实验一：实验环境配置与使用

1. 了解Linux操作系统及其基本操作
2. 掌握Linux下的C编程环境和工具
   ① 简单的vi使用
   ② GCC编译与链接
   ③ GDB调试

深圳大学-冯禹洪: ***yuhongf@szu.edu.cn***

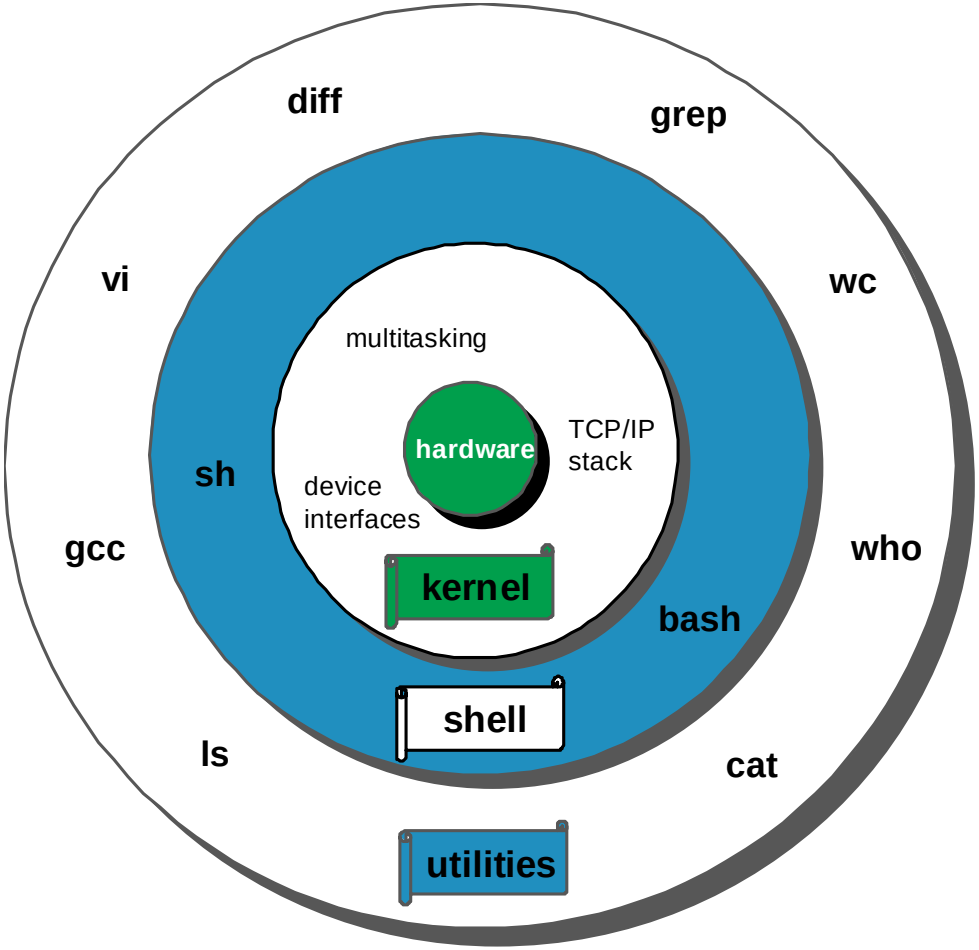
1.1 Linux操作系统特点

- 开放性
  - 遵循开放系统互连（OSI）国际标准。
- 多用户
  - 系统资源被不同用户使用，互不影响。
- 多任务
  - 计算机同时执行多个程序，互相独立。
- 良好的用户界面
  - 图形界面
  - 字符界面
  - 系统调用。
- 设备独立性
  - 所有外部设备统一当作成文件操纵和使用。
- 丰富的网络功能：完善的内置网络。
- 可靠的安全系统
  - 对读、写控制
  - 带保护的子系统
  - 审计跟踪
  - 核心授权。
- 良好的可移植性
  - 可运行于嵌入式设备、微型计算机到大型计算机。

1.2 Linux操作系统起源

- 1991年，芬兰大学生Linus Torvalds将Linux通过Internet发布
  - 所有的源码必须公开
  - 任何人均不得从Linux交易中获利。
- **纯粹的自由软件理想阻碍了Linux的普及和发展**
  - Linux开始转向GPL，成为GNU阵营中的主要一员
    - ◆Linux内核，即操作系统中允许用户的软件与硬件通信的那部分
    - ◆发行版本，Linux产商借网络爱好者升级的内核，通过优化、增加功能出售各个版本的linux操作系统

1.3 Linux操作系统结构



- Kernel
  - 系统启动时将内核装入内存
  - 管理系统各种资源
- Shell
  - 用户界面，提供用户与内核交互处理接口
  - 是命令解释器，提供强大的编程环境
  - bash, ash, pdksh, tcsh, ksh, sh, csh, zsh….
- Utility
  - 提供各种管理工具，应用程序

