



中南大学

CENTRAL SOUTH UNIVERSITY

Shell 程序设计 实验报告

学生姓名	王雅蓉
学 号	8206200602
专业班级	计科 2003
指导教师	沈海澜
学 院	计算机学院
完成时间	2023 年 04 月 04 日

计算机学院

2023 年 04 月

目录

- 一 实验概述3
 - (一) 实验目的 3
 - (二) 实验内容及要求..... 3
 - (1) 实验内容 3
 - (2) 实验要求..... 5
- 二 实验设计5
 - (一) 运行并调试程序..... 5
 - (1) fortest 及其调试 5
 - (2) fortest1 及其调试..... 5
 - (3) functest 及其调试 6
 - (4) paramtest 及其调试 7
 - (5) untiltest 及其调试 7
 - (6) whilettest 及其调试..... 7
 - (二) 脚本设计 1 7
 - (三) 脚本设计 2..... 8
- 三 代码及运行结果.....9
 - (一) 调试后的代码及运行结果 9
- 四 总结与收获.....13

一 实验概述

（一）实验目的

- (1) 掌握 Linux Shell 程序基本语法；
- (2) 掌握 Linux shell 程序编辑、运行、调试方法；
- (3) 掌握 Linux 环境变量及函数的使用；
- (4) 编写 Linux shell 程序；

（二）实验内容及要求

（1）实验内容

- 1、运行并调试以下程序，给出运行结果。

```
1 fortest
#!/bin/bash
for a in x y z
do
echo now a=$a
done

2 fortest1
#!/bin/bash
for a
do
echo now a=$a
done

3 functest
#!/bin/bash
setup()
{
echo setup
}
do_date()
{
date
}
chgdir()
{
```

```

cd $1
}
do_date
setup
chgdir

4 paramtest
#!/bin/bash
echo filename:$0
echo arguments:$*
echo number arg:$#
echo arg2:$2
shift
echo number arg:$#
echo arg2:$2
set hello,everone
echo args:$*
echo arg2:$2

5 untiltest
#!/bin/bash
ans=yes
until [ "$ans" = no ]
do
    echo Enter a name
    read name
    echo $name >> file.names
    echo "Continue?"
    echo Enter yes or no
    read ans
Done

6 whilettest
while test -r abc.txt
do
    echo "file abc.txt has not beed deleted !"
    sleep 10
done
    echo "file abc.txt has beed deleted !"

```

2、编写一个 shell 脚本，完成功能：

设计一个 shell 程序，添加一个新用户组为 class，然后添加属于这个组的 30 个用户，用户名的形式为 stdxx, 其中 xx 从 01 到 30。

3、编写一段 Shell 程序完成：根据从键盘输入的学生学号，到档案表文件中查找对应的学生信息，并进行显示。档案表的定义如下：

学号：姓名

(2) 实验要求

- (1) 掌握 Shell 程序的编辑、运行、调试方法
- (2) 完成实验内容要求实现的任务，并记录实验过程
- (3) 撰写实验报告，包括实验目的、实验内容、实验要求、实验代码及运行截图、实验心得等。
- (4) 从前面的 8 个题目中任选一道进行上机讲解及运行演示，提交讲解视频。

二 实验设计

(一) 运行并调试程序

(1) fortest 及其调试

该程序主要运用了 for.....do...done 结构，变量 a 被依次赋予 x、y、z，因此输出的 a 的引用也依次是 x、y、z。

(2) fortest1 及其调试

该程序同样运用了 for.....do...done 结构，但 for 循环后没有跟范围，所以当没有参数输入时，此时 a 不会被赋予任何值，故没有输出；当有参数输入时，此时的程序可等同于如下程序，会依次将传入参数赋值给 a 并进行输出。

```
#!/bin/bash
for a in $*
do
    echo now a=$a
done
```

(3) functest 及其调试

该程序主要使用了 `functionname { 若干命令行 }` 格式定义了 3 个 Shell 函数，并且 `chgdir()` 函数需要一个参数传入，所以如果想成功执行 `chgdir` 函数中的 `cd` 命令，需要向该函数传参。故需要对原程序作出如下修改：

```
#!/bin/bash
#!/bin/bash
setup()
{
echo setup
}
do_date()
{
date
}
chgdir()
{
cd $1
}
do_date
setup
chgdir $1
```

