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 exits
user@user-VirtualBox:~$ touch test.txt
user@user-VirtualBox:~$ chmod +x logic.sh
user@user-VirtualBox:~$ ./logic.sh test.txt
File 'test.txt' exits but not executable
user@user-VirtualBox:~$ touch extext.txt
user@user-VirtualBox:~$ chmod +x extext.txt
user@user-VirtualBox:~$ ./logic.sh extest.txt
File 'extest.txt' does not exits
user@user-VirtualBox:~$ ./logic.sh extext.txt
File 'extext.txt' exits and is executable
user@user-VirtualBox:~$
```

logic,sh

```
#!/bin/bash
if [ -f "$1" ] && [ -x "$1" ]; then
echo "File '$1' exits and is executable "
elif [ -f "$1" ] && [ ! -x "$1" ]; then
echo "File '$1' exits but not executable "
else
echo "File '$1' does not exits "
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
```

Open

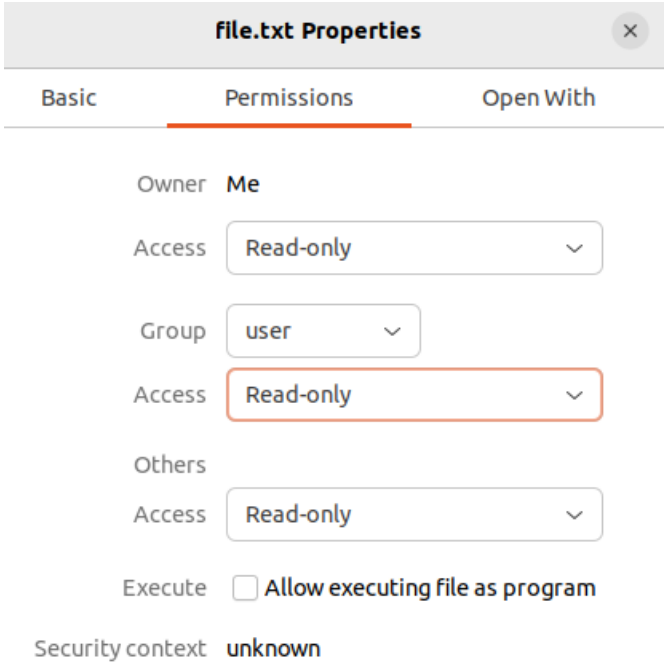
arith.sh
~/

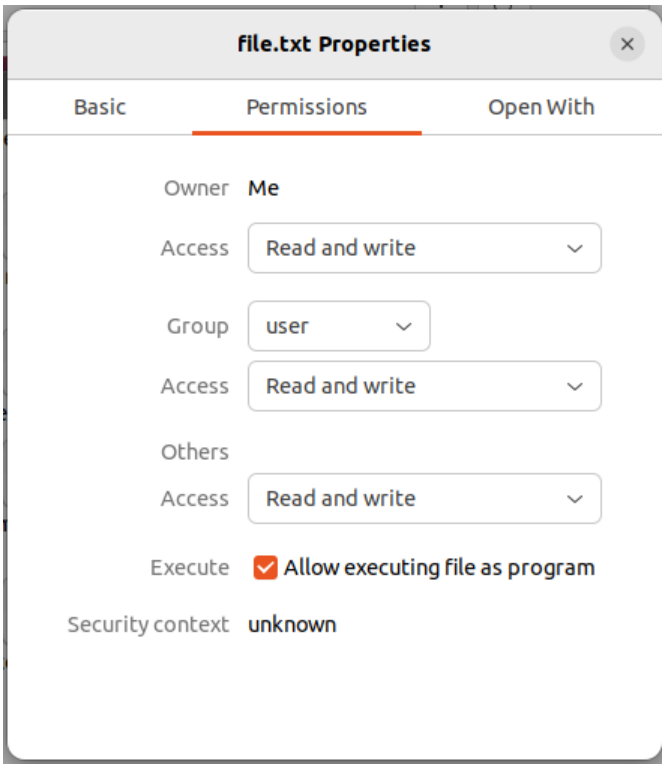
```
#!/bin/bash
echo "Enter the filename"
read y
x=`cat $y | wc -w`
        if [ $x -eq 0 ]; then
                echo "$y has 0 words"
        fi
        if [ $x -ne 0 ]; then
                echo "$y has $x words"
        fi

```

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:~$
```