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

1.4 Linux操作系统内核版本

版本号码

稳定版本
2.6.19-6

开发版本2.3.32

主版本号.稳定(**偶**）/开发版本（**奇**）.发布号-patch号

当前最新版本linux-4.15.10

https://www.kernel.org/

关于Linux的起源、发展与应用领域，请观看**Linux 20周年视频**

http://v.youku.com/v_show/id_XMjU3MTU2MTE2.html

1.5 开机

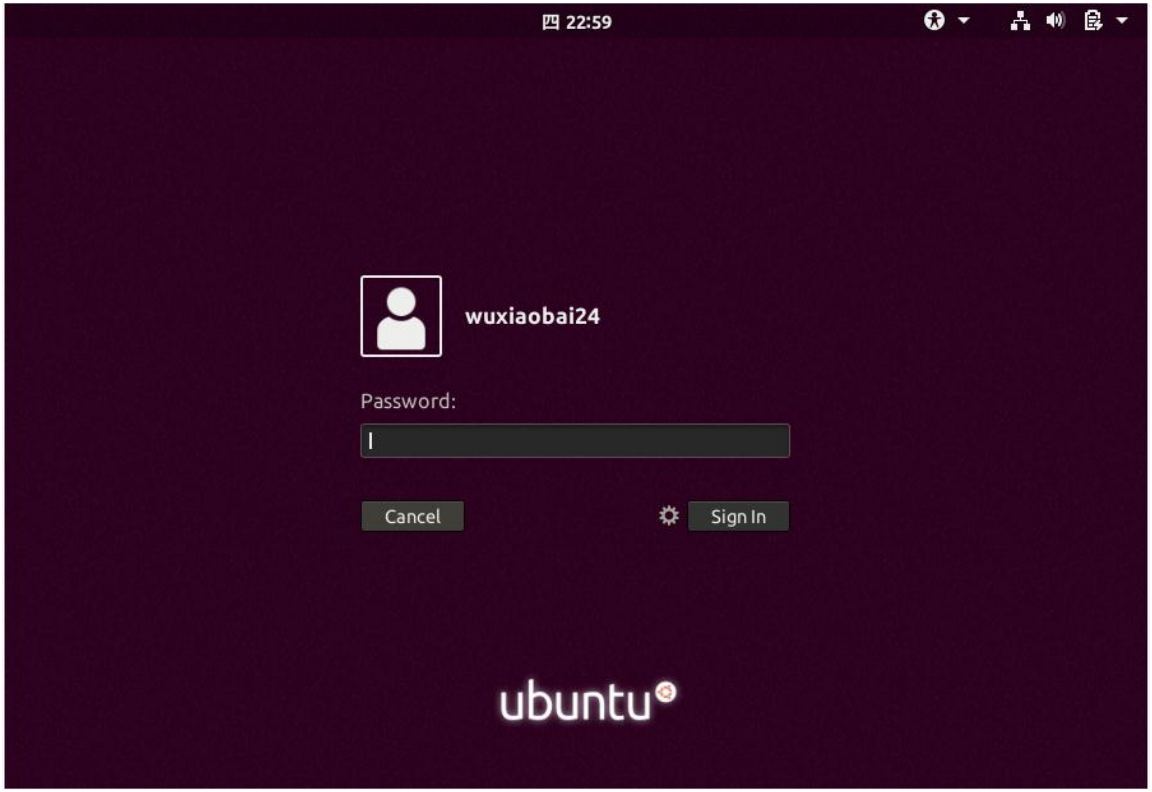
■桌面-计算机-E:\my virtual machines\

双击*.vmx ：开机

缺省账号：szu

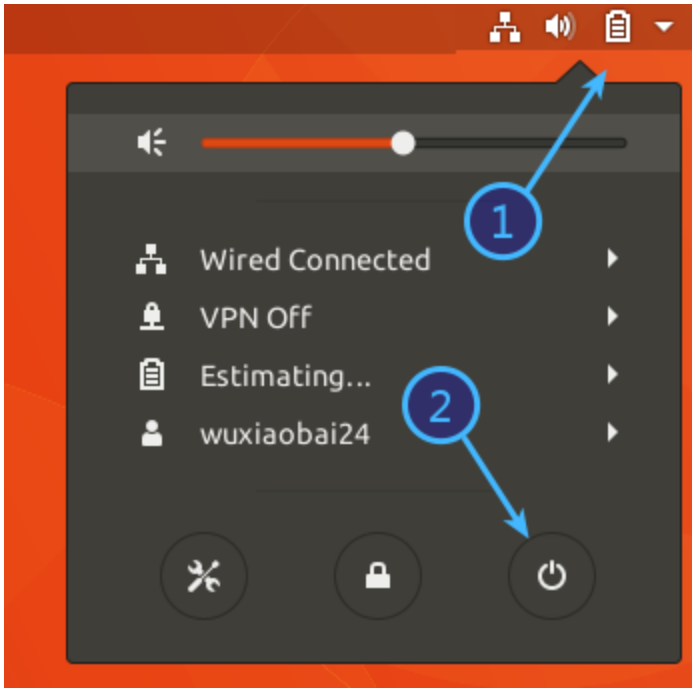
密码：123456

请不要更改此密码



1.6 关机和重新启动系统

■在切断计算机电源之前必须首先关闭Ubuntu系统
■不执行关闭Ubuntu系统就直接切断计算机的电源，会导致未存盘数据的丢失或者系统的损害



深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

1.7. 字符界面的启动

1）开机直接进入字符界面

2）在图形界面中使用字符终端：“应用程序”—“系统工具”—“终端”

或：右键选择“终端”

[szu@PC-LINUX ～]$

其中：szu：用户名

PC-LINUX：机器名

～：当前目录

$：一般用户

[root@PC-LINUX ～]#

其中：root：用户名

PC-LINUX：机器名

～：当前目录

#：超级用户

- **用户账户**
  - 超级用户/管理用户
  - 系统用户/服务用户
  - 普通用户

- **超级用户**:拥有root权限
  - 系统所有用户的管理权
  - 所有文件的处置权
  - 所有服务的使用权

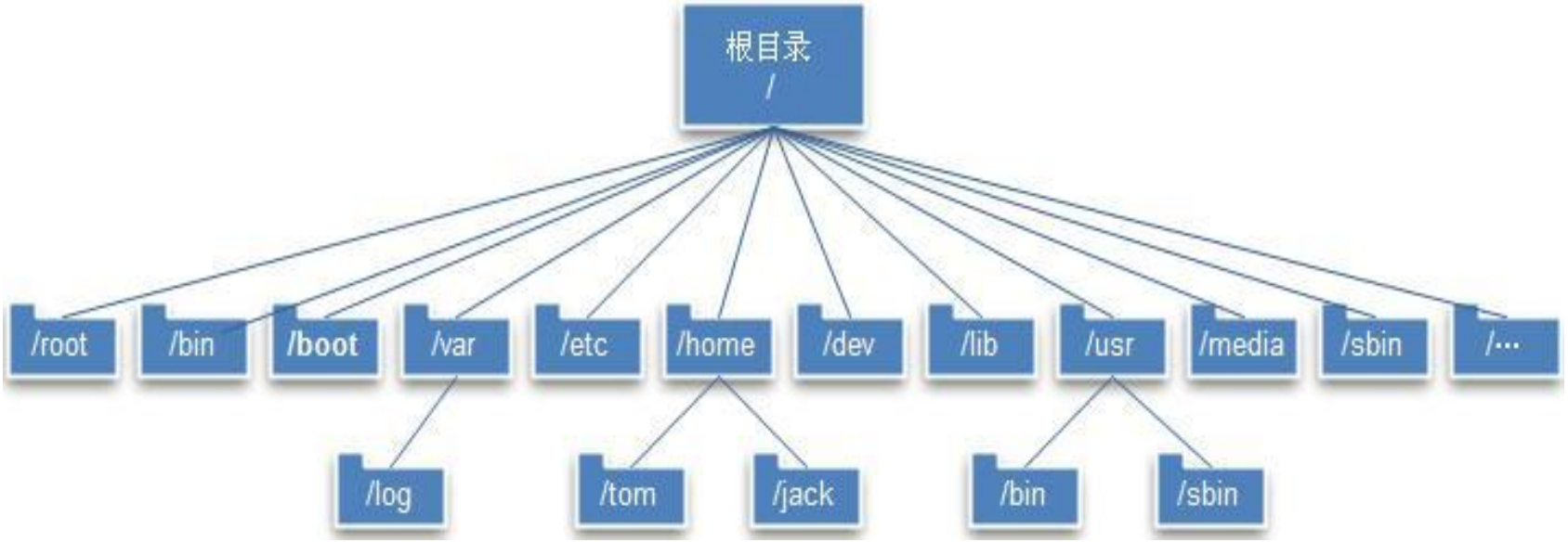
- 规范root权限使用
  - 普通操作，普通用户
  - 必要时，切换到超级用户

