Basic scripting in Linux

Nikita Neveditsin, SMU, 2019

nikita.neveditsin@smu.ca

Linux Shells

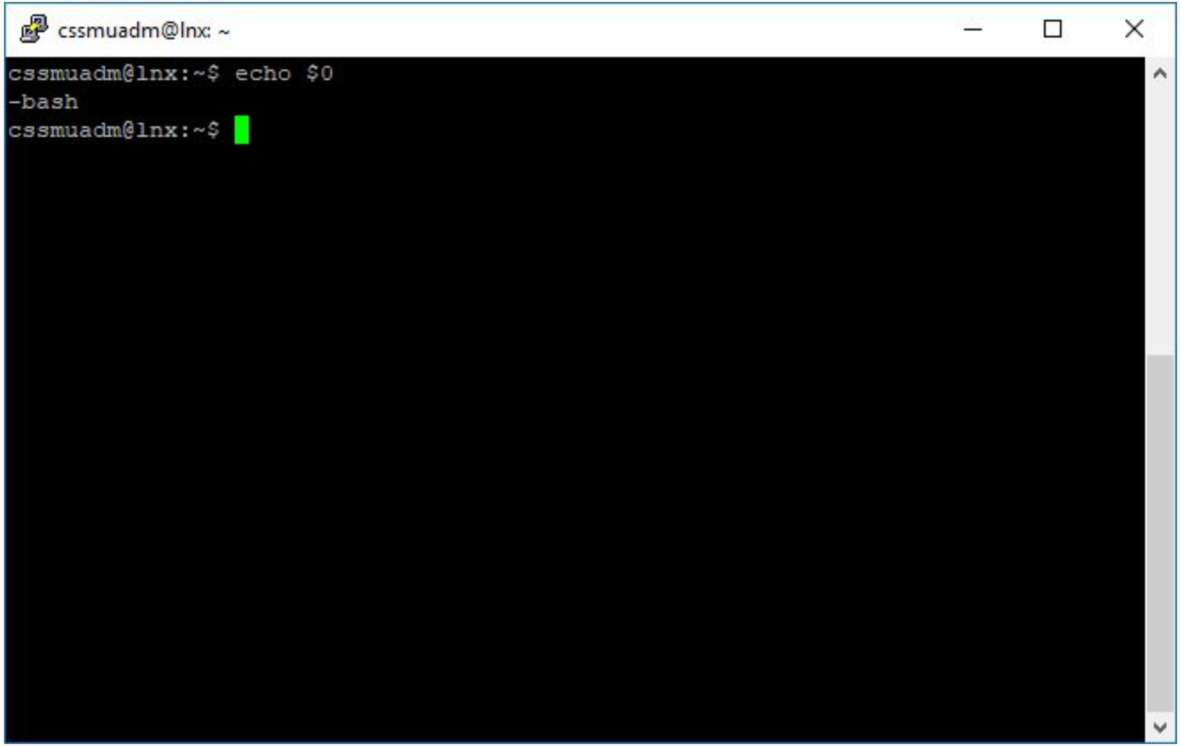
- Shell is just a program that allows user to interact with operating system: it takes commands from the keyboard and executes them
- There are few different shells for Linux. Most popular now is bash

[1]

Linux Shells (cont-d)

The default shell in most modern Linux distributions is **bash** (Bourne-again shell)

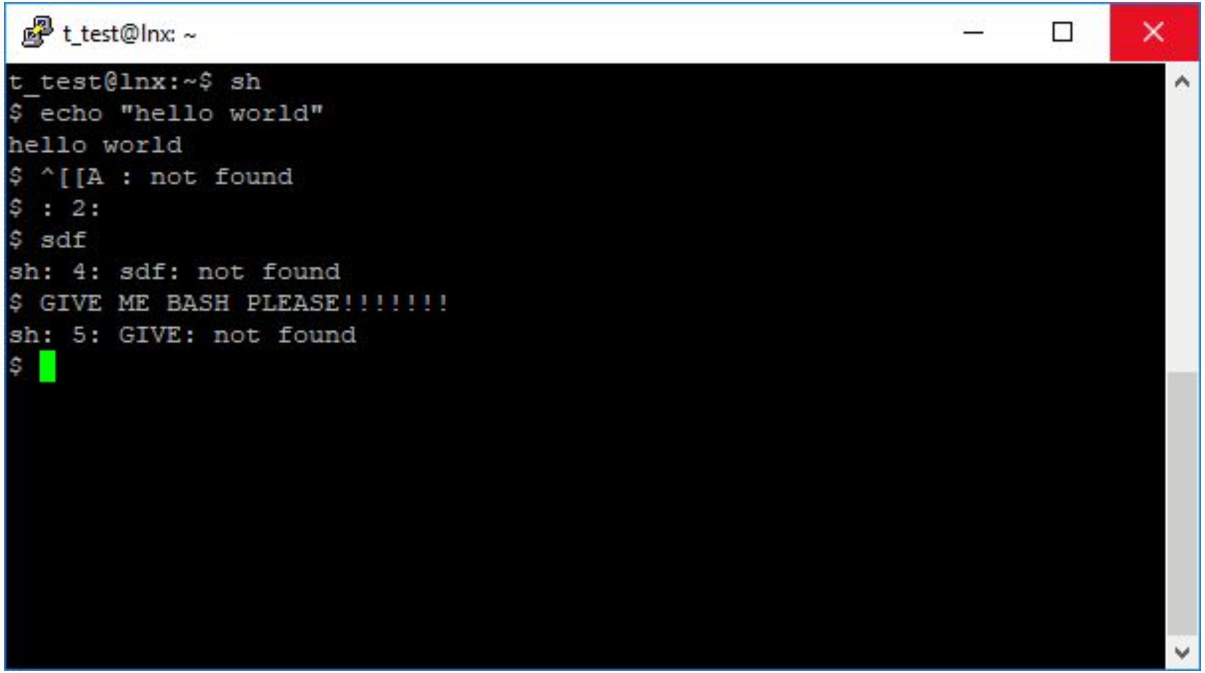
If you don't know which shell you are using, just type **echo $0**



```
cssmuadm@lnx:~$ echo $0
-bash
cssmuadm@lnx:~$
```

Linux Shells (cont-d)

Other shell that has been used since 1977 is **sh** (Bourne shell). It's one of the the oldest shells.

