

粒子物理与核物理实验中的 数据分析

王喆，杨振伟
清华大学

第一讲：Linux环境及shell
编程(1)

本讲摘要

- 什么是Linux
- 为什么使用Linux
- Linux简介(内核、shell以及目录结构等)
- Linux终端的常用命令(!!!)
- Linux终端的常用编辑器(vi, emacs)
- Linux环境变量与Shell 脚本(!!!)
- 常见Linux程序简介

什么是Linux

什么是Linux？

Linux是众多操作系统的一种



主要特点

源代码开放，自由软件/代码

强大的shell指令以及shell编程功能：

cd, ls, grep, find, sed...

大量的科学计算、数据分析处理的程序包
(CERN、FermiLab、KEK以及其它众多机构
提供支持)

Linux的各种版本

Linux有众多的发行版本，

(1) ubuntu: 与桌面系统结合得比较好，而且更新的最快，对新硬件适应最好

(2) Redhat发行Redhat Enterprise Linux(**RHEL**)发行版，新版较慢

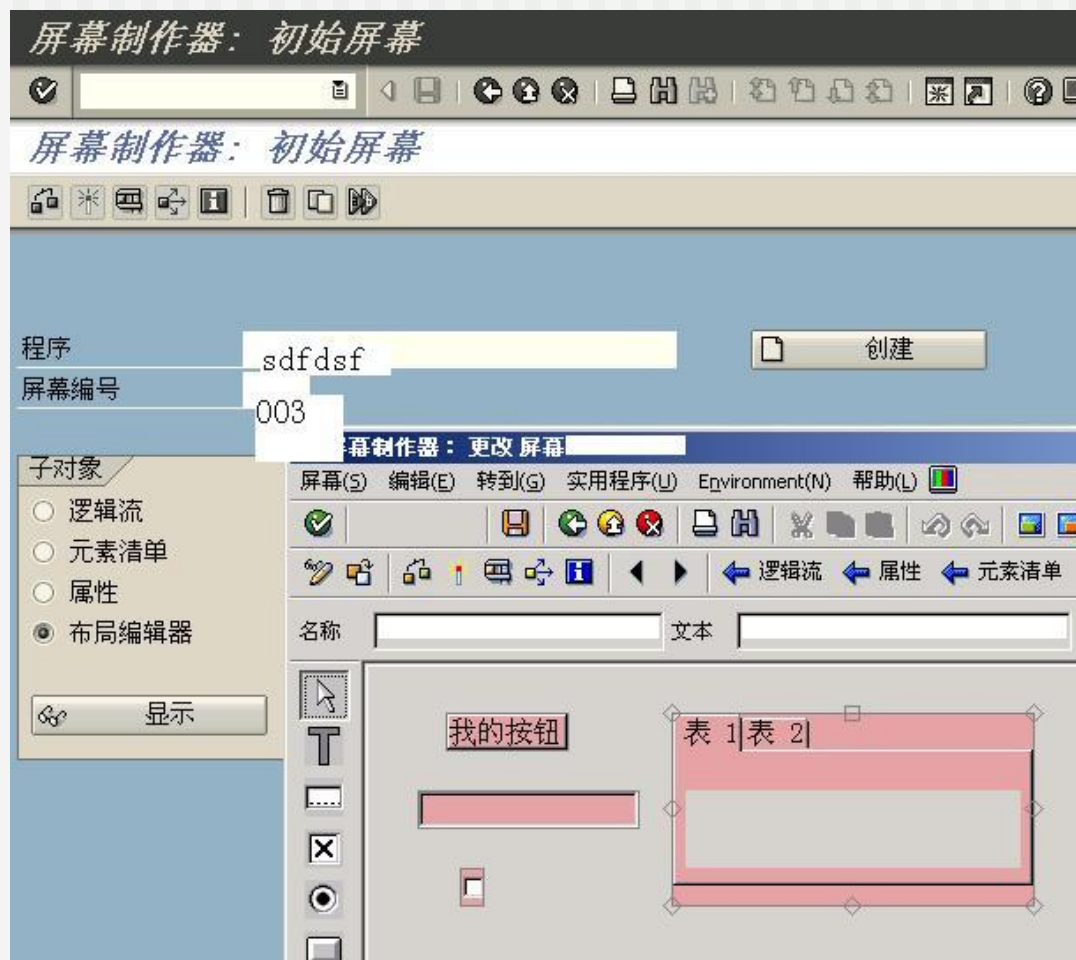
(3) 粒子与核物理界普遍使用的是 Scientific Linux CERN(SLC)或Scientific Linux(SL),