例一：用户dmtsai 变换身份为 root 。

```
[dmtsai@linux ~]$ su
Password: <==这里输入 root 的密码
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

1.8. Linux系统目录结构



/—根目录　/root —超级用户主目录　/bin－基本命令
/boot－kernel 和boot配置文件　/etc－各种配置文件
/usr－用户程序　/opt---　附加的应用软件包
/home－用户目录　/mnt--- 设备/文件系统挂载点
/tmp－临时文件
/var－可变信息区（file spool,logs,requests,mail,etc.）
/proc－进程信息　/dev－设备
/sbin－系统管理员执行程序　/lib--- 基本的共享库和核心模块

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

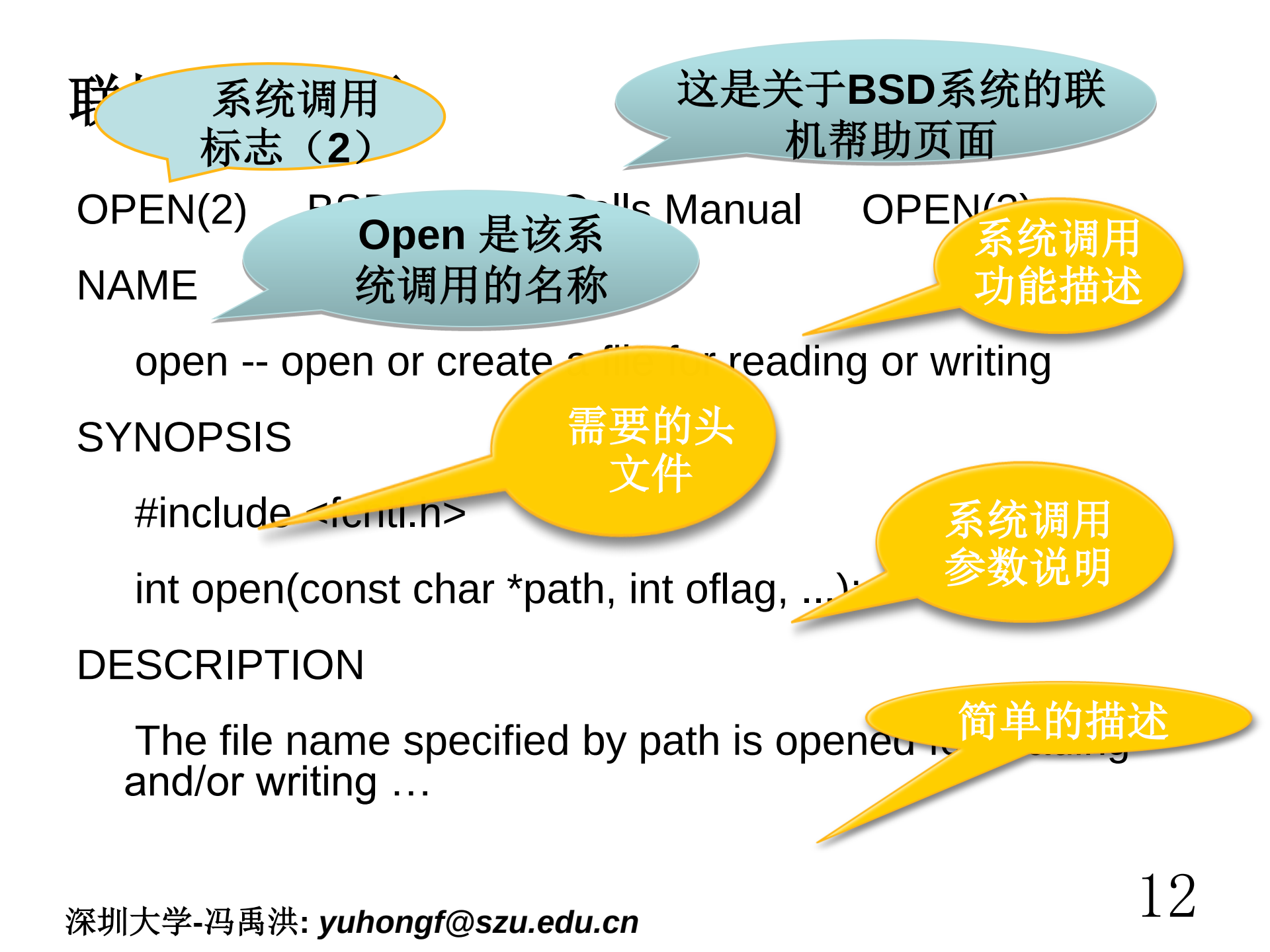
系统目录内容介绍

/sbin	系统启动时所需的二进制程序
/tmp	Temporary,存放暂存盘的目录
/usr	存放用户使用系统命令和应用程序等信息
/usr/bin	存放用户可执行程序，如grep,mdir等
/usr/doc	存放各式程序文件的目录
/usr/include	保存提供C语言加载的header文件
/usr/include/X11	保存提供X Windows程序加载的header文件
/usr/info	GNU程序文件目录
/usr/lib (/lib64)	函数库
/usr/lib(/lib64)/X11	函数库
/usr/local	提供自行安装的应用程序位置
/usr/share/man	存放在线说明文件目录
/usr/sbin	存放经常使用的程序，如showmount
/usr/src	保存系统的源码文件
/usr/bin/X11	存放X Windows System的执行程序
/var	Variable,具有变动性质的相关程序目录，如log

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

1.9. 常用命令

- man
- pwd
- cd
- ls
- touch
- cat
- more
- cp
- mv
- rm
- mkdir
- rmdir
- shutdown



深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

联机帮助例子

… (description snipped)…

RETURN VALUES

If successful… (snipped)

ERRORS

The named file is opened… (snipped)

SEE ALSO

chmod(2), close(2), dup(2), getdtablesize(2), lseek(2), read(2), umask(2), write(2)

HISTORY

An open() function call appeared in Version 6 AT&T UNIX

函数返回值

当异常发生

相关的联机帮助

相关标准描述等

pwd 命令

作用：显示当前用户所处工作目录

print working directory

格式：pwd

例子：

[yuhong@FedoraDVD13 ~]$pwd

/home/yuhong

cd 命令

作用：更改工作目录路径

格式：cd [目录名]

1. 绝对路径
   - 以/ 开头
   - /dev
   - /usr/bin
2. 相对路径
   - cd /usr
   - cd local/bin
   - pwd

     /usr/local/bin

使用”..”

作用：..目录是指向父目录的专门目录

例1：

```
$pwd
/usr/local/bin
$cd ..
```

例2：

```
$pwd
/usr/local
$cd ../share
$pwd
```

例3：

```
$cd ../bin/../bin
$pwd
```

使用”.”

作用：.目录指向当前目录，用来执行当前目录中的程序

例4：

```
$./a.out “Hello,world!”
```

使用”~”

作用：一个用户的主目录

例5：

```
$cat ~/.bashrc
$cat ~tina/.bashrc
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