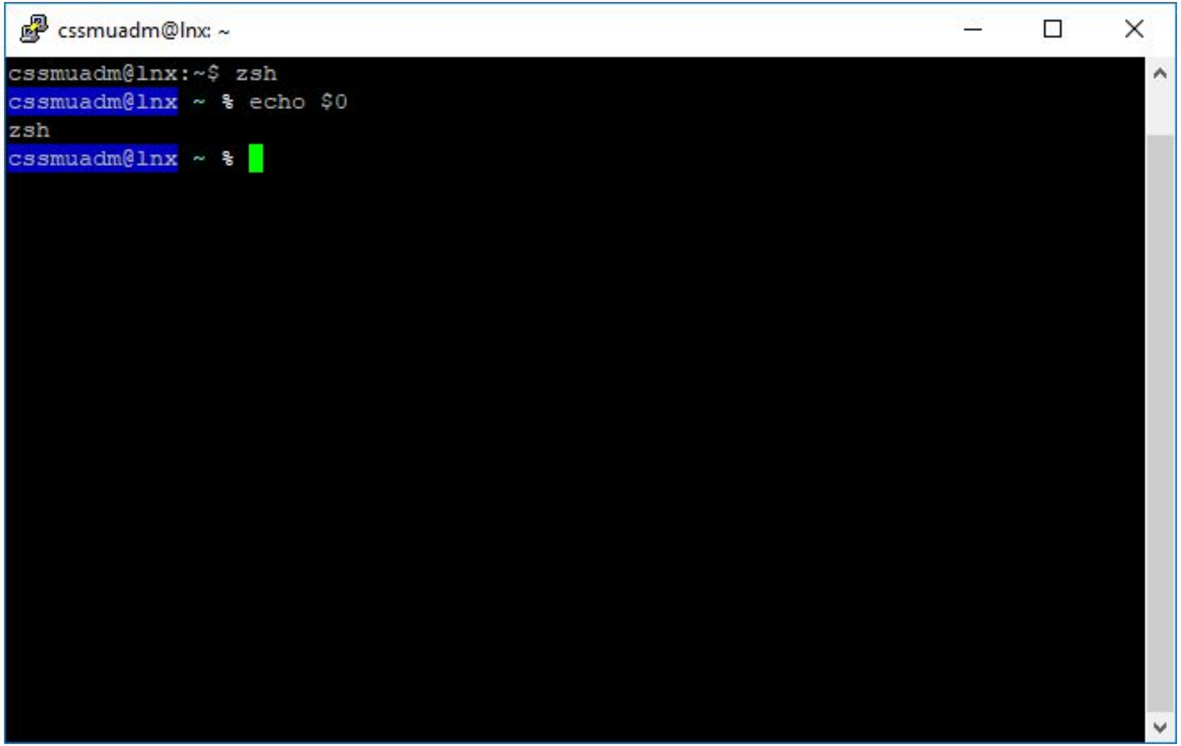


```
t_test@lnx:~$ sh
$ echo "hello world"
hello world
$ ^[[A : not found
$ : 2:
$ sdf
sh: 4: sdf: not found
$ GIVE ME BASH PLEASE!!!!!!!
sh: 5: GIVE: not found
$ 
```

Linux Shells (cont-d)

Few other examples: **zsh**

It has some nice features like “smart” auto-completion, spell check, etc.



```
cssmuadm@lnx:~$ zsh
cssmuadm@lnx ~ % echo $0
zsh
cssmuadm@lnx ~ %
```

Shell Scripts

- Shell script is a program
- Shell scripts are executed with a particular interpreter

Shell Scripts: #!/bin/bash or #!/bin/sh ?

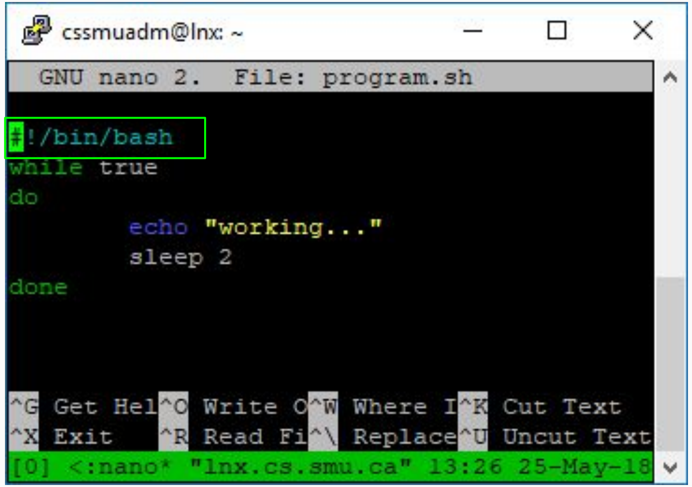
#! called shebang interpreter directive

- If you would like to make sure it's POSIX-compatible, use #!/bin/sh
- If you are sure that the system that runs your script has bash installed, use #!/bin/bash

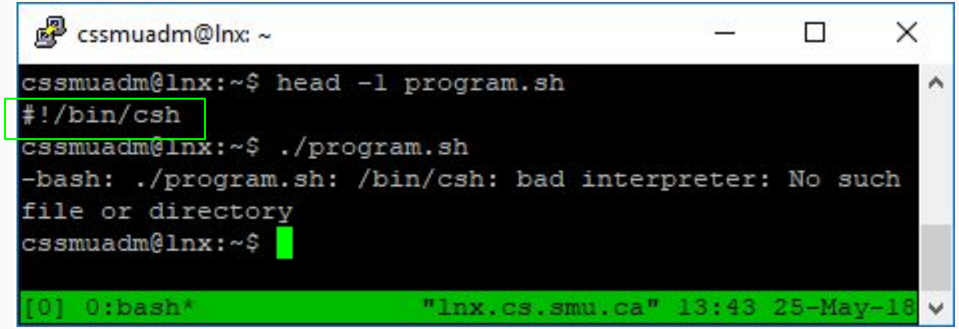
Bash scripts have more functionality and better syntax

You can specify other interpreter like #!/bin/csh (if you are sure that is installed on the target system)

NOTE: a script without shebang interpreter directive is just a sequence of commands!



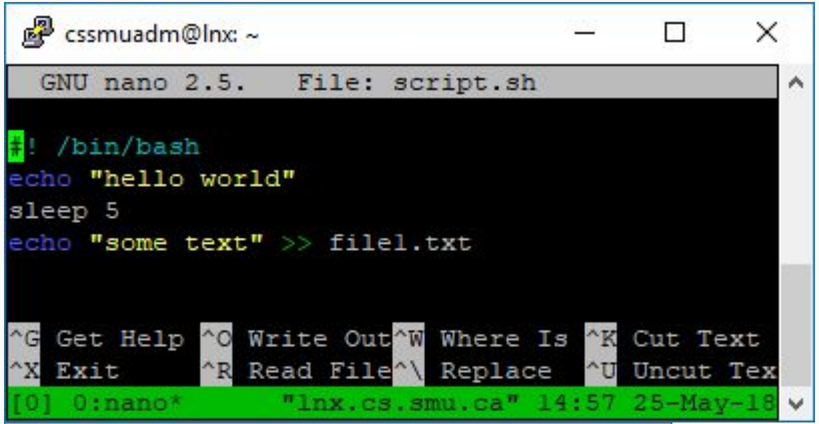
```
  GNU nano 2.   File: program.sh

#!/bin/bash
while true
do
        echo "working..."
        sleep 2
done



^G Get Hel^O Write O^W Where I^K Cut Text
^X Exit   ^R Read Fi^\ Replace^U Uncut Text
[0] <:nano* "lnx.cs.smu.ca" 13:26 25-May-18
```