(4) 与linux相关，相似: Android, Mac 等等

图形界面的应用困难

- 图形界面适合一些低频率和预设的操作。
- 图形界面的生成复杂，设计人员往往是先开发实际功能，然后再开发图形界面



为什么使用Linux

为什么使用Linux？

源代码开放，自由软件/代码（GNU license）

Linux的强势不在桌面、图形、游戏等方面

Linux的强势主要在于科学研究和开发方面

尤其是在需要大量计算或编程进行数据分析的科研工作中

强大的shell命令和脚本，多任务长时间的运算
各种编程语言的编译程序和环境

（C++，C，Fortran，java，python等）

提示：在Windows，Mac等下我们只有软件的成品，
并没有开发环境，要编程是需要购买编译器的。

Linux 的内核和shell

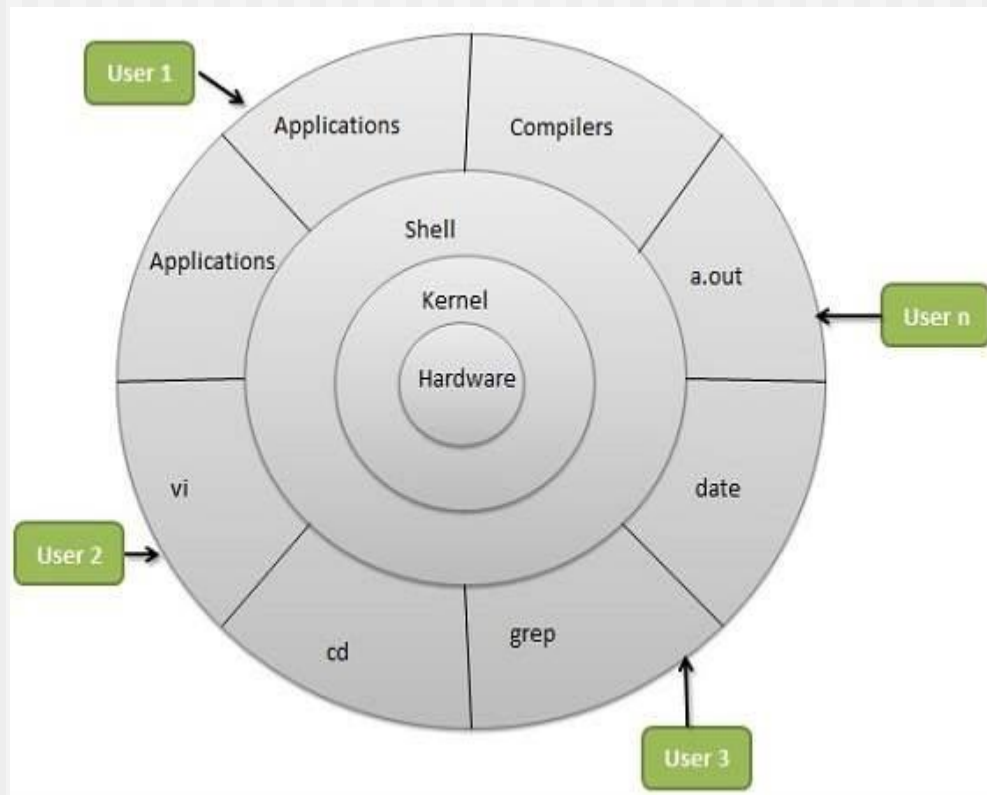
UNIX/Linux的任务可以简单地分为两部分：

1. 承担操作系统与计算机之间的互动工作

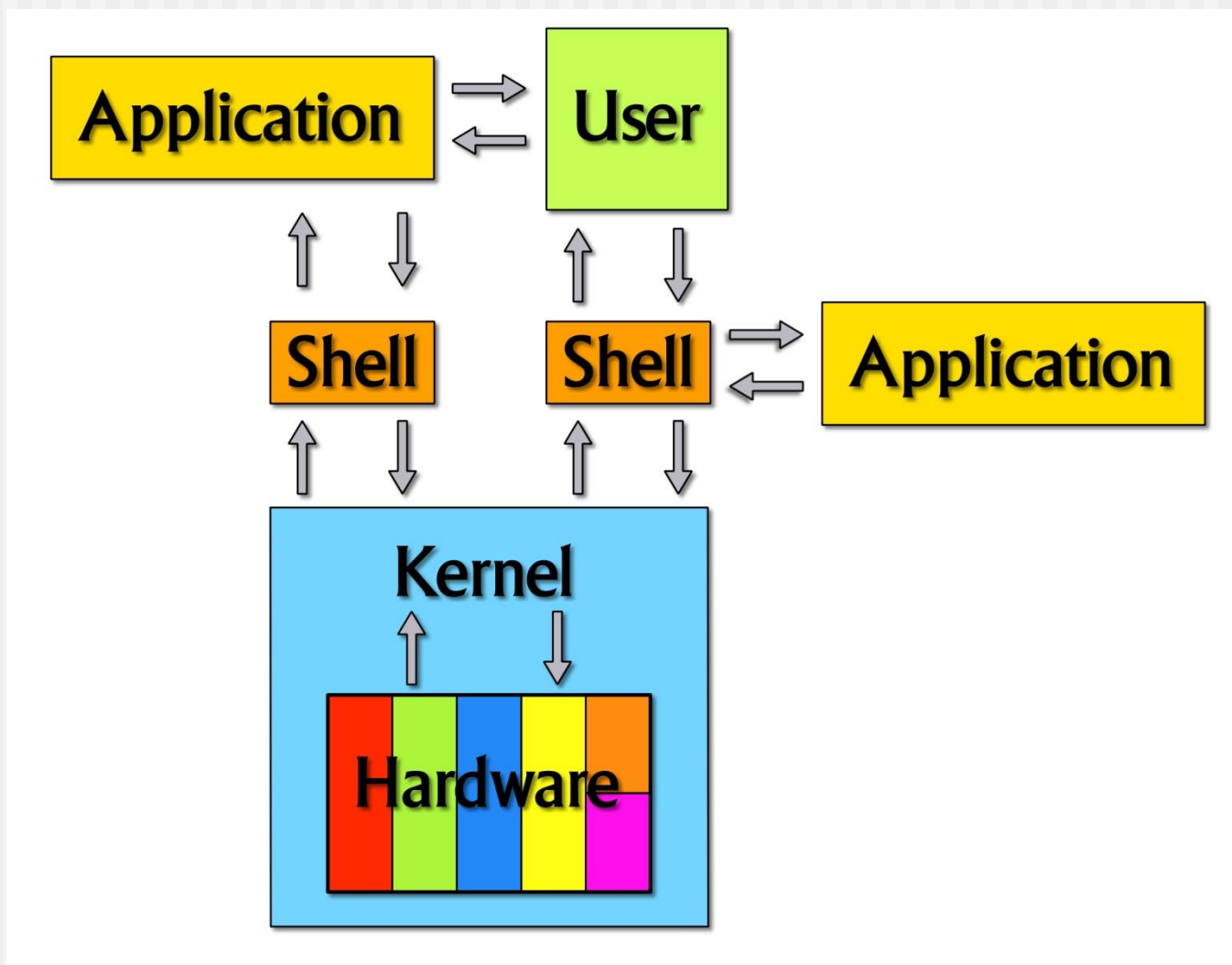
--内核(kernel)

2. 承担操作系统与用户之间的互动工作

--shell.

