实验报告

戴琪智
23020007013

1 运行shell脚本

作为可执行程序

```
a@ubuntu:~$ chmod +x ./test.sh
a@ubuntu:~$ ./test.sh
hello world !
a@ubuntu:~$
```

作为解释器参数

```
a@ubuntu:~$ /bin/sh test.sh
hello world !
a@ubuntu:~$
```

2 Shell传递参数

执行Shell脚本时，用$n向脚本传递参数，n代表第几个参数，$0为执行文件名,$#表示传递到脚本的参数个数，$*表示所有的参数。

```
echo $1
echo $2
echo $3
echo "传递到脚本的参数个数:$#"
echo $*
```

```
a@ubuntu:~$ /bin/sh test 1 2 3
1
2
3
传递到脚本的参数个数:3
1 2 3
a@ubuntu:~$
```

3 数组

关联数组使用declare 命令来声明，使用@和#可以获取数组中的所有元素和数组长度，在数组前加一个感叹号!可以获取数组的键。

```
#!bin/bash

declare -A arr
arr[A]=a
arr[B]=b
arr[C]=c

echo ${arr[A]}
echo ${!arr[@]}
echo ${arr[@]}
echo ${#arr[@]}
```

```
a@ubuntu:~$ /bin/sh test.sh
a
A B C
a b c
3
```

4 Shell判断给定的某一年是否是闰年。

```
#!/bin/bash
echo "Input a year number:"
read year
let "leap=$year%4==0&&$year%100!=0||$year%400==0"
if [ $leap -eq 0 ]
then
        echo "$year is not a leap year"
else
        echo "$year is a leap year"
fi
```

```
a@ubuntu:~$ /bin/sh test.sh
Input a year number:
1999
1999 is not a leap year
a@ubuntu:~$ /bin/sh test.sh
Input a year number:
2000
2000 is a leap year
a@ubuntu:~$
```

5　判断给定的参数是否是素数。

```bash
#!/bin/bash
read num
if [ $num -le 1 ]; then
  echo "$num is not a prime number."
  exit 0
fi
if [ $num -eq 2 ]; then
  echo "$num is a prime number."
  exit 0
fi
if [ $((num % 2)) -eq 0 ]; then
  echo "$num is not a prime number."
  exit 0
fi
for ((i=3; i*i<=num; i+=2))
do
  if [ $((num % i)) -eq 0 ]; then
    echo "$num is not a prime number."
    exit 0
  fi
done
echo "$num is a prime number."
```

```
a@ubuntu:~$ /bin/sh prime.sh
0
0 is not a prime number.
a@ubuntu:~$ /bin/sh prime.sh
1
1 is not a prime number.
a@ubuntu:~$ /bin/sh prime.sh
3
3 is a prime number.
a@ubuntu:~$ /bin/sh prime.sh
99
99 is not a prime number.
a@ubuntu:~$
```

6　显示Fibonacci数列的前10项及其总和。

```bash
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"
```

```
a@ubuntu:~$ /bin/bash fib.sh
Fibonacci Sequence:
0
1
2
3
5
8
13
21
34
55
Sum: 142
a@ubuntu:~$
```

7　输入两个正整数m和n，求其最大公约数

```
#!/bin/bash
read num1
read num2

gcd() {
  local a=$1
  local b=$2
  while [ $b -ne 0 ]; do
    local t=$b
    b=$((a % b))
    a=$t
  done
  echo $a
}
gcd=$(gcd$num1 $num2)
echo $gcd
```

```
a@ubuntu:~$ /bin/bash gcd.sh
9
6
3
a@ubuntu:~$
```

8　输入三个整数x、y、z，把这三个数由小到大输出。

```
#!/bin/bash

x=$1
y=$2
z=$3

if [ $x -gt $y ]; then
   t=$x
   x=$y
   y=$t
fi

if [ $x -gt $z ]; then
   t=$x
   x=$z
   z=$t
fi

if [ $y -gt $z ]; then
   t=$y
   y=$z
   z=$t
fi
echo "$x $y $z"
```



```
a@ubuntu:~$ /bin/bash compare.sh 21 12 9
9 12 21
a@ubuntu:~$
```

9　打印出菱形。

```
#!/bin/bash

for (( i=1; i<=4; i++ ))
do
  for (( j=4; j>i; j-- ))
  do
    echo -n " "
  done

  for (( k=1; k<=(2*i-1); k++ ))
  do
    echo -n "*"
  done
  echo
done

for (( i=3; i>=1; i-- ))
do
  for (( j=4; j>i; j-- ))
  do
    echo -n " "
  done

  for (( k=1; k<=(2*i-1); k++ ))
  do
    echo -n "*"
  done
  echo
done
```

```
a@ubuntu:~$ /bin/bash rho.sh
   *
  ***
 *****
*******
 *****
  ***
   *
a@ubuntu:~$
```