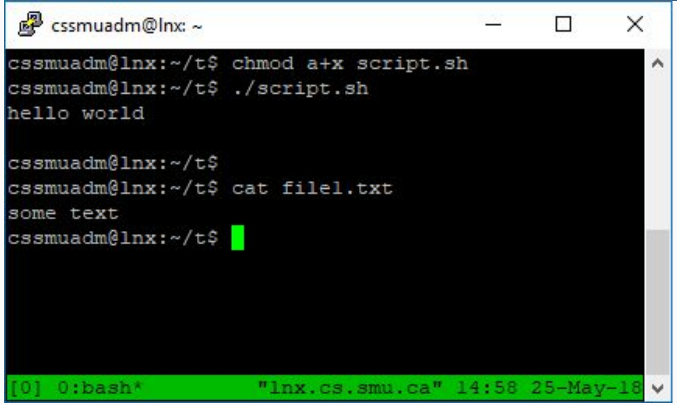
```
cssmuadm@lnx:~$ head -1 program.sh
#!/bin/csh
cssmuadm@lnx:~$ ./program.sh
-bash: ./program.sh: /bin/csh: bad interpreter: No such
file or directory
cssmuadm@lnx:~$

[0] 0:bash*              "lnx.cs.smu.ca" 13:43 25-May-18
```

Shell Scripts: the first bash script

- You can use any command in script that you use in shell with pipes and redirections if necessary
- Write commands (or groups of commands) line by line, no semicolons are necessary in the end of each line
- sleep command is used to make a pause in script execution (in seconds)



```
#! /bin/bash
echo "hello world"
sleep 5
echo "some text" >> file1.txt
```



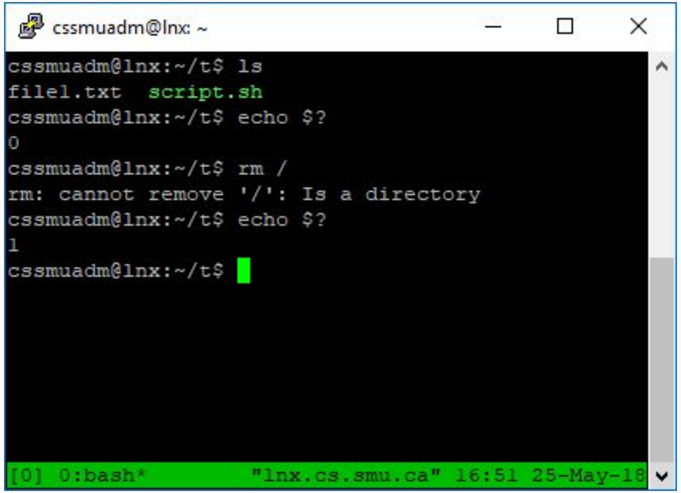
```
cssmuadm@lnx:~/t$ chmod a+x script.sh
cssmuadm@lnx:~/t$ ./script.sh
hello world

cssmuadm@lnx:~/t$
cssmuadm@lnx:~/t$ cat file1.txt
some text
cssmuadm@lnx:~/t$
```

Exercise

Create a simple script that writes current date and time in a file (**date** command), sleeps for 3 seconds and then displays contents of the file to standard output. Execute the script

Exit codes

- Each program gives an **exit code** upon completion
- $? Is used to get exit code of the last executed statement
- 0 exit code usually means that program exited without errors, other than 0 usually indicate errors



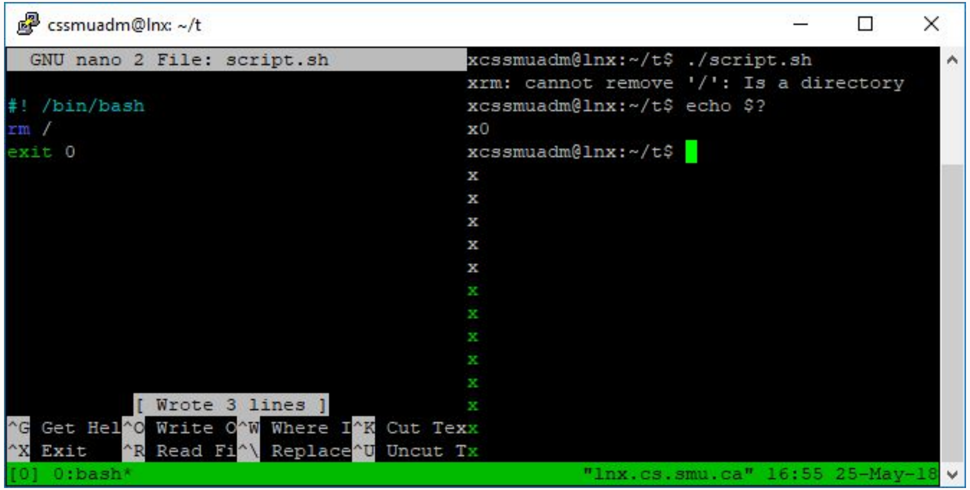
```
cssmuadm@lnx:~/t$ ls
file1.txt  script.sh
cssmuadm@lnx:~/t$ echo $?
0
cssmuadm@lnx:~/t$ rm /
rm: cannot remove '/': Is a directory
cssmuadm@lnx:~/t$ echo $?
1
cssmuadm@lnx:~/t$
[0] 0:bash*        "lnx.cs.smu.ca" 16:51 25-May-18
```

Exit codes (cont-d)

You can interrupt your script with

exit N

command where N is exit code. exit 0 is success, exit 1 (or other number) is error



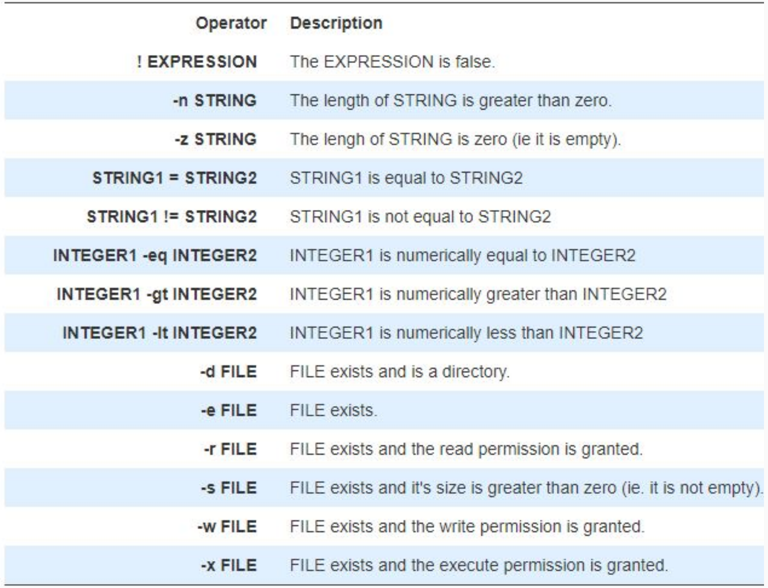
```
 GNU nano 2 File: script.sh             xcssmuadm@lnx:~/t$ ./script.sh
                                        xrm: cannot remove '/': Is a directory
#! /bin/bash                            xcssmuadm@lnx:~/t$ echo $?
rm /                                    x0
exit 0                                  xcssmuadm@lnx:~/t$
                                        x
                                        x
                                        x
                                        x
                                        x
                                        x
                                        x
                                        x
                                        x
                                        x
           [ Wrote 3 lines ]            x
^G Get Hel^O Write O^W Where I^K Cut Texx
^X Exit    ^R Read Fi^\ Replace^U Uncut Tx
[0] 0:bash*                                       "lnx.cs.smu.ca" 16:55 25-May-18
```

