

# MODULE 4- BASH TRAINING

## Logical Operators

1) Check whether the file exists and is executable using logical operators.

```
user@user-VirtualBox:~$ gedit logic.sh &
[1] 18076
user@user-VirtualBox:~$ chmod +x logic.sh
user@user-VirtualBox:~$ ./logic.sh
File '' does not exists
user@user-VirtualBox:~$ touch test.txt
user@user-VirtualBox:~$ chmod +x logic.sh
user@user-VirtualBox:~$ ./logic.sh test.txt
File 'test.txt' exists but not executable
user@user-VirtualBox:~$ touch extest.txt
user@user-VirtualBox:~$ chmod +x extest.txt
user@user-VirtualBox:~$ ./logic.sh extest.txt
File 'extest.txt' does not exists
user@user-VirtualBox:~$ ./logic.sh extest.txt
File 'extest.txt' exists and is executable
user@user-VirtualBox:~$
```

logic.sh

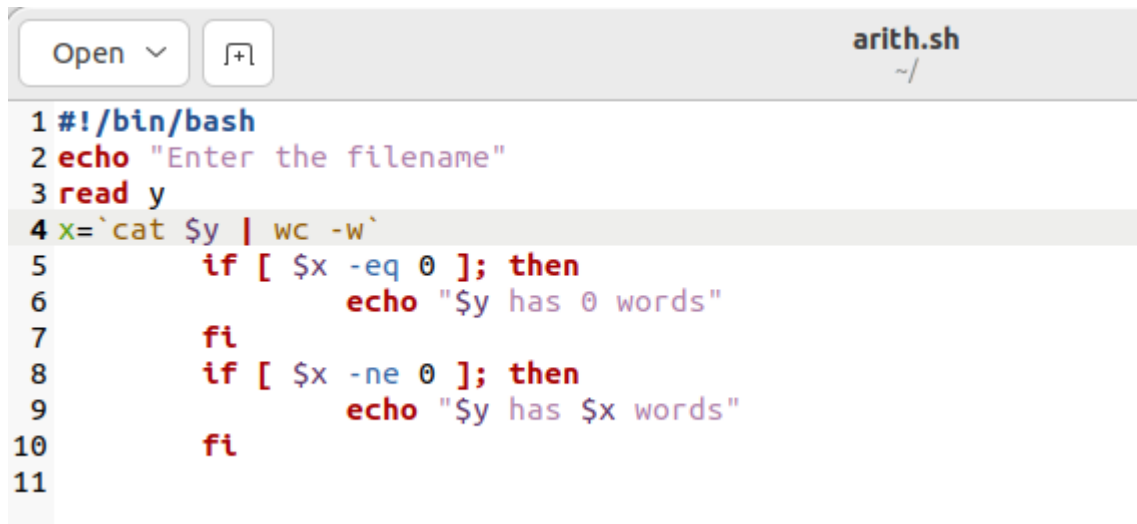
```
1 #!/bin/bash
2 if [ -f "$1" ] && [ -x "$1" ]; then
3 echo "File '$1' exists and is executable "
4 elif [ -f "$1" ] && [ ! -x "$1" ]; then
5 echo "File '$1' exists but not executable "
6 else
7 echo "File '$1' does not exists "
8 fi
```

Arithmetic Comparison

1) Write a program to demonstrate the use of not equal to operator.

Hint: -ne

```
user@user-VirtualBox:~$ gedit arith.sh &
[1] 18363
user@user-VirtualBox:~$ chmod +x arith.sh
user@user-VirtualBox:~$ touch list.txt
user@user-VirtualBox:~$ echo "YOU ARE USING BASH" > list.txt
user@user-VirtualBox:~$ chmod +x arith.sh
user@user-VirtualBox:~$ ./arith.sh
Enter the filename
list.txt
list.txt has 4 words
user@user-VirtualBox:~$ clear
```



