

Linux Assignment-3

1. Write a shell program to implement basic arithmetic operations using function.

Ans:

```
#!/bin/bash

add() {
    result=$(( $1 + $2 ))
    echo "addition: $result"
}

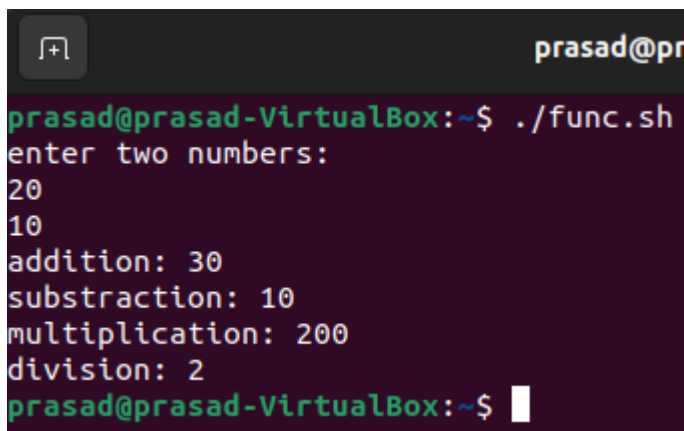
sub() {
    result=$(( $1 - $2 ))
    echo "subtraction: $result"
}

multi() {
    result=$(( $1 * $2 ))
    echo "multiplication: $result"
}

div() {
    result=$(( $1 / $2 ))
    echo "division: $result"
}

echo "Enter two numbers: "
read num1
read num2

add $num1 $num2
sub $num1 $num2
multi $num1 $num2
div $num1 $num2
```

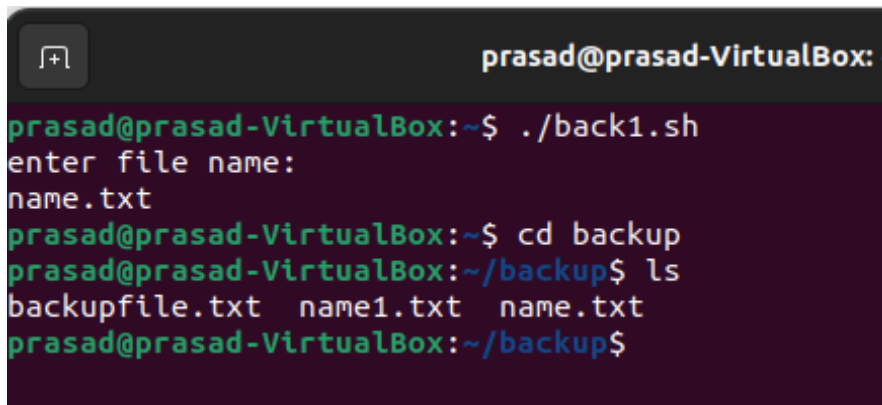
A terminal window with a dark background. The prompt is 'prasad@prasad-VirtualBox:~\$'. The user has run './func.sh'. The script prompts 'enter two numbers:'. The user enters '20' and '10' on separate lines. The script then outputs the results of four arithmetic operations: 'addition: 30', 'subtraction: 10', 'multiplication: 200', and 'division: 2'. The prompt returns to 'prasad@prasad-VirtualBox:~\$' with a cursor.

```
prasad@prasad-VirtualBox:~$ ./func.sh
enter two numbers:
20
10
addition: 30
subtraction: 10
multiplication: 200
division: 2
prasad@prasad-VirtualBox:~$
```

2. Run the command to take backup of file and share your output.

Ans:

```
echo "enter file name:"  
read fname  
for fname in *.txt  
do  
    [ -s $fname ] && cp $fname backup  
done
```

A terminal window titled 'prasad@prasad-VirtualBox: ~' with a dark background. The user runs './back1.sh' and enters 'name.txt'. Then they run 'cd backup' and 'ls', which shows 'backupfile.txt', 'name1.txt', and 'name.txt'.

```
prasad@prasad-VirtualBox: ~$ ./back1.sh  
enter file name:  
name.txt  
prasad@prasad-VirtualBox: ~$ cd backup  
prasad@prasad-VirtualBox: ~/backup$ ls  
backupfile.txt  name1.txt  name.txt  
prasad@prasad-VirtualBox: ~/backup$
```

3. How to take hardlink and softlink of a file in linux? Explain its differences.

Ans:

Hard Links:

A hard link is a direct reference to a file in the file system. It's basically another name for the same data on disk. Both the original file and the hard link point to the same inode on the disk means there's no original or copy they are basically the same file. Hard links can only be created within the same file system. Same size as the target file. Permissions and ownership are not relevant.