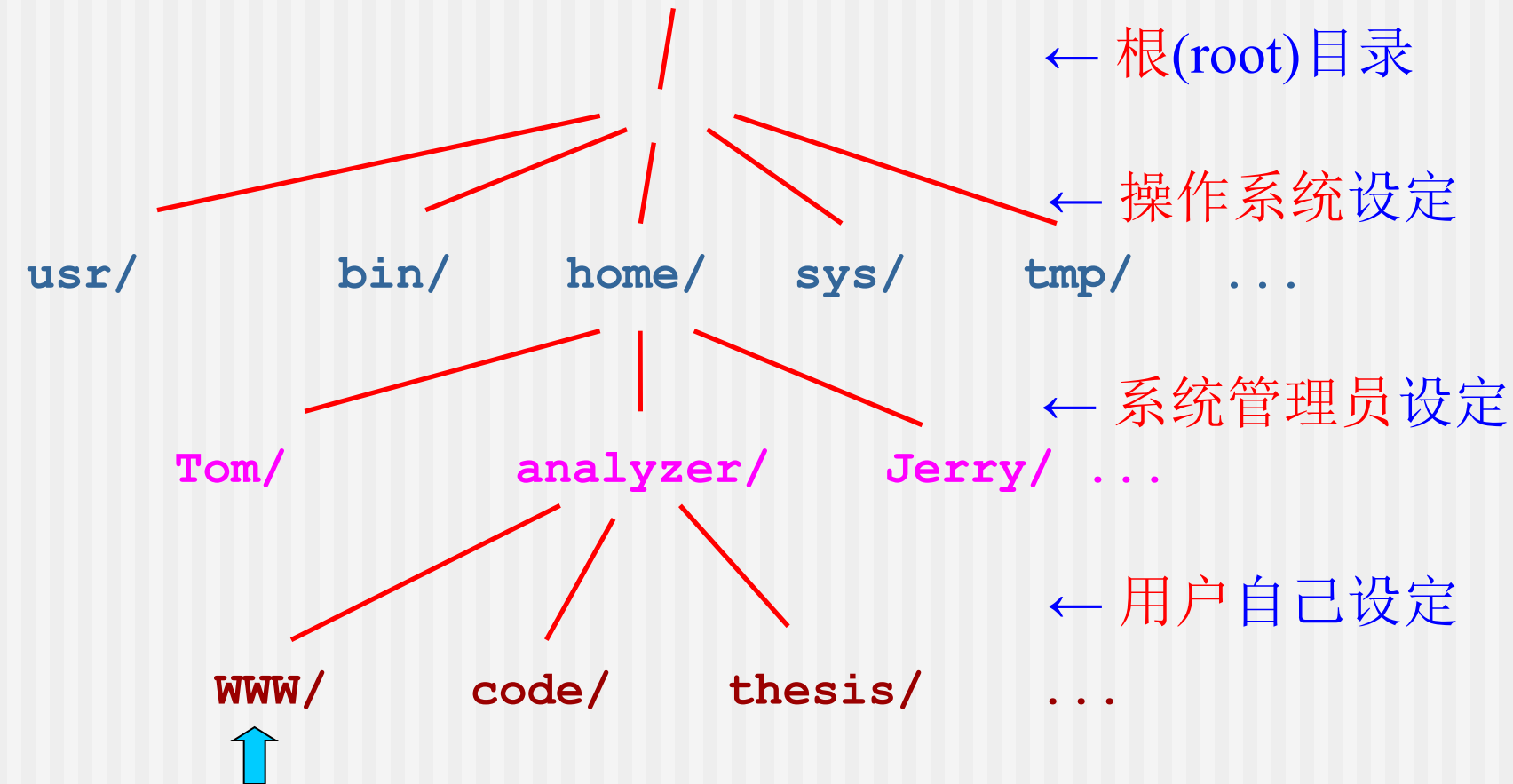


问题， 电脑响应输入指令的过程？



Linux的文件目录

UNIX/Linux对文件与目录的管理, 整体呈树状结构:



Linux: public_html/

2017/2/24

文件所有者、组和权限

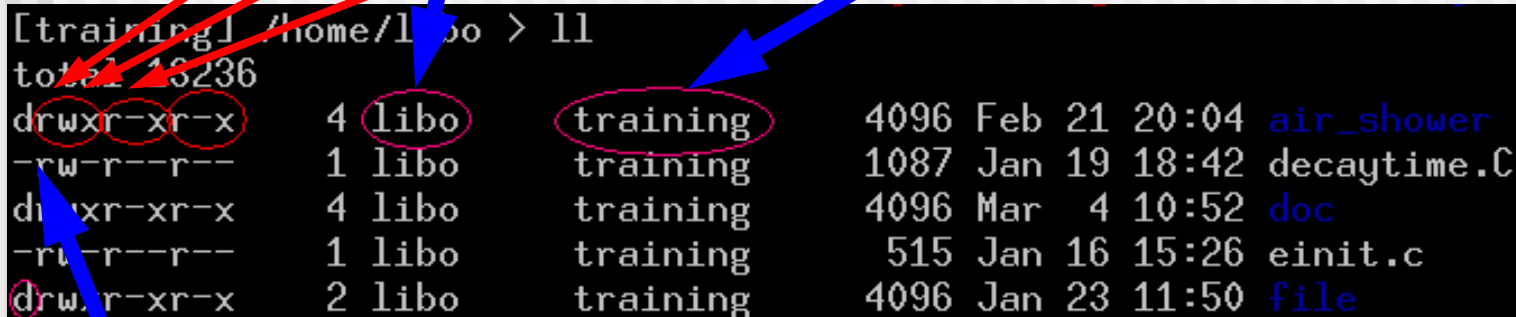
所有者与权限:

Linux文件都有特定的所有者和所有组。

文件具有3类权限: 可读(r)、可写(w)、可执行(x)

所有者有权规定3类用户对该文件的权限:

所有者自己(owner)、所有者同组成员(group)、其它用户(other)



```
[training] /home/libo > ll
total 18236
drwxr-xr-x  4 libo  training 4096 Feb 21 20:04 air_shower
-rw-r--r--  1 libo  training 1087 Jan 19 18:42 decaytime.C
drwxr-xr-x  4 libo  training 4096 Mar  4 10:52 doc
-rw-r--r--  1 libo  training  515 Jan 16 15:26 einit.c
drwxr-xr-x  2 libo  training 4096 Jan 23 11:50 file
```

最前面的字母"d"表示这是个目录
"- "表明没有该位置对应的权限

2017/2/24

Linux的绝对和相对路径

一个完整文件名应指定出其所处在的**路径** (path),
路径有**相对路径**和**绝对路径**

绝对路径: `/home/Tom/geom/geom.dat` (都是以"/"开始)

"~"符号可以用来简单表示home: (相对路径的例子)

`~/geom/geom.dat` ← 所登录的用户 (e.g. Tom)

`~Jerry/geom/result.dat` ← 不同的用户 (e.g. Jerry)

单个点"." 表示当前目录, 两个点".."表示上一层目录

`/home/Tom/geom` ← 当前目录

`../geom` ← 表示 `/home/Tom/geom`

这个知识对编写通用的 **shell** 脚本文件很有用。

常见的Shell

- 目前广泛应用的几大类shell(即指令集)：
`csch, tcsh, bash, sh`
- 目前一般系统默认的是bash，高能物理里常用的为tcsh。语法稍微有些区别，大同小异，习惯就好。
- shell都有一些内部指令和外部指令
 - 各种shell的好多内部指令都是一样的，例如ls, cd等偶尔不一样，例如setenv (tcsh) 和export (bash)
 - 所谓外部指令是一些独立的运行程序，例如ssh，也是各个shell通用的。
 - 如果有的外部命令你没有，可能是没有安装或路径找不到

Linux终端的常用shell命令(1)

pwd	显示当前目录(print working dir)
passwd	修改当前用户的密码
ls [-lahrt]	列出当前目录中的文件(list)
cd [dir]	进入指定目录或从当前目录回到用户的home目录 cd, cd foo, cd ../ cd /home/zhanghb/
mkdir dfoo	生成名为 dfoo 的子目录
rm [-rf] foo	删除文件 foo (参数rf表示强制删除文件夹, 慎用)
rmdir foo	删除名为 foo 的子目录(foo 应已经为空目录)
cp foo bar	拷贝文件 foo 到另一文件 bar
mv foo bar	更改文件 foo 的名称为 bar
man <command>	显示 command 指令说明
man -k <keyword>	寻找 “ keyword ”指令说明页
history	列出最近使用过的指令很有用
du	显示当前目录所用空间大小