ls命令

作用：打印指定目录(缺省为当前目录)里的文件和文件夹清单

格式：ls［选项］［目录或文件］

主要选项：

-a：列出目录下的所有的文件，包括以.开头的隐含文件

-A：显示除了”.”和”..”外的所有文件

-b：把文件名中不可输出的字符用反斜杠加字符编号的形式列出

-c：输出文件的i节点的修改时间，并以此排序

-d：将目录象文件一样显示，而不是显示其下的文件

-F：在每个文件名后附上一个字符以说明该文件的类型

*:可执行的普通文件　/: 目录　@:符号连接

|:表示FIFO　=:　套接字（sockets）

-i：输出文件的i节点索引信息

-l：列出文件的详细信息

ls 的应用例子

- 列举目录/文件的细节，包括权限（模式、属性）、所有者、组群、大小、创建日期、文件是否是到系统其他地方的连接，以及连接的方向

$ ls -l /bin/bash

-rwxr-xr-x 1 root wheel 430540 Dec 23 18:27 /bin/bash

文件属性	硬链接数/子目录数	文件拥有者	文件拥有者所在组	文件大小	创建月份	创建日期	创建时间	文件名

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

ls 的应用例子

- 列举目录/文件的细节，包括权限（模式、属性）、所有者、组群、大小、创建日期、文件是否是到系统其他地方的连接，以及连接的方向

$ ls -l /bin/bash

-rwxr-xr-x 1 root wheel 430540 Dec 23 18:27 /bin/bash

第一个符号：
d=目录
-=常规文件
b=块类型特殊文件
c=字符型特殊文件
s=套接字
l=链接
p=管道

设备文件
- 块特殊文件:磁盘设备
- 字符特殊文件

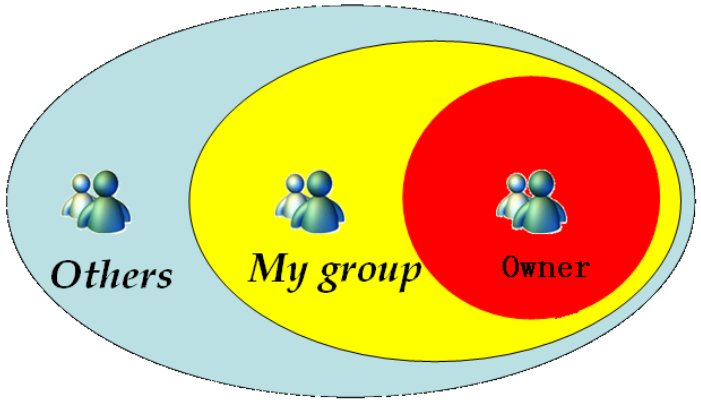
ls 的应用例子

- 列举目录/文件的细节，包括权限（模式、属性）、所有者、组群、大小、创建日期、文件是否是到系统其他地方的连接，以及连接的方向

  $ ls -l /bin/bash

-rwxr-xr-x 1 root wheel 430540 Dec 23 18:27 /bin/bash

接下来的3组：
（rwx) (rwx) (rwx)
owner group others



r 表示允许读（查看文件中的数据）

w 表示允许写（修改文件以及删除）

x 表示允许“执行”（运行程序）

touch命令

作用：创建空文件以及更改文件或目录的访问/修改时间

格式：touch [选项] [文件]

将文件的时间记录改为现在的时间。若文件不存在,系统会建立一个新的文件

```
$touch file1
$touch file1 file2 file3
```

cat命令

作用：读取文件内容并打印到标准输出，可同时读取多个文件

格式：cat [选项] 文件名

```
$touch file1
$cat file1
$echo “Hello World”>>file1
$cat file1
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

more命令

作用：分页显示文件内容

格式：more［选项］文件名

主要选项：

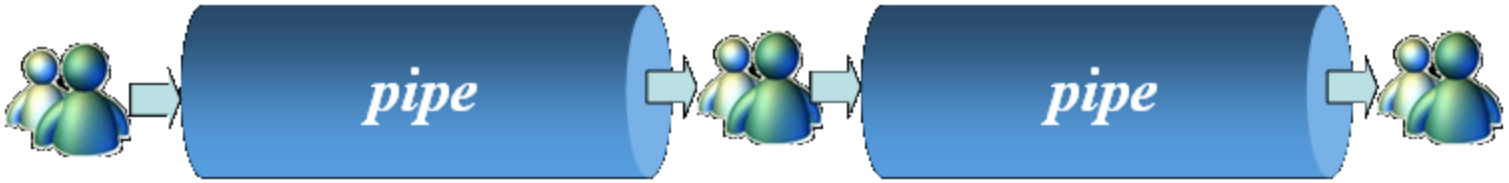
-num：一次显示的行数

+num：从第num行开始显示

-d：提示使用者，在画面下方显示［Press space to continue, q to quit.］，如果使用者按错键，则会显示［Press h for instructions.］

空白键：下一页
Ctrl+b：上一页
/　　：搜索字串
h　　：help

more命令的应用




```
$cd /etc
$ls -l
$ls -l | more
$cat filename | more
```

cp命令

作用：复制文件或者目录

格式：cp［选项］源文件或目录　目标文件或目录

主要选项：

-p：拷贝源文件的属性

-d：若源文件为连接文件的属性,则复制连接文件属性，而非复制文件本身

-r：递归持续复制目录内容及其子目录下的内容

-a：相当与-pdr

-f：强制复制，当有重复或其他疑问时，不会询问使用者

-i：交互方式操作。如果cp操作将导致对已存在的目标文件的覆盖，此时系统询问是否重写，要求用户回答y或n，这样可以避免误覆盖文件

-l：建立硬式连接，不是复制文件本身

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

cp命令应用例子

```
wuxiaobai24@wuxiaobai24-VirtualBox:~$ cd /var/log/
wuxiaobai24@wuxiaobai24-VirtualBox:/var/log$ ls -l wtmp
-rw-rw-r-- 1 root utmp 8448 3月  15 23:08 wtmp
wuxiaobai24@wuxiaobai24-VirtualBox:/var/log$ cd /tmp/
wuxiaobai24@wuxiaobai24-VirtualBox:/tmp$ ls
systemd-private-96c6eca1b4734f0cbb2ce7ca04a1dd10-colord.service-BKQtbL
systemd-private-96c6eca1b4734f0cbb2ce7ca04a1dd10-fwupd.service-TCzKrO
systemd-private-96c6eca1b4734f0cbb2ce7ca04a1dd10-rtkit-daemon.service-7n8dOF
systemd-private-96c6eca1b4734f0cbb2ce7ca04a1dd10-systemd-resolved.service-t4pWlY
wuxiaobai24@wuxiaobai24-VirtualBox:/tmp$ cp /var/log/wtmp .
wuxiaobai24@wuxiaobai24-VirtualBox:/tmp$ ls -l /var/log/wtmp
-rw-rw-r-- 1 root utmp 8448 3月  15 23:08 /var/log/wtmp
wuxiaobai24@wuxiaobai24-VirtualBox:/tmp$ ls -l wtmp
-rw-r--r-- 1 wuxiaobai24 wuxiaobai24 8448 3月  16 00:49 wtmp
wuxiaobai24@wuxiaobai24-VirtualBox:/tmp$ cp -a /var/log/wtmp .
wuxiaobai24@wuxiaobai24-VirtualBox:/tmp$ ls -l wtmp
-rw-rw-r-- 1 wuxiaobai24 wuxiaobai24 8448 3月  15 23:08 wtmp
wuxiaobai24@wuxiaobai24-VirtualBox:/tmp$
```

mv命令

作用：1. 为文件或目录改名

2. 将文件由一个目录移入另一个目录中

格式：mv [选项] 源文件或目录　目标文件或目录

主要选项：-i,-f

将文件hello.c改名为echoHello.c

$touch hello.c

$mv hello.c echoHello.c

rm命令

rm命令：删除文件

格式：rm [选项] 文件或目录

主要选项：-f，-i，-r

$rm -i echoHello.c

$touch echoHello.c

$rm echoHello.c

mkdir命令

作用：创建指定的目录

格式：mkdir［选项］目录名

主要选项：

-m：对新建目录设置存取权限，也可以用chmod命令设置

-p：可以是一个路径名称。此时若路径中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

rmdir命令

作用：删除空目录

格式：rmdir［选项］目录名

主要选项：

-p：删除目录及其先驱目录

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

mkdir应用例子