Soft Links:

A symbolic link, also known as a soft link, is a special type of file that contains a reference to another file or directory. It's like a pointer to the original file. Unlike hard links, soft links do not directly reference the inode. Instead, they contain the path to the target file or directory. The original file and the symbolic link have different inode numbers. Soft links can point to files or directories on different file systems. Very small in size. Has its own permissions and ownership.

4. Run commands (ps,fg, bg, find, du, df, head, tail, less, more) in your terminal and share the results.

Ans:

ps:

```
prasad@prasad-VirtualBox: ~  
prasad@prasad-VirtualBox:~$ ps  
  PID TTY          TIME CMD  
 11597 pts/0    00:00:00 bash  
 12239 pts/0    00:00:00 ps  
prasad@prasad-VirtualBox:~$
```

fg:

```
prasad@prasad-VirtualBox:~$ gcc -o infi infi.c  
prasad@prasad-VirtualBox:~$ ./infi  
^Z  
[1]+  Stopped                  ./infi  
prasad@prasad-VirtualBox:~$ bg ./infi  
[1]+  ./infi &  
prasad@prasad-VirtualBox:~$ jobs  
[1]+  Running                  ./infi &  
prasad@prasad-VirtualBox:~$ fg ./infi  
./infi  
█
```

bg:

```
prasad@prasad-VirtualBox: ~  
prasad@prasad-VirtualBox:~$ gcc -o infi infi.c  
prasad@prasad-VirtualBox:~$ ./infi  
^Z  
[1]+  Stopped                  ./infi  
prasad@prasad-VirtualBox:~$ bg ./infi  
[1]+  ./infi &  
prasad@prasad-VirtualBox:~$ jobs  
[1]+  Running                  ./infi &  
prasad@prasad-VirtualBox:~$ █
```

find:

```
prasad@prasad-VirtualBox: ~  
prasad@prasad-VirtualBox:~$ find . -empty  
./cache/evolution/sources/trash  
./cache/evolution/tasks/trash  
./cache/evolution/calendar/trash  
./cache/evolution/mail/trash  
./cache/evolution/memos/trash  
./cache/evolution/addressbook/trash  
./cache/ibus-table  
./cache/tracker3/files/errors  
./cache/tracker3/files/.meta.isrunning  
./sudo_as_admin_successful
```

df:

```
prasad@prasad-VirtualBox: ~  
prasad@prasad-VirtualBox:~$ df prime.sh  
Filesystem      1K-blocks    Used Available Use% Mounted on  
/dev/sda3        40453376 13572264   24793996  36% /  
prasad@prasad-VirtualBox:~$
```

du:

```
prasad@prasad-VirtualBox: ~  
prasad@prasad-VirtualBox:~$ du prime.sh  
4      prime.sh  
prasad@prasad-VirtualBox:~$
```

head:

```
prasad@prasad-VirtualBox: ~  
prasad@prasad-VirtualBox:~$ head -n 2 prime.sh  
#!/bin/bash  
  
prasad@prasad-VirtualBox:~$
```

tail:

```
prasad@prasad-VirtualBox:~$ tail -n 2 prime.sh
    echo "not prime"
fi
prasad@prasad-VirtualBox:~$
```

less:

```
prasad@prasad-VirtualBox:~$ less prime.sh
#!/bin/bash

echo "enter number:"
read number

p=1
i=2

if [ $number -lt 2 ];
then
    echo "number is not prime"
    exit
fi

while [ $i -le $((number/2)) ];
do
    if [ $((number % i)) -eq 0 ];
    then
        p=0
        break
    fi
    i=$((i + 1))
done
prime.sh
```

more:

```
prasad@prasad-VirtualBox:~$ more prime.sh
#!/bin/bash

echo "enter number:"
read number

p=1
i=2

if [ $number -lt 2 ];
then
    echo "number is not prime"
    exit
fi

while [ $i -le $((number/2)) ];
do
    if [ $((number % i)) -eq 0 ];
    then
        p=0
        break
    fi
    i=$((i + 1))
done
--More-- (78%)
```