注意指令字母大小写。Linux区分一切大小写, 指令, 文件名, 目录名

Linux终端的常用命令(2)

more foo	显示名为foo 的文件(按空格键换页)
less foo	与 more foo类似, 但可以往回翻页(按q 退出)
emacs foo &	用emacs 编辑名为 foo 的文件(& 为提交后台进程)
vi, pico, nano, ... 这些命令都以可编辑方式打开文件	
ps	显示正在运行的进程
kill 345	删除进程 345 (如果不行可尝试使用 kill -9)
./foo	在当前目录运行可执行文件 foo
ctrl-c	中断目前在前台执行的进程
ln -s source linkname	为source建立一个符号链接linkname
locate foo	在所有目录中寻找有文件名 foo 的路径
find . -name file1	在当前目录中寻找文件名为file1的路径
grep TH1F foo	显示文件 foo 中含 “TH1F” 的每一行
sed -e "s/str1/str2/g" foo > bar	将文件 foo 中字符串 “str1”改为 “str2”并将修改后的文件写到新文件 bar , foo 保持不变。

Linux终端的常用命令(3)

chmod 755 <file>	更改文件file的属性,1:x 2:w 4:r 5:rx 7:rwx
chmod ug+x foo	使文件 foo 对用户与同小组成员增加执行权限
diff file1 file2	比较文件file1和file2的不同
tar -cvfz 1.tgz file1 file2	压缩file1, file2为1. tgz
tar -xvfz 1.tgz	解压缩1. tgz
gcc test.c -o try1	用C编译器编译test.c, 生成可执行文件try
g++ test.cpp -o try2	用C++编译器编译程序
date	显示系统当前时间
sleep 10	暂停10秒钟
wc [-lw] file	显示file的行数/字数等信息
echo "Welcome to Linux World!"	屏幕显示指定字符串
file file1	显示文件file1的属性

注: Linux有些特殊字符, 比如 >, |, &等符号

ls > list.txt 将ls的结果写入list.txt, 即重定向

ps aux | grep yangzw 显示跟用户yangzw有关的进程, 即通道

Emacs、Vi编辑器的基本指令

emacs: 很好很强大

打开文件 `emacs [filename]`

不要窗口 `emacs -nw [filename]`

保存文件 `Ctrl+x Ctrl+s` (连续两次组合键)

退出文件 `Ctrl+x Ctrl+c` (连续两次组合键)

vi (vim): 古老，不过也很强大

打开文件 `vi [filename]`

保存文件 `:w` (注意：是输入冒号然后输入w或q或q!)

退出文件 `:q`

不存退出 `:q!`

注：vi有两种模式，命令模式和输入模式

按小写字母“i”进入输入模式，按“esc”键进入命令模式

在命令模式中可以输入命令很方便的进行编辑修改

讲义最后列了一些vi的常用命令

熟练使用任何一种编辑器都可以极大提高工作效率，建议多多练习。

Shell、环境变量和脚本(1)

shell中有很多环境变量，有的是系统的环境变量，有的是用户自己定义的环境变量，为系统和用户程序服务。

环境变量一般用大写字母定义(有些类似于C语言的宏定义)

比如**PATH**，**PWD**，**USER**，**GROUP**等都是系统环境变量。

PATH: 可执行程序的搜索路径集合，

如果里面不包含“.”，则当前目录下的可执行程序是找不到，并不能被执行的。

查看所有环境变量: **env**或者**printenv**

查看环境变量**PATH**的值: **echo \$PATH**或**printenv PATH**

定义环境变量

export ANADIR=/home/analyzer (bash)

setenv ANADIR /home/analyzer (tcsh)

取消环境变量 **unset ANADIR**

Shell、环境变量和脚本(2)

什么是脚本(script):

脚本就是用于实现某种目的的命令集合。

这些命令集合放在一个文件中，由shell来解析执行。

为什么需要用脚本:

很多工作是重复性的，脚本可以让你更高效。

比如用脚本循环修改程序的某一部分，自动运行。

目 标:

1)知道什么是shell脚本(script)，如何写自己的脚本

2)可以看懂别人的脚本

执行shell脚本，比如有脚本test.sh:

>test.sh 或者 > ./test.sh

注：运行前确保用户对test.sh有可执行权限，否则，

chmod u+x test.sh

Shell、环境变量和脚本(3)

例：最简单的一个脚本

编写一个shell脚本test.csh:

```
#!/bin/tcsh
# This is a simple test shell script

echo "Hello everyone!"
ls /projects/$USER
date
echo $PWD
```

注：1) 标准的脚本都以“#!”开头，后面跟随bash/tcsh或其它脚本程序的路径(用which bash指令可以查看bash的路径)

2) 注释行以“#”开头(第一行的#!除外)

3) 需要执行的指令(一般每行一个指令)

Shell、环境变量和脚本(4)

```
#!/bin/bash
# Another test shell script
```

```
####for循环####
```

```
for i in `ls /home/analyzer`
do
    echo $i
done
```

```
####while循环###
```

```
num=1
DIR="testDir"
while (( $num < 5 ))
do
    if [ -d $DIR$num ]; then
        echo "$DIR$num exist!!"
    else
        mkdir $DIR$num
    fi
    let num+=1
done
```

```
#!/bin/tcsh
# Another test shell script
```

```
####for循环####
```

```
foreach i `ls /home/analyzer`
    echo $i
end
```

```
####while循环###
```

```
num=1
DIR="testDir"
while ( $num < 5 )
    if ( -d $DIR$num ) then
        echo "$DIR$num exist!!"
    else
        mkdir $DIR$num
    endif
    set num=`expr $num +1`
end
```

脚本中变量和循环的例子: bash vs tcsh

Python入门

python功能极强，它的脚本更友好，当下很流行。

启动和退出

python

>>>[ctrl+d]

* 命令行

python

>>> import os

>>> os.listdir('.')

>>> [ctrl+d]

python的问题：

语法检查松散，
变量不需要声明，
可引入模块丰富，
引用方法丰富，语
法说明少，查错效
率低，有的时候效
率低。

练习一个python脚本

- 生成并保存一个文件 **a.py**
- 文件内容如下：

```
#!/usr/bin/env python
import os
if __name__ == "__main__":

    currDir = os.getcwd()
    files = os.listdir( currDir )
    for f in files:
        print f
```

- 添加文件可执行权限 **chmod a+x a.py**
- 运行 **> a.py**

ssh

- ssh: Secure Shell 的缩写，需要安装
- ssh用来连接远程的linux系统，比如一个大型的运算服务器。
- 本地端称做client，远程端称作server