```
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$ ls
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$ mkdir dirA
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$ mkdir -m 775 dirB
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$ ls -l
total 8
drwxr-xr-x 2 wuxiaobai24 wuxiaobai24 4096 3月  16 00:56 dirA
drwxrwxr-x 2 wuxiaobai24 wuxiaobai24 4096 3月  16 00:56 dirB
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$ mkdir dirC/hello
mkdir: cannot create directory ‘dirC/hello’: No such file or directory
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$ mkdir -p dirC/hello
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$ ls -R
.:
dirA  dirB  dirC

./dirA:

./dirB:

./dirC:
hello

./dirC/hello:
wuxiaobai24@wuxiaobai24-VirtualBox:~/mkdir$
```

shutdown命令

命令语法：

shutdown［选项］［时间］［警告信息］

主要选项：

-k，-r，-h，-f，-c，-n，[time]

马上重启机器

#shutdown -r now

2. Linux上的C编程环境及工具

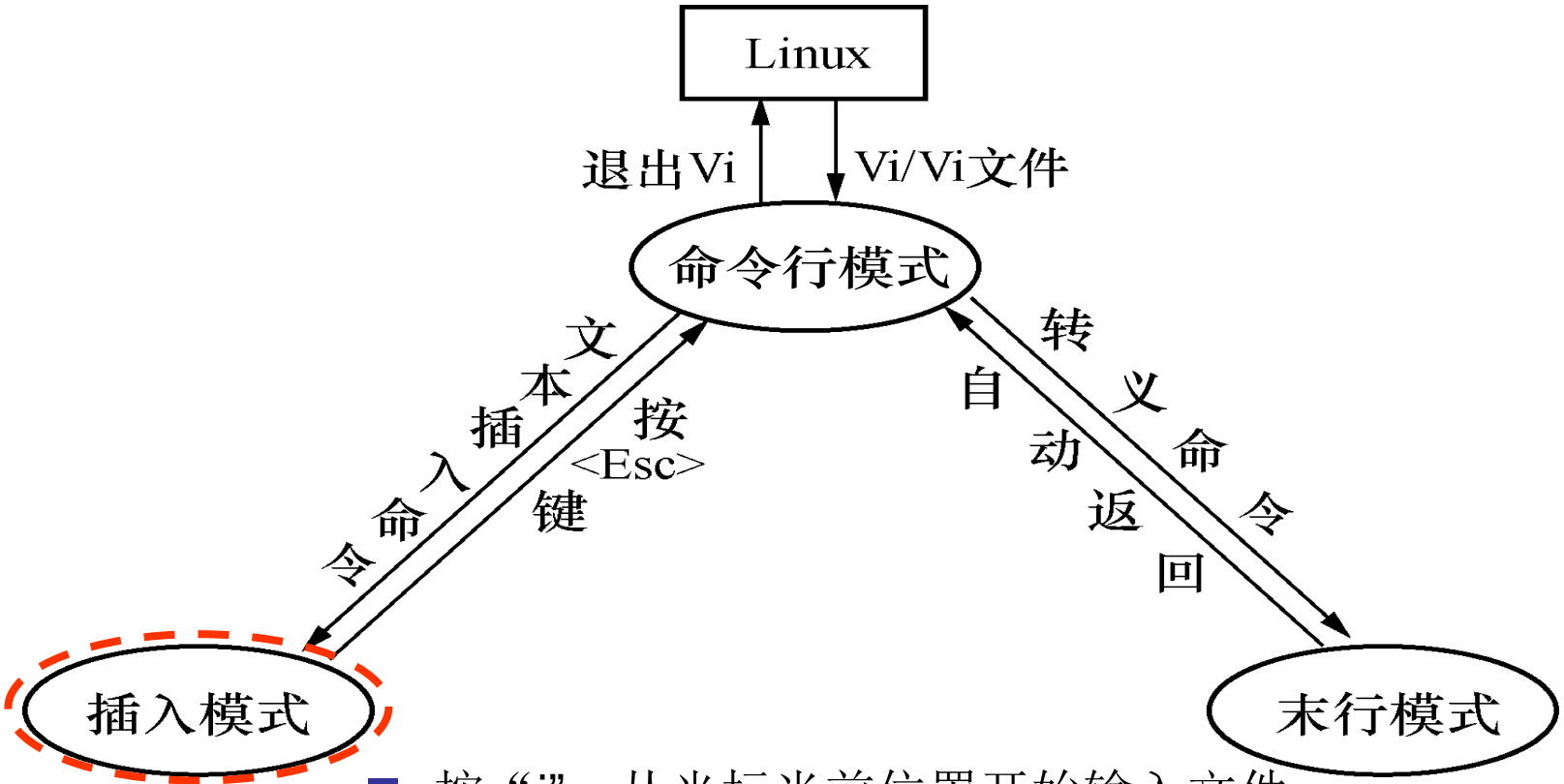
2.1. 编辑器:emacs/vi

- vi编辑器是所有Linux系统的标准编辑器，用于编辑任何ASCII文本，可以对文本进行创建、查找、替换、删除、复制和粘贴等操作

- vi编辑器有3种基本工作模式，分别是命令行模式、插入模式和末行模式。

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

vi编辑器的使用（续）

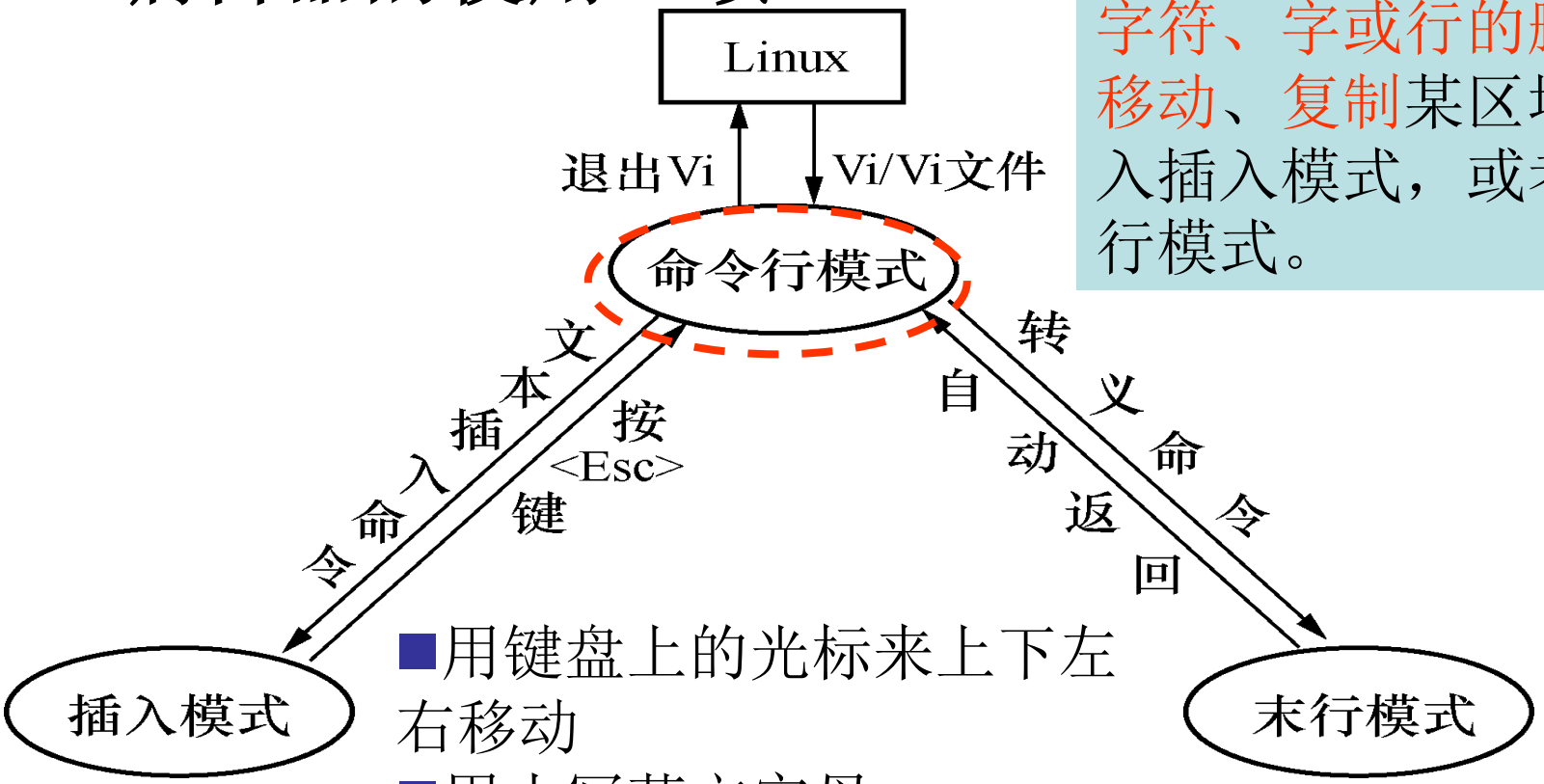


- 按“i”：从光标当前位置开始输入文件。
- 按“I”：在光标所在行的行首插入。
- 按“a”：从目前光标所在位置的下一个位置开始输入文字。
- 按“A”：在光标所在行的行末插入。

