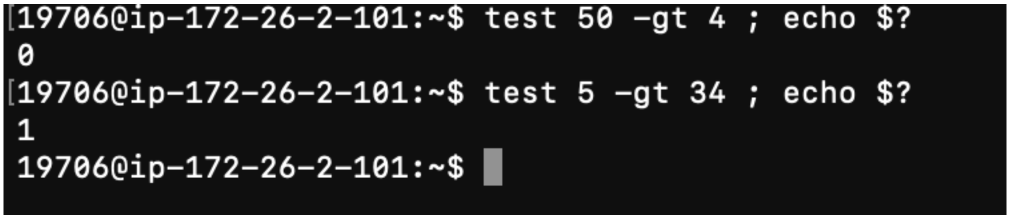
Go to this website and finish all the steps in Tutorial Eight. This tutorial talks about variables and path.
http://www.ee.surrey.ac.uk/Teaching/Unix/unix8.html

⇨ Webminal_ScriptingLoops

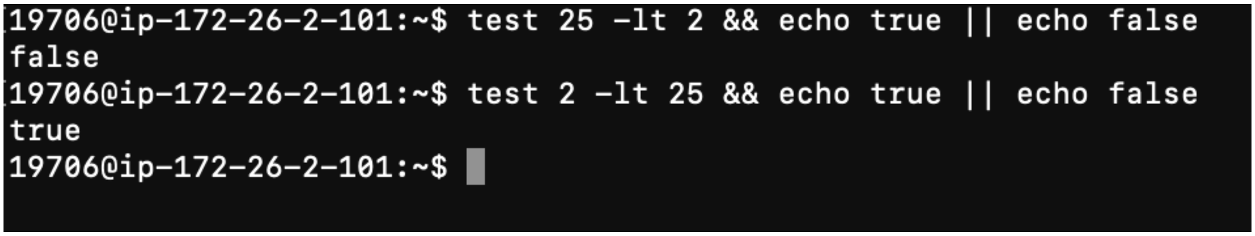
1. Test [] :
   It can test whether the command is true or false. If the output you provide is actual, it gives you output as 0; if it is wrong, it will supply output as 1.

   a) The first example is whether the first number is greater than the second.



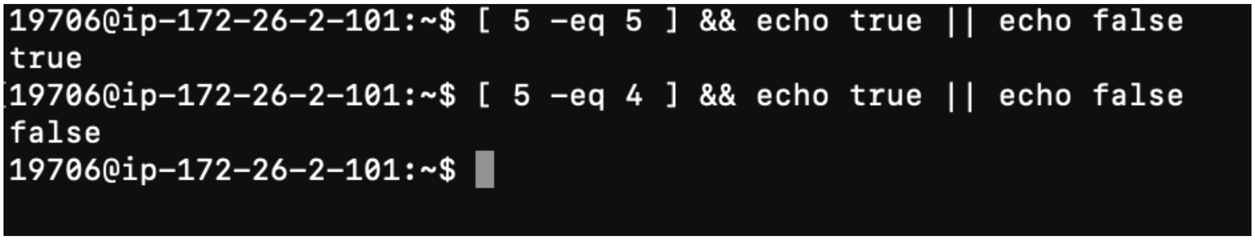
```
[19706@ip-172-26-2-101:~$ test 50 -gt 4 ; echo $?
0
[19706@ip-172-26-2-101:~$ test 5 -gt 34 ; echo $?
1
19706@ip-172-26-2-101:~$ 
```

   b) The second example is whether the first number is less than the second.



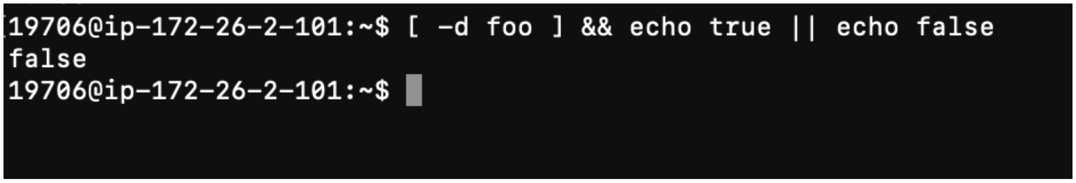
```
19706@ip-172-26-2-101:~$ test 25 -lt 2 && echo true || echo false
false
[19706@ip-172-26-2-101:~$ test 2 -lt 25 && echo true || echo false
true
19706@ip-172-26-2-101:~$ 
```

   c) The third example is whether the first number is equal to the second.



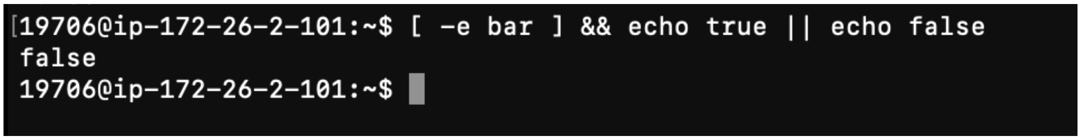
```
19706@ip-172-26-2-101:~$ [ 5 -eq 5 ] && echo true || echo false
true
19706@ip-172-26-2-101:~$ [ 5 -eq 4 ] && echo true || echo false
false
19706@ip-172-26-2-101:~$ 
```

d) Does the directory foo exist?



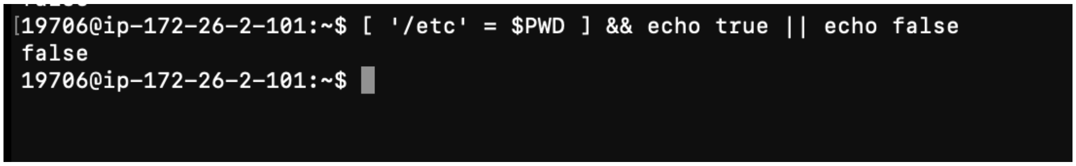
```
19706@ip-172-26-2-101:~$ [ -d foo ] && echo true || echo false
false
19706@ip-172-26-2-101:~$ 
```

e) Does the file bar exist?



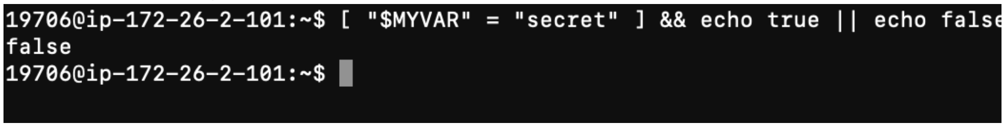
```
[19706@ip-172-26-2-101:~$ [ -e bar ] && echo true || echo false
 false
 19706@ip-172-26-2-101:~$ 
```

f) Is the string / etc. equal to the variable $PWD?



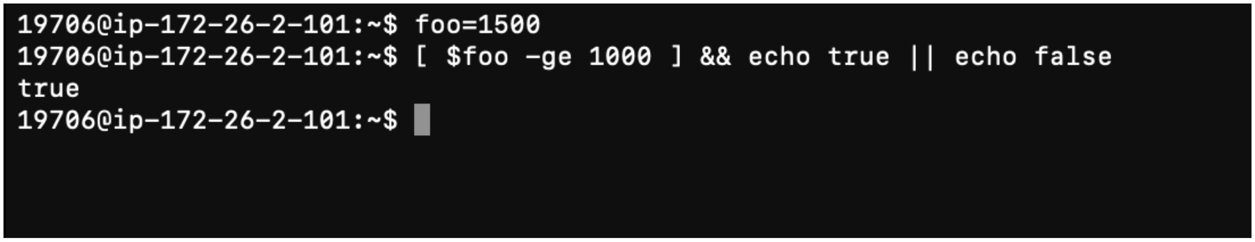
```
[19706@ip-172-26-2-101:~$ [ '/etc' = $PWD ] && echo true || echo false
 false
 19706@ip-172-26-2-101:~$ 
```

g) Is the first parameter different from the secret?