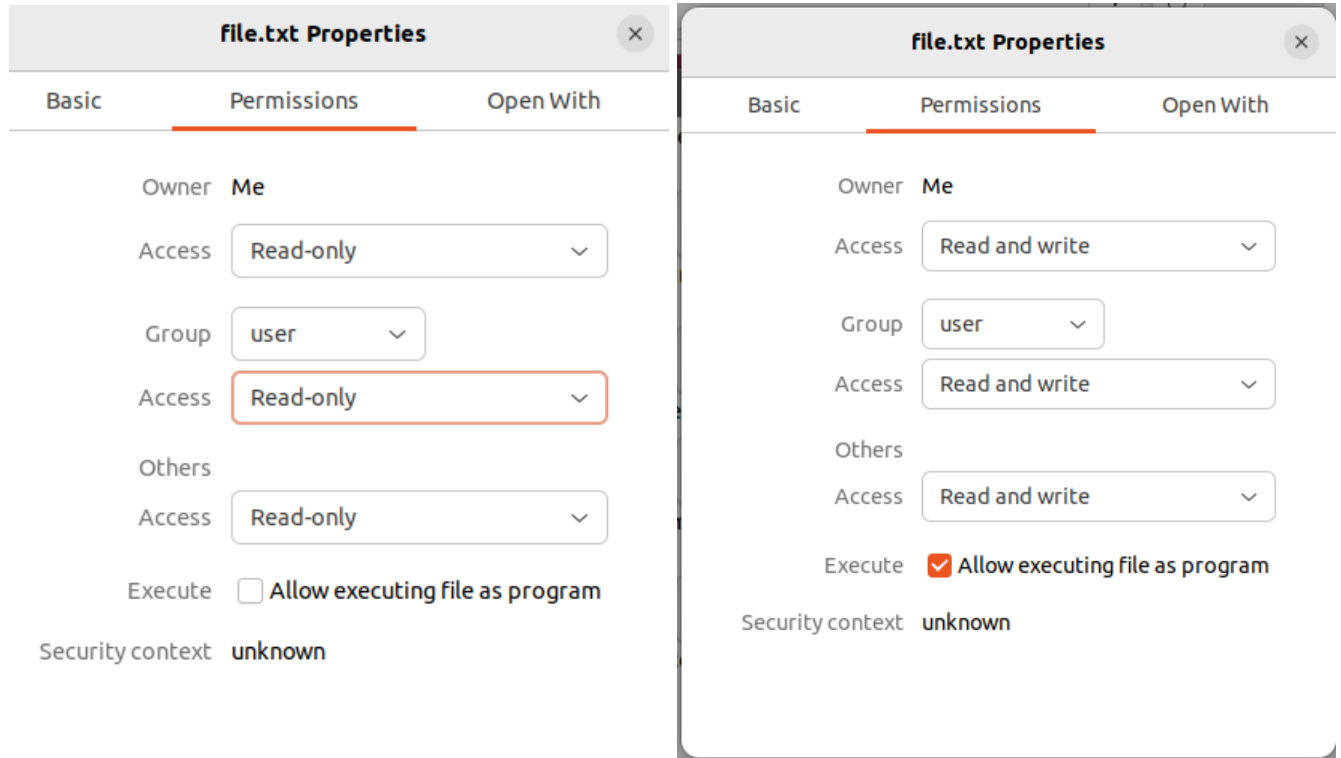
```
1 #!/bin/bash
2 echo "Enter the filename"
3 read y
4 x=`cat $y | wc -w`
5     if [ $x -eq 0 ]; then
6         echo "$y has 0 words"
7     fi
8     if [ $x -ne 0 ]; then
9         echo "$y has $x words"
10    fi
11
```

String and File attributes

1) Explore some more attributes

```
-r
-x
-o
```

Changing the properties of the file



```
user@user-VirtualBox:~$ ./atts.sh
File exist in read only format
user@user-VirtualBox:~$
```

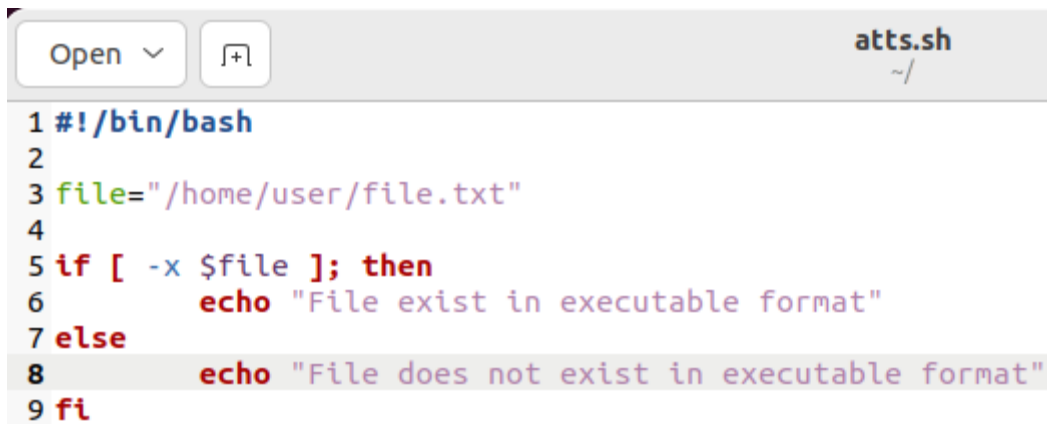
```

1 #!/bin/bash
2
3 file="/home/user/file.txt"
4
5 if [ -r $file ]; then
6     echo "File exist in read only format"
7 else
8     echo "File does not exist in read only format"
9 fi