Open *atts.sh ~/ Save

```
#!/bin/bash

file="/home/user/file.txt"

if [ -r $file ]; then
        echo "File exist in read only format"
else
        echo "File does not exist in read only format"
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
```

Open

atts.sh
~/

```
#!/bin/bash

file="/home/user/file.txt"

if [ -x $file ]; then
        echo "File exist in executable format"
else
        echo "File does not exist in executable format"
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:~$
```

```bash
#!/bin/bash

# Function to check if a number is prime
is_prime() {
    num=$1
    if [ $num -le 1 ]; then
        return 1
    fi
    for ((i=2; i*i<=num; i++)); do
        if [ $((num % i)) -eq 0 ]; then
            return 1
        fi
    done
    return 0
}

# Read 'n' from user input
read -p "Enter the value of 'n': " n

sum=0
count=0
num=2

while [ $count -lt $n ]; do
    if is_prime $num; then
        sum=$((sum + num))
        count=$((count + 1))
    fi
    num=$((num + 1))
done

echo "Sum of the first $n prime numbers is: $sum"
```

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 ~/

```
#!/usr/bin/env bash
for dir in test*; do
echo "Files in $dir directory"
echo ""
files=("$dir")
file_count=${#files[@]}
outer=0
while [ $outer -lt $file_count ]; do
file="${files[$outer]}"
        echo $file
        (( outer++ ))
        done
        echo "............................"
done
```

```
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:~$
```

```bash
#!/usr/bin/env bash
echo "SIMPLE CALCULATOR !"
echo "Enter your choice"
echo "1. Addition"
echo "2. Subraction"
echo "3. Multiplication"
echo "4. Division"
echo "5. Quit"
read -p "Enter the first number " a
read -p "Enter the second number " b
read -p "Choose the operator(1/2/3/4/5) " op
case $op in
1)
result=$((a + b))
        echo "The sum is $result"
        ;;
2)
result=$((a - b))
        echo "The difference is $result"
        ;;
3)
result=$((a * b))
        echo "The product is $result"
        ;;
4)
if [ $b -eq 0 ]; then
        echo "Error ! division by zero is not allowed."
else
result=$((a / b))
        echo "The answer is $result"
fi
        ;;
5)echo "Goodbye !"
        exit 0;;
*)echo "INVALID CHOICE";;
esac
```