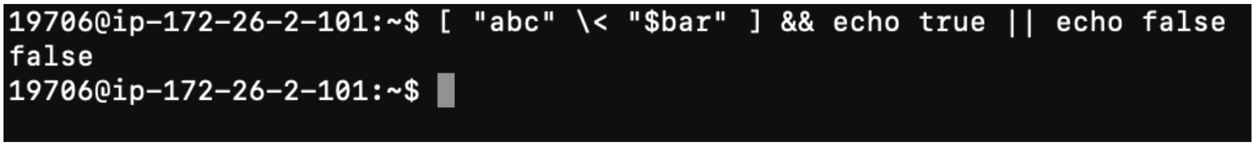
```
19706@ip-172-26-2-101:~$ [ "$MYVAR" = "secret" ] && echo true || echo false
false
19706@ip-172-26-2-101:~$ 
```

h) Is the value of $foo greater or equal to 1000?



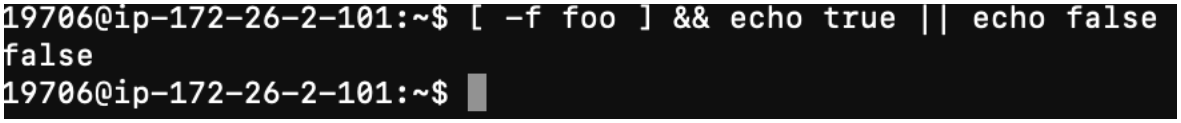
```
19706@ip-172-26-2-101:~$ foo=1500
19706@ip-172-26-2-101:~$ [ $foo -ge 1000 ] && echo true || echo false
true
19706@ip-172-26-2-101:~$ 
```

i) Does abc sort before the value of $bar?



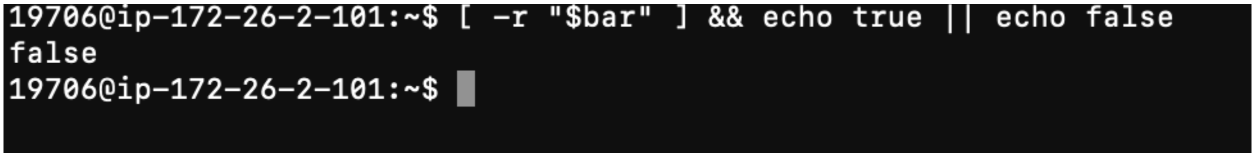
```
19706@ip-172-26-2-101:~$ [ "abc" \< "$bar" ] && echo true || echo false
false
19706@ip-172-26-2-101:~$ 
```

j) Is foo a regular file?



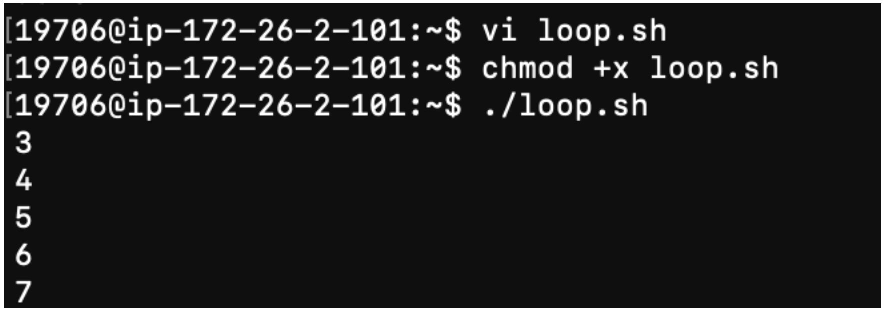
```
19706@ip-172-26-2-101:~$ [ -f foo ] && echo true || echo false
false
19706@ip-172-26-2-101:~$ 
```

k) Is the bar a readable file?

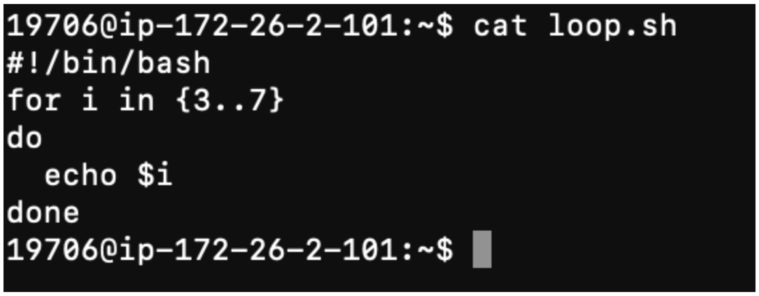


```
19706@ip-172-26-2-101:~$ [ -r "$bar" ] && echo true || echo false
false
19706@ip-172-26-2-101:~$ 
```

2. Write a script that uses a for loop to count from 3 to 7.

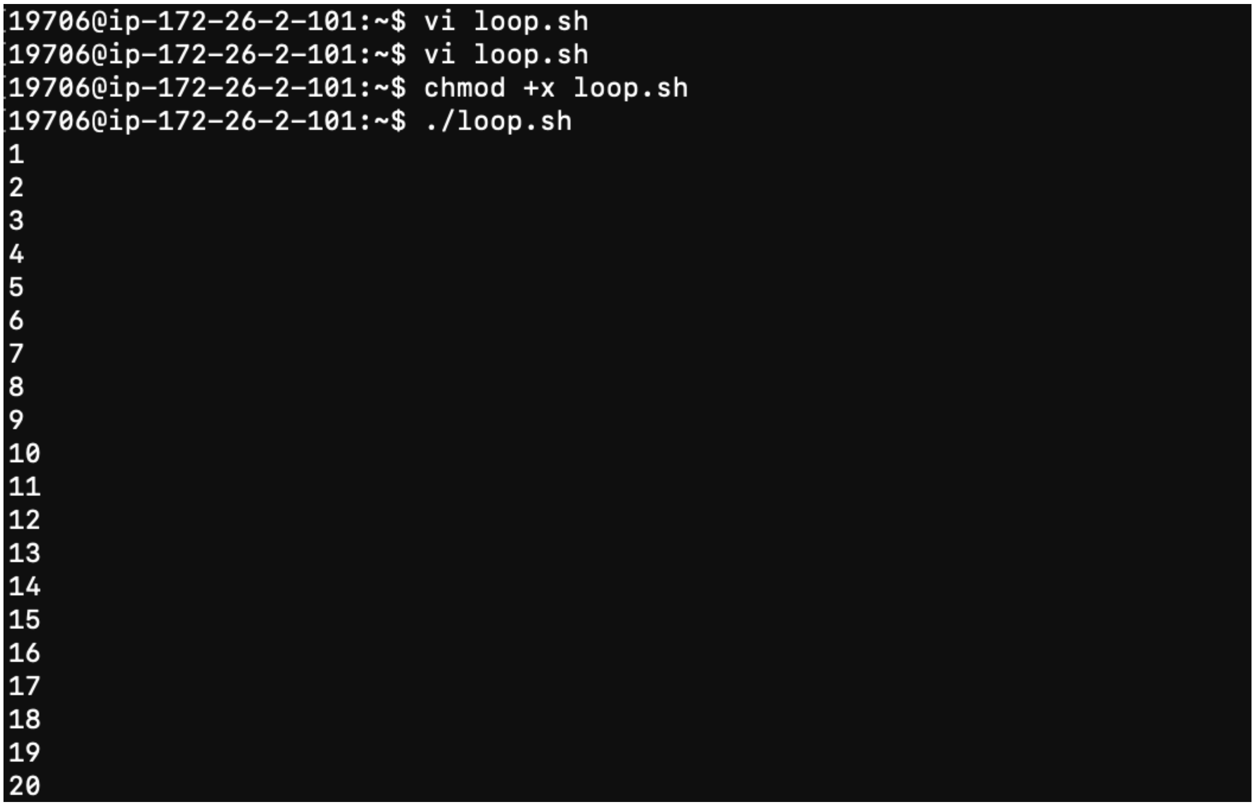


```
[19706@ip-172-26-2-101:~$ vi loop.sh
[19706@ip-172-26-2-101:~$ chmod +x loop.sh
[19706@ip-172-26-2-101:~$ ./loop.sh
 3
 4
 5
 6
 7
```