```

```
user@user-VirtualBox:~$ ./atts.sh
File exist in executable format
user@user-VirtualBox:~$
```

```
user@user-VirtualBox:~$ ./atts.sh
File exist in read only format
user@user-VirtualBox:~$ ./atts.sh
File exist in executable format
user@user-VirtualBox:~$ ./atts.sh
You are not the owner of the file
```



The screenshot shows a code editor window with a title bar containing 'Open', a file icon, and the filename 'atts.sh' with a tilde '~/' indicating the current directory. The editor contains a shell script with the following lines:

```
1 #!/bin/bash
2
3 file="/home/user/file.txt"
4
5 if [ -x $file ]; then
6     echo "File exist in executable format"
7 else
8     echo "File does not exist in executable format"
9 fi
```

Conditional Loops

1) Find the sum of first n prime numbers.

```
user@user-VirtualBox:~$ chmod +x prime.sh
user@user-VirtualBox:~$ ./prime.sh
Enter Number : 6
6 is not a prime number.
user@user-VirtualBox:~$ ./prime.sh
Enter the value of 'n': 6
Sum of the first 6 prime numbers is: 41
user@user-VirtualBox:~$
```

```
1 #!/bin/bash
2
3 # Function to check if a number is prime
4 is_prime() {
5     num=$1
6     if [ $num -le 1 ]; then
7         return 1
8     fi
9     for ((i=2; i*i<=num; i++)); do
10         if [ $((num % i)) -eq 0 ]; then
11             return 1
12         fi
13     done
14     return 0
15 }
16
17 # Read 'n' from user input
18 read -p "Enter the value of 'n': " n
19
20 sum=0
21 count=0
22 num=2
23
24 while [ $count -lt $n ]; do
25     if is_prime $num; then
26         sum=$((sum + num))
27         count=$((count + 1))
28     fi
29     num=$((num + 1))
30 done
31
32 echo "Sum of the first $n prime numbers is: $sum"
33
34
```

More on Loops

- 1) Retype nested-for.sh bash script using nested while loop
- 2) Save your program with the name: nestedwhile.sh

Open
+

\*nestedwhile.sh  
~/

```

1 #!/usr/bin/env bash
2 for dir in test*; do
3 echo "Files in $dir directory"
4 echo ""
5 files=("$dir")
6 file_count=${#files[@]}
7 outer=0
8 while [ $outer -lt $file_count ]; do
9 file="${files[$outer]}"
10     echo $file
11     (( outer++ ))
12 done
13     echo "....."
14 done
15

```

```

user@user-VirtualBox:~$ ./nestedwhile.sh
Files in test1.txt directory
test1.txt
.....
Files in test2.txt directory
test2.txt
.....
Files in testa.txt directory
testa.txt
.....
Files in testb.txt directory
testb.txt
.....
Files in test_link.txt directory
test_link.txt
.....
Files in test.txt directory
test.txt
.....
user@user-VirtualBox:~$

```

**Case statement**

- 1) Write a menu driven program for mathematical calculation
  - a. It should take user inputs a and b
  - b. It should ask for mathematical operator (+, -, / and \*).
  - c. Do the calculation
  - d. Print the output

```
user@user-VirtualBox:~$ ./case.sh
SIMPLE CALCULATOR !
Enter your choice
1. Addition
2. Subraction
3. Multiplication
4. Division
5. Quit
Enter the first number 2
Enter the second number 3
Choose the operator(1/2/3/4/5) 1
The sum is 5
user@user-VirtualBox:~$ ./case.sh
SIMPLE CALCULATOR !
Enter your choice
1. Addition
2. Subraction
3. Multiplication
4. Division
5. Quit
Enter the first number 2
Enter the second number 3
Choose the operator(1/2/3/4/5) 2
The difference is -1
user@user-VirtualBox:~$ ./case.sh
SIMPLE CALCULATOR !
Enter your choice
1. Addition
2. Subraction
3. Multiplication
4. Division
5. Quit
Enter the first number 2
Enter the second number 3
Choose the operator(1/2/3/4/5) 3
The product is 6
user@user-VirtualBox:~$ ./case.sh
SIMPLE CALCULATOR !
Enter your choice
1. Addition
```

```

user@user-VirtualBox:~$ ./case.sh
SIMPLE CALCULATOR !
Enter your choice
1. Addition
2. Subraction
3. Multiplication
4. Division
5. Quit
Enter the first number 9
Enter the second number 3
Choose the operator(1/2/3/4/5) 4
The answer is 3
user@user-VirtualBox:~$