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

vi编辑器的使用（续）

控制屏幕光标的移动，字符、字或行的删除，移动、复制某区域及进入插入模式，或者到末行模式。



- 用键盘上的光标来上下左右移动
- 用小写英文字母
  - “h”：光标左移一格
  - “j”：光标下移一格
  - “k”：光标上移一格
  - “l”，光标右移一格。

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

vi编辑器的使用

■**删除文字:**

“x”：每按一次，删除光标所在位置的一个字符。

“nx”：删除光标所在位置开始的n个字符。

■**删除文字:**

“dd”：删除光标所在行。

“ndd”：从光标所在行开始删除n行。

■**复制:**

“yy”：复制光标所在行到缓冲区。

“nyy”：复制从光标所在行开始的n行字符。

“p”：将缓冲区内的内容写到光标所在位置。

vi编辑器的使用（续）

■**替换：**

“r”：替换光标所在处的字符。

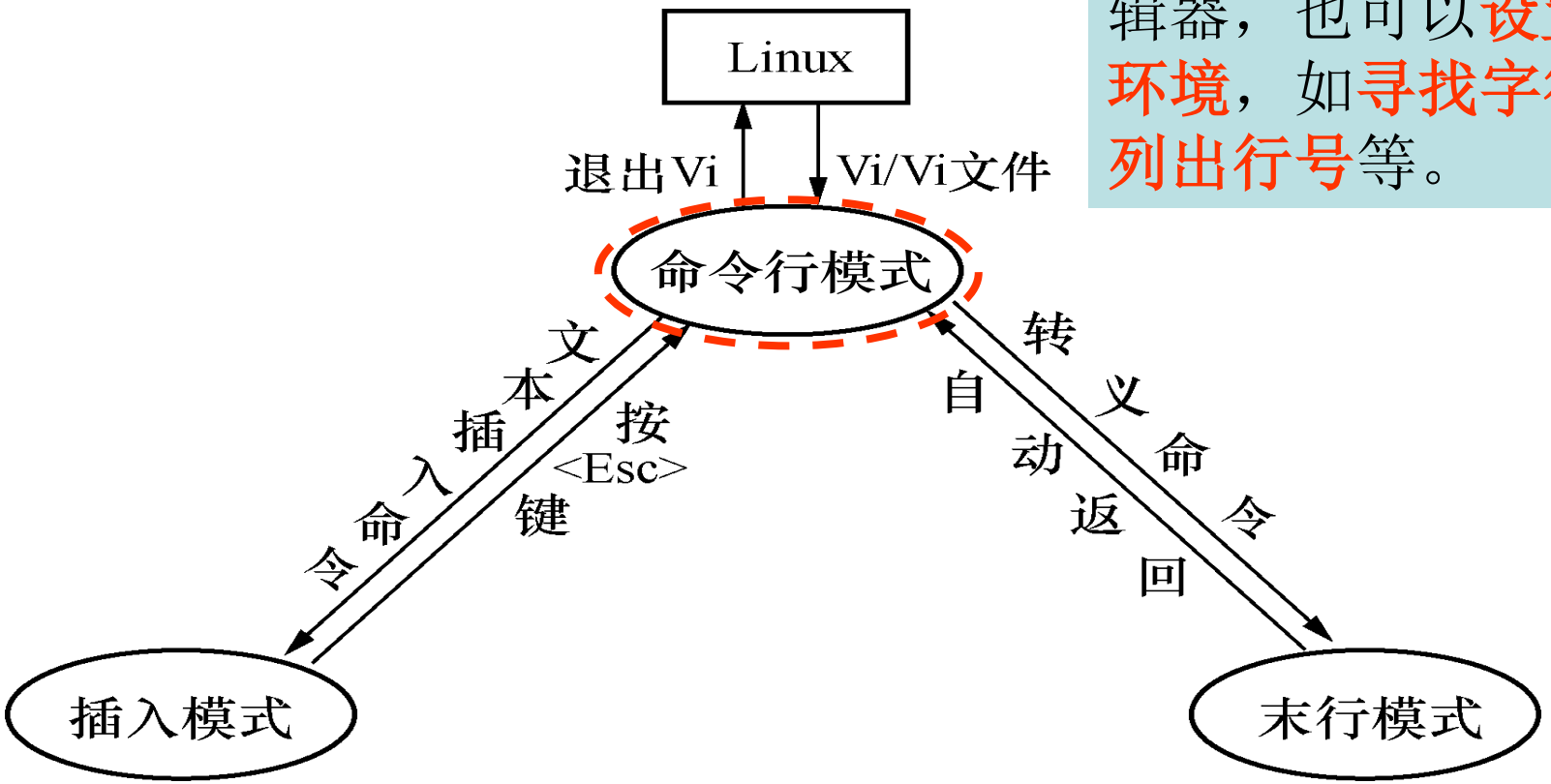
“R”：替换光标所到之处的字符，直到按下“Esc”键为止。

■**撤销上一次操作：**

“u”：回到上一个操作。按多次“u”可以执行多次撤销。

vi编辑器的使用

将文件**保存**或**退出**vi编辑器，也可以**设置编辑环境**，如**寻找字符串**、**列出行号**等。



进入末行模式操作：

先按“Esc”键确定已经处于命令行模式后，再按冒号“：”即可进入末行模式。

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

vi编辑器的使用

■查找字符操作：

“:/关键字”：查找关键字，按“n”往后查找下一个。

“:?关键字”：查找关键字，按“n”往前查找下一个。

■存盘不退出

:w

■存盘退出

:wq

■存盘时，忘了输入文件名

：w fileName(具体文件名）

■显示行号：

:set number

■取消行号显示

:set nonumber

■设置文件只读

:set readonly

vi编辑器的使用

- 替换字符串
  - 在一行内替换头一个字符串old为新的字符串new
    :s/old/new
  - 在一行内替换所有的字符串old为新的字符串new
    :s/old/new/g
  - 在两行内替换所有的字符串old为新的字符串new
    :#,#s/old/new/g
  - 在文件内替换所有的字符串old为新的字符串new
    :%s/old/new/g
  - 进行全文替换时询问用户确认每个替换需添加c选项
    :%s/old/new/gc

2.2.编辑以下C程序

reverse.h

