GNU/Linux Shell 编程

GNU/Linux
Shell 编程

Shell 脚本作用

shell 脚本可以完成工作中日常的工作内容及可以自动的得到想要的结果

GNU/Linux
Shell 编程

Shell 变量

read 从键盘上读取输入的内容传递给某个变量

```
read input
read  -s  -n 1
```

readonly

将一个用户定义的 shell 变量标识为不可变

命令格式 : readonly 变量名

只执行 readonly 会显示所有不可变的 shell 变量

GNU/Linux
Shell 编程

Shell 退出

只执行 readonly 会显示所有不可变的 shell 变量

wait

shell 等待后台启动的所有子进程结束 ,wait 的返回值永远为真

exit

退出程序并返回一个值 0 或非 0 。 0 表示正常退出 , 非 0 则表示 , 非正常退出

GNU/Linux
Shell 编程

Shell 脚本

Shell 再有些语句中需要确定值的真假。可以用下面的值作为真或假

true	无条件返回 0
false	无条件返回非 0

GNU/Linux Shell 编程

判断语句可以使用下面的控制语句进行

(1) 数值测试

-eq　　等于则为真

-ne　　不等于则为真

-gt　　大于则为真

-ge　　大于等于则为真

-lt　　小于则为真

-le　　小于等于则为真

GNU/Linux Shell 编程

判断语句可以使用下面的控制语句进行

(2) 字串测试

=	等于则为真
!=	不等于则为真
-z字串	字串长度伪为真
-n 字串	字串长度不伪为真

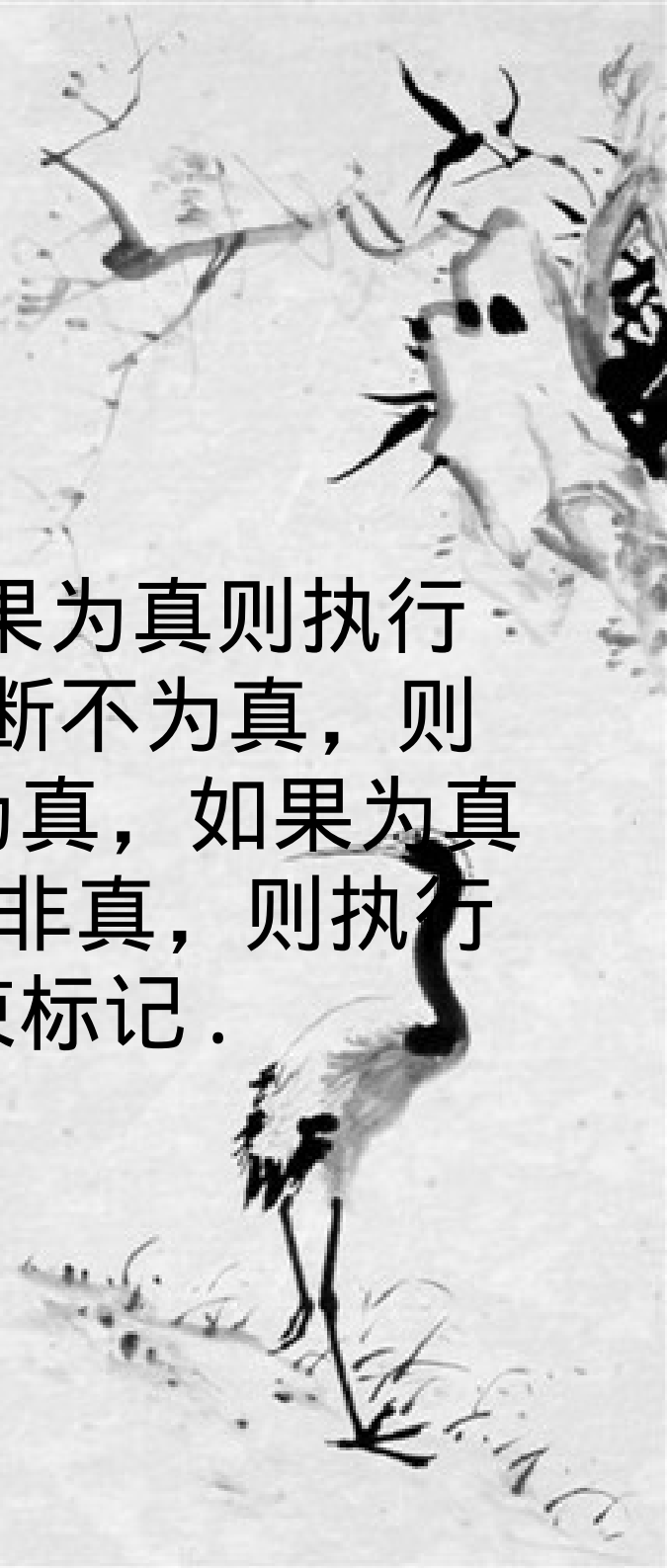
GNU/Linux Shell 编程

判断语句可以使用下面的控制语句进行

(3) 文档测试

-e	文件名	如果文档存在为真
-r	文件名	如果文档存在而且可读为真
-w	文件名	如果文档存在而且可写为真
-d	目录	如果目录存在为真
-x	文件名	如果文档存在且可执行为真
-b	文件名	如果文档存在且为块设备
-c	文件名	如果文档存在且为字符设备
-s	文件名	如果文档存在且大于零字节

GNU/Linux Shell 编程



判断语句

if 语句

if 判断所给予的值是否为真，如果为真则执行 then 的命令，如果 if 值在第一个判断不为真，则进入 elif 的第二个判断，确认是否为真，如果为真则执行 then 下的命令。如果结果为非真，则执行 else 后的命令，以 fi 是 if 判断的结束标记 .

GNU/Linux Shell 编程

格式：

```
if [ 判断 1 是否为真 ]
then
 command ←- 为真则执行此命令
elif [ 判断 2 是否为真 ]
 then
 command
elif [ 判断 3 是否为真 ]
 then
 command
else
 command
fi
```

GNU/Linux
Shell 编程

shell 脚本的编写与执行

1. 合手的 vi 或者 emacs 这样的文本编辑器

第一行以 #!/bin/bash 开始