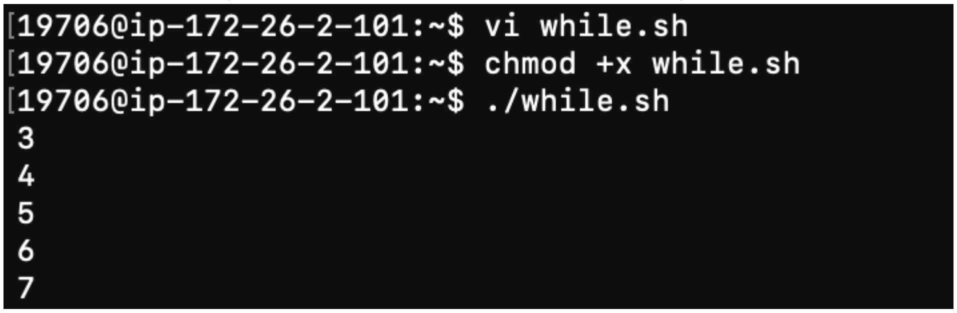
```
19706@ip-172-26-2-101:~$ cat loop.sh
#!/bin/bash
for i in {3..7}
do
  echo $i
done
19706@ip-172-26-2-101:~$ 
```

3. Write a script that uses a for loop to count from 1 to 17000.

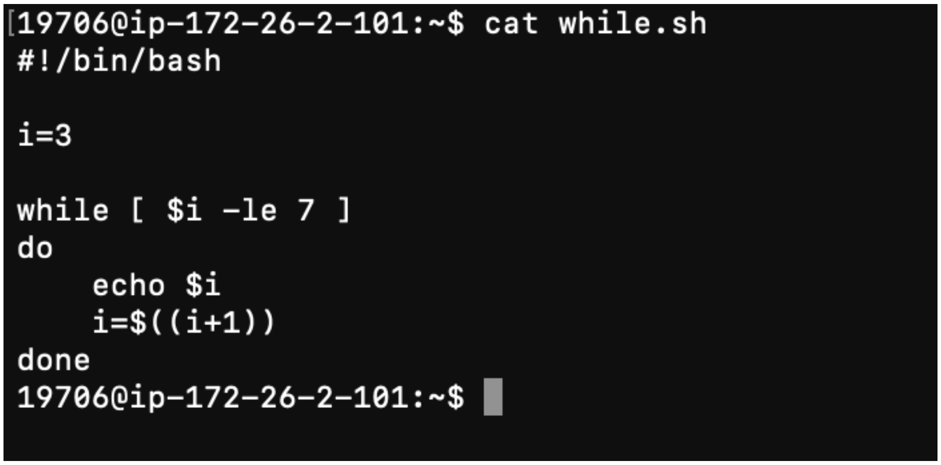


```
[19706@ip-172-26-2-101:~$ vi loop.sh
[19706@ip-172-26-2-101:~$ vi loop.sh
[19706@ip-172-26-2-101:~$ chmod +x loop.sh
[19706@ip-172-26-2-101:~$ ./loop.sh
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

4. Write a script that uses a while loop to count from 3 to 7.

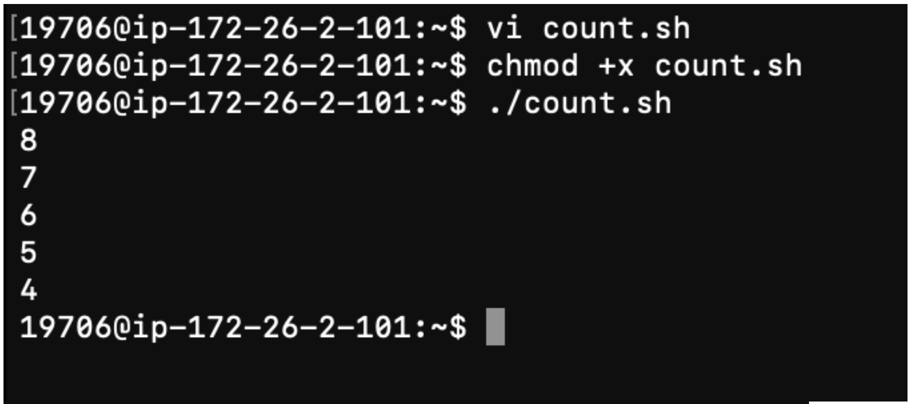


```
[19706@ip-172-26-2-101:~$ vi while.sh
[19706@ip-172-26-2-101:~$ chmod +x while.sh
[19706@ip-172-26-2-101:~$ ./while.sh
 3
 4
 5
 6
 7
```

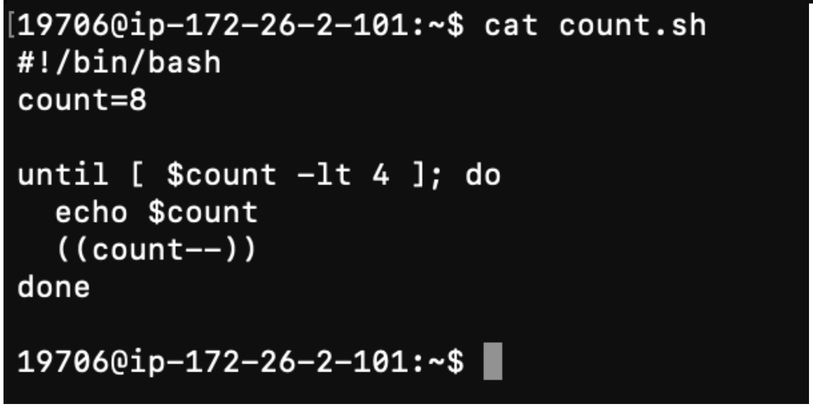


```
[19706@ip-172-26-2-101:~$ cat while.sh
#!/bin/bash

i=3

while [ $i -le 7 ]
do
    echo $i
    i=$((i+1))
done
19706@ip-172-26-2-101:~$ █
```

5. Write a script that uses an until loop to count from 8 to 4.

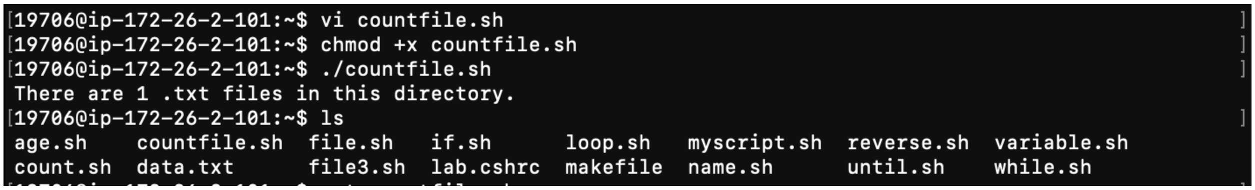


```
[19706@ip-172-26-2-101:~$ vi count.sh
[19706@ip-172-26-2-101:~$ chmod +x count.sh
[19706@ip-172-26-2-101:~$ ./count.sh
8
7
6
5
4
19706@ip-172-26-2-101:~$ █
```

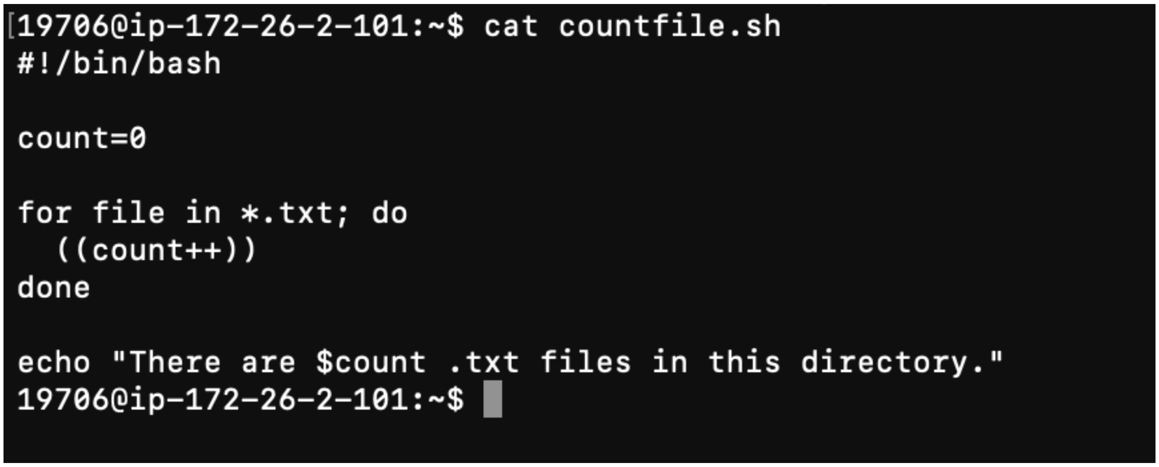


