Task 1

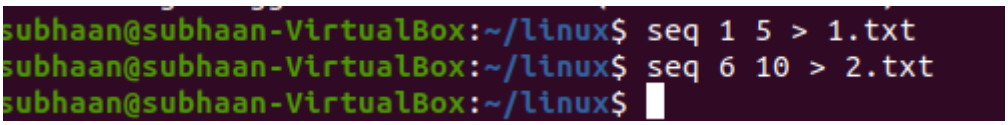
Task 1.1

Script code:

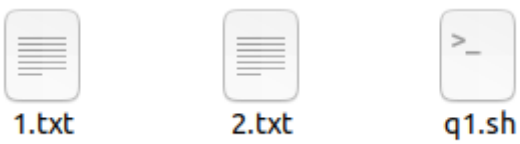
```
#!/bin/bash
if [ -z "$2" ]; then
        echo "Please enter two inputs."
else
        if [ -f "$1" ] && [ -f "$2" ]; then
                numOfLines=$(cat "$1" "$2" | wc -l)
                echo "Total number of lines is $numOfLines."
        elif [ -d "$1" ]; then
                echo "$1 is a directory"
        elif [ -d "$2" ]; then
                echo "$2 is a directory"
        else
                echo "Error: the inputs are not valid files."
        fi
fi
```

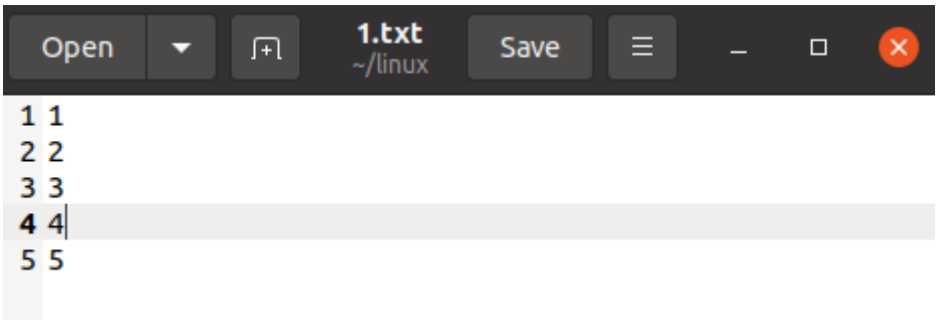
Test case 1

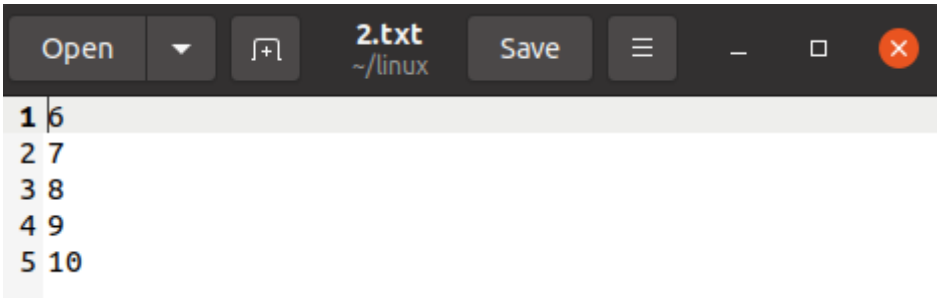
seq 1 5 > 1.txt	(To create the file “1.txt” with numbers 1-5)
seq 6 10 > 2.txt	(To create the file “2.txt” with numbers 6-10)
bash q1.sh 1.txt 2.txt	(To run the script with the arguments “1.txt” and “2.txt”)



After running the following commands, the files “1.txt” and “2.txt” are created and stored in the “linux” directory:







From the screenshots, it is visible that the files have been created successfully. The total number of lines in both files is 10, therefore the script should output 10.

```
subhaan@subhaan-VirtualBox:~/linux$ bash q1.sh 1.txt 2.txt
Total number of lines is 10.
subhaan@subhaan-VirtualBox:~/linux$
```

You can see that the script successfully output that the number of lines is 10.

