**Morning**

1. 脚本：一个**可以运行**的文件，可以实现某种功能。（必须给予文件执行权限）
2. Shell脚本的一般组成

#！环境声明（以下代码有哪个程序进行翻译，默认/bin/bash，Python的解释器为/usr/bin/python）

#注释文本

可执行代码

1. Shell简单技巧

-管道|

-重定向输出

>：只收集前面命令的正确输出信息，写入文本文件中

2>：只收集前面命令的错误输出信息，写入文本文件中

&>：收集前面命令的正确与错误输出信息，写入文本文件中

-黑洞设备（/dev/null）：专用于收集无用的信息

cat /etc/passwd &> /dev/null

1. 变量的概念

-作用：为了增加脚本的灵活度，适用多变的环境

-变量：以不变的名称，存储可以变化的值

-注意事项：

1）若指定变量已存在，相当于给变量赋值

2）等号两边不能有空格

3）只能由字母/数字/下划线开头，区分大小写

4）不能以数字开头，不要使用关键字和特殊字符

tmp=$user #将user变量的当前值赋给tmp变量

-基本格式：

-引用变量值：$变量名

-查看变量值：echo $变量名、echo ${变量名}

1. 捕捉用户输入

- read

1. 产生交互
2. 记录用户再键盘上的输入
3. 将记录的信息赋值给变量

例：read -p '请输入您要创建的用户' name

**Afternoon**

1、变量的种类：

1）环境变量：一般都大写，由系统定义完成，用户直接使用

-$USER：永远储存当前系统登陆的用户名

-$PATH：储存命令程序的路径（通过echo $PATH查看）

PATH=/usr/local/bin:/usr/local/sbin #修改PATH变量

2）位置变量：由系统定义完成，用户直接使用，做到非交互式

-在执行脚本时，提供命令行参数

-表示为$n，n为数字。如$1,$2...${11},${12}

1. 预定义变量：由系统定义完成，用户直接使用

-用来保存脚本程序的执行信息

-直接使用这些变量，不能直接为这些变量赋值

$# 已加载的位置变量的个数（计算用户输入变量的个数）

$\* 所有位置变量的值

$? 上一次程序退出后的状态（返回值），0表示正常，其他值异常。

1. 自定义变量：用户自主设置、修改及使用

# 单引号’ ‘：取消所有特殊字符的含义

1. 常用的检测选项
2. 检查文档状态（0为真）

-e：文档存在为真 -d：文档存在，且为目录为真

-f：文档存在，且为文件为真 -r：文档存在，且具备读取权限为真

-w：文档存在，且具备写入权限为真

-x：文档存在，且具备执行权限为真

例：[ -e /etc ] ； echo $? 输出0

1. 比较整数大小

-gt：大于 -ge：大于等于 -eq：等于 -ne：不等于

-lt：小于 -le：小于等于

1. 字符串比对

==：两个字符串相等为真 !=：不等为真

1. If多分支处理

if [ 条件测试 ];then

命令序列xx

elif [ 条件测试 ];then

命令序列yy

else

命令序列zz

fi

1. for循环处理

for 变量名 in 值列表

do

命令序列

done

-造数工具：{起始数值..结束数值}

echo ‘ ‘ >&2：将echo变为错误输出

exit n（n为非0值）：将上一条命令的返回值设为n

$()或反撇号` `：将命令的输出，作为另外一个命令的参数

例：echo `date +%F` 或者 echo $(date +%F)

#输出2018-09-12

mkdir nsd-$(date +%F) #创建目录nsd-2018-09-12

**作业**

案例4：编写一个判断脚本

在server0上创建/root/foo.sh脚本

1. 当运行/root/foo.sh redhat，输出为fedora
2. 当运行/root/foo.sh fedora，输出为redhat
3. 当没有任何参数或者参数不是redhat或者fedora时，其错误输出产生以下信息：/root/foo.sh redhat|fedora

#!/bin/bash

if [ $# -eq 0 ];then

echo '/root/foo.sh redhat|fedora' >&2

exit 1

elif [ $1 == redhat ];then  
 echo fedora

elif [ $1 == fedora ];then

echo redhat

else

echo '/root/foo.sh redhat|fedora' >&2 #将echo变成错误输出

exit 2

fi

案例5：编写一个批量添加用户脚本

在server0上创建/root/batchusers脚本

1. 此脚本要求提供用户名列表文件作为参数
2. 如果没有提供参数，此脚本应该给出提示Usage:/root/batchusers，退出并返回相应值
3. 如果提供一个不存在的文件，此脚本应该给出提示Input file not found，退出并返回相应值
4. 新用户的登陆Shell为/bin/false，无需设置密码
5. 用户列表测试文件：<http://classroom/pub/materials/userlist>

#!/bin/bash

if [ $# -eq 0 ];then

echo 'Usage:/root/batchusers' >&2

exit 1

elif [ ! -f $1 ];then

#文件不存在时为真

echo 'Input file not found' >&2

exit 2

else

for n in $(cat $1)

do

useradd -s /bin/false $n &> /dev/null

if [ $? -eq 0 ] ;then

echo 已创建$n

else

echo $n用户已存在

fi

done

fi