```
[19706@ip-172-26-2-101:~$ cat count.sh
#!/bin/bash
count=8

until [ $count -lt 4 ]; do
  echo $count
  ((count--))
done

19706@ip-172-26-2-101:~$ █
```

6. Write a script that counts the files ending in .txt in the current directory.

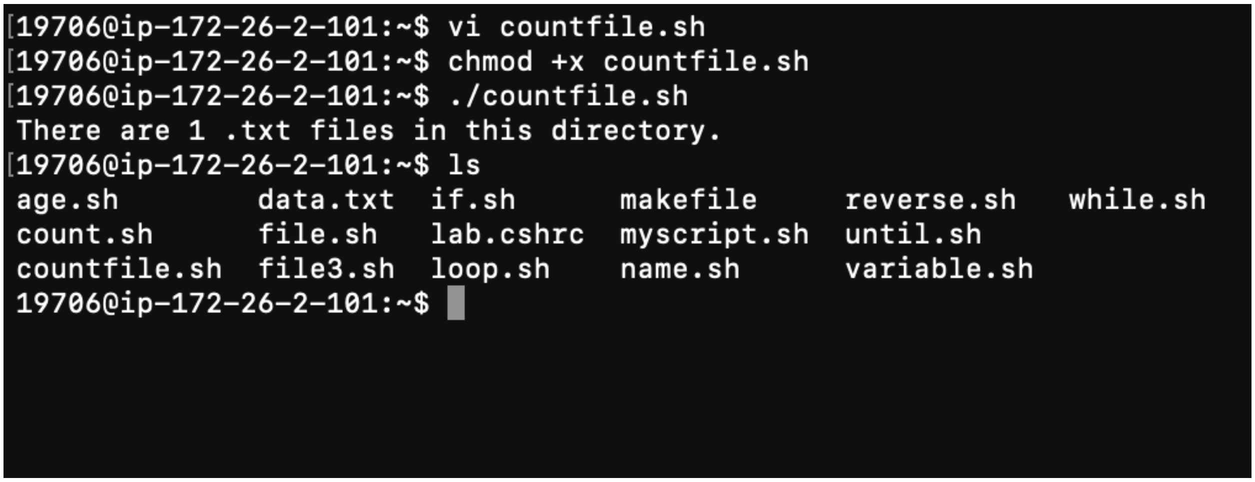


```
[19706@ip-172-26-2-101:~$ vi countfile.sh
[19706@ip-172-26-2-101:~$ chmod +x countfile.sh
[19706@ip-172-26-2-101:~$ ./countfile.sh
There are 1 .txt files in this directory.
[19706@ip-172-26-2-101:~$ ls
age.sh    countfile.sh  file.sh   if.sh      loop.sh   myscript.sh  reverse.sh  variable.sh
count.sh  data.txt      file3.sh  lab.cshrc  makefile  name.sh      until.sh    while.sh
```

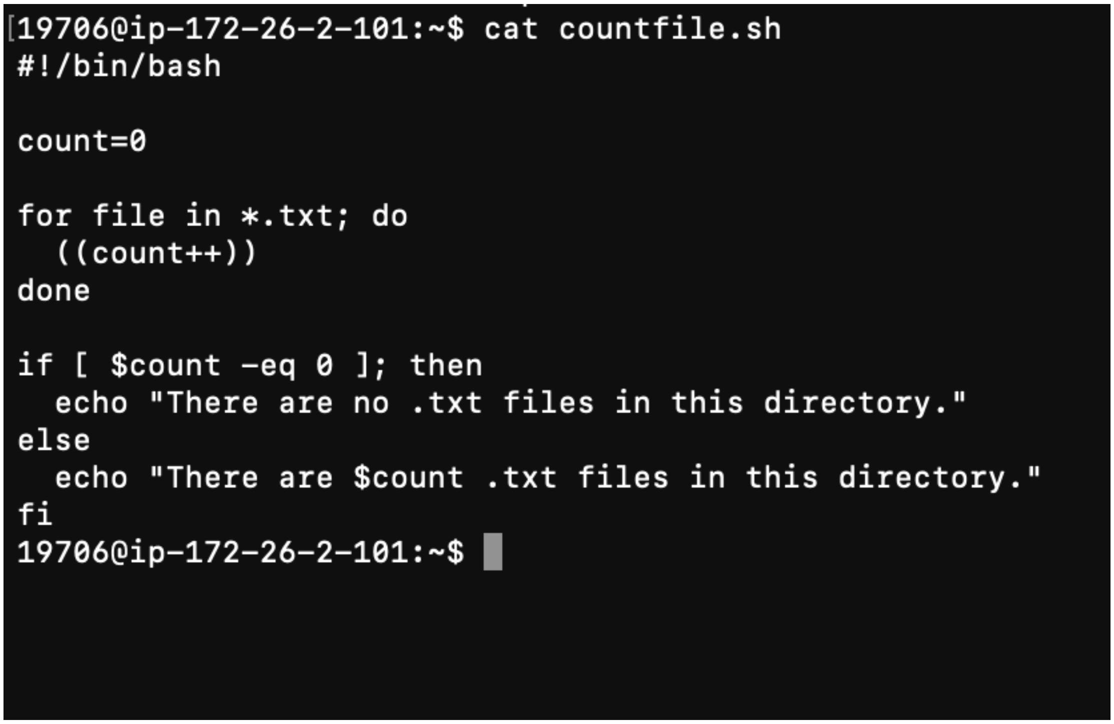