常用的ssh命令：

```
> ssh username@hostname
```

```
> ssh tom@166.111.145.89
```

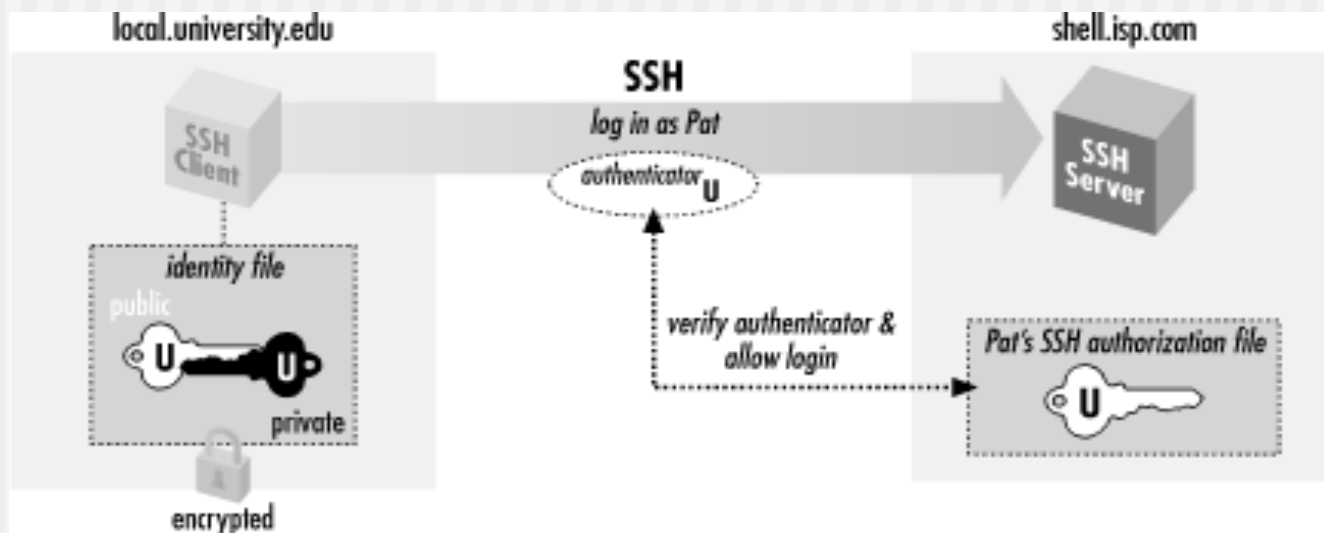
在Linux下：

```
> ssh -Y jerry@177.102.4.33    -Y 打开图形接收
```

还有重要的scp命令：（远程文件传输）

```
> scp scott@jin.tsinghua.ac:/home/scott/work ./
```

ssh登录的key认证机制



- 先使用 `ssh-keygen` (Linux命令) 生成一对 key, private key和public key
- public key必须事先通过可靠通道传递到服务器上
- 在登录验证时, 一组key必须符合

rsync

- 提供一种增量式的文件传输方案，实现文件同步，备份等功能。
- 它可以智能的选择新增文件，或改动文件进行传输，大大的减少了同步的负担
- 它还可以选择是全同模式（同步删除），或保留模式（不同步删除）