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:~$
```

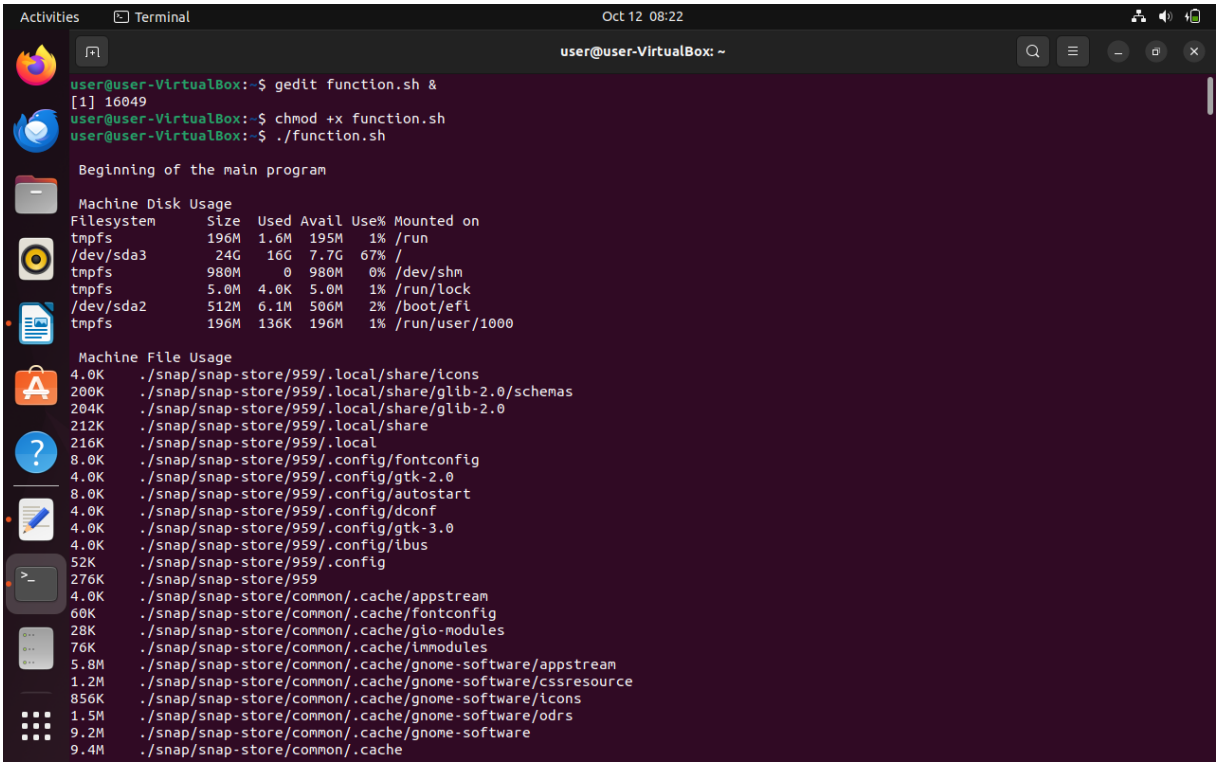
Open

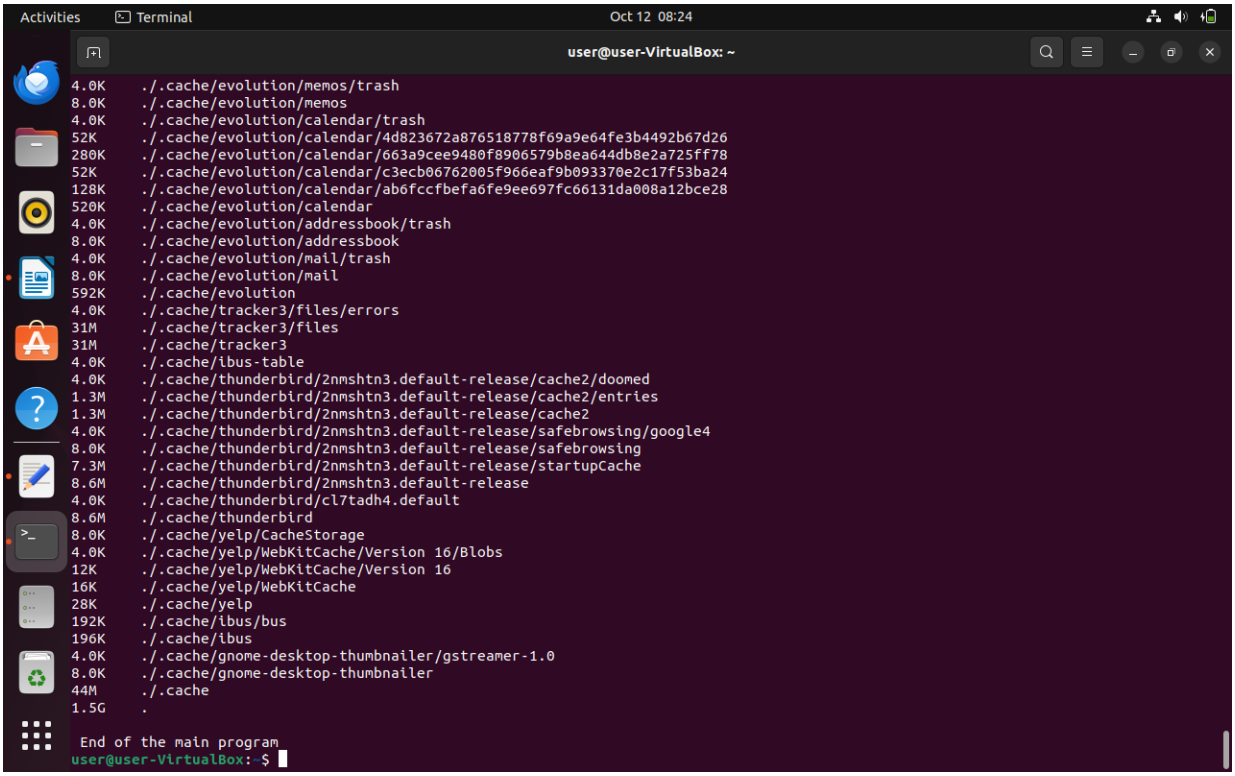
readtest.sh
~/

```
#!/usr/bin/env bash
exec 3< test.txt
cat <&3
exec 3<&-
```

Basics of functions

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





Open

function.sh
~/

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

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