```
[19706@ip-172-26-2-101:~$ cat countfile.sh
#!/bin/bash

count=0

for file in *.txt; do
  ((count++))
done

echo "There are $count .txt files in this directory."
19706@ip-172-26-2-101:~$ 
```

7. Wrap an if statement around the script to correct when zero files end in.txt.



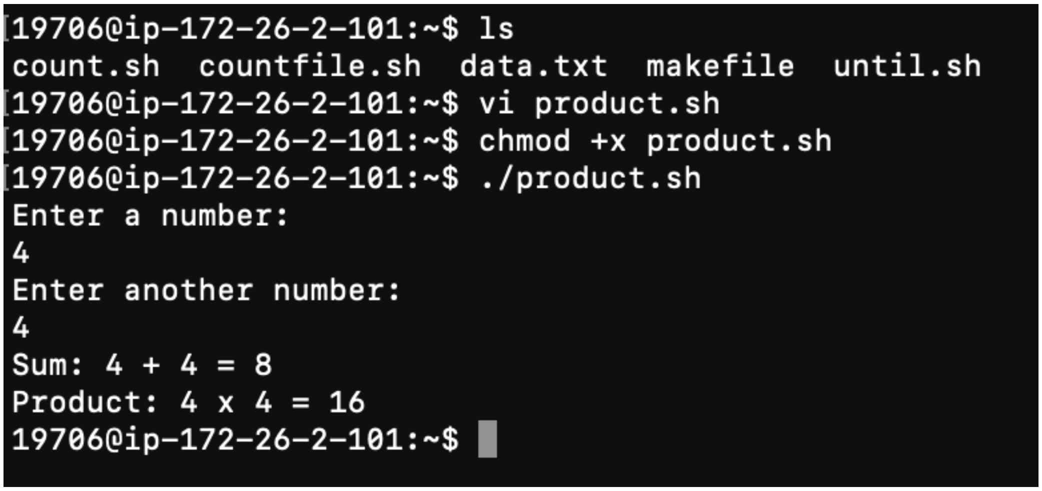
```
[19706@ip-172-26-2-101:~$ vi countfile.sh
[19706@ip-172-26-2-101:~$ chmod +x countfile.sh
[19706@ip-172-26-2-101:~$ ./countfile.sh
There are 1 .txt files in this directory.
[19706@ip-172-26-2-101:~$ ls
age.sh        data.txt  if.sh      makefile     reverse.sh   while.sh
count.sh      file.sh   lab.cshrc  myscript.sh  until.sh
countfile.sh  file3.sh  loop.sh    name.sh      variable.sh
19706@ip-172-26-2-101:~$ 
```



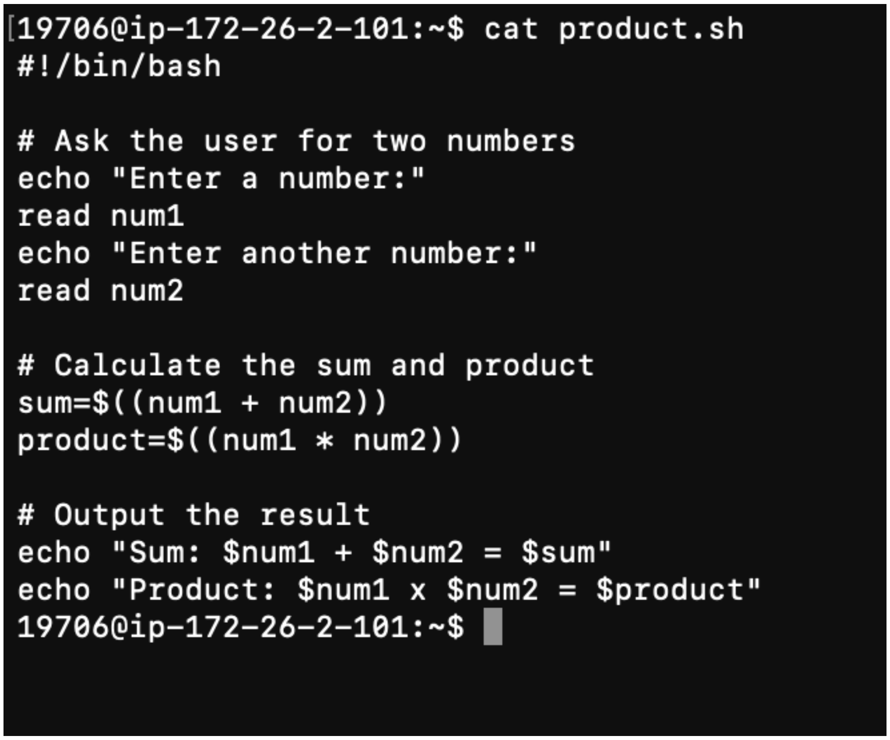
```
[19706@ip-172-26-2-101:~$ cat countfile.sh
#!/bin/bash

count=0

for file in *.txt; do
  ((count++))
done

if [ $count -eq 0 ]; then
  echo "There are no .txt files in this directory."
else
  echo "There are $count .txt files in this directory."
fi
19706@ip-172-26-2-101:~$
```

⇨ Webminal_Scripting_Misc

1. Write a script that asks for two numbers and outputs the sum and product.

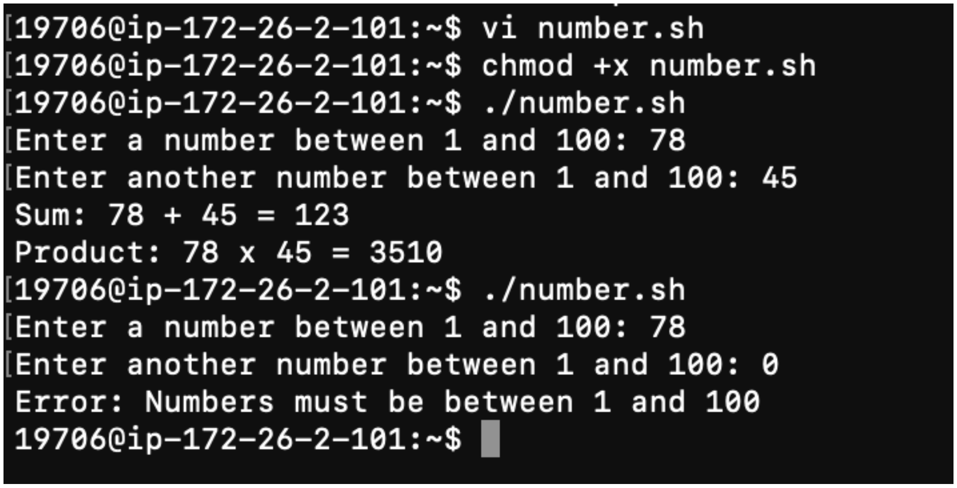


```
[19706@ip-172-26-2-101:~$ ls
count.sh  countfile.sh  data.txt  makefile  until.sh
[19706@ip-172-26-2-101:~$ vi product.sh
[19706@ip-172-26-2-101:~$ chmod +x product.sh
[19706@ip-172-26-2-101:~$ ./product.sh
Enter a number:
4
Enter another number:
4
Sum: 4 + 4 = 8
Product: 4 x 4 = 16
19706@ip-172-26-2-101:~$
```

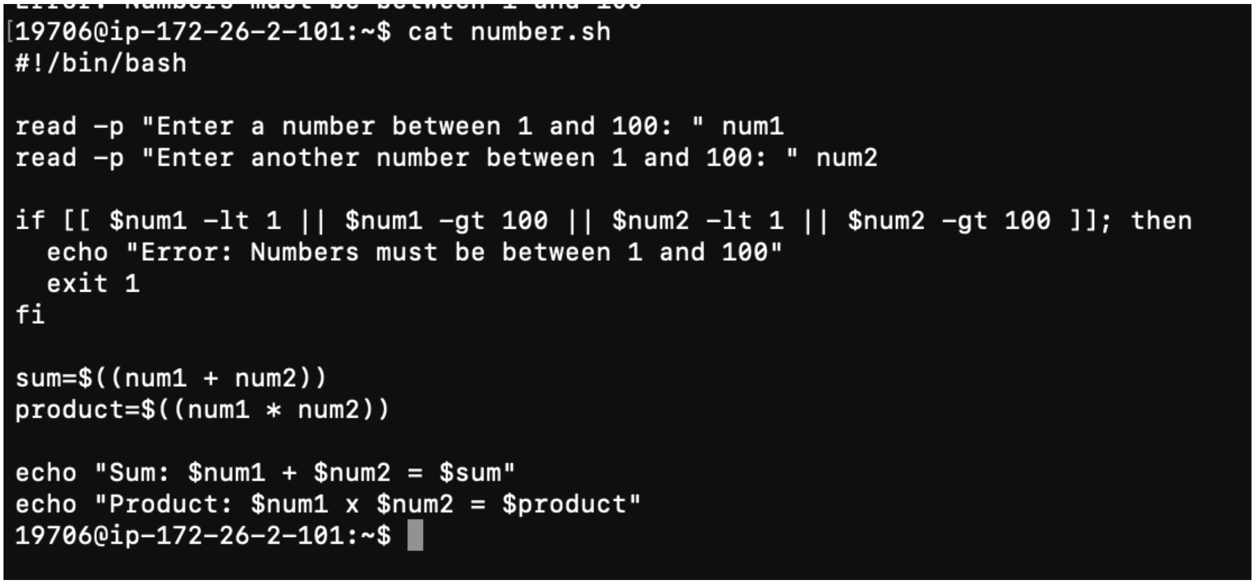


