

# 南昌大学实验报告

姓名： 谢志彬 学号： 6103115112

邮箱地址： [siliconx@163.com](mailto:siliconx@163.com)

专业班级： 计算机科学与技术153

实验日期： 2018/04/09

课程名称： Linux程序设计实验

## 实验项目名称

### Bash Programming

## 实验目的

- 1.编写shell脚本解决实际问题
- 2.通过系统调用实现一些Linux的实用工具

## 实验基础

### shell脚本的使用

## 实验步骤

### 目录结构

```
siliconx@Lenovo:~/code/linux/shell$ ls
1.sh 2.sh 3.sh 4.sh 5.sh 6.sh 7.sh poem1.txt poem2.txt test words.txt
```

### T1

```
#!/bin/sh

if [ $# != 3 ]; then
    echo "USAGE: $0 FILENAME FROM(int) TO(int)"
    exit 1
fi

echo "FILENAE: $1"
echo "FROM: $2"
echo "TO: $3"

echo "===== target text ====="
cat $1 | head -n $3 | tail -n +$2
```

```
echo "\n===== done ====="
```

```
siliconx@Lenovo:~/code/linux/shell$ ./1.sh
USAGE: ./1.sh FILENAME FROM(int) TO(int)
siliconx@Lenovo:~/code/linux/shell$ ./1.sh poem1.txt 1 4
FILENAME: poem1.txt
FROM: 1
TO: 4
===== target text =====
A reentrant function,
if interrupted,
will return a result,
which is not perturbed.

===== done =====
siliconx@Lenovo:~/code/linux/shell$ |
```

## T2

```
#!/bin/sh

if [ $# != 2 ]; then
    echo "ARGS ERROR"
    echo "USAGE: $0 FILENAME TARGET_WORD"
    exit 1
fi

echo "FILENAME: $1"
echo "TARGET WORD: $2"

echo "===== original text ====="
more $1

echo "===== new text ====="
sed -e "/$2/d" $1 > $1.witout-$2
more $1.witout-$2
```

```
siliconx@Lenovo:~/code/linux/shell$ ./2.sh poem1.txt if
FILENAME: poem1.txt
TARGET WORD: if
===== original text =====
A reentrant function,
if interrupted,
will return a result,
which is not perturbed.
int global_int;int is_not_reentrant(int x) { int x = x; return global_int + x
,
depends on a global variable,
which may change during execution.
int global_int;int is_reentrant(int x) { int saved = global_int; return saved
x; },
mitigates external dependency,
it is reentrant, though not thread safe.
===== new text =====
A reentrant function,
will return a result,
which is not perturbed.
int global_int;int is_not_reentrant(int x) { int x = x; return global_int + x
,
depends on a global variable,
which may change during execution.
int global_int;int is_reentrant(int x) { int saved = global_int; return saved
x; },
mitigates external dependency,
it is reentrant, though not thread safe.
siliconx@Lenovo:~/code/linux/shell$ |
```

T3

```
#!/bin/sh
```

```
# reference https://blog.csdn.net/beyondlpf/article/details/46426513
find -type f -perm -700
```

```
siliconx@Lenovo:~/code/linux/shell$ ./3.sh
./3.sh
./5.sh
./6.sh
./4.sh
./2.sh
./7.sh
./1.sh
siliconx@Lenovo:~/code/linux/shell$ |
```

T4

```
#!/bin/sh
```

```
for i in $@; do
    if [ -f $i ]; then
        # wc -- get the number of lines of a file
        echo $i 'is a FILE, LINES:' `wc -l < $i`
```

```

    elif [ -d $i ]; then
        echo $i 'is a DIR'

    else
        echo $i 'NOTFOUND'
    fi
done

```

```

siliconx@Lenovo:~/code/linux/shell$ ./4.sh poem1.txt test 1.sh 2.sh abcd
poem1.txt is a FILE, LINES: 9
test is a DIR
1.sh is a FILE, LINES: 15
2.sh is a FILE, LINES: 18
abcd NOTFOUND
siliconx@Lenovo:~/code/linux/shell$ |

```

T5

```

#!/bin/bash

# present the result in a form of matrix
# ROW: each file
# COL: occurrence of each word

printf "WORD \ FILE"
for i in $@; do
    printf "%11s" $i
done
echo

for word in $(<$1); do # read words_file word by word
    printf "%11s" $word

    for i in $@; do # counting
        printf "%11s" `grep -w "$word" $i | wc -l`
    done
    echo
done

```

```

siliconx@Lenovo:~/code/linux/shell$ ./5.sh words.txt poem1.txt poem2.txt
WORD \ FILE  words.txt  poem1.txt  poem2.txt
    if         1         1         1
    an         1         0         3
    and        1         0         4
    not        1         2         0
    who        1         0         0
    what       1         0         2
    which      1         2         0
siliconx@Lenovo:~/code/linux/shell$ |

```

T6

```
#!/bin/bash
```

```
ls -d */ # list dir only
```

```
siliconx@Lenovo:~/code/linux/shell$ ./6.sh  
test/  
siliconx@Lenovo:~/code/linux/shell$ |
```

## T7

```
#!/bin/bash
```

```
# `seq` gen expression, `bc` calculating  
seq -s "*" $1 | bc
```

```
siliconx@Lenovo: ~/code/linux/shell |  
siliconx@Lenovo:~/code/linux/shell$ ./7.sh 3  
3! = 6  
siliconx@Lenovo:~/code/linux/shell$ ./7.sh 4  
4! = 24  
siliconx@Lenovo:~/code/linux/shell$ |
```

## 实验思考

1.需要多练习Linux命令

2.注意命令参数匹配

## 参考资料

《Linux程序设计》