```

```

1 #!/usr/bin/env bash
2 echo "SIMPLE CALCULATOR !"
3 echo "Enter your choice"
4 echo "1. Addition"
5 echo "2. Subraction"
6 echo "3. Multiplication"
7 echo "4. Division"
8 echo "5. Quit"
9 read -p "Enter the first number " a
10 read -p "Enter the second number " b
11 read -p "Choose the operator(1/2/3/4/5) " op
12 case $op in
13 1)
14 result=$((a + b))
15     echo "The sum is $result"
16     ;;
17 2)
18 result=$((a - b))
19     echo "The difference is $result"
20     ;;
21 3)
22 result=$((a * b))
23     echo "The product is $result"
24     ;;
25 4)
26 if [ $b -eq 0 ]; then
27     echo "Error ! division by zero is not allowed."
28 else
29 result=$((a / b))
30     echo "The answer is $result"
31 fi
32     ;;
33 5)echo "Goodbye !"
34     exit 0;;
35 *)echo "INVALID CHOICE";;
36 esac
37
38


```



### Using File Descriptors

- 1) Try to append few lines to a file test.txt using file descriptor.
- 2) Display the content of the file using file descriptor.

```
user@user-VirtualBox:~$ gedit test.sh
user@user-VirtualBox:~$ chmod +x test.sh
user@user-VirtualBox:~$ cat test.txt
user@user-VirtualBox:~$ ./test.txt
bash: ./test.txt: Permission denied
user@user-VirtualBox:~$ ./test.sh
user@user-VirtualBox:~$ cat test.txt
Welcome to Bash
Today's date and time is
Wednesday 11 October 2023 09:55:28 PM IST
user@user-VirtualBox:~$ gedit readtest.sh &
[1] 25686
user@user-VirtualBox:~$ chmod +x readtest.sh
[1]+  Done                  gedit readtest.sh
user@user-VirtualBox:~$ ./readtest.sh
Welcome to Bash
Today's date and time is
Wednesday 11 October 2023 09:55:28 PM IST
user@user-VirtualBox:~$
```



```
Open  [icon] readtest.sh
~ /
1 #!/usr/bin/env bash
2 exec 3< test.txt
3 cat <&3
4 exec 3<&-
```

**Basics of functions**

- 1) Write a program with two functions:
  - a. The first function should display disk space usage in human readable form.  
(Hint: `df -h`)
  - b. The second function should display filesystem usage in human readable form.  
(Hint: `du -h`)

```

user@user-VirtualBox: ~
user@user-VirtualBox: $ gedit function.sh &
[1] 16049
user@user-VirtualBox: $ chmod +x function.sh
user@user-VirtualBox: $ ./function.sh

Beginning of the main program

Machine Disk Usage
Filesystem      Size  Used Avail Use% Mounted on
tmpfs            196M   1.6M  195M   1% /run
/dev/sda3        24G    16G   7.7G  67% /
tmpfs            980M    0   980M   0% /dev/shm
tmpfs            5.0M   4.0K   5.0M   1% /run/lock
/dev/sda2        512M   6.1M  506M   2% /boot/efi
tmpfs            196M  136K  196M   1% /run/user/1000

Machine File Usage
4.0K  ./snap/snap-store/959/.local/share/icons
200K  ./snap/snap-store/959/.local/share/glib-2.0/schemas
204K  ./snap/snap-store/959/.local/share/glib-2.0
212K  ./snap/snap-store/959/.local/share
216K  ./snap/snap-store/959/.local
8.0K  ./snap/snap-store/959/.config/fontconfig
4.0K  ./snap/snap-store/959/.config/gtk-2.0
8.0K  ./snap/snap-store/959/.config/autostart
4.0K  ./snap/snap-store/959/.config/dconf
4.0K  ./snap/snap-store/959/.config/gtk-3.0
4.0K  ./snap/snap-store/959/.config/ibus
52K   ./snap/snap-store/959/.config
276K  ./snap/snap-store/959
4.0K  ./snap/snap-store/common/.cache/appstream
60K   ./snap/snap-store/common/.cache/fontconfig
28K   ./snap/snap-store/common/.cache/glo-modules
76K   ./snap/snap-store/common/.cache/lnmodules
5.8M  ./snap/snap-store/common/.cache/gnome-software/appstream
1.2M  ./snap/snap-store/common/.cache/gnome-software/cssresource
856K  ./snap/snap-store/common/.cache/gnome-software/icons
1.5M  ./snap/snap-store/common/.cache/gnome-software/odrs
9.2M  ./snap/snap-store/common/.cache/gnome-software
9.4M  ./snap/snap-store/common/.cache