Test case 2

seq 1 5 > 1.txt (To create the file “1.txt” with numbers 1-5)

mkdir test (To create a directory called “test”)

bash q1.sh 1.txt test (To run the script with the arguments “1.txt” and “test”)

The arguments contain a directory, therefore the script should output that test is a directory:

```
subhaan@subhaan-VirtualBox:~/linux$ seq 1 5 > 1.txt
subhaan@subhaan-VirtualBox:~/linux$ mkdir test
subhaan@subhaan-VirtualBox:~/linux$ bash q1.sh 1.txt test
test is a directory
subhaan@subhaan-VirtualBox:~/linux$
```

Test case 3

bash q1.sh (To run the script with no arguments)

Since there are no arguments, the script should prompt the user to provide two inputs:

```
subhaan@subhaan-VirtualBox:~/linux$ bash q1.sh
Please enter two inputs.
subhaan@subhaan-VirtualBox:~/linux$
```

Task 1.2

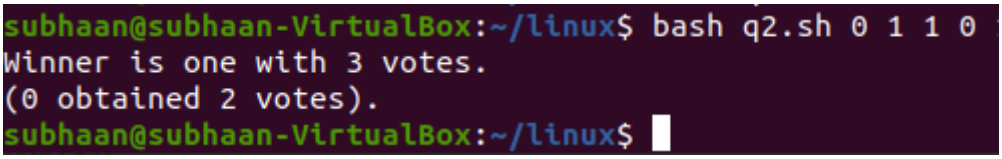
Script code:

```
#!/bin/bash
zero=0
one=0
flag=0
while [ $# -gt 0 ]; do
        if [ "$1" -eq 0 ]; then
                let zero=zero+1
                let flag=1
        elif [ "$1" -eq 1 ]; then
                let one=one+1
                let flag=1
        else
                echo "Please enter either zeros or ones."
        fi
        shift
done

if [ $zero -gt $one ]; then
        echo "Winner is zero with $zero votes."
        echo "(1 obtained $one votes)"
elif [ $one -gt $zero ]; then
        echo "Winner is one with $one votes."
        echo "(0 obtained $zero votes)"
else
        if [ $one==$zero ]; then
                if [ $flag -eq 1 ]; then
                        echo "Vote is a tie. Each obtained $zero votes."
                else
                        echo "You have not entered a valid input."
                fi
        fi
fi
```

Test case 1

bash q2.sh 0 1 1 0 1 (To run the script with the arguments 0 1 1 0 1)



Test case 2

bash q2.sh 0 0 0 0 (To run the script with the arguments 0 0 0 0)

```
subhaan@subhaan-VirtualBox:~/linux$ bash q2.sh 0 0 0 0
Winner is zero with 4 votes.
(1 obtained 0 votes)
subhaan@subhaan-VirtualBox:~/linux$
```

Test case 3

bash q2.sh 500 (To run the script without a valid argument – not a 0 or 1)

```
subhaan@subhaan-VirtualBox:~/linux$ bash q2.sh 500
Please enter either zeros or ones.
You have not entered a valid input.
subhaan@subhaan-VirtualBox:~/linux$
```

The user is notified that they have not provided a valid input. To do this, I have used a nested if loop to check if any of the values were zeroes or ones. If no zeroes or ones are met, the message is output.

Test case 4

bash q2.sh 0 0 1 1 0 1 1 0 (To run the script with the same amount of ones and zeroes)

```
subhaan@subhaan-VirtualBox:~/linux$ bash q2.sh 0 0 1 1 0 1 1 0
Vote is a tie. Each obtained 4 votes.
subhaan@subhaan-VirtualBox:~/linux$
```