```
[19706@ip-172-26-2-101:~$ cat product.sh
#!/bin/bash

# Ask the user for two numbers
echo "Enter a number:"
read num1
echo "Enter another number:"
read num2

# Calculate the sum and product
sum=$((num1 + num2))
product=$((num1 * num2))

# Output the result
echo "Sum: $num1 + $num2 = $sum"
echo "Product: $num1 x $num2 = $product"
19706@ip-172-26-2-101:~$ 
```

2. Improve the previous script to test the numbers between 1 and 100 and exit with an error if necessary.

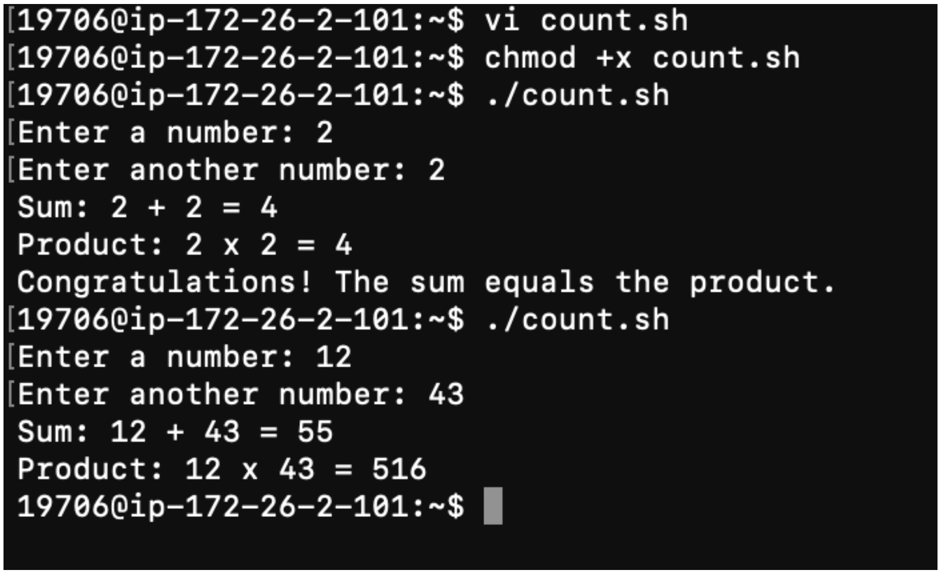


```
[19706@ip-172-26-2-101:~$ vi number.sh
[19706@ip-172-26-2-101:~$ chmod +x number.sh
[19706@ip-172-26-2-101:~$ ./number.sh
[Enter a number between 1 and 100: 78
[Enter another number between 1 and 100: 45
Sum: 78 + 45 = 123
Product: 78 x 45 = 3510
[19706@ip-172-26-2-101:~$ ./number.sh
[Enter a number between 1 and 100: 78
[Enter another number between 1 and 100: 0
Error: Numbers must be between 1 and 100
19706@ip-172-26-2-101:~$ 
```

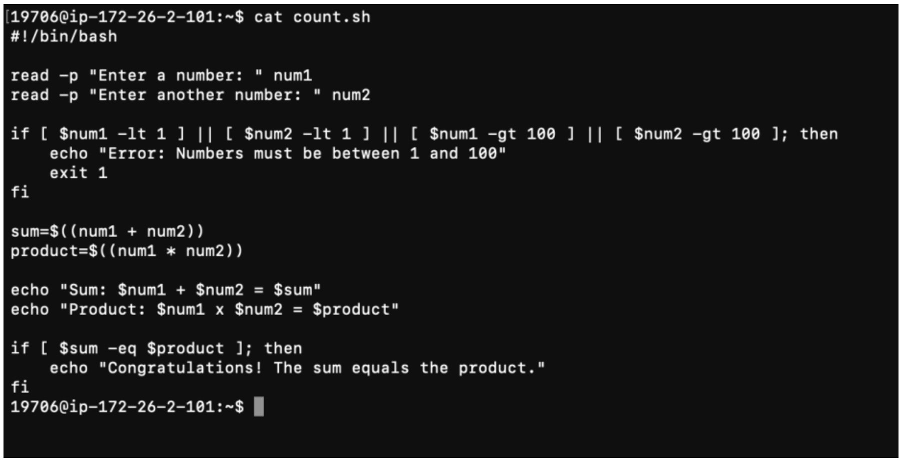


```
Error: Numbers must be between 1 and 100
[19706@ip-172-26-2-101:~$ cat number.sh
#!/bin/bash

read -p "Enter a number between 1 and 100: " num1
read -p "Enter another number between 1 and 100: " num2

if [[ $num1 -lt 1 || $num1 -gt 100 || $num2 -lt 1 || $num2 -gt 100 ]]; then
  echo "Error: Numbers must be between 1 and 100"
  exit 1
fi

sum=$((num1 + num2))
product=$((num1 * num2))

echo "Sum: $num1 + $num2 = $sum"
echo "Product: $num1 x $num2 = $product"
19706@ip-172-26-2-101:~$ 
```

3. Improve the previous script to congratulate the user if the sum equals the product.

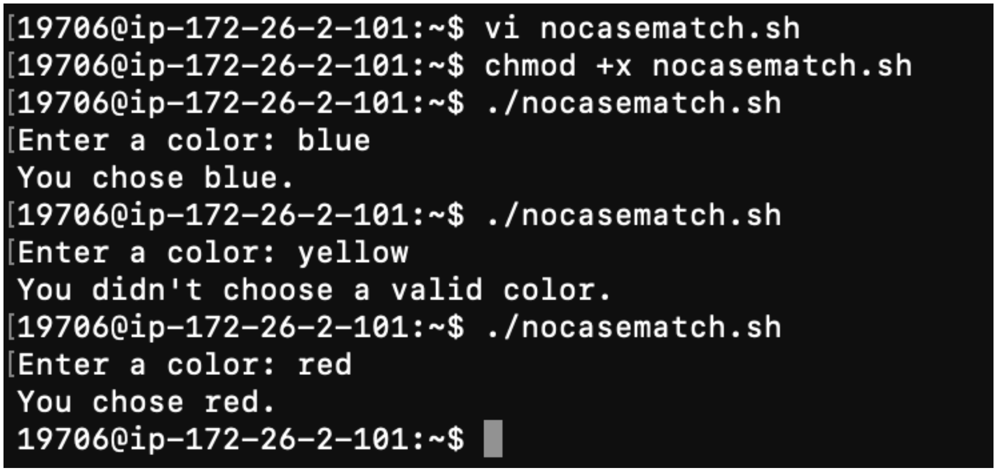


```
[19706@ip-172-26-2-101:~$ vi count.sh
[19706@ip-172-26-2-101:~$ chmod +x count.sh
[19706@ip-172-26-2-101:~$ ./count.sh
[Enter a number: 2
[Enter another number: 2
 Sum: 2 + 2 = 4
 Product: 2 x 2 = 4
 Congratulations! The sum equals the product.
[19706@ip-172-26-2-101:~$ ./count.sh
[Enter a number: 12
[Enter another number: 43
 Sum: 12 + 43 = 55
 Product: 12 x 43 = 516
 19706@ip-172-26-2-101:~$ 
```

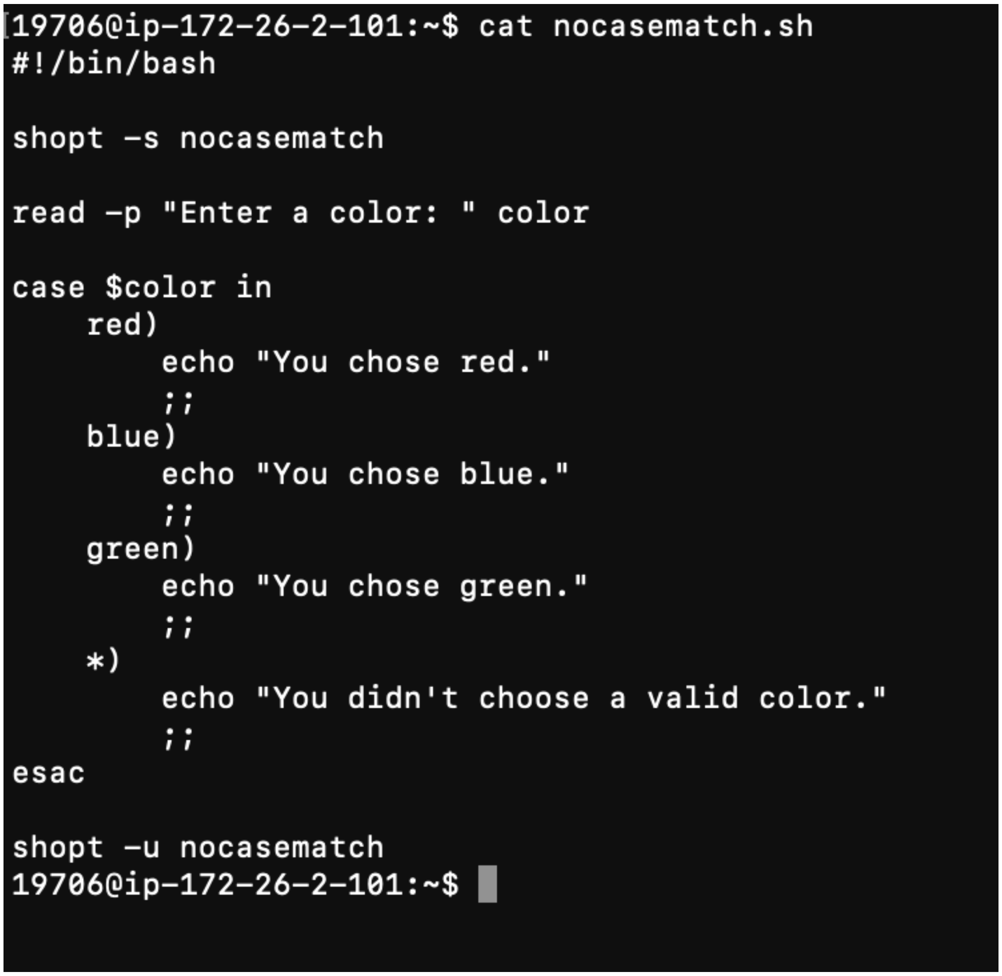