5. With a shell program show the difference of break and continue statements

Ans:

```
#!/bin/bash
```

```
echo "break statement:"
```

```
for i in {1..5}
```

```
do
```

```
    echo " $i"
```

```
    if [ $i -eq 3 ]
```

```
    then
```

```
        echo "break"
```

```
        break
```

```
    fi
```

```
done
```

```
echo "finished"
```

```
echo -e "\ncontinue statement:"
```

```
for i in {1..5}
```

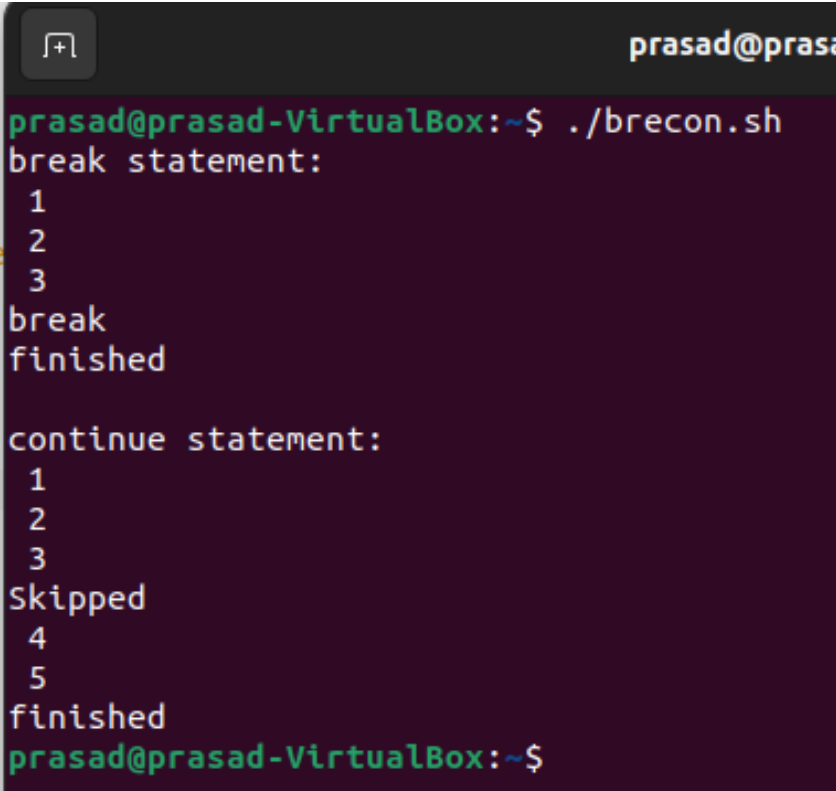
```
do
```

```
    echo " $i"
```

```
    if [ $i -eq 3 ]
```

```
    then
```

```
        echo "Skipped "  
        continue  
    fi  
done  
echo "finished"
```

A terminal window with a dark purple background. The title bar shows a window icon and the text "prasad@prasad". The terminal content shows a script being executed. The prompt is "prasad@prasad-VirtualBox:~\$". The command is "./brecon.sh". The output is "break statement:", followed by a list of numbers 1, 2, 3, then "break", then "finished". Then "continue statement:", followed by a list of numbers 1, 2, 3, then "Skipped", then 4, 5, then "finished". The prompt returns to "prasad@prasad-VirtualBox:~\$".

```
prasad@prasad-VirtualBox:~$ ./brecon.sh  
break statement:  
1  
2  
3  
break  
finished  
  
continue statement:  
1  
2  
3  
Skipped  
4  
5  
finished  
prasad@prasad-VirtualBox:~$
```