Task 1.3

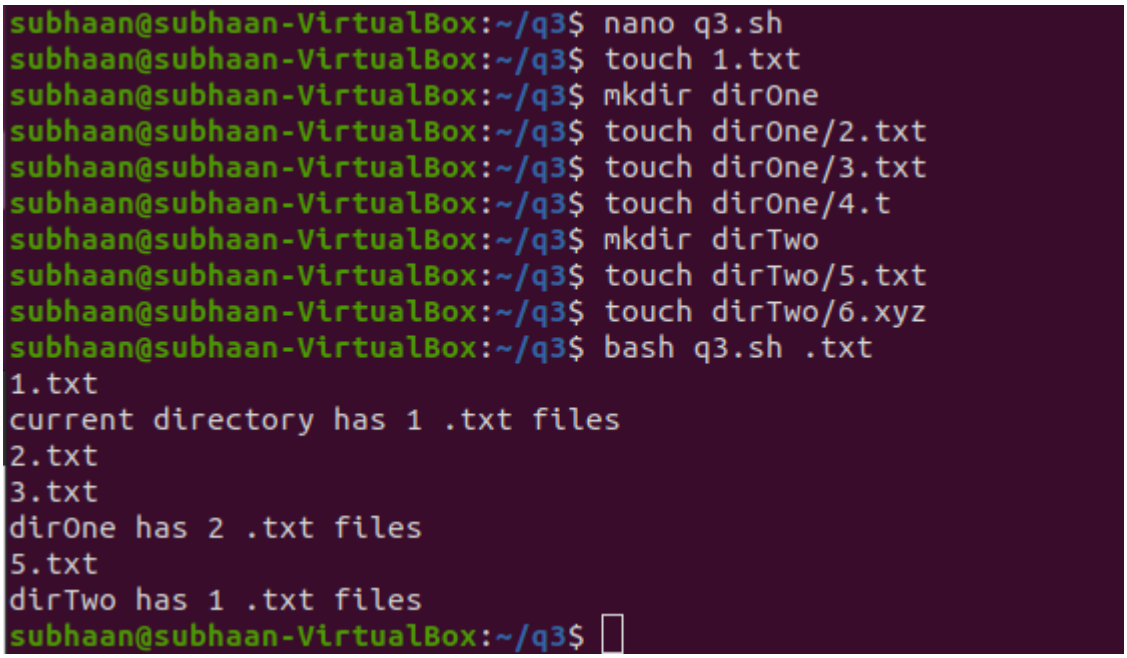
Script code:

```
#!/bin/bash
extension="$1"
fileCount=0
if [ -z "$extension" ]; then
        echo "Please enter an extension to search for (such as .txt)"
        read extension
fi
ls | grep $extension
fileCount=$(ls | grep $extension | wc -l)
echo "current directory has $fileCount $extension files"
for file in *; do
        if [ -d "$file" ]; then
                cd $file
                ls | grep $extension
                fileCount=$(ls | grep $extension | wc -l)
                echo "$file has $fileCount $extension files"
                cd ..
```

```
        fi
done
```

Test case 1

```
touch 1.txt
mkdir dirOne
touch dirOne/2.txt
touch dirOne/3.txt
touch dirOne/4.t
mkdir dirTwo
touch dirTwo/5.txt
touch dirTwo/6.xyz
bash q3.sh .txt
```



Test case 2

```
touch one.test
mkdir firstDir
touch firstDir/second.txt
touch firstDir/third.test
mkdir secondDir
touch secondDir/fourth.txt
touch secondDir/fifth.test
mkdir thirdDir
```

touch thirdDir/sixth.test
bash q3.sh
.test

```
subhaan@subhaan-VirtualBox:~/test2$ touch one.test
subhaan@subhaan-VirtualBox:~/test2$ mkdir firstDir
subhaan@subhaan-VirtualBox:~/test2$ touch firstDir/second.txt
subhaan@subhaan-VirtualBox:~/test2$ touch firstDir/third.test
subhaan@subhaan-VirtualBox:~/test2$ mkdir secondDir
subhaan@subhaan-VirtualBox:~/test2$ touch secondDir/fourth.txt
subhaan@subhaan-VirtualBox:~/test2$ touch secondDir/fifth.test
subhaan@subhaan-VirtualBox:~/test2$ mkdir thirdDir
subhaan@subhaan-VirtualBox:~/test2$ touch thirdDir/sixth.test
subhaan@subhaan-VirtualBox:~/test2$ bash q3.sh
Please enter an extension to search for (such as .txt)
.test
one.test
current directory has 1 .test files
third.test
firstDir has 1 .test files
fifth.test
secondDir has 1 .test files
sixth.test
thirdDir has 1 .test files
subhaan@subhaan-VirtualBox:~/test2$
```

Test case 3

bash q3.sh .qwerty

```
subhaan@subhaan-VirtualBox:~/test2$ bash q3.sh .qwerty
current directory has 0 .qwerty files
subhaan@subhaan-VirtualBox:~/test2$
```

Because there are no files with the .qwerty extension, the script outputs “current directory has 0 .qwerty files”.

Task 2

Creating admin account:

useradd -m admin
passwd admin
password
password

```
root@slax:~# useradd -m admin
root@slax:~# passwd admin
Changing password for admin
Enter the new password (minimum of 5 characters)
Please use a combination of upper and lower case letters and numbers.
New password:
Re-enter new password:
passwd: password changed.
root@slax:~#
```

groupadd sudo
gpasswd -a admin sudo

```
root@slax:~# groupadd sudo
root@slax:~# gpasswd -a admin sudo
Adding user admin to group sudo
root@slax:~#
```

visudo
Remove # from line 81

```
## Uncomment to allow members of group sudo to execute any command
%sudo<->ALL=(ALL) ALL
```