即指定用 bash 解释下面的语句

GNU/Linux
Shell 编程

shell 脚本的编写

2 . 运行脚本有 3 种方法：

1. bash 脚本名　　脚本需要有可读权限
2. bash < 脚本名　　脚本需要有可读权限
3. ./path/ 脚本名　　脚本需要有 , 执行权限 (+x)

GNU/Linux Shell 编程

shell 脚本的编写

3 . 调试 shell 脚本：

格式为　bash　- 选项　　脚本名

常用的选项有

-e	如果一个命令失败就立即退出
-n	读入命令，但不执行命令
-u	置换时把未设置的变量看做出错
-v	执行过程中显示内容
-x	执行命令是把命令的参数显示出来

GNU/Linux Shell 编程

判断语句

case 语句

case 判断所给予的值是否为真，如果等于值 1 则执行值 1 的命令，如果等于值 2 则执行值 2 的命令，否则则执行没有值为真的命令

GNU/Linux Shell 编程

格式：

```
case   变量    in
匹对 1)
        Commands
        ;;
匹对 2)
        Commands;;
匹对 3)
        Commands;;
*)
        Commands;;
esac
```

GNU/Linux Shell 编程

循环语句 :for 语句

格式 :

```
for  变量   in 关键字
     do
          commands
done
```

在这个 for 语句中 , 在变量中符合关键字的候 , for 循环直至值全部循环完成才退出循环。 for 每次循环都将执行一次 commands, 直接值结束。有多少个值即执行多少次 commands 。其中 commands 代表的一组命令。

GNU/Linux Shell 编程

循环语句 :while 语句

格式 :

```
    while [  值是否为真  ]
do
  command
done
```

while 循环 :

当值为真时 , 则执行 command 命令一直循环，直至值为假退出 ;

GNU/Linux
Shell 编程

Shell 脚本

对于循环可以用下面的语句进行进行控制

break　　用于立即终止当前循环

continue　用于不执行当前循环而跳到下一个循环中

GNU/Linux Shell 编程

循环语句 :until 语句

格式 :

```
    until [  值是否为真  ]
do
  command
done
```

until 循环 :

当值为假时 , 则执行 command 命令一直循环，直至值为真退出 ;

GNU/Linux Shell 编程

循环语句 :select 语句

格式 :

```
    select    变量  in  值
do
    command 1
    command n
done
```

Select： 将值中的每一项做成类似表单，以交互的方式执行 do 和 done 之间的 command 1... ... 一直到 command n 一般会把 case 套嵌在 select 中使用 .