10　打印出1到1000所有的”水仙花数”

```
#!/bin/bash

for((i = 0;i < 1000; i++))
do
   x=$((i % 10))
   y=$(((i / 10) % 10))
   z=$(((i / 100) % 10))
   sum=$((x**3 + y**3 + z**3))
   if [$i -eq $sum ]; then
           echo $i
   fi
done
```

```
a@ubuntu:~$ /bin/bash nar.sh
0
1
153
370
371
407
```

11 vim打开文件

vim 文件路径光标置于上次退出时的位置。

```
a@ubuntu:~$ vim fib.sh
```

```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"
```

vim 文件路径+n 光标位于第n行

```
a@ubuntu:~$ vim fib.sh +5
```

```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"

~
~
~
~
~
~
~
"fib.sh" 16L, 183C                                    5,1           全部
```

vim 文件路径+ 光标位于末行的行首

```
a@ubuntu:~$ vim fib.sh +
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"

~
~
~
~
~
~
~
"fib.sh" 16L, 183C                                    16,0-1        全部
```

12 vim查找

进入vim默认为普通模式，按下“:”键进入末行模式。\ + 查找内容，从光标所在行向下查找，？ + 查找内容，从光标所在行向上查找。n下一个，N上一个。

```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"
```

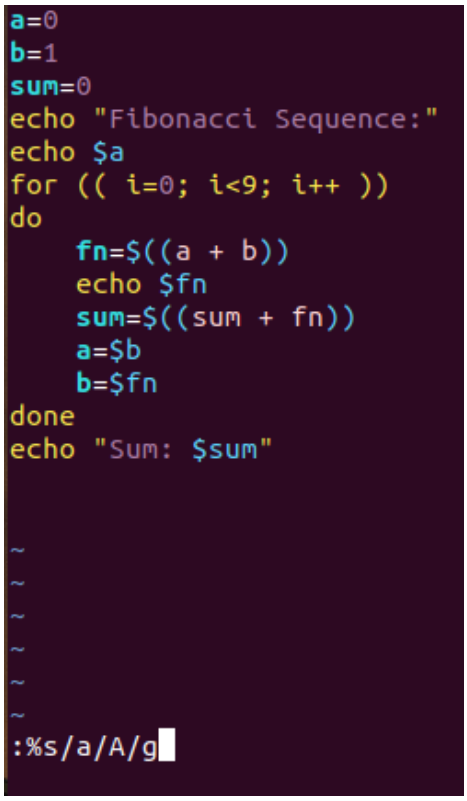
```
:/do
```

```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"
```

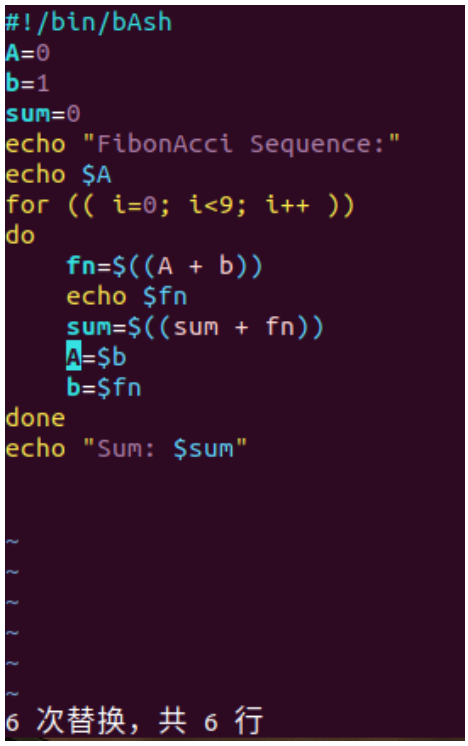
```
:?sum
```

13 vim替换

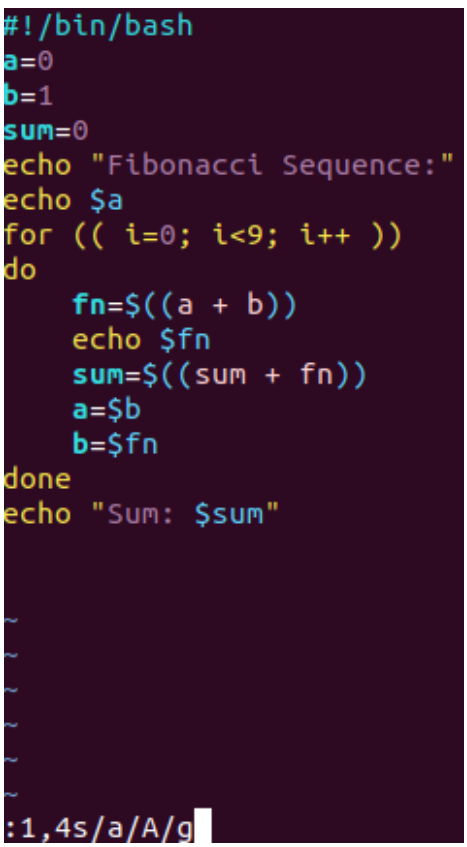
:%s/search/replace/g - 在整个文件中替换所有匹配项。
:n1,n2s/search/replace/g - 在指定行范围（n1 到n2）内替换。