If statement

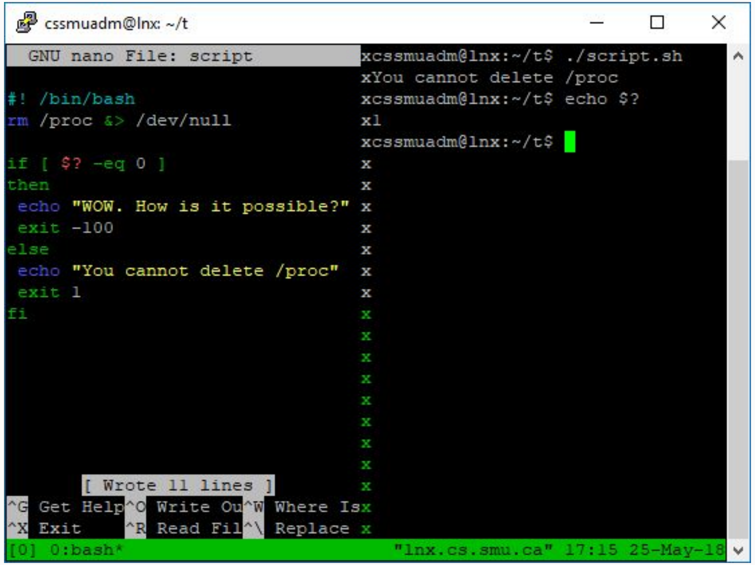
```
if [ test ]
then
        command1
        command2
fi
```

```
if [ test ]
then
        command1
        command2
else
        command3
fi
```

Some tests:



Operator	Description
! EXPRESSION	The EXPRESSION is false.
-n STRING	The length of STRING is greater than zero.
-z STRING	The lengh of STRING is zero (ie it is empty).
STRING1 = STRING2	STRING1 is equal to STRING2
STRING1 != STRING2	STRING1 is not equal to STRING2
INTEGER1 -eq INTEGER2	INTEGER1 is numerically equal to INTEGER2
INTEGER1 -gt INTEGER2	INTEGER1 is numerically greater than INTEGER2
INTEGER1 -lt INTEGER2	INTEGER1 is numerically less than INTEGER2
-d FILE	FILE exists and is a directory.
-e FILE	FILE exists.
-r FILE	FILE exists and the read permission is granted.
-s FILE	FILE exists and it's size is greater than zero (ie. it is not empty).
-w FILE	FILE exists and the write permission is granted.
-x FILE	FILE exists and the execute permission is granted.



cssmuadm@lnx: ~/t

```
  GNU nano File: script          xcssmuadm@lnx:~/t$ ./script.sh
                                 xYou cannot delete /proc
#! /bin/bash                     xcssmuadm@lnx:~/t$ echo $?
rm /proc &> /dev/null            x1
                                 xcssmuadm@lnx:~/t$
if [ $? -eq 0 ]                  x
then                             x
 echo "WOW. How is it possible?" x
 exit -100                       x
else                             x
 echo "You cannot delete /proc"  x
 exit 1                          x
fi                               x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
       [ Wrote 11 lines ]        x
^G Get Help^O Write Ou^W Where Isx
^X Exit    ^R Read Fil^\ Replace x
[0] 0:bash*                        "lnx.cs.smu.ca" 17:15 25-May-18
```

1

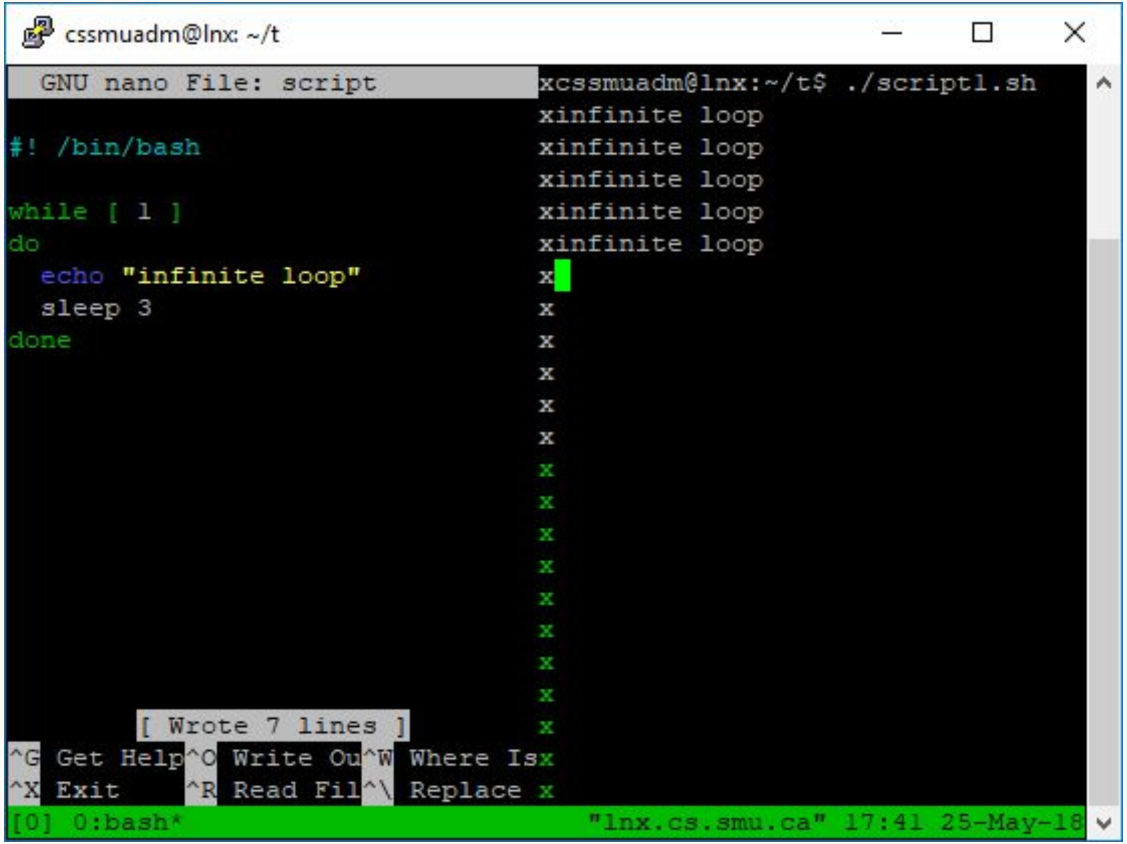
If statement (cont-d)

```
if [ test1 ] && [ test2 ]
then
        command1
        command2
fi
```

```
if [ test1 ] || [ test2 ]
then
        command1
        command2
fi
```

While loop

```
while [ test  ]
do
        command1
        command2
done
```



cssmuadm@lnx: ~/t

```
 GNU nano File: script           xcssmuadm@lnx:~/t$ ./script1.sh
                                 xinfinite loop
#! /bin/bash                     xinfinite loop
                                 xinfinite loop
while [ 1 ]                      xinfinite loop
do                               xinfinite loop
  echo "infinite loop"           x
  sleep 3                        x
done                             x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
                                 x
        [ Wrote 7 lines ]        x
^G Get Help^O Write Ou^W Where Isx
^X Exit    ^R Read Fil^\ Replace x
[0] 0:bash*                                  "lnx.cs.smu.ca" 17:41 25-May-18
```

