

SUMMER SCHOOL 2021

Bash Scripting Part 4

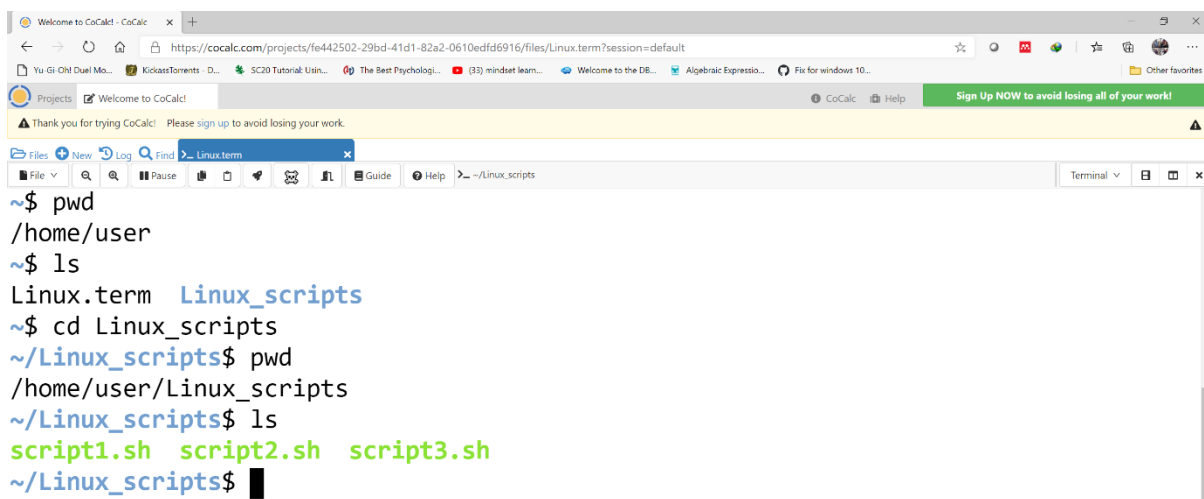
Dr Krishna K. Govender

Summary of commands used in this lecture

Command	Meaning
ls	List items in current directory
nano	Command line text editor in Linux
Ctrl+X > Y > Enter	Sequence used when saving changes to a file created/edited in nano
cat	Concatenate a file/s
chmod	Modify file permissions
#!/bin/bash	Shebang
echo	Print text to screen
\$var	Print the contents of the variable var
for var in XX; do ...; done	Perform an iterative task

- As in the previous lectures all commands provided in **BOLD** and center aligned from here onwards are commands that can be typed/executed directly in a terminal.
- If you are in your home directory first make sure you change directory to the Linux_scripts before continuing:

```
pwd
ls
cd Linux_scripts
pwd
ls
```

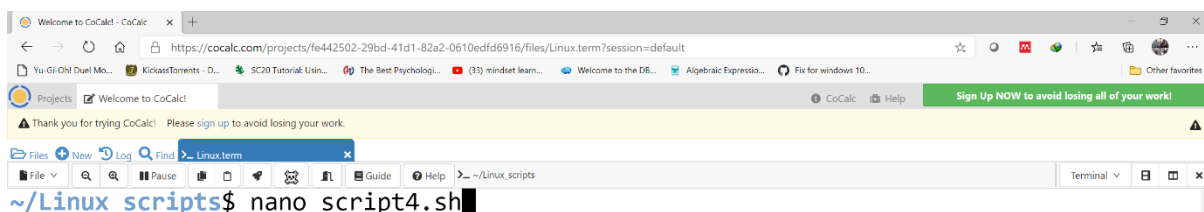


The screenshot shows a web browser window with a CoCalc terminal. The terminal output is as follows:

```
~$ pwd
/home/user
~$ ls
Linux.term  Linux_scripts
~$ cd Linux_scripts
~/Linux_scripts$ pwd
/home/user/Linux_scripts
~/Linux_scripts$ ls
script1.sh  script2.sh  script3.sh
~/Linux_scripts$
```

- You should see the scripts from the previous lectures and if you completed **Exercise 9** you will also have a file called number.txt contained in your directory.
- You can have a loop that continues until a value specified by a user:

nano script4.sh



The screenshot shows the same CoCalc terminal window. The command `nano script4.sh` has been entered at the prompt, and the cursor is at the end of the command.

```
~/Linux_scripts$ nano script4.sh
```

```
GNU nano 4.8 script4.sh
#!/bin/bash

# This script will iterate up to a value requested by the user
echo 'How many lines would you like'
read line
for((outer=1;outer<=$line;outer++))
do
    echo $outer
done
```

[Read 9 lines]

^G Get Help	^O Write Out	^W Where Is	^K Cut Text	^J Justify	^C Cur Pos
^X Exit	^R Read File	^\ Replace	^U Paste Text	^T To Spell	^_ Go To Line

- The loop in this case has a variable called **outer** which starts with a value of 1 and it continues until the variable **outer** is either less then or equal to the variable **line** (the value obtained from the users' input).

The last part in the for statement increments the value of **outer** by 1.

- Use **Ctrl+X > Y > Enter** to exit **nano**, save the changes and end up back on the terminal.

