echo equal; fi
```

例3:

```
if [ $foo -eq 0 ]; then echo equal; fi
```

Shell流程控制for

for流程控制

例1:

```
for var in item1 item2 ... itemN
do
    command1
    command2
    ...
    commandN
done
```

例2:

```
for var in item1 item2 ... itemN; do command1;
    command2... done;
```

例3:

```
for i in {1..100}; do command1; command2... done;
```

Shell流程控制while

while流程控制

例1:

```
while condition  
do  
    command  
done
```

例2, 无限循环:

```
while true  
do  
    command  
done
```

或:

```
for (( ; ; ))
```

awk脚本编程： 实例¹⁰

格式：awk 'pattern {action}' file(s)

```
ps -a > ps.txt
```

```
awk '$1==65108 || NR==1 {print $2}' ps.txt 过滤
```

```
awk 'BEGIN{FS=":"} {print $1,$3,$6}' /etc/passwd 指定分隔符
```

```
awk -F: '{print $1,$3,$6}' /etc/passwd
```

```
awk -F: '{print $1,$3,$6}' OFS="\t" /etc/passwd
```

```
awk '$6 ~ /stanley/ || NR==1 {print NR,$4,$5,$6}' OFS="\t" ps.txt 字符串匹配
```

```
awk '/LISTEN/' ps.txt
```

```
awk 'NR!=1{print $1,$2 > "2.txt"}' ps.txt 拆分输出
```

```
ls -l *.txt | awk '{sum+=$5} END {print sum}' 统计大小
```

```
ps aux | awk 'NR!=1{a[$1]+=$6;} END { for(i in a) print i, " a[i]"KB;}'
```

¹⁰<https://coolshell.cn/articles/9070.html>

awk脚本编程

```
cat score.txt
```

Marry	2143	78	84	77
Jack	2321	66	78	45
Tom	2122	48	77	71
Mike	2537	87	97	95
Bob	2415	40	57	62

awk脚本编程

```
cat cal.awk
```

```
#!/bin/awk -f
BEGIN {
    math = 0
    english = 0
    computer = 0

    printf "NAME      NO.      MATH    ENGLISH  COMPUTER  TOTAL\n"
    printf "-----\n"
}
{
    math+=$3
    english+=$4
    computer+=$5
    printf "%-6s %-6s %4d %8d %8d\n", $1, $2, $3,$4,$5, $3+$4+$5
}
END {
    printf "-----\n"
    printf "      TOTAL:%10d %8d %8d \n", math, english, computer
    printf "AVERAGE:%10.2f %8.2f %8.2f\n", math/NR, english/NR, computer/NR
}
```


awk脚本编程

```
awk -f cal.awk score.txt
```

NAME	NO.	MATH	ENGLISH	COMPUTER	TOTAL
Marry	2143	78	84	77	239
Jack	2321	66	78	45	189
Tom	2122	48	77	71	196
Mike	2537	87	97	95	279
Bob	2415	40	57	62	159

TOTAL:		319	393	350	
AVERAGE:		63.80	78.60	70.00	

awk脚本编程

```
source cal2.awk
```

```
#!/bin/awk
```

```
x=5
```

```
y=10
```

```
export y
```

```
awk -v val=$x '{print $1, $2, $3, $4+val, $5+ENVIRON["y"]}' OFS="\t" score.txt
```

```
Marry 2143 78 89 87
```

```
Jack 2321 66 83 55
```

```
Tom 2122 48 82 81
```

```
Mike 2537 87 102 105
```

```
Bob 2415 40 62 72
```

awk脚本编程

从file文件中找出长度大于80的行

```
awk 'length>80' file
```

按连接数查看客户端IP

```
netstat -ntu | awk '{print $5}' | cut -d: -f1 | sort  
| uniq -c | sort -nr
```

打印99乘法表

```
seq 9 | sed 'H;g' | awk -v RS=' ' '{for(i=1;i<=NF;i++)  
printf("%dx%d=%d%s", i, NR, i*NR, i==NR?"\n":"\t")  
}'
```

```
1x1=1
1x2=2 2x2=4
1x3=3 2x3=6 3x3=9
1x4=4 2x4=8 3x4=12 4x4=16
1x5=5 2x5=10 3x5=15 4x5=20 5x5=25
1x6=6 2x6=12 3x6=18 4x6=24 5x6=30 6x6=36
1x7=7 2x7=14 3x7=21 4x7=28 5x7=35 6x7=42 7x7=49
1x8=8 2x8=16 3x8=24 4x8=32 5x8=40 6x8=48 7x8=56 8x8=64
1x9=9 2x9=18 3x9=27 4x9=36 5x9=45 6x9=54 7x9=63 8x9=72 9x9=81
```

sed脚本编程： 实例¹¹

```
cat pets.txt
```

```
This is my cat
  my cat's name is betty
This is my dog
  my dog's name is frank
This is my fish
  my fish's name is george
This is my goat
  my goat's name is adam
```

```
sed "s/my/NJU's/g" pets.txt > NJU_pets.txt
```

```
sed -i "s/my/NJU's/g" pets.txt 直接修改
```

```
sed "3,6s/my/your/g" pets.txt 指定替换
```

```
This is NJU's cat
  NJU's cat's name is betty
This is NJU's dog
  NJU's dog's name is frank
This is NJU's fish
  NJU's fish's name is george
This is NJU's goat
  NJU's goat's name is adam
```

¹¹<https://coolshell.cn/articles/9104.html>

sed脚本编程

一段网页源文件内容：

```
<b>This</b> is what <span style="text-decoration:
underline;">I</span> meant. Understand?
```

使用sed过滤获得内容：

```
sed 's/<[^>]*>//g' html.txt
```

This is what **I** meant. Understand?

sed脚本编程

多个匹配

```
sed '1,3s/my/your/g; 3,$s/This/That/g' my.txt  
sed -e '1,3s/my/your/g' -e '3,$s/This/That/g' my.txt
```

添加行

```
sed "1 i This is my monkey, my monkey's name is  
wukong" my.txt  
sed "$ a This is my monkey, my monkey's name is  
wukong" my.txt  
sed "/fish/a This is my monkey, my monkey's name is  
wukong" my.txt
```

sed脚本编程

替换行

```
sed "2 c This is my monkey, my monkey's name is  
wukong" my.txt  
sed "/fish/c This is my monkey, my monkey's name is  
wukong" my.txt
```

删除行

```
sed '/fish/d' my.txt  
sed '2d' my.txt  
sed '2,$d' my.txt
```

Thank You!

南京大学操作系统与Linux程序设计课程所有权保留