F2 to save, F10 to quit
vi /home/admin/.bashrc
PATH=$PATH:/sbin:/usr/sbin

```
# Put your user specific aliases and functions here

PATH=$PATH:/sbin:/usr/sbin
```

exit
admin
password
sudo -l
password

```
slax login: admin
Password:
Linux 3.6.11.
admin@slax:~$ sudo -l

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

Password:
User admin may run the following commands on this host:
    (ALL) ALL
admin@slax:~$ _
```

Aliases and UMASK

sudo vi +64 /etc/profile
change UMASK to 002

```
# Default umask.  A umask of 022 prevents new files from being created group
# and world writable.
umask 002
```

sudo vi /etc/skel/.bashrc
alias help="man"
alias rename="mv"
alias dir="ls"

```
# .bashrc

# Put your user specific aliases and functions here

alias help="man"
alias rename="mv"
alias dir="ls"


# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
~
```

You can see these commands are working using a test user:

```
slax login: test
Password:
Linux 3.6.11.
test@slax:~$ help
What manual page do you want?
test@slax:~$ mv
mv: missing file operand
Try 'mv --help' for more information.
test@slax:~$ rename
mv: missing file operand
Try 'mv --help' for more information.
test@slax:~$ dir
Desktop/  Documents/  Downloads/  Music/  Pictures/  Public/  Templates/  Videos/
test@slax:~$
```

```
admin@slax:~$ ls -l /home/
total 0
drwx------ 11 admin admin     340 May 20 19:34 admin/
drwx------  2 ed    ed        140 May 20 19:41 ed/
drwx------  2 john  john      140 May 20 19:39 john/
drwxrws---  2 root  johnanded  40 May 20 19:59 johnanded/
drwx------  2 mel   mel       140 May 20 19:42 mel/
drwx------  2 myron myron     140 May 20 19:42 myron/
admin@slax:~$ _
```

User accounts

sudo useradd -m john
sudo useradd -m ed
sudo useradd -m myron
sudo useradd -m mel
sudo passwd john
sudo passwd ed
sudo passwd myron
sudo passwd mel

```
admin:$5$1MS9xgAnlD/$iuZ/G2Tr1UIPgRcKjZ2MQBG2D76a5qe.wbRMgZEZmC7:18767:0:99999:7:::
john:$5$LMiMi/2.$obVwmQzxXHwTXc8WtIEw2kS4hYF8RhYSDI2KcFLy9pC:18767:0:99999:7:::
ed:$5$pjyEt/tptgZ/6RR1$87hUyu5999IUolm.FN5kFdDlcyvOkr23Q9sdHpLTeC5:18767:0:99999:7:::
myron:$5$tw10e/sQQHK/3D0$EgJ6Y3gdZ9AcbwfU2jzlFqCB6vC.B1CeqKUAHLtUZ51:18767:0:99999:7:::
mel:$5$1FTXesqmfr$Sw3n4GIqdYp/LsVCdcMr89WuMhKYcwuutKUDo2/eOL3:18767:0:99999:7:::
```

sudo chmod 700 /home/*
sudo mkdir /home/johnanded
sudo groupadd johnanded
sudo gpasswd -a john johnanded
sudo gpasswd -a ed johnanded

```
sudo:x:1002:admin
john:x:1003:
ed:x:1004:
myron:x:1005:
mel:x:1006:
johnanded:x:1007:john,ed
admin@slax:~$ _
```

sudo chgrp johnanded /home/johnanded/
sudo chmod 2770 /home/johnanded/
sudo mkdir /home/myronandmel
sudo groupadd myronandmel
sudo gpasswd -a myron myronandmel
sudo gpasswd -a mel myronandmel
sudo chngrp myronandmel /home/myronandmel
sudo chmod 2775 /home/myronandmel

Testing

Testing as John

John should not have access to anyone else's home directory except his own. To ensure John can access his own home directory, I will try creating a text file:

```
"hometest.txt" 1L, 27C written
john@slax:~$ _
```

To view the contents of the file, I can use cat:

```
john@slax:~$ cat /home/john/hometest.txt
this is john in /home/john
john@slax:~$
```

John should not be able to access anyone else's home directory:

```
john@slax:~$ ls /home/ed/
/bin/ls: cannot open directory /home/ed/: Permission denied
john@slax:~$ _
```

John should be able to read and write in the *johnanded* directory:

```
"/home/johnanded/test.txt" 1L, 26C written
john@slax:~$ _
```

However, John should only be able to read in the *myronandmel* directory:

```
"/home/myronandmel/test.txt" [readonly] 2L, 56C
```