Variables

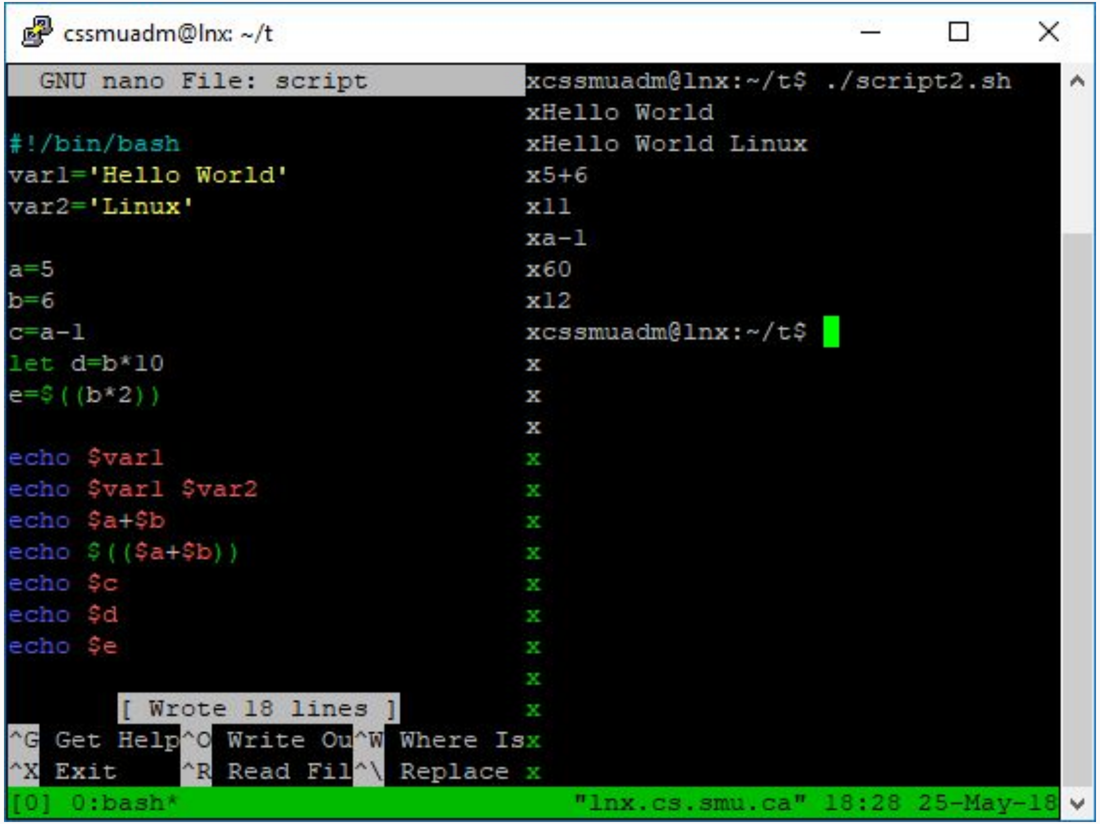
var1 = value_a
var2 = value_b

echo $var1 $var2

Numeric evaluations:

- let a=b+c
- a=$((b+c))

note: variables are untyped



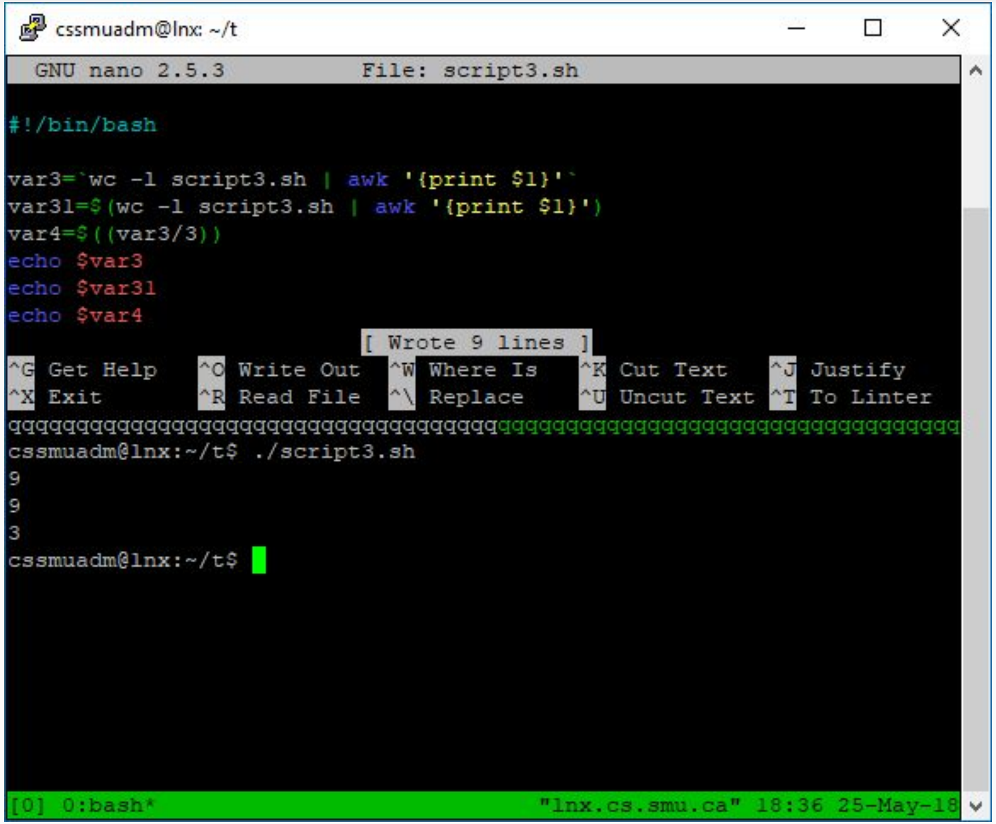
```
  GNU nano File: script          xcssmuadm@lnx:~/t$ ./script2.sh
                                 xHello World
#!/bin/bash                      xHello World Linux
var1='Hello World'               x5+6
var2='Linux'                     x11
                                 xa-1
a=5                              x60
b=6                              x12
c=a-1                            xcssmuadm@lnx:~/t$
let d=b*10                       x
e=$((b*2))                       x
                                 x
echo $var1                       x
echo $var1 $var2                 x
echo $a+$b                       x
echo $(($a+$b))                  x
echo $c                          x
echo $d                          x
echo $e                          x
                                 x
       [ Wrote 18 lines ]        x
^G Get Help^O Write Ou^W Where Isx
^X Exit    ^R Read Fil^\ Replace x
[0] 0:bash*                         "lnx.cs.smu.ca" 18:28 25-May-18
```

Variables (cont-d)

To put result of some command into variable use:

- var=`command arg`
- var=$(command arg)