```
/*
   * 声明函数但不给出定义
   */
int reverse(char *str);
```

reverse.c

```
#include <stdio.h>
#include "reverse.h"

/*****************************************************************/
int reverse(str)
        char *str;
{
        int i;
        int len;
        char c;
        len = strlen(str);
        for(i = 0;i < len/2; i++)
        {
                c = *str + i;
                *(str+1) = *str + len -i -1;
                *(str + len -i -1) = c;
        }
}

int main(void)
{
        char str[1024];
        printf("Give me a word to reverse:\n");
        scanf("%s", &str);
        reverse(str);
        printf("REVERSED:%s\n", str);
}
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

2.2 编译/联接器 gcc 的使用

- 编译出一个名为a.out的程序

```
$gcc reverse.c
```

- 编译出一个名为reverse(可以另命名）的程序

```
$gcc reverse.c  -o reverse
```

- 只生成目标文件

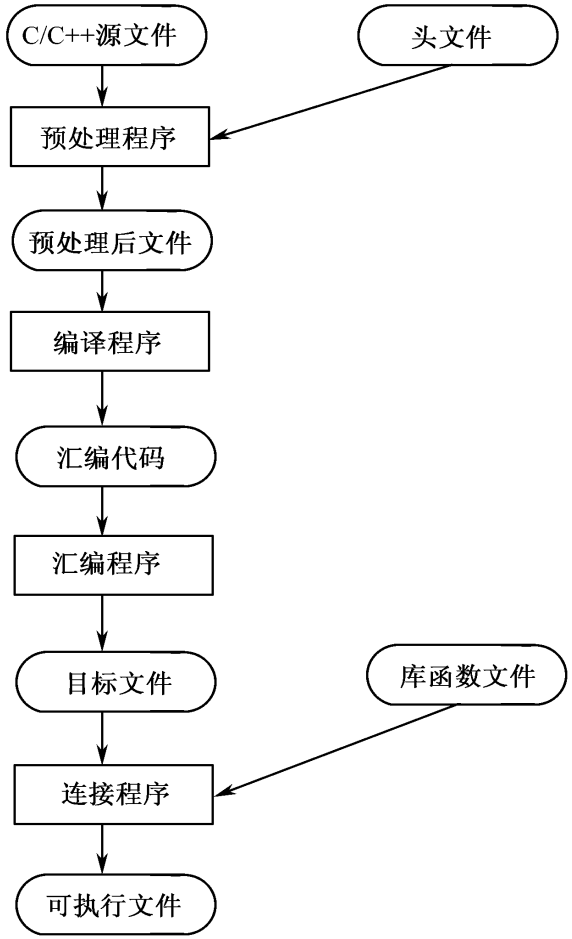
```
$gcc  -c reverse.c
```

- 生成所有的警告信息

```
$gcc  -Wall reverse.c  -o reverse
```

- 运行可执行程序

```
$./reverse
```



深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

gcc编译系统(续)

- 目前Linux平台上最常用的C语言编译系统是gcc（GNU Compiler Collection）
- 常用文件名后缀及其表示的文件类型

文件名后缀	文 件 类 型
.c	C源文件
.i	预处理后的C源文件
.ii	预处理后的C++源文件
.h	C或C++头文件
.C .cc .cp .cpp .c++ .cxx	C++源文件
.s	汇编程序文件
.S	必须预处理的汇编程序文件
.o	目标文件
.a	静态链接库
.so	动态链接库

gcc编译系统

- gcc的四个编译阶段：预处理选项、编译选项、优化选项和连接选项。

1．预处理选项

选 项 格 式	功　能
-C	在预处理后的输出中保留源文件中的注释
-D name	预定义一个宏**name**，而且其值为**1**
-D name=definition	预定义一个宏**name**，并指定其值为**definition**所指定的值。其作用等价于在源文件中使用宏定义指令：**#define name definition**。但**-D**选项比宏定义指令的优先级高，它可以覆盖源文件中的定义
-U name	取消先前对**name**的任何定义，不管是内置的，还是由**-D**选项提供的
-I dir	指定搜索头文件的路径**dir**。先在指定的路径中搜索要包含的头文件，若找不到，则在标准路径（**/usr/include, /usr/lib**及当前工作目录）上搜索
-E	只对指定的源文件进行预处理，不做编译，生成的结果送到标准输出

gcc编译系统(续)

2．编译程序选项

选 项 格 式	功　　能
-c	只生成目标文件，不进行连接。用于对源文件的分别编译
-S	只进行编译，不做汇编，生成汇编代码文件格式，其名与源文件相同，但扩展名为**.s**
-o file	将输出放在文件**file**中。如果未使用该选项，则可执行文件放在**a.out**中
-g	指示编译程序在目标代码中加入供调试程序**gdb**使用的附加信息
-v	在标准出错输出上显示编译阶段所执行的命令，即编译驱动程序及预处理程序的版本号

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

3．优化程序选项

- 优化分为对中间代码的优化和针对目标码生成的优化。

4．连接程序选项

选 项 格 式	功　　能
object-file-name	不以专用后缀结尾的文件名就认为是目标文件名或库名。连接程序可以根据文件内容来区分目标文件和库
-c -S -E	如果使用其中任何一个选项，那么都不运行连接程序，而且目标文件名不应该用做参数
-llibrary	连接时搜索由library命名的库。连接程序按照在命令行上给定的顺序搜索和处理库及目标文件。实际的库名是liblibrary.a
-static	在支持动态连接的系统中，它强制使用静态链接库，而阻止连接动态库；而在其他系统中不起作用
-Ldir	把指定的目录dir加到连接程序搜索库文件的路径表中，即在搜索-l后面列举的库文件时，首先到dir下搜索，找不到再到标准位置下搜索
-Bprefix	该选项规定在什么地方查找可执行文件、库文件、包含文件和编译程序本身数据文件
-o file	指定连接程序最后生成的可执行文件名称为file，不是默认的a.out

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

gcc常用参数

- 对程序进行优化编译、连接，提高可执行文件的执行效率可以提高。

  $gcc -O

  $gcc -O2 //比-O更好的优化编译、连接。

- C程序中的头文件包含两种情况：

A)#include <stdio.h>

B)#include “myinc.h”