The [readonly] indicator notifies John that he can only read the file, and not make changes to it. The commands help, rename and dir should be available:

```
john@slax:~$ help
What manual page do you want?
john@slax:~$ rename
mv: missing file operand
Try 'mv --help' for more information.
john@slax:~$ dir
Desktop/    Downloads/  Pictures/  Templates/  hometest.txt
Documents/  Music/      Public/    Videos/     hometest.txt~
john@slax:~$ _
```

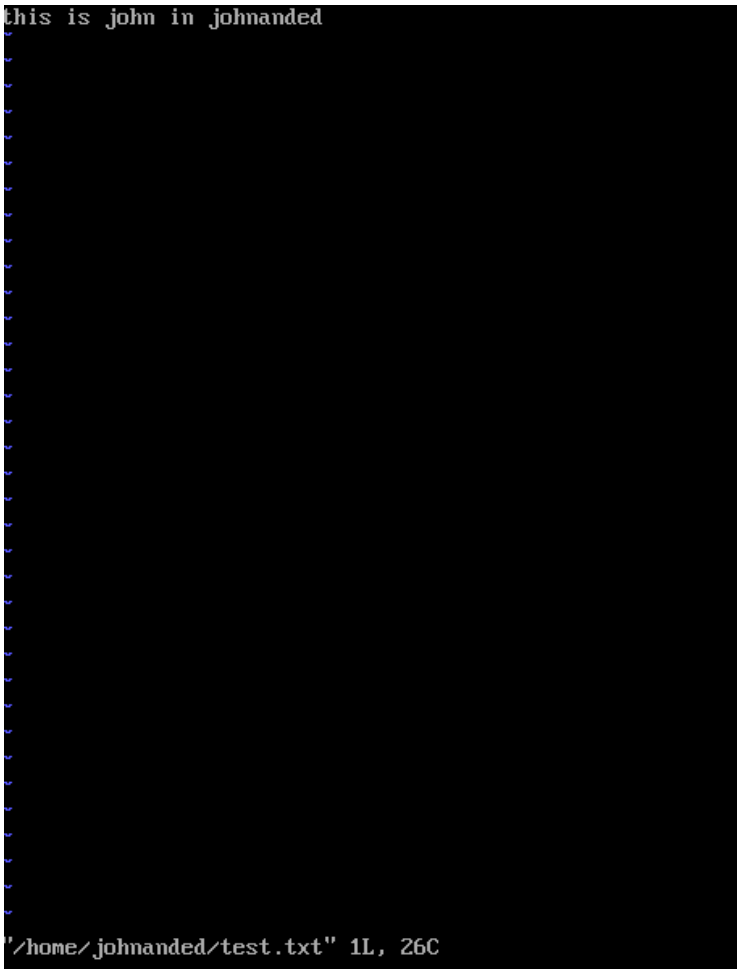
Testing as Ed

Ed should not have access to anyone else's home directory. Ed should have read and write access to *johnanded* (the directory I created for John and Ed's collaboration work).