GNU/Linux Shell 编程

函数：在 shell 中这个函数实际上也是由若干条 shell 命令组成的，因此与 shell 程序形式上基本相似，不同的是它不是一个单独的进程，而是 shell 程序的一部分 .

function( )

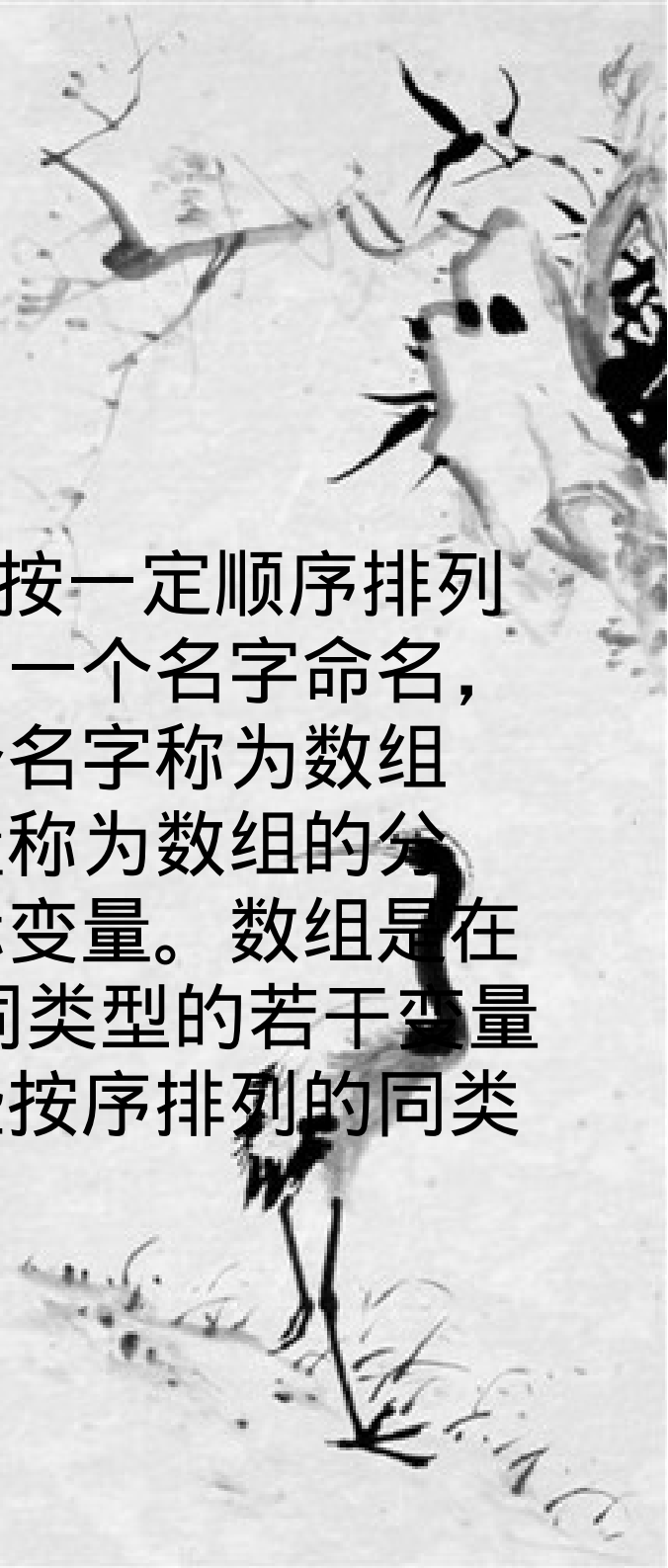
{command 1

command n}

函数调用时不带 ( ) 格式：函数名 参数 1 参数 2

函数中的变量均为全局变量，不存在局部变量，函数中的参数，调用函数时可以传递参数，参数在函数中用 $1,$2,$3 来引用

GNU/Linux Shell 编程



数组

定义

所谓数组，就是相同数据类型的元素按一定顺序排列的集合，就是把有限个类型相同的变量用一个名字命名，然后用编号区分他们的变量的集合，这个名字称为数组名，编号称为下标。组成数组的各个变量称为数组的分量，也称为数组的元素，有时也称为下标变量。数组是在程序设计中，为了处理方便， 把具有相同类型的若干变量按有序的形式组织起来的一种形式。这些按序排列的同类数据元素的集合称为数组。