$gcc -I dirname

//将dirname所指出的目录加入到程序头文件目录列表中

gcc常用参数

使用外部函数库

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define MAX_INPUT 25

int main(int agrc, char **argv)
{
        char input[MAX_INPUT];
        double angle;

        printf("Give me an angle (in radians) ==>");
        if (!fgets(input, MAX_INPUT, stdin)){    perror("an error occurred.\n");    }
        angle = strtod(input, NULL);

        printf("sin(%e) = %e\n", angle, sin(angle));
        return 0;
}
```

```
$gcc  -o trig  -lm trig.c

//“-lm”选项告诉gcc查看系统提供的数学库libm
//函数库一般在/lib 或/usr/lib中
```

2.3 GDB, GNU调试工具

- 使用语法

  gdb [<programfile> [<corefile>|<pid>]]

  <programfile>　可执行的二进制文件

  <corefile>　核心转储程序（ core dump，crash records， 故障记录)

  <pid>　运行程序的进程id

- 编译时候用上 –g 选项

  $gcc -g <programfile>

基本的 gdb 命令

- 常用命令:

file [<file>]	装入想要调试的可执行程序 **<file>**
run [<args>]	运行选择的程序并将参数**<args>**传给它
attach <pid>	将**gdb attach**到一个运行时的进程 **<pid>**，并调试它
kill	终止正在调试的程序
quit	终止**gdb**
help [<topic>]	帮助

- 单步跟踪和从断点开始继续执行:

c[ontinue]	从断点开始继续执行
s[tep]	执行下一个源程序行，必要时进入下一个函数
n[ext]	在不单步执行进入其他函数的情况下，向前执行一行源代码
finish	运行程序，直到当前函数完成返回。并打印函数返回时的堆栈地址和返回值及参数值等信息。

深圳大学-冯禹洪: ***yuhongf@szu.edu.cn***

gdb 断点

- **Useful breakpoint commands:**

b[reak] [<where>]	在代码里设置断点, 这将使程序执行到这里时被挂起. **<where>** 可以16进制地址、函数名、行号、相对的行位移。 **b CMSTask.c:200** **b buildPubWinTask**
[r]watch <expr>	为表达式（变量） **expr** 设置一个观察点。一旦表达式值被写[或读]时，马上停住程序
info break[points]	列出当前所有的断点
clear [<where>]	去除一个指定的断点

用gdb观察数据变化

- **查看相关的命令:**

list [<where>]	列出产生执行文件的源代码部分 <where>
search <regexp>	在源文件中搜索正规表达式 <regexp>
backtrace [<n>]	显示程序中的当前位置和表示如何到达当前位置的栈跟踪 <n>
info [<what>]	显示与该程序有关的各种信息 <what> (可以是局部变量和函数参数) info breakpoints/threads/locals……
p[rint] [<expr>]	打印变量或是表达式的值 <expr>

- **修改数据和控制路径的命令:**

set <name> <expr>	给变量或参数赋值
jump <where>	在源程序中的另一点<where>开始运行

深圳大学-冯禹洪: ***yuhongf@szu.edu.cn***

gdb 例子

$ gcc -g reverse.c -o reverse1

1）启用GDB调试——

```
wukunhan_2015170297@wuxiaobai24-VirtualBox:~/gdbdebug$ gdb ./reverse1
GNU gdb (Ubuntu 8.0.1-0ubuntu1) 8.0.1
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./reverse1...done.
(gdb)
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

gdb 例子

2）键入list,查看源代码并根据行号/函数名设置断点

```
(gdb) break main
Breakpoint 1 at 0x84d: file reverse.c, line 23.
(gdb) break 16
Breakpoint 2 at 0x7ce: file reverse.c, line 16.
(gdb) break 17
Breakpoint 3 at 0x7df: file reverse.c, line 17.
(gdb) break 18
Breakpoint 4 at 0x808: file reverse.c, line 18.
(gdb) break reverse
Breakpoint 5 at 0x7b6: file reverse.c, line 13.
(gdb) info break
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x000000000000084d in main at reverse.c:23
2       breakpoint     keep y   0x00000000000007ce in reverse at reverse.c:16
3       breakpoint     keep y   0x00000000000007df in reverse at reverse.c:17
4       breakpoint     keep y   0x0000000000000808 in reverse at reverse.c:18
5       breakpoint     keep y   0x00000000000007b6 in reverse at reverse.c:13
(gdb) run
Starting program: /home/wukunhan_2015170297/gdbdebug/reverse1
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

gdb 例子

3）由断点结果变量值推测错误

```
Give me a word to reverse:
abcd

Breakpoint 5, reverse (str=0x7fffffffdb50 "abcd") at reverse.c:13
13              len = strlen(str);
(gdb) c
Continuing.

Breakpoint 2, reverse (str=0x7fffffffdb50 "abcd") at reverse.c:16
16                      c = *str + i;
(gdb) c
Continuing.

Breakpoint 3, reverse (str=0x7fffffffdb50 "abcd") at reverse.c:17
17                      *(str+i) = *str + len -i -1;
(gdb) c
Continuing.

Breakpoint 4, reverse (str=0x7fffffffdb50 "dbcd") at reverse.c:18
18                      *(str + len -i -1) = c;
(gdb) c
Continuing.

Breakpoint 2, reverse (str=0x7fffffffdb50 "dbca") at reverse.c:16
16                      c = *str + i;
(gdb) c
Continuing.

Breakpoint 3, reverse (str=0x7fffffffdb50 "dbca") at reverse.c:17
17                      *(str+i) = *str + len -i -1;
(gdb) c
Continuing.

Breakpoint 4, reverse (str=0x7fffffffdb50 "dfca") at reverse.c:18
18                      *(str + len -i -1) = c;
(gdb) c
Continuing.
REVERSED:dfea
```

断点中的16、17行位置有错误。

即 c = *str+i;和 *(str+i) = *str+len-i-1 两句有错误。

深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

gdb 例子

4）仔细观察变量的值和思考程序的逻辑

```
Breakpoint 3, reverse (str=0x7fffffffdb50 "dbca") at reverse1.c:17
17                      *(str+i) = *(str + len -i -1);
(gdb) c
Continuing.

Breakpoint 4, reverse (str=0x7fffffffdb50 "dcca") at reverse1.c:18
18                      *(str + len -i -1) = c;
(gdb) p c
$1 = 101 'e'
(gdb)
```

- 程序逻辑：0~n/2-1，对称位置，0 ：n-1，1：n-2，…交换字符值
- 此时测试的c应为’b’，但结果显示c为‘e’，说明以下语句有误

```
c = *str+i;
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*

gdb 例子

5）修改代码

```c
/* REVERSE2.c */
#include <stdio.h>
#include <string.h>
#include "reverse.h"

/*****************************************************************/
int reverse(str)
        char *str;
{
        int i;
        int len;
        char c;
        len = strlen(str);
        for(i = 0;i < len/2; i++)
        {
                c = *(str + i);
                *(str+i) = *(str + len -i -1);
                *(str + len -i -1) = c;
        }
}

int main(void)
{
        char str[1024];
        printf("Give me a word to reverse:\n");
        scanf("%s", &str);
        reverse(str);
        printf("REVERSED:%s\n", str);
}
```

6）验证结果

```
wukunhan_2015170297@wuxiaobai24-VirtualBox:~/gdbdebug$ vim reverse1.c
wukunhan_2015170297@wuxiaobai24-VirtualBox:~/gdbdebug$ vim reverse1.c
wukunhan_2015170297@wuxiaobai24-VirtualBox:~/gdbdebug$ gcc -g reverse1.c -o test
wukunhan_2015170297@wuxiaobai24-VirtualBox:~/gdbdebug$ ./test
Give me a word to reverse:
abcde
REVERSED:edcba
wukunhan_2015170297@wuxiaobai24-VirtualBox:~/gdbdebug$
```


深圳大学-冯禹洪: *yuhongf@szu.edu.cn*