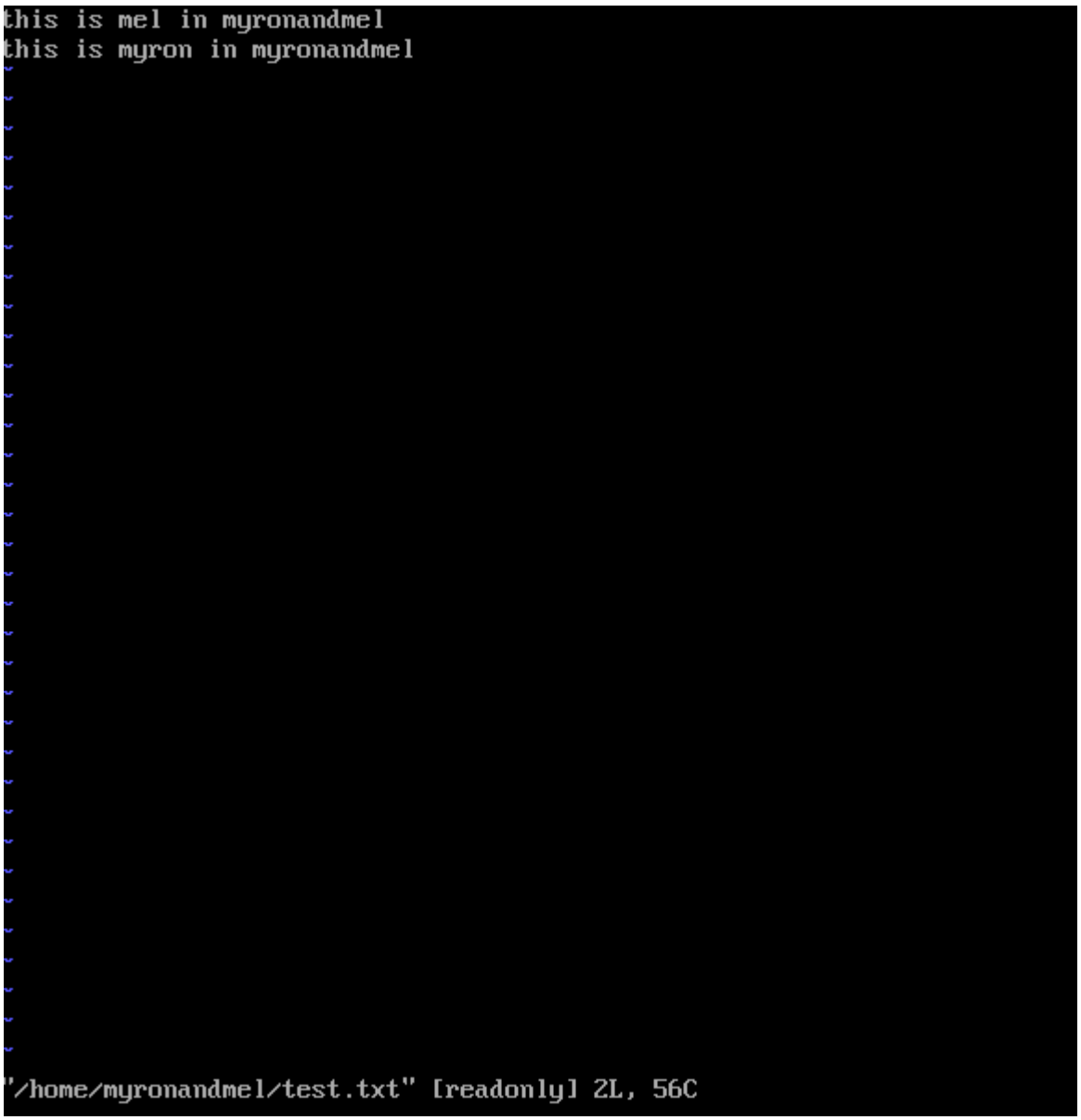
```
slax login: ed
Password:
Linux 3.6.11.
ed@slax:~$ ls /home/john
/bin/ls: cannot open directory /home/john: Permission denied
ed@slax:~$
```

```
slax login: ed
Password:
Linux 3.6.11.
ed@slax:~$ ls /home/john
/bin/ls: cannot open directory /home/john: Permission denied
ed@slax:~$ cat /home/john/hometest.txt
cat: /home/john/hometest.txt: Permission denied
ed@slax:~$
```

vi /home/johnanded/test.txt



Ed should have access to read the files located in myronandmel:



From the screenshot, you can see that this access is readonly.

Furthermore, the commands help, rename and dir should be available:

Testing as Mel

```
mel@slax:~$ cat /home/john/hometest.txt
cat: /home/john/hometest.txt: Permission denied
mel@slax:~$
```

```
mel@slax:~$ ls /home/johnanded/
/bin/ls: cannot open directory /home/johnanded/: Permission denied
mel@slax:~$ _
```

```
mel@slax:~$ cat /home/johnanded/test.txt
cat: /home/johnanded/test.txt: Permission denied
mel@slax:~$
```

Mel cannot access John's hometest.txt file (which is in his home directory). This means that home directories are successfully private.
Furthermore, Mel cannot access the *johnanded* directory because it is limited to John and Ed. However, Mel can access the *myronandmel* directory.

vi /home/myronandmel/test.txt (To create a test text file in the *myronandmel* directory)