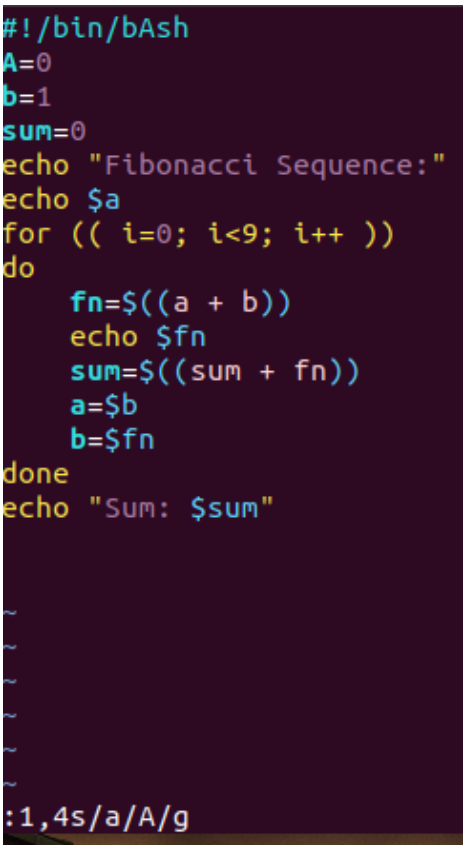
```
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"



~
~
~
~
~
~
:%s/a/A/g
```



```
#!/bin/bAsh
A=0
b=1
sum=0
echo "FibonAcci Sequence:"
echo $A
for (( i=0; i<9; i++ ))
do
    fn=$((A + b))
    echo $fn
    sum=$((sum + fn))
    A=$b
    b=$fn
done
echo "Sum: $sum"



~
~
~
~
~
~
6 次替换，共 6 行
```



```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"


~
~
~
~
~
~
:1,4s/a/A/g
```



```
#!/bin/bAsh
A=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"


~
~
~
~
~
~
:1,4s/a/A/g
```

14 导入其他文件的内容

:r 文件名把文件内容导入到光标位置.

```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"


~
~
~
~
~
~
:r nar.sh
```

```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"


#!/bin/bash

for((i = 0;i < 1000; i++))
do
   x=$((i % 10))
   y=$(((i / 10) % 10))
"nar.sh" 12L, 191C
```

15 在vim 中执行系统命令

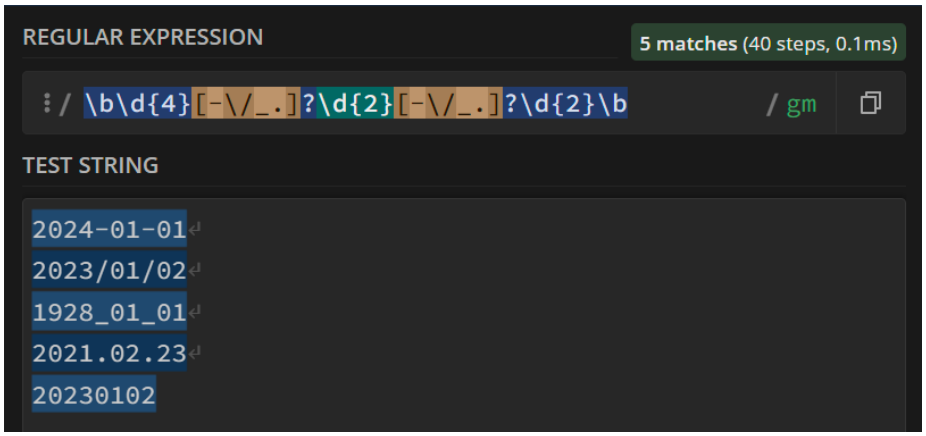
:! 执行系统命令。

```
#!/bin/bash
a=0
b=1
sum=0
echo "Fibonacci Sequence:"
echo $a
for (( i=0; i<9; i++ ))
do
    fn=$((a + b))
    echo $fn
    sum=$((sum + fn))
    a=$b
    b=$fn
done
echo "Sum: $sum"



~
~
~
~
~
~
:!ls
```