此时，执行脚本时传入的第一个参数会传给 `chgdir` 函数作为其第一个参数。但当修改结束后会发现此时采用 `sh` 执行该文件仍然不能正确执行 `cd` 命令，经过搜索学习后得知原因如下：

- 在 `shell` 在执行脚本的时候，会创建一个子 `shell`，并在子 `shell` 中逐条执行脚本中的指令。
- 子 `shell` 会从父 `shell` 中继承了环境变量，但是执行后不会改变父 `shell` 的环境变量。
- 在子 `shell` 中的操作和环境变量不会影响父进程，在执行完 `shell` 后又回到了父进程。

解决方法：可以采取 `source` 命令执行脚本，`source` 命令可以在当前的 `shell` 环境下执行脚本，不会创建子 `shell`，直接影响父进程。

(4) paramtest 及其调试

该程序主要实现了在函数内部对参数的不同读取形式，如下表所示：

读取形式	读取内容
\$0	执行的文件名（包含路径）
\$*	以一个单字符串显示所有向脚本传递的参数。
\$#	传递到脚本的参数个数
\$2	执行脚本的第二个参数

shift 命令作用：将当前所有参数值提前一位（如\$1 变量被销毁，\$2 替代\$1），\$0 不变，参数总个数减少一位。所以在本程序中，当执行了 shift 命令后，\$#的值会减少 1，\$2 会变为原先的\$3，\$*中也会去掉原先的\$1。

(5) untiltest 及其调试

本程序主要实现了对 until 结构的使用，当 ans 变量为“yes”时，以追加形式输入变量 name 到 file.names，到用户输入“no”，则下一轮循环不会被开启。此程序中有一处错误导致无法正确运行：方括号包围，两括号两侧、内侧以及等号两侧没有正确使用空格，修改后程序可正确运行。

(6) whilettest 及其调试

本程序主要实现了对 while...do...done 结构的使用，结合 test 命令，每 10 秒检查 abc.txt 是否仍然在当前路径，如果在输出“file abc.txt has not beed deleted !”，如果该文件已经删除则输出“file abc.txt has beed deleted !”并退出循环。

（二）脚本设计 1

功能需求：设计一个 shell 程序，添加一个新用户组为 class，然后添加属于这个组的 30 个用户，用户名的形式为 stdxx,其中 xx 从 01 到 30.

- 添加新用户组 class：此设计在脚本中可添加如下判断语句。查看当前是否有

名为 class 的用户组，如果有则 grep 命令返回 0，没有则返回非零。当 grep 命令返回非 0 的时候，使用 groupadd 命令添加用户组。

```
#用户组不存在则创建，存在则添加用户
```

```
grep '^class' /etc/group
```

```
if [ $? -ne 0 ]
```

```
then
```

```
    groupadd $group;
```

```
fi
```

- 添加 30 个用户：添加用户采用 useradd 命令
- 指定用户名为 stdXX，其中 XX 从 01 到 30：采取 for 循环，每次循环调用 useradd 命令，设定用户组为 class，用户名为 \$user\$i。同时为了保证 01-09 的输出第一位为 0，在 for 循环时采用 seq 命令，并且设置 -w 选项，这样就可以保证 01-09 输出宽度也为 2。

（三）脚本设计 2

功能需求：3、编写一段 Shell 程序完成：根据从键盘输入的学生学号，到档案表文件中查找对应的学生信息，并进行显示。档案表的定义如下：

学号：姓名

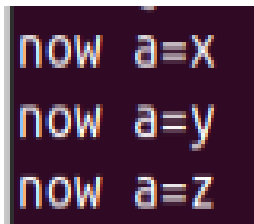
- 读取键盘输入的学生学号：read stu_num 命令
- 查找对应学生：使用 grep 命令，由于学生信息是一行一行存储的，故只需要找到以 stu_num 为首的行即可。
- 同时使用 if...then...else...fi 结构，当存在这样的学生时则输出对应行，不存在时则输出 “student not exists”。