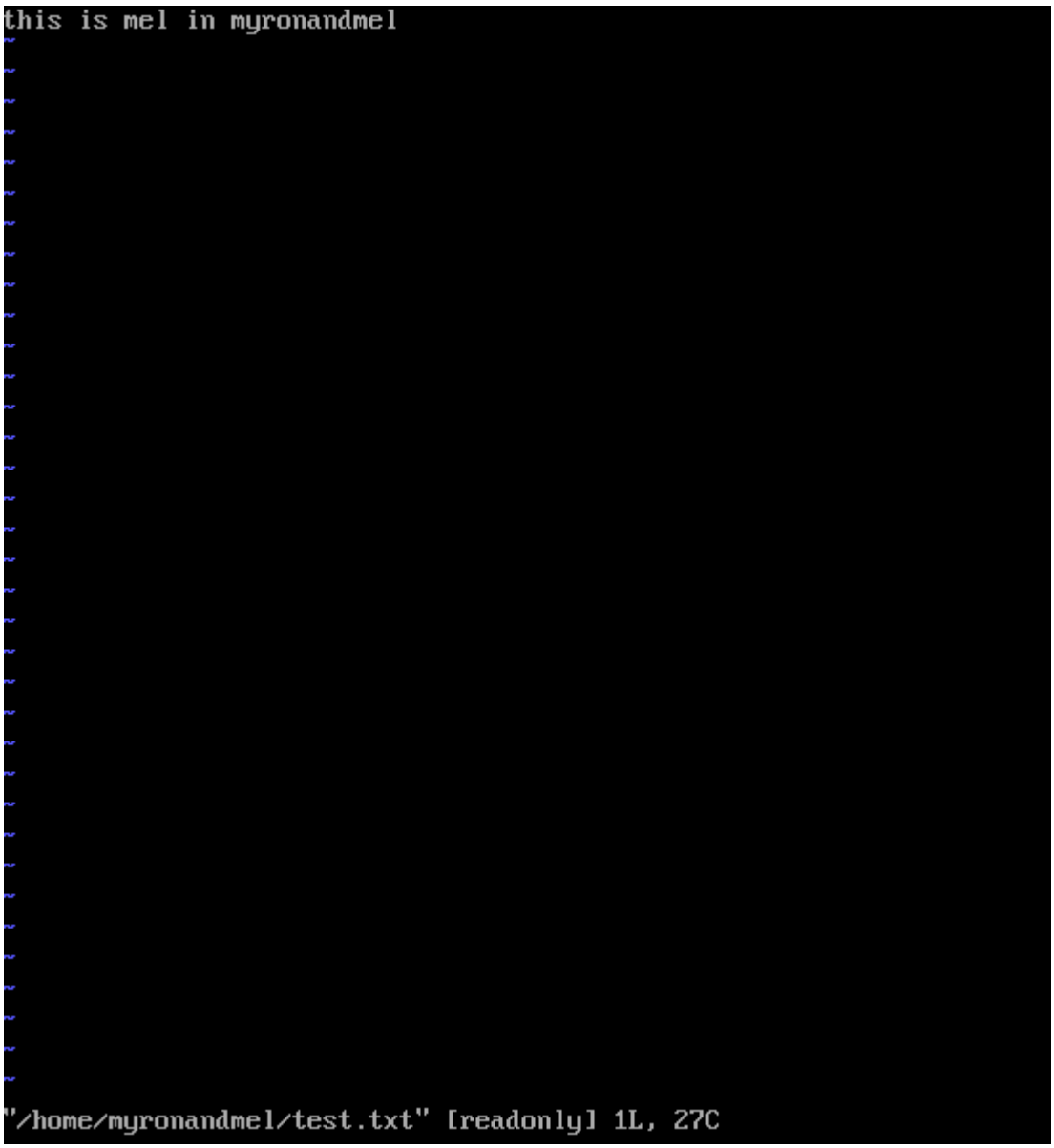
```
"/home/myronandmel/test.txt" [New] 1L, 27C written
mel@slax:~$ _
```

According to the specification, users not part of the *myronandmel* group should still be able to read files in the directory.

To test this, I will login as John and try to read the test.txt file located in *myronandmel*.

After logging in as John, executing the following command opens the test.txt in vi:

vi /home/myronandmel/test.txt



From the screenshot, you can see at the bottom that the file is readonly.
To test that the file can be edited by the other member of the *myronandmel* group, Myron, I will login to Myron and try to edit the text file.