Both `cmd` and $(cmd) do the same thing with different syntax



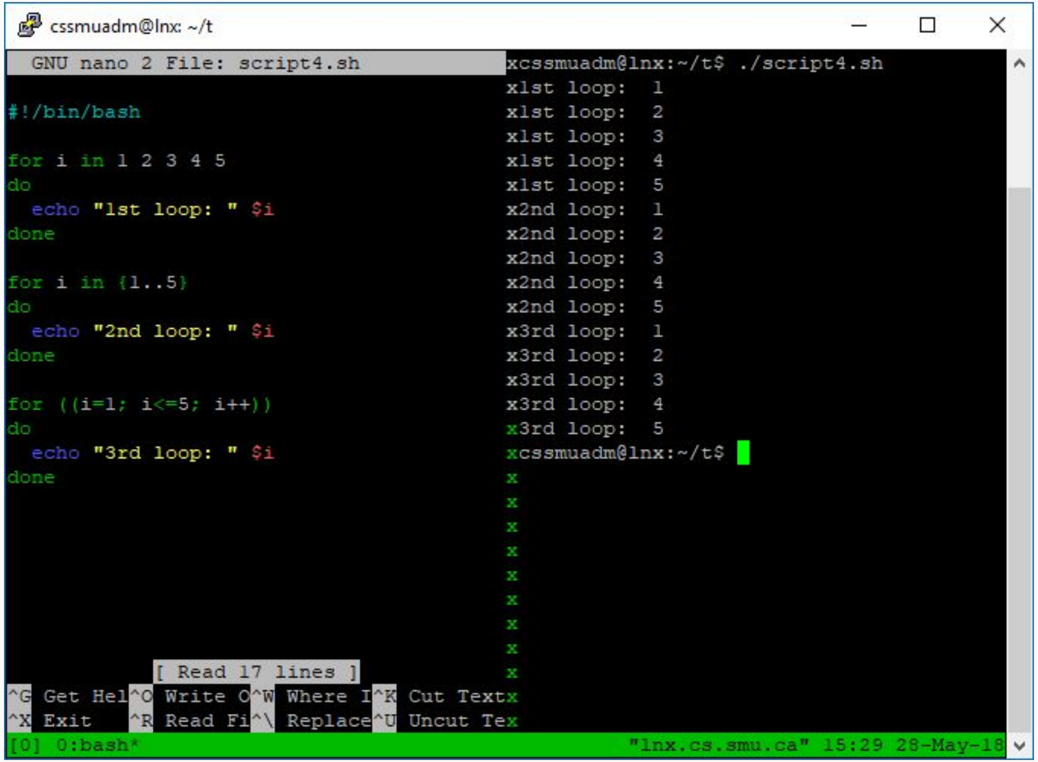
```
  GNU nano 2.5.3                File: script3.sh

#!/bin/bash

var3=`wc -l script3.sh | awk '{print $1}'`
var31=$(wc -l script3.sh | awk '{print $1}')
var4=$((var3/3))
echo $var3
echo $var31
echo $var4
                                   [ Wrote 9 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Linter
qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqq
cssmuadm@lnx:~/t$ ./script3.sh
9
9
3
cssmuadm@lnx:~/t$

[0] 0:bash*                               "lnx.cs.smu.ca" 18:36 25-May-18
```

For loops

```
for i in 1 2 3 4 5
do
        echo $i
        command1
done
```

```
for i in {1..5}
do
        echo $i
        command2
done
```

```
for ((i=1; i<=5; i++))
do
        echo $i
        command3
done
```



```
  GNU nano 2 File: script4.sh

#!/bin/bash

for i in 1 2 3 4 5
do
  echo "1st loop: " $i
done

for i in {1..5}
do
  echo "2nd loop: " $i
done

for ((i=1; i<=5; i++))
do
  echo "3rd loop: " $i
done

                   [ Read 17 lines ]
^G Get Hel^O Write O^W Where I^K Cut Text
^X Exit   ^R Read Fi^\ Replace^U Uncut Tex
```

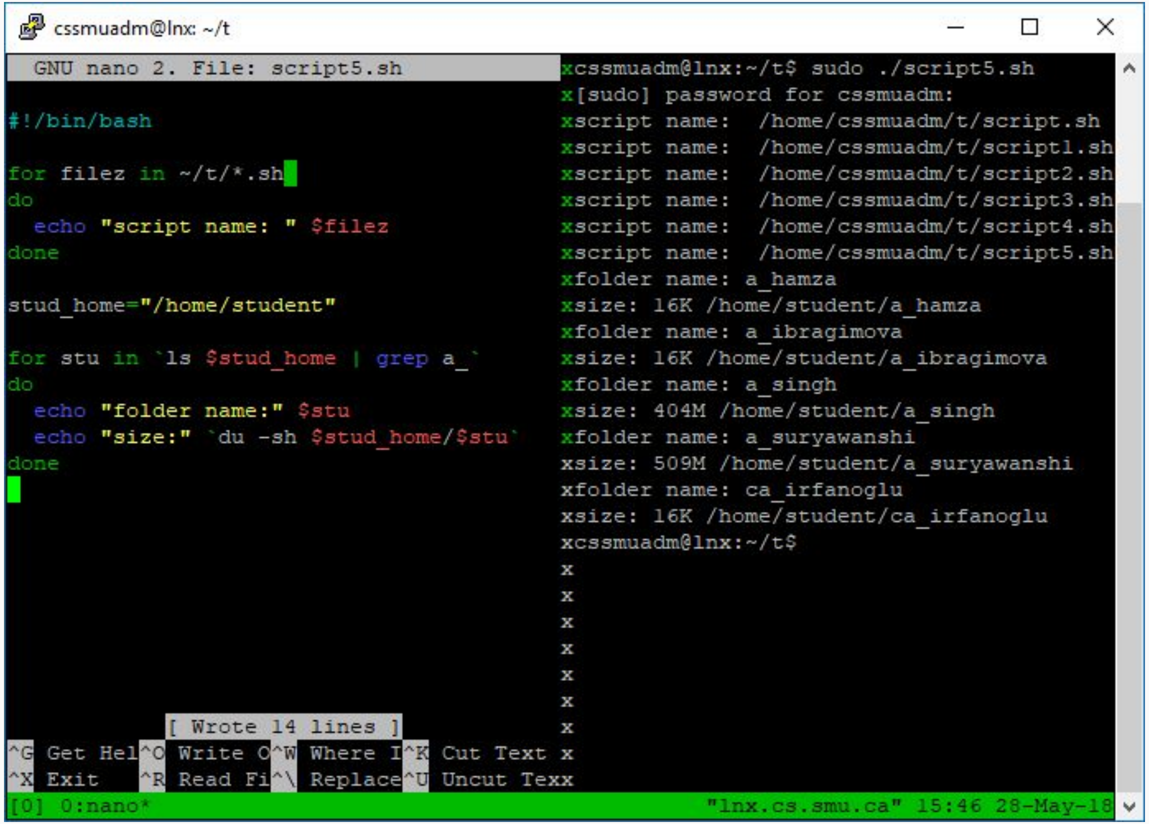
```
xcssmuadm@lnx:~/t$ ./script4.sh
x1st loop:  1
x1st loop:  2
x1st loop:  3
x1st loop:  4
x1st loop:  5
x2nd loop:  1
x2nd loop:  2
x2nd loop:  3
x2nd loop:  4
x2nd loop:  5
x3rd loop:  1
x3rd loop:  2
x3rd loop:  3
x3rd loop:  4
x3rd loop:  5
xcssmuadm@lnx:~/t$
```

```
[0] 0:bash*                    "lnx.cs.smu.ca" 15:29 28-May-18
```