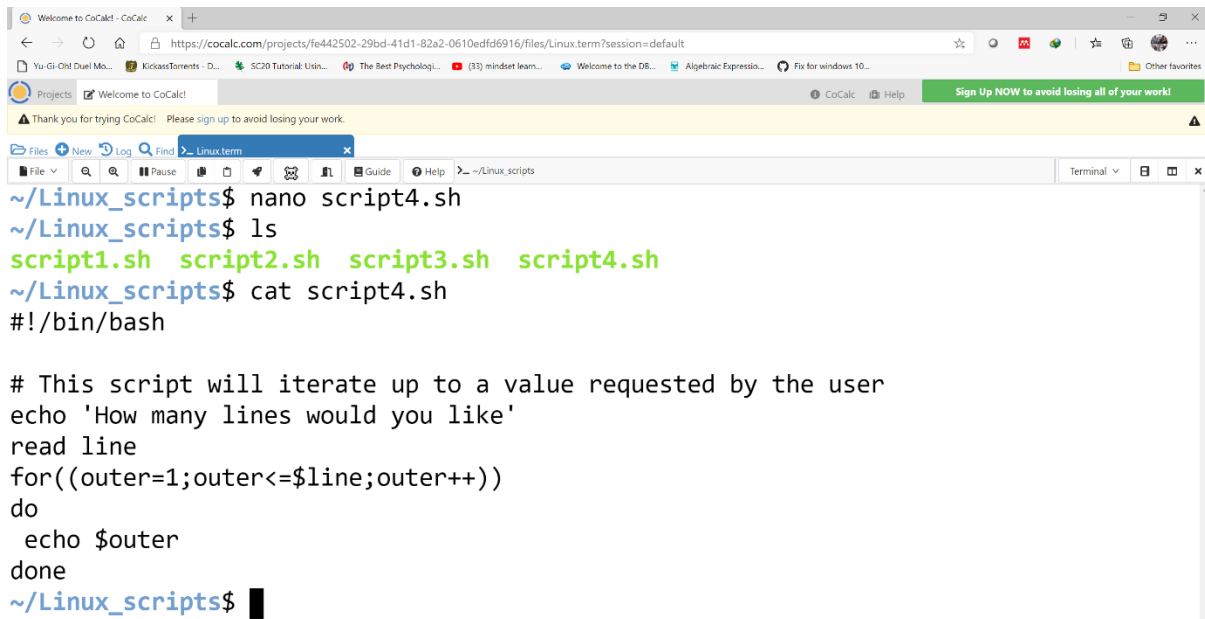
Check that the file exists, that it has the information we require and make it executable:

ls

cat script4.sh

chmod u+x script4.sh

Note the double brackets used in the for loop as this is required when creating a loop where you declare a variable and increment that variable by 1 all in the same statement.

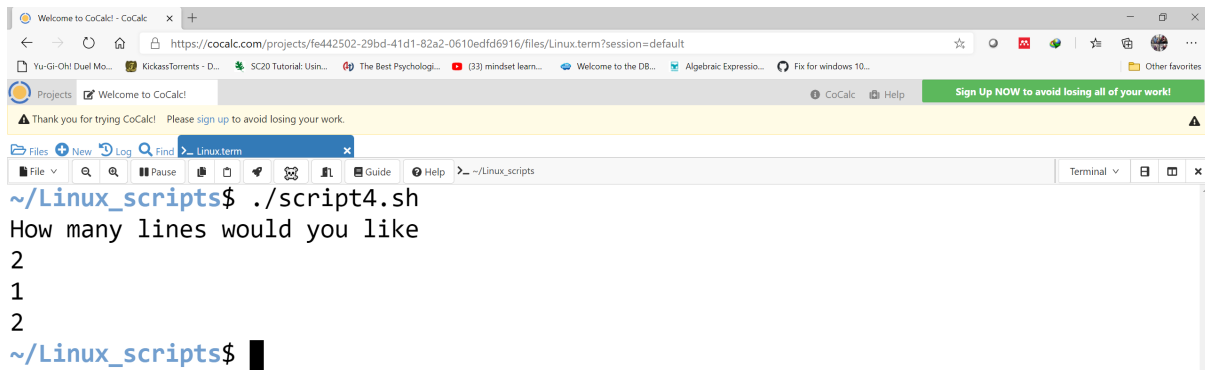


```
~/Linux_scripts$ nano script4.sh
~/Linux_scripts$ ls
script1.sh script2.sh script3.sh script4.sh
~/Linux_scripts$ cat script4.sh
#!/bin/bash

# This script will iterate up to a value requested by the user
echo 'How many lines would you like'
read line
for((outer=1;outer<=$line;outer++))
do
    echo $outer
done
~/Linux_scripts$
```

- Now we will run this script to understand how the loop works:

./script4.sh



```
~/Linux_scripts$ ./script4.sh
How many lines would you like
2
1
2
~/Linux_scripts$
```

- I entered a value of 2, so when the script gets to the loop it starts with a value of **outer** equal to 1. This is less than or equal to **line** (which is the value of 2 that I entered) so the **do** statement gets executed and we end up with the value 1 being printed to screen. **Outer** is then incremented by 1 to become 2, which is also less than or equal to **line** and once again the **do** statement gets executed this time printing the value of 2. **Outer** gets incremented by 1 again to become 3, which is NOT less than or equal to **line** causing the loop to stop executing.
- This brings us to the end of **Bash Scripting Part 4**.
- At this point you should try to expand on the script above by completing **Exercise 10**.