The commands help, rename and dir should be available:

```
mel@slax:~$ help
What manual page do you want?
mel@slax:~$ rename
mv: missing file operand
Try 'mv --help' for more information.
mel@slax:~$ dir
Desktop/  Documents/  Downloads/  Music/  Pictures/  Public/  Templates/  Videos/
mel@slax:~$
```

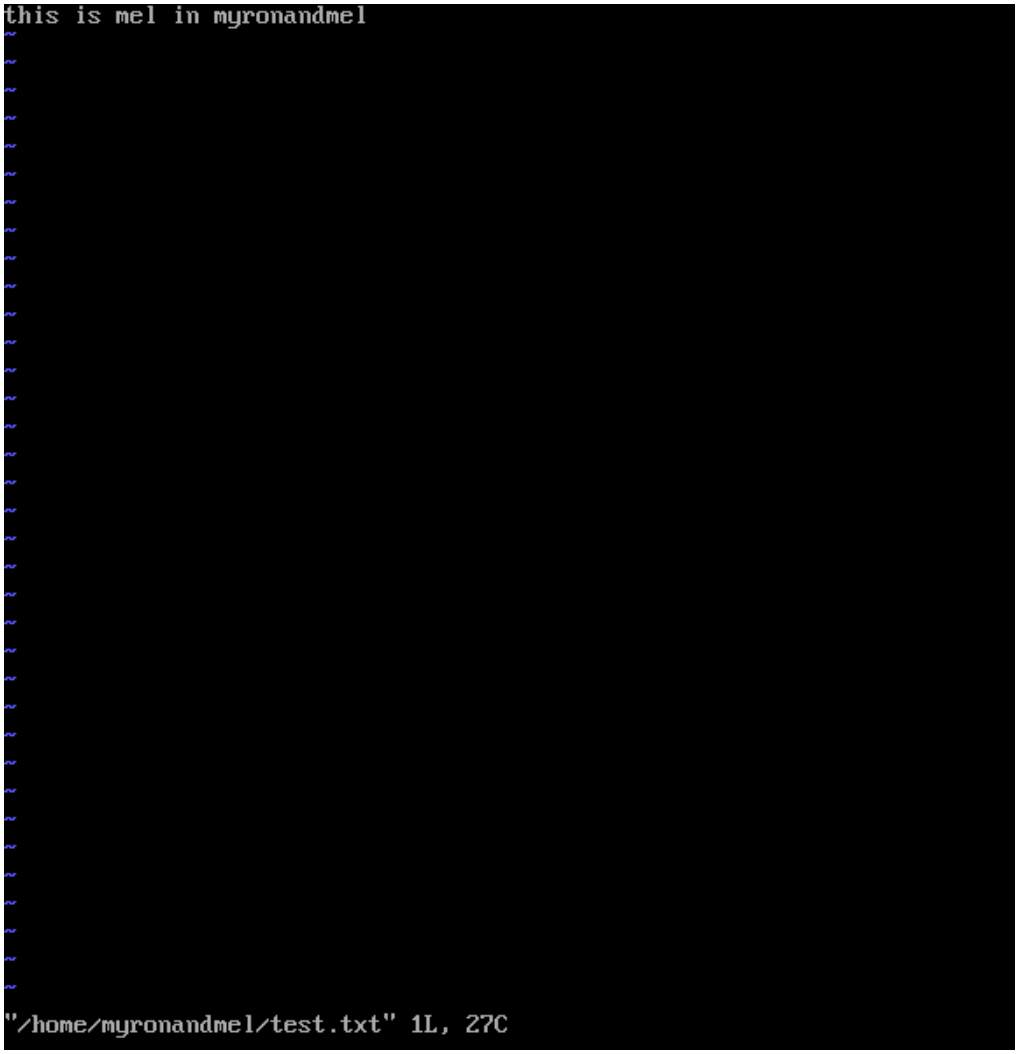
Testing as Myron

Myron should not be able to access other member's home directories. To test this, I will try and access John's home directory:

```
myron@slax:~$ ls /home/john
/bin/ls: cannot open directory /home/john: Permission denied
myron@slax:~$
```

However, Myron should be able to edit the test file located in *myronandmel*.

vi /home/myronandmel/test.txt



You can see that the [readonly] indicator is not present at the bottom when logged in as Myron.

```
"/home/myronandmel/test.txt" 2L, 56C written
myron@slax:~$
```

Furthermore, I was able to update the text file to the following:

```
myron@slax:~$ cat /home/myronandmel/test.txt
this is mel in myronandmel
this is myron in myronandmel
myron@slax:~$ _
```

Myron should not have access to the files in *johnanded*:

```
myron@slax:~$ ls /home/johnanded/
/bin/ls: cannot open directory /home/johnanded/: Permission denied
myron@slax:~$
```

The commands help, rename and dir should be available:

```
myron@slax:~$ help
What manual page do you want?
myron@slax:~$ rename
mv: missing file operand
Try 'mv --help' for more information.
myron@slax:~$ dir
Desktop/  Documents/  Downloads/  Music/  Pictures/  Public/  Templates/  Videos/
myron@slax:~$
```