```
[19706@ip-172-26-2-101:~$ cat count.sh
#!/bin/bash

read -p "Enter a number: " num1
read -p "Enter another number: " num2

if [ $num1 -lt 1 ] || [ $num2 -lt 1 ] || [ $num1 -gt 100 ] || [ $num2 -gt 100 ]; then
    echo "Error: Numbers must be between 1 and 100"
    exit 1
fi

sum=$((num1 + num2))
product=$((num1 * num2))

echo "Sum: $num1 + $num2 = $sum"
echo "Product: $num1 x $num2 = $product"

if [ $sum -eq $product ]; then
    echo "Congratulations! The sum equals the product."
fi
19706@ip-172-26-2-101:~$
```

4. Write a script with a case-insensitive case statement using the shot nocasematch option. The nocasematch option is reset to its value before the scripts start.



```
[19706@ip-172-26-2-101:~$ vi nocasematch.sh
[19706@ip-172-26-2-101:~$ chmod +x nocasematch.sh
[19706@ip-172-26-2-101:~$ ./nocasematch.sh
[Enter a color: blue
 You chose blue.
[19706@ip-172-26-2-101:~$ ./nocasematch.sh
[Enter a color: yellow
 You didn't choose a valid color.
[19706@ip-172-26-2-101:~$ ./nocasematch.sh
[Enter a color: red
 You chose red.
 19706@ip-172-26-2-101:~$
```



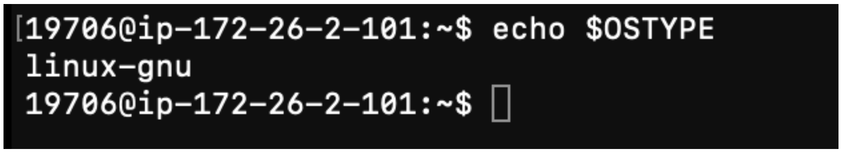
```
19706@ip-172-26-2-101:~$ cat nocasematch.sh
#!/bin/bash

shopt -s nocasematch

read -p "Enter a color: " color

case $color in
    red)
        echo "You chose red."
        ;;
    blue)
        echo "You chose blue."
        ;;
    green)
        echo "You chose green."
        ;;
    *)
        echo "You didn't choose a valid color."
        ;;
esac

shopt -u nocasematch
19706@ip-172-26-2-101:~$
```

2. Go through Webminal_ScriptingLoops.pdf and Webminal_Scripting_Misc.pdf. These are labs taken from Webminal.

⇨

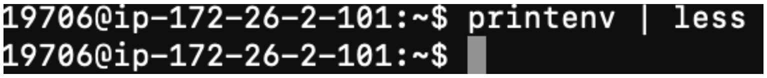
1. Environment variable:

a) OSTYPE variable:

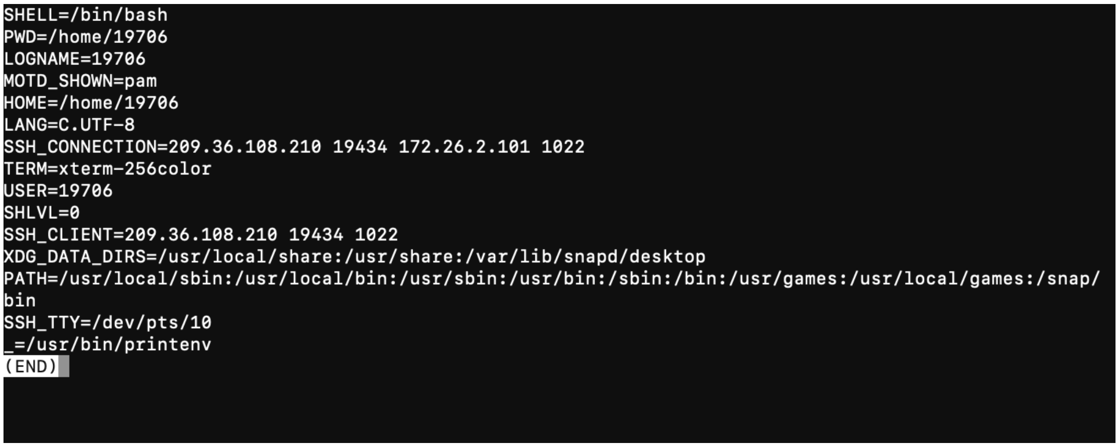


```
19706@ip-172-26-2-101:~$ echo $OSTYPE
linux-gnu
19706@ip-172-26-2-101:~$
```

b) Printenv | less:



```
19706@ip-172-26-2-101:~$ printenv | less
19706@ip-172-26-2-101:~$ 
```

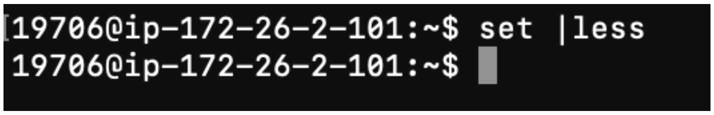


```
SHELL=/bin/bash
PWD=/home/19706
LOGNAME=19706
MOTD_SHOWN=pam
HOME=/home/19706
LANG=C.UTF-8
SSH_CONNECTION=209.36.108.210 19434 172.26.2.101 1022
TERM=xterm-256color
USER=19706
SHLVL=0
SSH_CLIENT=209.36.108.210 19434 1022
XDG_DATA_DIRS=/usr/local/share:/usr/share:/var/lib/snapd/desktop
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/
bin
SSH_TTY=/dev/pts/10
_=/usr/bin/printenv
(END)
```

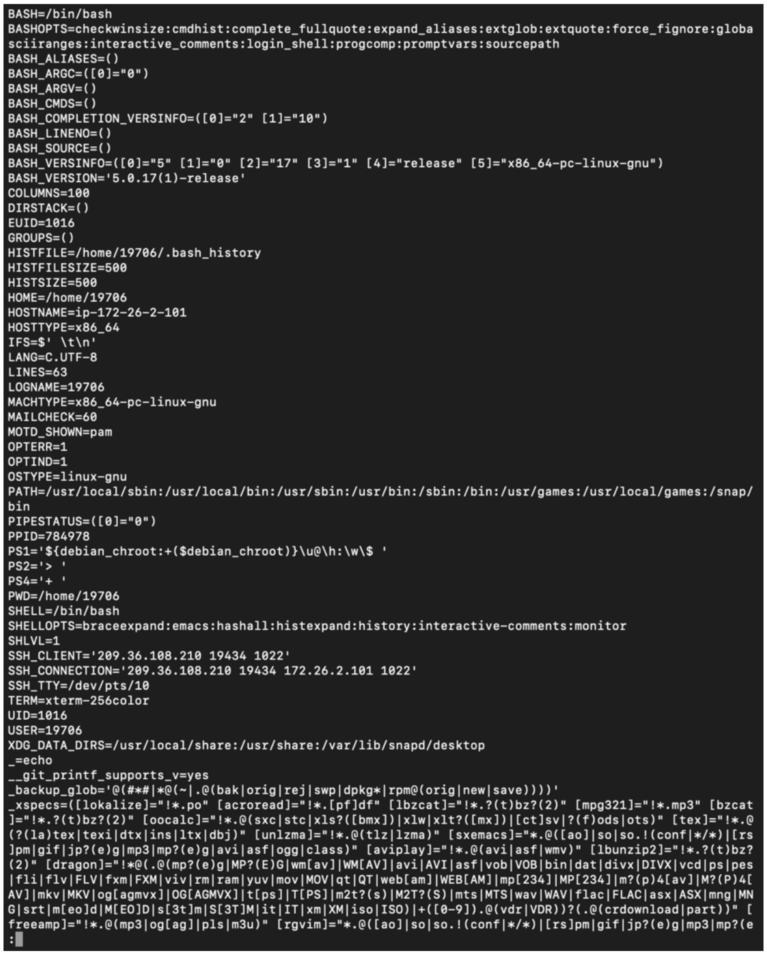
2. Shell Variable:

a) echo $history

b) Set |less:



```
19706@ip-172-26-2-101:~$ set |less
19706@ip-172-26-2-101:~$ 
```

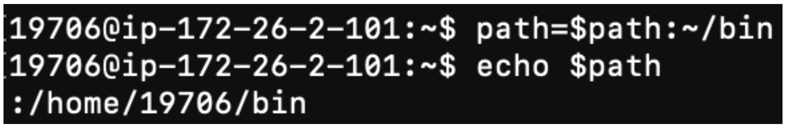


```
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliases:extglob:extquote:force_fignore:globa
sciiranges:interactive_comments:login_shell:progcomp:promptvars:sourcepath
BASH_ALIASES=()
BASH_ARGC=([0]="0")
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION_VERSINFO=([0]="2" [1]="10")
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="5" [1]="0" [2]="17" [3]="1" [4]="release" [5]="x86_64-pc-linux-gnu")
BASH_VERSION='5.0.17(1)-release'
COLUMNS=100
DIRSTACK=()
EUID=1016
GROUPS=()
HISTFILE=/home/19706/.bash_history
HISTFILESIZE=500
HISTSIZE=500
HOME=/home/19706
HOSTNAME=ip-172-26-2-101
HOSTTYPE=x86_64
IFS=$' \t\n'
LANG=C.UTF-8
LINES=63
LOGNAME=19706
MACHTYPE=x86_64-pc-linux-gnu
MAILCHECK=60
MOTD_SHOWN=pam
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/
bin
PIPESTATUS=([0]="0")
PPID=784978
PS1='${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
PS2='> '
PS4='+ '
PWD=/home/19706
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:history:interactive-comments:monitor
SHLVL=1
SSH_CLIENT='209.36.108.210 19434 1022'
SSH_CONNECTION='209.36.108.210 19434 172.26.2.101 1022'
SSH_TTY=/dev/pts/10
TERM=xterm-256color
UID=1016
USER=19706
XDG_DATA_DIRS=/usr/local/share:/usr/share:/var/lib/snapd/desktop
_=echo
__git_printf_supports_v=yes
_backup_glob='@(#*#|*@(~|.@(bak|orig|rej|swp|dpkg*|rpm@(orig|new|save))))'
_xspecs=([lokalize]="!*.po" [acroread]="!*.[pf]df" [lbzcat]="!*.?(t)bz?(2)" [mpg321]="!*.mp3" [bzcat
]="!*.?(t)bz?(2)" [oocalc]="!*.@(sxc|stc|xls?([bmx])|xlw|xlt?([mx])|[ct]sv|?(f)ods|ots)" [tex]="!*.@
(?(la)tex|texi|dtx|ins|ltx|dbj)" [unlzma]="!*.@(tlz|lzma)" [sxemacs]="*.@([ao]|so|so.!(conf|*/*)|[rs
]pm|gif|jp?(e)g|mp3|mp?(e)g|avi|asf|ogg|class)" [aviplay]="!*.@(avi|asf|wmv)" [lbunzip2]="!*.?(t)bz?
(2)" [dragon]="!*@(.@(mp?(e)g|MP?(E)G|wm[av]|WM[AV]|avi|AVI|asf|vob|VOB|bin|dat|divx|DIVX|vcd|ps|pes
|fli|flv|FLV|fxm|FXM|viv|rm|ram|yuv|mov|MOV|qt|QT|web[am]|WEB[AM]|mp[234]|MP[234]|m?(p)4[av]|M?(P)4[
AV]|mkv|MKV|og[agmvx]|OG[AGMVX]|t[ps]|T[PS]|m2t?(s)|M2T?(S)|mts|MTS|wav|WAV|flac|FLAC|asx|ASX|mng|MN
G|srt|m[eo]d|M[EO]D|s[3t]m|S[3T]M|it|IT|xm|XM|iso|ISO)|+([0-9]).@(vdr|VDR))?(.@(crdownload|part))" [
freeamp]="!*.@(mp3|og[ag]|pls|m3u)" [rgvim]="*.@([ao]|so|so.!(conf|*/*)|[rs]pm|gif|jp?(e)g|mp3|mp?(e
:
```

3. What are the differences between the path and PATH?

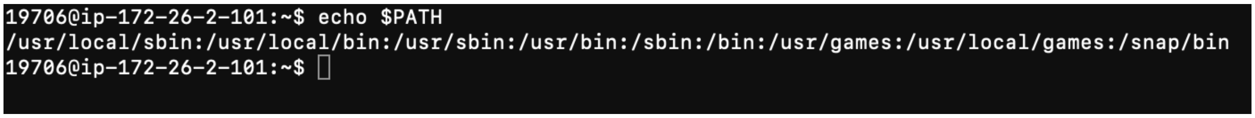
⇨ Path is a user-defined variable that can specify additional directories to search for executables within the current shell session. At the same time, PATH is a system-defined variable that determines the search path for executables used by the entire system.

⇨ Path:



```
19706@ip-172-26-2-101:~$ path=$path:~/bin
19706@ip-172-26-2-101:~$ echo $path
:/home/19706/bin
```

⇨ PATH:

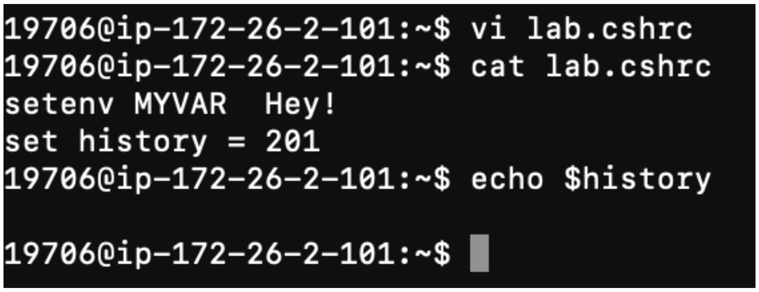


```
19706@ip-172-26-2-101:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
19706@ip-172-26-2-101:~$ 
```

4. Shell variable

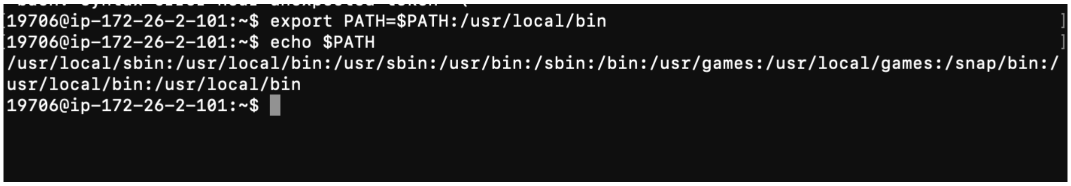
a) Set history:

It didn’t work in my terminal because it is



```
19706@ip-172-26-2-101:~$ vi lab.cshrc
19706@ip-172-26-2-101:~$ cat lab.cshrc
setenv MYVAR  Hey!
set history = 201
19706@ip-172-26-2-101:~$ echo $history

19706@ip-172-26-2-101:~$ 
```

b) Set PATH:



```
-bash: syntax error near unexpected token `('
19706@ip-172-26-2-101:~$ export PATH=$PATH:/usr/local/bin
19706@ip-172-26-2-101:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/
usr/local/bin:/usr/local/bin
19706@ip-172-26-2-101:~$ 
```