```

```

user@user-VirtualBox: ~
4.0K  ./cache/evolution/memos/trash
8.0K  ./cache/evolution/memos
4.0K  ./cache/evolution/calendar/trash
52K   ./cache/evolution/calendar/4d823672a876518778f69a9e64fe3b4492b67d26
280K  ./cache/evolution/calendar/663a9cee9480f8906579b8ea644db8e2a725ff78
52K   ./cache/evolution/calendar/c3ecb06762005f966eaf9b093370e2c17f53ba24
128K  ./cache/evolution/calendar/ab6fccfbef6a6fe9ee697fc66131da008a12bce28
520K  ./cache/evolution/calendar
4.0K  ./cache/evolution/addressbook/trash
8.0K  ./cache/evolution/addressbook
4.0K  ./cache/evolution/mail/trash
8.0K  ./cache/evolution/mail
592K  ./cache/evolution
4.0K  ./cache/tracker3/files/errors
31M   ./cache/tracker3/files
31M   ./cache/tracker3
4.0K  ./cache/ibus-table
4.0K  ./cache/thunderbird/2nmshtn3.default-release/cache2/ doomed
1.3M  ./cache/thunderbird/2nmshtn3.default-release/cache2/entries
1.3M  ./cache/thunderbird/2nmshtn3.default-release/cache2
4.0K  ./cache/thunderbird/2nmshtn3.default-release/safebrowsing/google4
8.0K  ./cache/thunderbird/2nmshtn3.default-release/safebrowsing
7.3M  ./cache/thunderbird/2nmshtn3.default-release/startupCache
8.6M  ./cache/thunderbird/2nmshtn3.default-release
4.0K  ./cache/thunderbird/cl7tadh4.default
8.6M  ./cache/thunderbird
8.0K  ./cache/yelp/cacheStorage
4.0K  ./cache/yelp/WebKitCache/Version 16/Blobs
12K   ./cache/yelp/WebKitCache/Version 16
16K   ./cache/yelp/WebKitCache
28K   ./cache/yelp
192K  ./cache/ibus/bus
196K  ./cache/ibus
4.0K  ./cache/gnome-desktop-thumbnailer/gstreamer-1.0
8.0K  ./cache/gnome-desktop-thumbnailer
44M   ./cache
1.5G  .

End of the main program
user@user-VirtualBox: $

```

Open ▾ 

function.sh  
~/

```
1#!/usr/bin/env bash
2function machine
3{
4echo -e " \n Machine Disk Usage "; df -h
5echo -e " \n Machine File Usage "; du -h
6}
7echo -e " \n Beginning of the main program "
8machine
9echo -e " \n End of the main program "
```

**More on functions**

- 1) Write a program,
  - a. where the function accepts two arguments.
  - b. The function should multiply the two arguments.
  - c. Make 3 function calls with arguments - (1, 2), (2, 3) and (3, 4)

```
user@user-VirtualBox:~$ ./function_call.sh
```

```
Multiplication of 1 and 2 is 2
```

```
Multiplication of 2 and 3 is 6
```

```
Multiplication of 3 and 4 is 12
```

```
user@user-VirtualBox:~$
```

```
1 #!/usr/bin/env bash
2 add(){
3     array=( $@ )
4     echo -e "\n Array elements are ${array[@]}"
5     local sum=0
6     for elements in "$@"; do
7         sum=$((sum+elements))
8     done
9     echo "Sum of elements $sum"
10 }
11 add 1 2 3
12 add 4 5 6
```

Arrays and functions

- 1) Write a program,
  - a. Where a function adds all the elements in an array.
  - b. The function should display the sum of elements.
  - c. Make 2 function calls with array elements- (1, 2, 3) and (4, 5, 6).

```
user@user-VirtualBox:~$ chmod +x arrayfun.sh
user@user-VirtualBox:~$ ./arrayfun.sh
```

```
Array elements are 1 2 3
Sum of elements 6
```

```
Array elements are 4 5 6
Sum of elements 15
user@user-VirtualBox:~$
```

```
1 #!/usr/bin/env bash
2 fun_call (){
3 result=$(( $1 * $2 ))
4 echo -e "\n Multiplication of $1 and $2 is $result"
5 }
6 fun_call 1 2
7 fun_call 2 3
8 fun_call 3 4
9
10
```