三 代码及运行结果

（一）调试后的代码及运行结果

(1) fortest

```
#!/bin/bash
for a in x y z
do
echo now a=$a
done
```

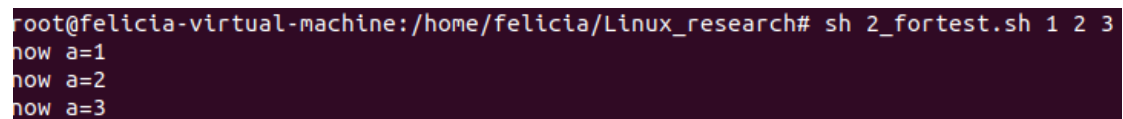


```
now a=x
now a=y
now a=z
```

(2) fortest1

```
#!/bin/bash
for a
do
echo now a=$a
done
```

依次输入 1 2 3 后结果如下：



```
root@felicia-virtual-machine:/home/felicia/Linux_research# sh 2_fortest.sh 1 2 3
now a=1
now a=2
now a=3
```

(3) funtest

```
#!/bin/bash
setup()
{
echo setup
}
do_date()
{
date
}
chgdir()
{
cd $1
```

```
}  
do_date  
setup  
chkdir $1
```

输入 “. ./3_fortest.sh insider” 后结果如下：

```
root@felicia-virtual-machine:/home/felicia/Linux_research# . ./3_fortest.sh insider  
+08 2023 оны Гуравдугаар сар 31, Ба 21:38:12  
setup  
root@felicia-virtual-machine:/home/felicia/Linux_research/insider#
```

(4) paramtest

```
#!/bin/bash  
echo filename:$0  
echo arguments:$*  
echo number arg:$#  
echo arg2:$2  
shift  
echo number arg:$#  
echo arg2:$2  
set hello,everone  
echo args:$*  
echo arg2:$2
```

依次输入 1 2 后结果如下：

```
root@felicia-virtual-machine:/home/felicia/Linux_research# sh 4_fortest.sh 1 2  
filename:4_fortest.sh  
arguments:1 2  
number arg:2  
arg2:2  
number arg:1  
arg2:  
args:hello,everone  
arg2:
```

(5) untiltest

```
#!/bin/bash  
ans=yes  
until [ "$ans" = no ]  
do  
    echo Enter a name  
    read name  
    echo $name >> file.names  
    echo "Continue?"  
    echo Enter yes or no  
    read ans
```

Done

依次输入“yes”、“no”后输出结果如下：

```
root@felicia-virtual-machine:/home/felicia/Linux_research# sh 5_foritest.sh
Enter a name
yes
Continue?
Enter yes or no
no
```

(6) whiletest

```
while test -r abc.txt
do
    echo "file abc.txt has not beed deleted !"
    sleep 10
done
echo "file abc.txt has beed deleted !"
```

当初始时没有建立 abc.txt 时，输出如下：

```
root@felicia-virtual-machine:/home/felicia/Linux_research# sh 6_foritest.sh
file abc.txt has not beed deleted !
```

当初始时建立了 abc.txt，并且 15 秒后删除了该文件，输出如下：

```
root@felicia-virtual-machine:/home/felicia/Linux_research# sh 6_foritest.sh
file abc.txt has not beed deleted !
file abc.txt has not beed deleted !
file abc.txt has beed deleted !
```

2、编写脚本完成功能

```
user=std
group=class
#用户组不存在则创建，存在则添加用户
grep '^class' /etc/group
if [ $? -ne 0 ]
then
    groupadd $group;
fi
#创建 30 个用户 -w 为默认补位操作
for i in `seq -w 1 30`
do
    useradd -g $group $user$i
    echo "successfully add $user$i"
```

运行该程序后，输出结果如下：

```
successfully add std01
successfully add std02
successfully add std03
successfully add std04
successfully add std05
successfully add std06
successfully add std07
successfully add std08
successfully add std09
successfully add std10
successfully add std11
successfully add std12
successfully add std13
successfully add std14
successfully add std15
successfully add std16
successfully add std17
successfully add std18
successfully add std19
```