使用方法：

Pull: `rsync [OPTION...] [USER@]HOST:SRC... [DEST]`

Push: `rsync [OPTION...] SRC... [USER@]HOST:DEST`

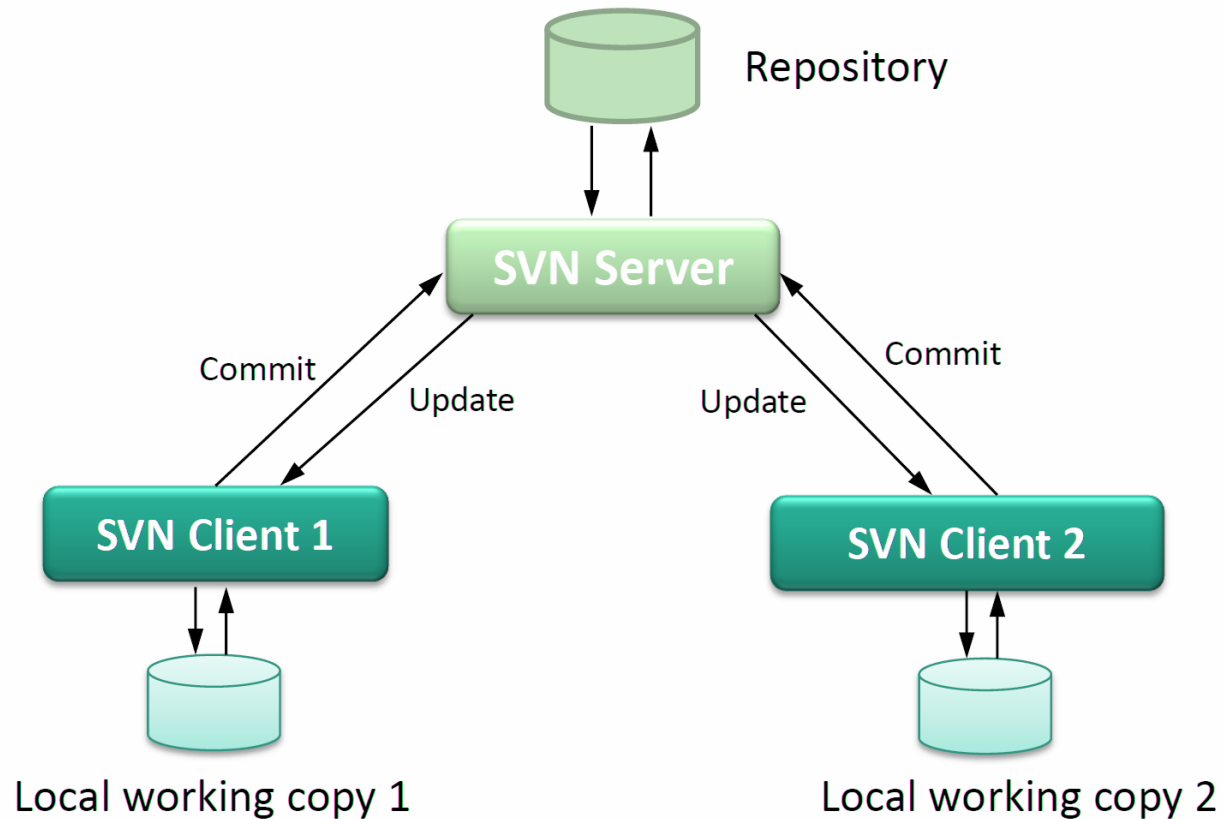
例如：

`rsync -avz Lee@foo:src/bar /data/tmp` （远程备份）

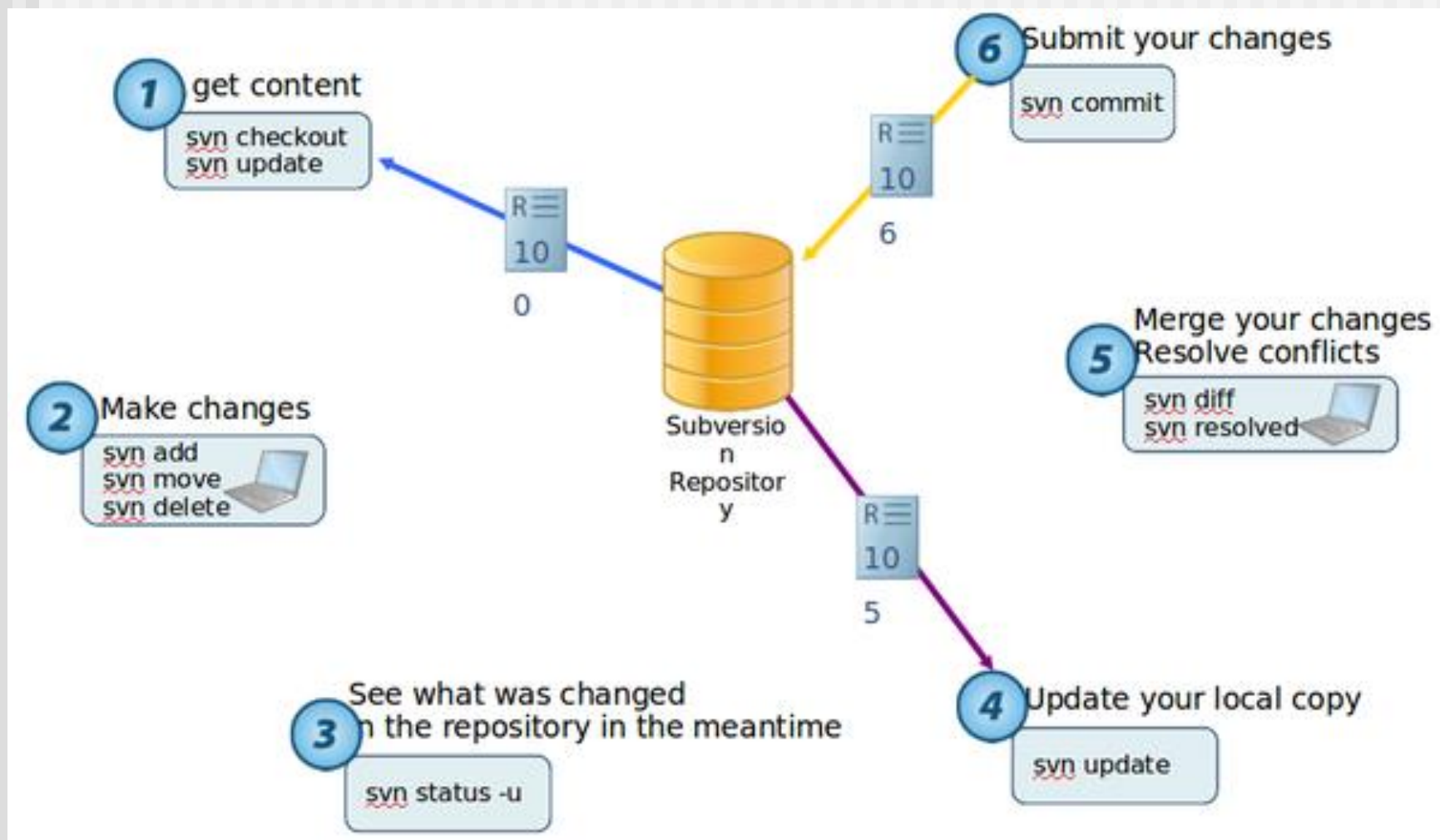
`rsync -avz /src/bar /data/tmp` （同机备份）

svn（全名subversion）

用于多人开发同一个软件（文档）项目时的工作协同



svn可以有效地避免工作冲突



websvn

websvn是独立于svn的附属产品，可以直接图示的看到svn中的历史进展

Path	Last modification	Log	RSS
<input type="checkbox"/> Analysis/	29 139d 20h wangzhe	Log	RSS
<input type="checkbox"/> LoopExample/	29 139d 20h wangzhe	Log	RSS
<input type="checkbox"/> TWin/	28 139d 20h wangzhe	Log	RSS
<input type="checkbox"/> Calibration/	6 177d 23h guozy	Log	RSS
<input type="checkbox"/> CommonLib/	94 50d 00h guozy	Log	RSS
<input type="checkbox"/> DataType/	80 71d 06h guozy	Log	RSS
<input type="checkbox"/> DetectorStructure/	90 62d 02h guozy	Log	RSS
<input type="checkbox"/> Document/	93 50d 09h wanly	Log	RSS
<input type="checkbox"/> EventDisplay/	77 79d 07h wanly	Log	RSS
<input type="checkbox"/> Generator/	77 79d 07h wanly	Log	RSS
<input type="checkbox"/> JPSim/	95 50d 00h guozy	Log	RSS
<input type="checkbox"/> Reconstruction/	77 79d 07h wanly	Log	RSS

Compare Paths

REV 36 → REV 37

Ignore whitespace

/CommonLib/CMakeLists.txt

46,7 → 46,7

```
#add_library(${PACKAGE} SHARED ${sources}  
G_${PACKAGE}.cxx)
```

```
• FILE(GLOB UTILSHEADER RELATIVE ${PROJECT_SOURCE_DIR}  
  ${PROJECT_SOURCE_DIR}/*.hh)  
• FILE(GLOB UTILSHEADER RELATIVE ${PROJECT_SOURCE_DIR}  
  ${PROJECT_SOURCE_DIR}/*.h*)
```

```
FOREACH(utilesheader ${UTILSHEADER})  
  #MESSAGE( ${utilesheader} )  
  configure_file(  

```

mysql - 数据库

连接一个远程数据库

> mysql -h db2.hep.ac.cn -u dayabay -p

查看都有哪些数据库

mysql> show databases;

使用其中的一个数据库

mysql> use aDataBase;

查看其中的表格

mysql> show tables;

退出


mysql> quit;