GNU/Linux Shell 编程

数组使用

可以整体定义数组：

```
ARRAY_NAME=(value0 value1 value2 value3 ...)
```

或者：

```
ARRAY_NAME=(
value0
value1
value2
value3
...
)
```

此时数组的下标默认是从 0 开始的

GNU/Linux Shell 编程

数组使用

还可以单独定义数组的各个分量：
ARRAY_NAME[0]=value0

ARRAY_NAME[1]=value1
ARRAY_NAME[n]=valuen
...
可以不使用连续的下标，而且下标的范围没有限制。

GNU/Linux Shell 编程

expr

是对整型变量进行算术运算的命令

expr 3 + 5

expr 8 / 4

注意格式 + 号两边必需是 space 或 tab 。

+ : 加 *: 乘 %: 取模 -: 减 /: 除

GNU/Linux Shell 编程

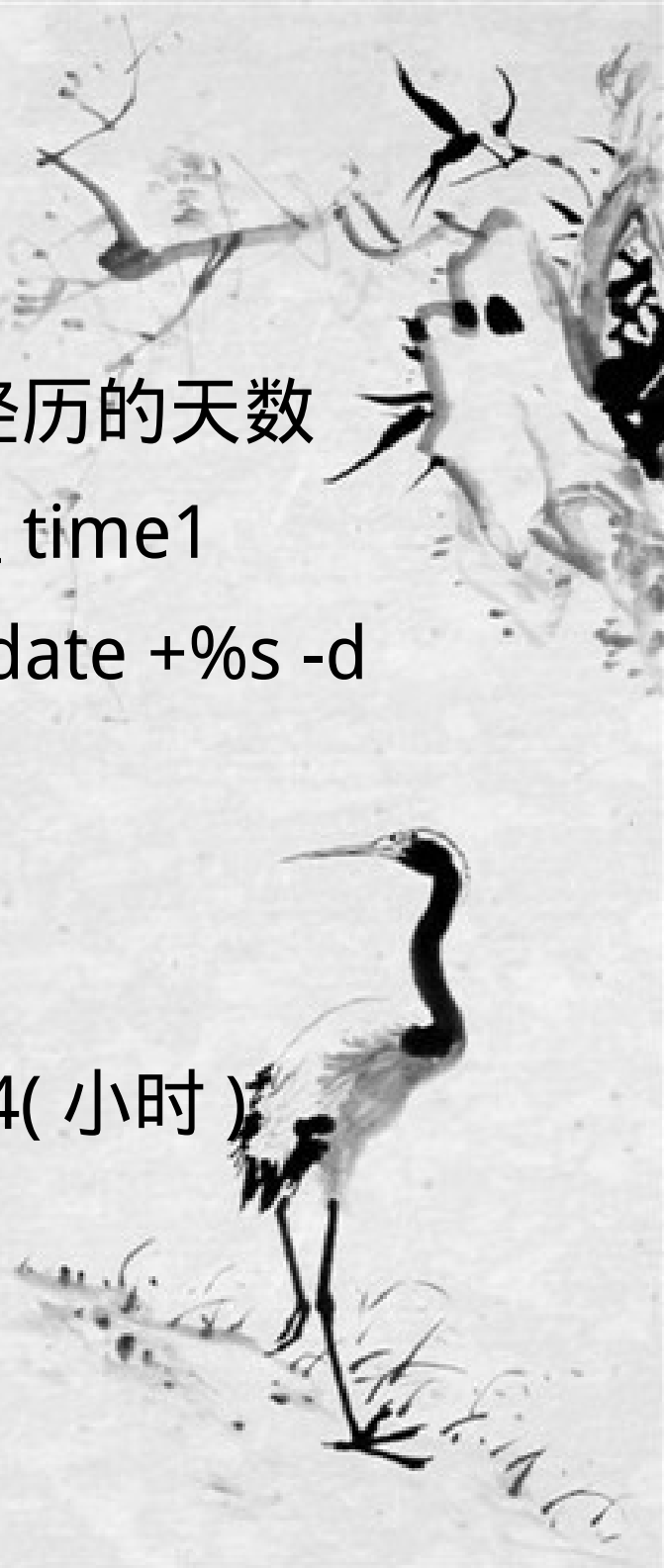
计算从 1945 年 8 月 15 日到当前时间所经历的天数

1. 计算时间所经历的时间，并赋值给变量 time1

#time1=$(($(date +%s -d '2014-9-27')-$(date +%s -d '1945-08-15')))

2. 计算经历的天数

#expr $time1 / 60(秒) / 60(分钟) / 24(小时)



GNU/Linux Shell 编程

计算从 1945 年 8 月 15 日 7:00 到当前时间 20:00 所经历的分钟数

2. 计算时间所经历的时间 , 并赋值给变量 time1

time1=$(($(date +%s -d '2014-9-27 20:00')-$(date +%s -d '1945-08-15 07:00')))

3. 计算经历的分钟数

#expr $time1 / 60(秒)

GNU/Linux Shell 编程

命令 :sleep

功能 :Linux 系统延迟指定时间

说明 :sleep 常用于 Linux Shell 脚本中控制延迟时间

格式 :sleep <n> [s] [m] [h] [d]

GNU/Linux Shell 编程

参数

n: 支持小数，代表时间

s: 秒

m: 分钟

h: 小时

d: 天数

示例 :

#sleep 1.5m