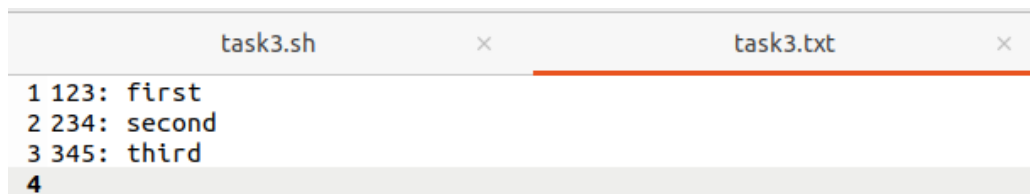
此时若重复执行该程序，输出结果如下：

```
root@felicia-virtual-machine:/home/felicia/Linux_research# sh task2.sh
class:x:1001:
useradd: 用户“std01”已存在
successfully add std01
useradd: 用户“std02”已存在
successfully add std02
useradd: 用户“std03”已存在
successfully add std03
useradd: 用户“std04”已存在
successfully add std04
useradd: 用户“std05”已存在
successfully add std05
useradd: 用户“std06”已存在
successfully add std06
useradd: 用户“std07”已存在
successfully add std07
useradd: 用户“std08”已存在
successfully add std08
useradd: 用户“std09”已存在
successfully add std09
useradd: 用户“std10”已存在
```

3、编写脚本完成功能

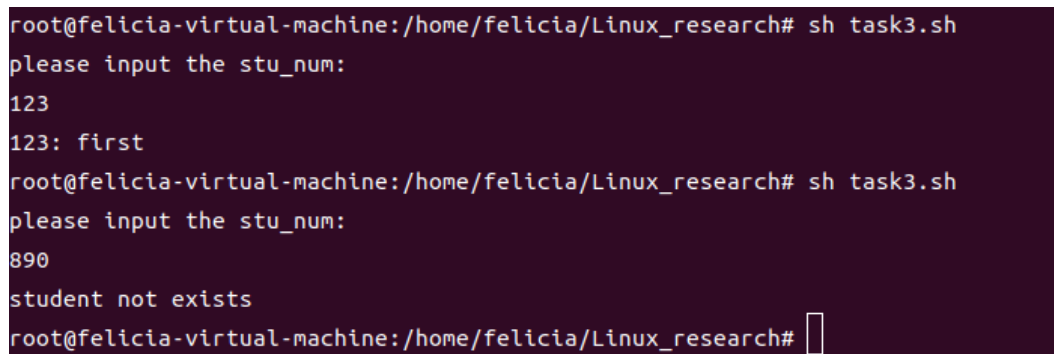
```
echo "please input the stu_num:"
read stu_num
grep "^$stu_num" task3.txt >& /dev/null
if [ $? -eq 0 ]
then
grep "^$stu_num: " task3.txt
else
echo "student not exists"
fi
```

Task3.txt 文件内容如下：



```
task3.sh × task3.txt ×
1 123: first
2 234: second
3 345: third
4
```

分别输入 123 及 890，得到的结果依次如下：



```
root@felicia-virtual-machine:/home/felicia/Linux_research# sh task3.sh
please input the stu_num:
123
123: first
root@felicia-virtual-machine:/home/felicia/Linux_research# sh task3.sh
please input the stu_num:
890
student not exists
root@felicia-virtual-machine:/home/felicia/Linux_research#
```

四 总结与收获

本次 shell 程序设计实验总结并应用到了过去几周的知识点，在本次实验前我对 shell 命令的掌握基本熟练，但还有许多的细节部分仍然需要加强学习。本次实验刚好弥补了我在过去 shell 程序设计中的许多不足，实验中很多需要调试的地方，也是我自身经常犯错的地方。

本次实验中令我学习到最多的就是第三个程序，在此之前我知识学会了运用 shell 脚本传参，但此程序的调试让我真正明白了参数的传递的来龙去脉。

对我来说还是一个新知识的还有第三个程序中的 cd 命令的执行，在此之前我并没有深入学习过 cd 命令。也是在学习了为什么使用 sh 执行脚本时，脚本中的 cd 指令并不能改变系统的当前路径，经过搜索才知道其中的原因。我感受到了 shell 脚本的强大以及细节之多。

在未来的 Linux 系统与应用的学习中，我会更加深入学习课堂上没有深入讲解的内容，更注重自身知识的深度与厚度积累，积极探索，认真学习，在今后的实验中精益求精的要求自己。