数据库中的table

Table structure

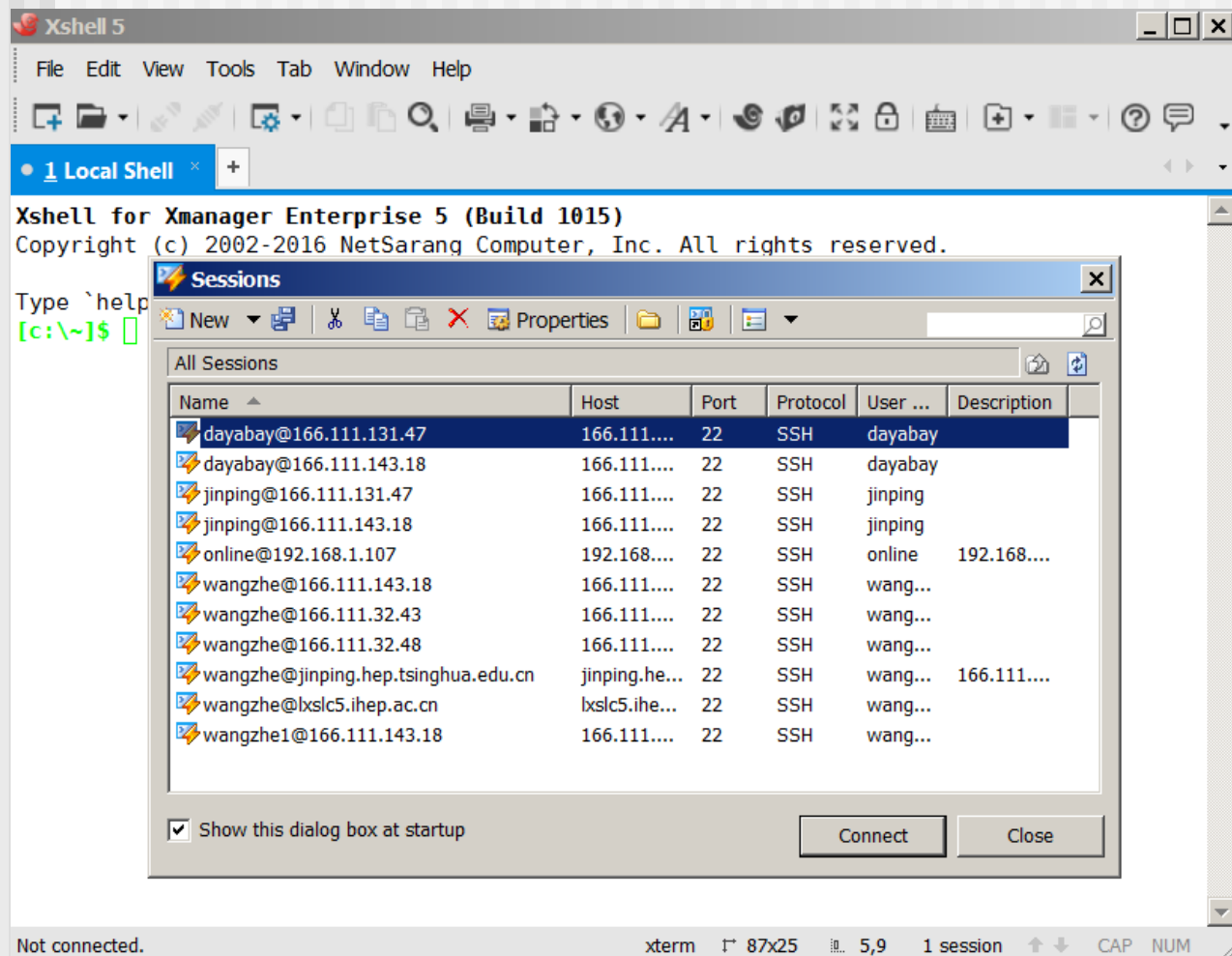
Relation view

按照这个table的数据库

#	Name	Type	Collation	ID	Name	CountryCode	District	Population
<input type="checkbox"/> 1	ID 	int(11)		1	Kabul	AFG	Kabol	1780000
<input type="checkbox"/> 2	Name	char(35)	latin1	2	Qandahar	AFG	Qandahar	237500
<input type="checkbox"/> 3	CountryCode	char(3)	latin1	3	Herat	AFG	Herat	186800
<input type="checkbox"/> 4	District	char(20)	latin1	4	Mazar-e-Sharif	AFG	Balkh	127800
<input type="checkbox"/> 5	Population	int(11)		5	Amsterdam	NLD	Noord-Holland	731200
				6	Rotterdam	NLD	Zuid-Holland	593321
				7	Haag	NLD	Zuid-Holland	440900
				8	Utrecht	NLD	Utrecht	234323
				9	Eindhoven	NLD	Noord-Brabant	201843
				10	Tilburg	NLD	Noord-Brabant	193238
				11	Groningen	NLD	Groningen	172701

XManager, Putty等

这些软件解决了如何从Windows登录到Linux

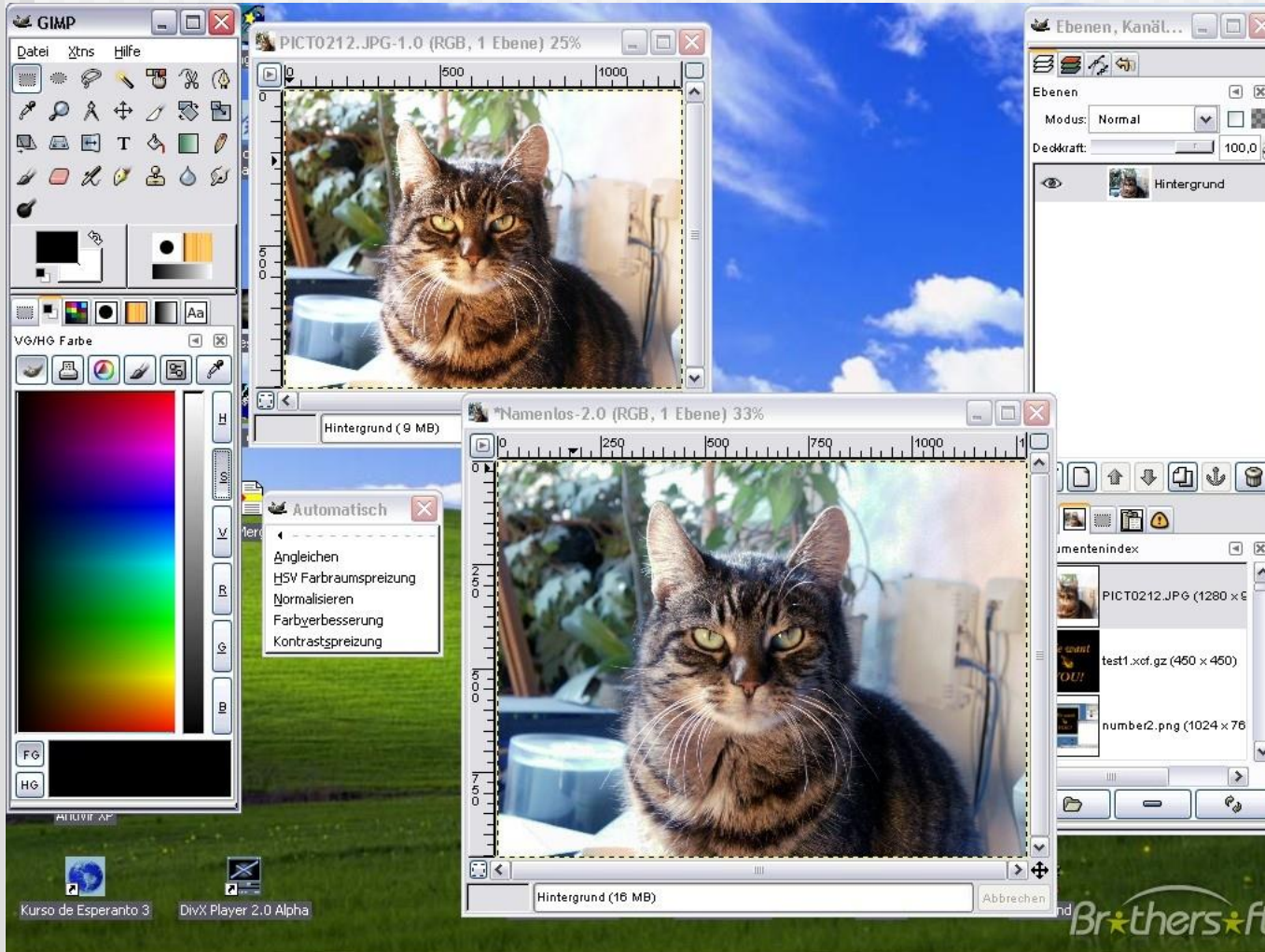


注意，利用一些软件可以实现远程登陆，但无法打开图形界面。

因为：图形界面是需要本地软件作为客户端配合的，有些登录软件没有这一功能

gimp

<https://www.gimp.org/>



可以说是
Linux下的
Photoshop

总结

- 介绍Linux操作系统以及常用命令
- Linux终端的常用编辑器(vi, emacs, pico, etc)
- 介绍shell、环境变量和脚本
- Python