For loops (cont-d)

```
for f in ~/*.sh
do
        echo $f
        command1
done
```

```
for i in `command`
do
        echo $i
        command2
done
```



```
  GNU nano 2. File: script5.sh

#!/bin/bash

for filez in ~/t/*.sh
do
  echo "script name: " $filez
done

stud_home="/home/student"

for stu in `ls $stud_home | grep a_`
do
  echo "folder name:" $stu
  echo "size:" `du -sh $stud_home/$stu`
done



                 [ Wrote 14 lines ]
^G Get Hel^O Write O^W Where I^K Cut Text
^X Exit   ^R Read Fi^\ Replace^U Uncut Texx
```

```
xcssmuadm@lnx:~/t$ sudo ./script5.sh
x[sudo] password for cssmuadm:
xscript name:  /home/cssmuadm/t/script.sh
xscript name:  /home/cssmuadm/t/script1.sh
xscript name:  /home/cssmuadm/t/script2.sh
xscript name:  /home/cssmuadm/t/script3.sh
xscript name:  /home/cssmuadm/t/script4.sh
xscript name:  /home/cssmuadm/t/script5.sh
xfolder name: a_hamza
xsize: 16K /home/student/a_hamza
xfolder name: a_ibragimova
xsize: 16K /home/student/a_ibragimova
xfolder name: a_singh
xsize: 404M /home/student/a_singh
xfolder name: a_suryawanshi
xsize: 509M /home/student/a_suryawanshi
xfolder name: ca_irfanoglu
xsize: 16K /home/student/ca_irfanoglu
xcssmuadm@lnx:~/t$
x
x
x
x
x
x
x
x
```

```
[0] 0:nano*                                        "lnx.cs.smu.ca" 15:46 28-May-18
```

For loops (cont-d)

NOTES:

- You can break execution of loop with **break** keyword
- Or continue with **continue** keyword (skip the current iteration)

Exercise

Create a script:
for each file in **/bin** directory do:

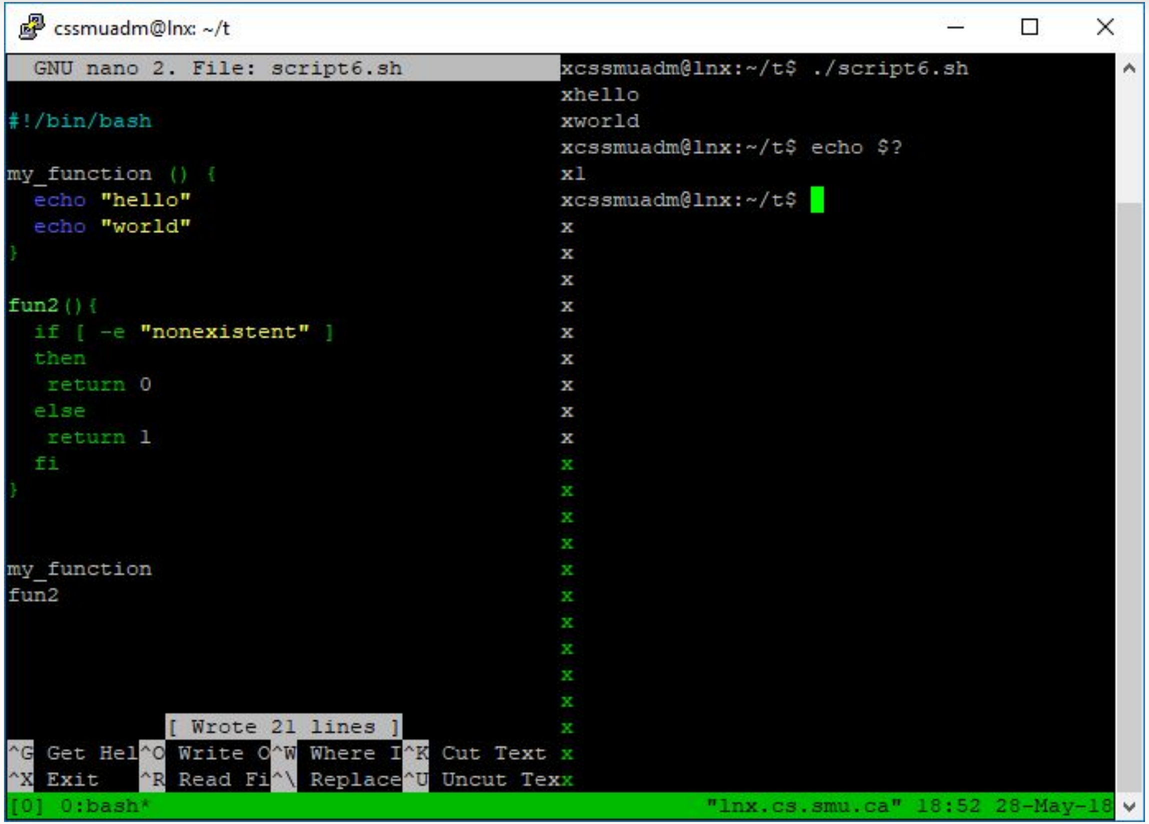
- if name of the file starts with **b** or **z** then append the filename to “**filebz**” file, otherwise append filename to file “**other**” file

Execute the script

Functions

```
my_function () {
        command1
        command2
        function1
}
```

```
my_function2() {
        command1
        command2
        function1
        return ret_val
}
```



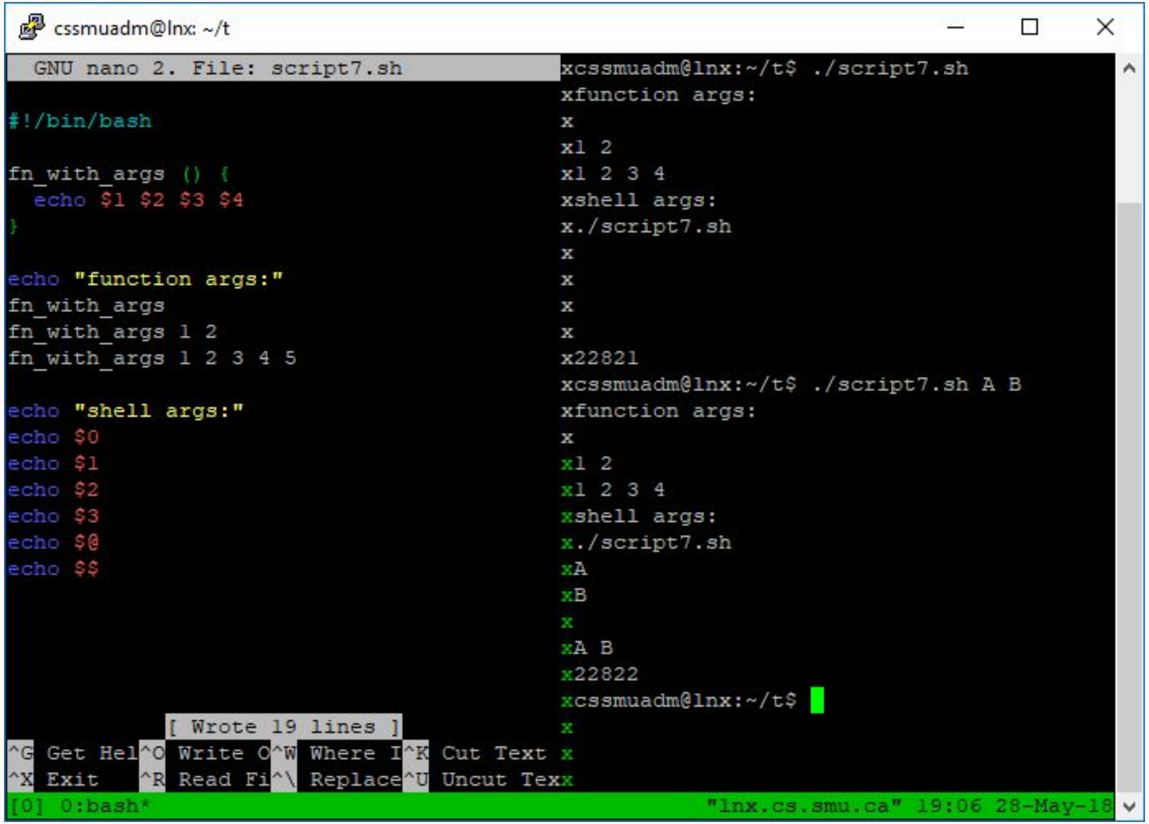
```
  GNU nano 2. File: script6.sh

#!/bin/bash

my_function () {
  echo "hello"
  echo "world"
}

fun2 () {
  if [ -e "nonexistent" ]
  then
   return 0
  else
   return 1
  fi
}


my_function
fun2




            [ Wrote 21 lines ]
^G Get Hel^O Write O^W Where I^K Cut Text
^X Exit   ^R Read Fi^\ Replace^U Uncut Tex
```

```
xcssmuadm@lnx:~/t$ ./script6.sh
xhello
xworld
xcssmuadm@lnx:~/t$ echo $?
x1
xcssmuadm@lnx:~/t$ 
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
x
```

```
[0] 0:bash*                              "lnx.cs.smu.ca" 18:52 28-May-18
```