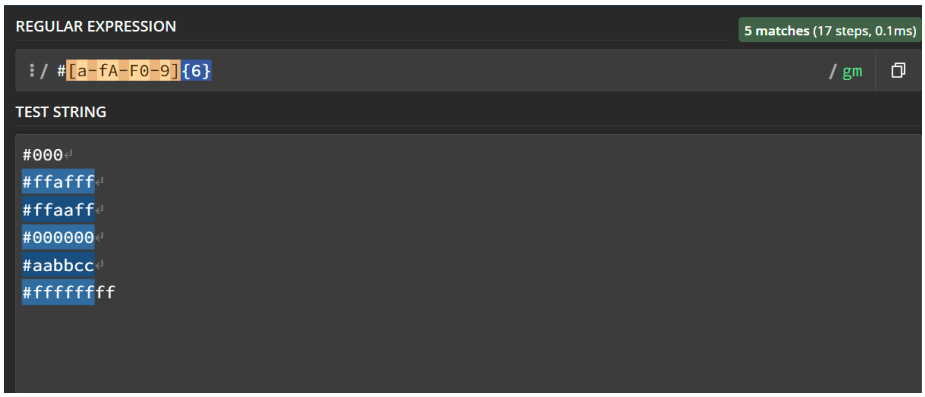
```
compare.sh  Documents  fib.sh  Music    Pictures  Public  Templates  test.sh
Desktop     Downloads  gcd.sh  nar.sh   prime.sh  rho.sh  test       Videos

请按 ENTER 或其它命令继续
```

16 正则表达式懒惰匹配

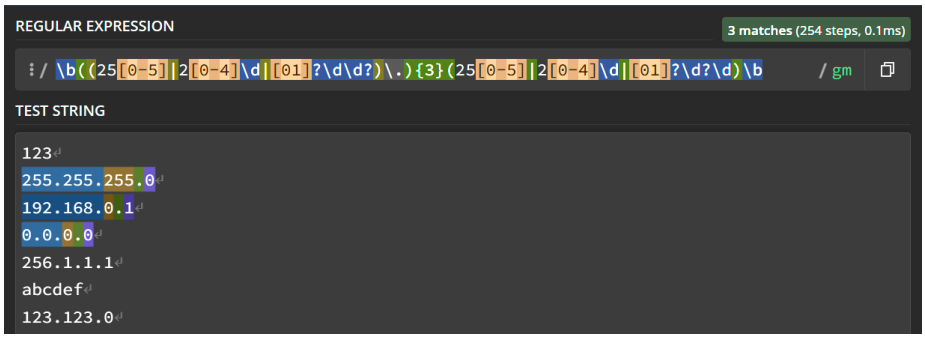
17 正则表达式匹配日期



18 正则表达式RGB颜色值匹配

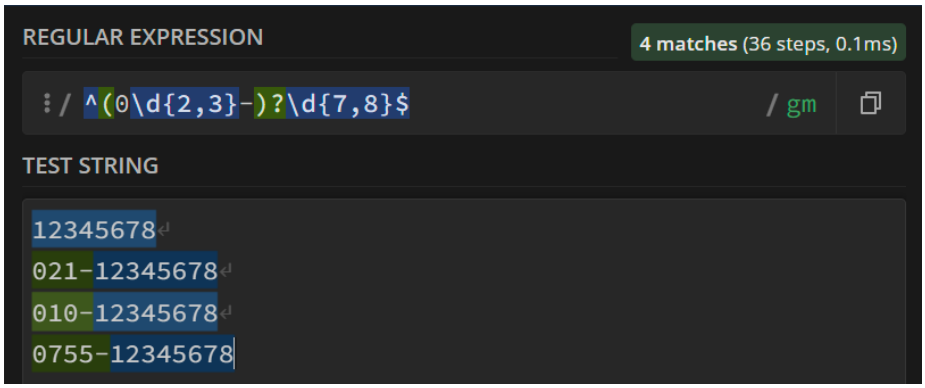


19 正则表达式IPv4地址匹配



20 正则表达式中国固定电话号码匹配

中在中国，固定电话号码通常由区号和电话号码组成。区号是一个3到4位的数字，电话号码是一个7到8位的数字。



感悟

学习shell、Vim 和正则表达式这些工具，可以让我在处理文本和自动化任务时变得更加高效和灵活。熟练使用shell 命令可以极大地提高我处理文件和执行系统任务的速度。通过命令行，我可以批量处理文件，无需使用图形界面。Vim 是一个功能强大的文本编辑器，特别是对于那些经常编辑文本的人来说。它的多模式编辑和命令行输入使得文本操作更加高效。正则表达式是一个强大的文本处理工具，可以用来搜索、替换、提取和修改文本。正则表达式允许你精确地描述我想要匹配的模式，这对于处理复杂的文本数据非常有用。

https://github.com/asd279/myrepo