目的： 了解Linux操作系统的基本概念
熟练掌握Linux操作系统的常用命令
了解Shell脚本编程的基本概念

参考资料

1. 鸟哥的Linux私房菜
2. Linux从入门到精通, Phil Cornes著; 童寿彬等译
3. RedHat Linux 9系统管理, 李蔚泽编著
4. 精通Shell编程, Sriranga Veeraraghavan著; 卢涛译
5. Linux命令、编辑器与Shell编程, Mark Sobell著, 杨明军等译

6. Bing and/or Baidu !!!

课上练习

科研中可能会遇到的小问题(程序相关):

- 1.在某文件夹下有很多文件和子文件夹，需要将所有的
.cpp文件中的Charge_int改成Charge_float
- 2.同样该文件夹，有很多文件中可能包含某个函数，比如Fit_Landau()，需要找到函数的定义及使用
- 3.查看某个文本文件a.txt一共有多少行
- 4.某程序，需要变更其中的参数进行多次运行。

解决方法

问题1可以用下面一行命令解决：

```
find . -name "*.cpp" -exec sed -i 's/Charge_int/Charge_float/g' {} \;
```

即用find命令在当前目录寻找(递归)所有的cpp文件，找到后执行sed命令，其中-name和-exec是find命令的参数，分别表示按文件名寻找和执行指令。sed指令在文件中寻找(s)字符串“Charge_int”，替换为“Charge_float”。find后面的“.”表示在当前目录寻找，也可以改成其它想寻找的目录，比如/home/analyzer/mywork

问题2可以用grep命令解决：

```
grep -srn "Fit_Landau" /home/analyzer/mywork
```

即用grep命令递归查找/home/analyzer/mywork目录里面的所有文件，打印出所有包含“Fit_Landau”字符串的文件名称，以及该字符串出现的行号和该行的内容。其中-srn是grep的参数，s表示忽略文件不存在或无法读取等错误信息，r表示在文件夹中递归查找，n表示打印出字符串出现的行号。这些参数可以组合使用。

解决方法

问题3可以用下面一行命令解决：

```
wc -l a.txt
```

即用wc命令，计算a.txt文件有多少行，其中-l参数表示计算行号。如果改成-w，则表示计算有多少word。

问题4可以用shell脚本快速解决，详见shell脚本编程。

这仅仅是几个简单的例子，Linux提供的这种指令不计其数。一般通过bing或者baidu都可以查到如何实现你需要的功能。

思考题：如果在linux下获得系统当前时间并截取时间中的月份？可以搜索试一下。

作业

1. 打开你的电脑，打开**virtual box**的**Linux**虚拟机（在桌面上）。
打开一个**terminal** (**Application->system tools**)
在当前目录建立一个新目录“**workarea**”
设置环境变量**WORKDIR**为“**workarea**”的绝对路径
然后转到“**workarea**”目录，创建目录**dir1**, **dir2**, **dir3**,
以及文件**file1**, **file2**, **index1.htm**, **index2.htm**, **test1.txt**
2. 检查环境变量，查看自己正在使用的**shell**, **echo \$SHELL**
3. 编写脚本**myscript1.sh**, 要求:
 - 1) 显示开始运行的时间;
 - 2) 在屏幕上打印出当前目录，当前用户名以及**SHELL**类型
 - 3) 显示**WORKDIR**的值
 - 4) 显示当前**\$WORKDIR**目录所用磁盘空间
 - 5) 间隔5秒钟之后再显示出当前时间

作业

4. 完成一个练习脚本，计数`workarea`下的目录数目和文件数目。
5. 利用`python`，列出当前目录中的目录名，不包含文件。

vi常用指令

vi 的常用技巧

- | | | | |
|--------------------|---------------------------------------|---------------------------------------------|----------------------|
| 1. 显示行号
:se nu | 13. 删除光标至行尾
D | 22. 将3-9行的" Abc"替换为"ABC"
:3,9s/Abc/ABC/g | 33. 全文加亮光标当前变量
gd |
| 2. 移动光标到第5行
:5 | 14. 用某字母(如"k")替换光标所在字符
r k | 23. 将3-6行复制到第9行
:3,6 co 9 | 34. 保存文件
:w |
| 3. 移动光标到行首
^ | 15. 向下新增一行
o | 24. 将3-6行移动到第9行
:3,6 m 9 | 35. 保存退出
:wq |
| 4. 移动光标到行尾
\$ | 16. 向上新增一行
O | 25. 删除3-6行
:3,6 d | 36. 不保存退出
:q! |
| 5. 移动光标到文件头
gg | 17. 复制光标所在行
yy | 26. 3-6行行首加上"ABC"
:3,6s/^/ABC/g | 37. 进入输入模式
i |
| 6. 移动光标到文件尾
G | 18. 将复制的行粘贴到光标所在行下方
p | 27. 3-6行行首加上"//", 即C++注释
:3,6s/^/\\/V/g | 38. 进入命令模式
ESC |
| 7. 向后移动3个字
3w | 19. 将复制的行粘贴到光标所在行上方
P | 28. 3-6行行尾添加"ABC"
:3,6s/\$/ABC/g | 39. 在行首进入输入模式
I |
| 8. 向前移动4个字
4b | 20. 查找字符串"Abc"
/Abc | 29. 将光标的下一行连接到光标所在行
J | 40. 在行尾进入输入模式
A |
| 9. 删除光标所在字
dw | 21. 全局替换"Abc"为" ABC"
:%s/Abc/ABC/g | 30. 将光标所在处字母变更大小写
~ | |
| 10. 删除光标所在字符
x | | 31. 取消操作(undo)
u | |
| 11. 删除行
dd | | 32. 重复操作(redo)
. | |
| 12. 删除光标后3行
3dd | | | |

2017/2/24

emacs常用指令

C = Control

M = Meta = Alt|Esc

1. C-x C-s save the file
2. C-x C-w write the text to an alternate name
3. C-z suspend emacs
4. C-X C-c close down emacs
5. C-A go to line begin
6. C-E go to line end
7. C-s Search forward
8. C-r search backward
9. C-_ undo
10. C-g go to line number
11. C-x r r copy rectangle to register
12. C-x r k kill rectangle
13. C-x r y yank rectangle