Arguments

- **$1, $2, $3 -** Nth arguments
- **$@ -** all arguments
- **$# -** the number of args
- **$$ -** PID of the current shell
- **$0 -** name of the shell or shell script.
- **$? -** most recent exit code




```
  GNU nano 2. File: script7.sh

#!/bin/bash

fn_with_args () {
  echo $1 $2 $3 $4
}

echo "function args:"
fn_with_args
fn_with_args 1 2
fn_with_args 1 2 3 4 5

echo "shell args:"
echo $0
echo $1
echo $2
echo $3
echo $@
echo $$

            [ Wrote 19 lines ]
^G Get Hel^O Write O^W Where I^K Cut Text
^X Exit   ^R Read Fi^\ Replace^U Uncut Tex
```

```
xcssmuadm@lnx:~/t$ ./script7.sh
xfunction args:
x
x1 2
x1 2 3 4
xshell args:
x./script7.sh
x
x
x
x
x22821
xcssmuadm@lnx:~/t$ ./script7.sh A B
xfunction args:
x
x1 2
x1 2 3 4
xshell args:
x./script7.sh
xA
xB
x
xA B
x22822
xcssmuadm@lnx:~/t$
x
x
x
```

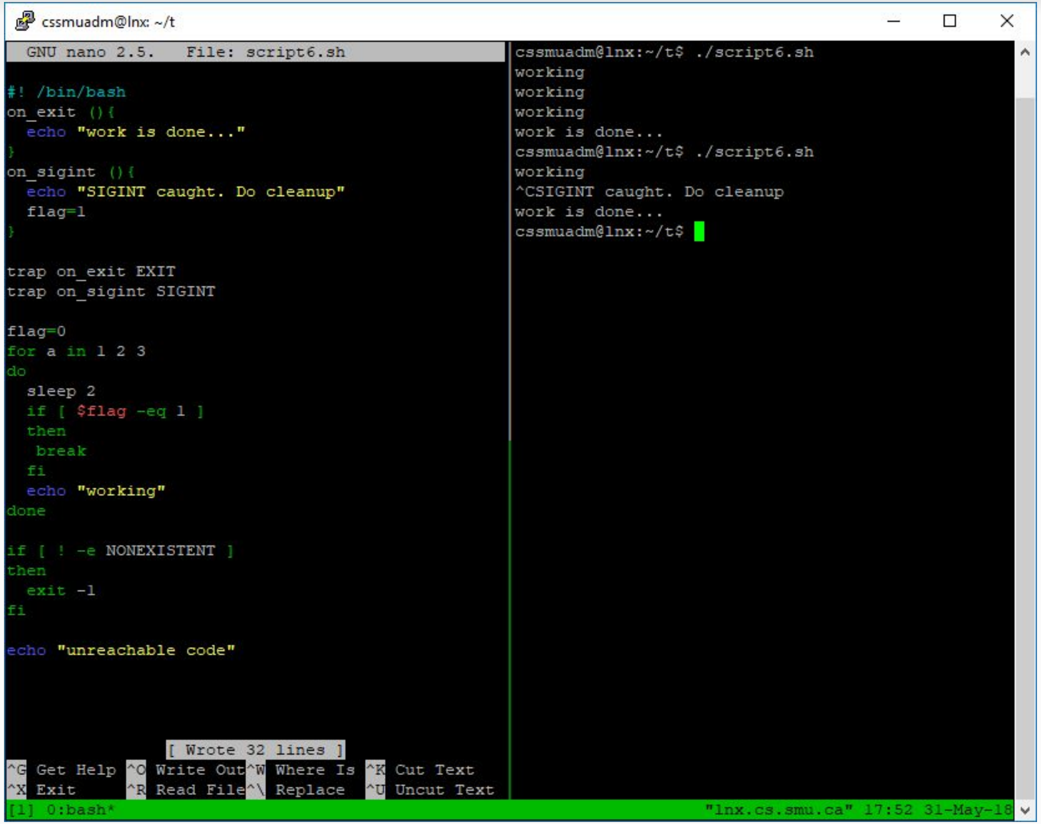
```
[0] 0:bash*                                          "lnx.cs.smu.ca" 19:06 28-May-18
```

Traps

```
#!/bin/bash

on_exit (){
 #some cleanup code

}
trap on_exit EXIT
```

NOTES:

- With traps you can also handle **signals**
- Traps should be put in the **beginning** of a script before code



```
  GNU nano 2.5.    File: script6.sh

#! /bin/bash
on_exit (){
  echo "work is done..."
}
on_sigint (){
  echo "SIGINT caught. Do cleanup"
  flag=1
}

trap on_exit EXIT
trap on_sigint SIGINT

flag=0
for a in 1 2 3
do
  sleep 2
  if [ $flag -eq 1 ]
  then
   break
  fi
  echo "working"
done

if [ ! -e NONEXISTENT ]
then
  exit -1
fi

echo "unreachable code"

                [ Wrote 32 lines ]
^G Get Help  ^O Write Out^W Where Is  ^K Cut Text
^X Exit      ^R Read File^\ Replace   ^U Uncut Text
```

```
cssmuadm@lnx:~/t$ ./script6.sh
working
working
working
work is done...
cssmuadm@lnx:~/t$ ./script6.sh
working
^CSIGINT caught. Do cleanup
work is done...
cssmuadm@lnx:~/t$
```

```
[1] 0:bash*                                                "lnx.cs.smu.ca" 17:52 31-May-18
```

Exercise

Create a script:

Write two functions:

- the first one sets up an **SSH tunnel** for MySQL@dev server
- The second one “kills” the SSH tunnel
- Try to use a trap to call the second function on exit

Execute it