Contents

1 Display Permission

Problem 1 *Write a program to a shell script that will read a directory name from the terminal and will display only the name and permission of the files*

Code.

```
#!/bin/bash
read -p "Enter directory name:" direc
ls -l $direc | awk '{print $1, $9}'
```

Output.

```
-rw-rw-r--
-rw-rw-r--
drwxrwxr-x
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
-rw-rw-r--
116.193.143.137.html
124442-Pulse-Glass.tar.gz
16___1366x768_wallpaper_pack_by_jhasenfusphoto-d4ddjpd
16___1366x768_wallpaper_pack_by_jhasenfusphoto-d4ddjpd.zip
17017156_1892189880877953_1020699469635668422_o.jpg
171217-Breeze-GRUB2.tar.gz
240P_400K_113924551.mp4
7b0cff72-9a21-4567-8353-5d87c4696e26.html
AdbeRdr9.5.5-1_i386linux_enu.deb
android-ndk-r13b-linux-x86_64.zip
android-studio-ide-143.3101438-linux.zip
assignment-1.doc
```

2 Maximum

Problem 2 *Write a shell script that will find the maximum from the given three nos.*

Code.

```
#!/bin/bash
read -p "Enter first number:" a
echo ""
```

```
read -p "Enter second number:" b
echo ""
read -p "Enter third number:" c
echo ""
if [ $a -gt $b ]
then if [ $a -gt $c ]
then echo "$a is the greatest"
else echo "$c is the greatest"
fi
elif [ $b -gt $c ]
then echo "$b is the greatest"
else echo "$c is the greatest"
fi
```

Output.

```
Enter first number:25
Enter second number:-3
Enter third number:5
25 is the greatest
```

3 Directory checking

Problem 3 *Write a shell script that will read a file/directory name from the terminal, check whether that file/directory is in the current directory. If it exists in the current directory, display whether it is file or directory.*

Code.

```
#!/bin/bash
read -p "Enter a name of directory or file:" name
echo ""
if [ -f $name ]
then echo "File is in directory"
elif [ -d $name ]
then echo "It is a sub-directory"
else echo "Doesn't exist"
fi
```

Output.

```
Enter a name of directory or file:mouri
Doesn't exist
Enter a name of directory or file:exist.sh
File is in directory
Enter a name of directory or file:prog
It is a sub-directory
```

4 Directories in sorted order

Problem 4 *Write a shell script that will read a directory name from the terminal then it will display all the directories followed by the files in the sorted order.*

Code.

```
#!/bin/bash
read -p "Enter a direc:" direc
ls --group-directories-first $direc
```

Output.

```
Downloaded by Variety
images
Wallpapers
Webcam
15416884_1239745149428837_1970874045_n (3rd copy).jpg
15416884_1239745149428837_1970874045_n (another copy).jpg
15416884_1239745149428837_1970874045_n (copy).jpg
15416884_1239745149428837_1970874045_n.jpg
```

5 GCD if two numbers

Problem 5 *Write a shell script that computes the gcd of two numbers.*

Code.

```
#!/bin/bash
gcd()
{
read -p "Enter first: " a
read -p "Enter second number: " b
r=1
until [ $r -eq 0 ]
do
let "r= $a % $b "
a=$b
b=$r
done
echo "HCF is: "$a
}
gcd $a $b
```

Output.

```
Enter first: 5
Enter second number: 25
HCF is: 5
```

6 Fibonacci numbers

Problem 6 *Write a shell script to generate a Fibonacci series of length 'n' with the first two nos of the series being 3 and 5 respectively.*

Code.

```
#!/bin/bash
fibo()
{
a=3
b=5
read -p "Enter no of terms to generate: " n
echo -n "$a "
echo -n "$b "
n=$((n-2))
until [ $n -eq 0 ]
do
c=$(($a+$b))
echo $c|bc
a=$b
b=$c
n=$((n-1))
done
echo ""
}
fibo $n
```

Output.

```
3 5 8 13 21 34 55 89 144 233
```

7 Factorial

Problem 7 *Write a shell scripts to calculate the factorial of a integer 'n'.*

Code.

```
#!/bin/bash
read -p "Enter a number:" a
seq -s "*" 1 $a | bc
```

Output.

```
120
```

8 Sort a list of n numbers

Problem 8 *Write a shell program to sort a list of 'n' no.*

Code.

```
#!/bin/bash
arr=(8 7 9)
sorted=( $( printf "%s\n" "${arr[@]}" | sort -n ) )
echo ${sorted[*]}
```

Output.

```
7 8 9
```

9 Change background and foreground color

Problem 9 *Write a shell program to change the foreground and background color of terminal*

Code.

```
read -p "Enter foreground color: " foregrd
read -p "Enter background color: " bckgrd
setterm -term linux -back $bckgrd -fore $foregrd -clear
```

Output.

10 Use different options of grep command

Problem 10 *Write a shell program to demonstrate various use of "grep" command*

Code.

```
read -p "Enter the file name: " file
read -p "Enter the pattern: " key
grep -i $key $file #prints numbers of lines ignoring case
echo ""
grep "R*" $file #prints lines where string is starting with    R
echo ""
grep -A 3 "Rohit" $file #prints three lines after line containing    Rohit
echo ""
grep -w "R" $file #prints
echo ""
grep -c $key $file
echo ""
grep -n $key $file
echo ""
```

Output.

```
Enter the file name: inp.txt
Enter the pattern: R
Rudra doesn't like Tokon
Rohit a doesn't like tokon
Rahul doesn't like tokon
Sumitra doesn't like tokon
Rudra doesn't like Tokon
Rohit a doesn't like tokon
Debaa doesn't like tokon
Rahul doesn't like tokon
Sumitra doesn't like tokon
Supi like tokon
LOLOLOLOLOL
Rohit a doesn't like tokon
Debaa doesn't like tokon
Rahul doesn't like tokon
Sumitra doesn't like tokon
```

11 Search from a list of n numbers

Problem 11 *Write a shell program that will accept 'n' nos from the terminals and will search the position of a given no in the supplied nos*

Code.

```
#!/bin/bash
read -p "Enter limit of numbers: " n
flag=1
for (( i = 0; i < n; i++ )); do
read a[i]
done
read -p "Enter no to be searched: " key
for (( i = 0; i < n; i++ )); do
if [ "$key" -eq ${a[i]} ]
then
echo " Key found in " $i
flag=0
fi
done
if [ "$flag" -eq "1" ]
then
echo "not found"
fi
```

Output.

```
Enter limit of numbers: 6
1 3 5 12 6 -1
Enter no to be searched: -1
Key found in 5
Enter limit of numbers: 6
1 3 5 12 6 -1
Enter no to be searched: 2
not found
```

12 Show contents of a file

Problem 12 *Write a shell program that will show the contents of a file*

Code.

```
read -p "Enter the file name: " file$
cat -A $file$
```

Output.

```
Enter the file name: hello.txt
Hello World!
```

13 Show only even numbers from an aray

Problem 13 *Show only even numbers from an aray*

Code.

```
#!bin/bash
a="1 2 3 4 5 6 7 8 9 10"
for var in $a
do
even=$((var%2))
if [ "$even" -eq 0 ]
then
echo $var
fi
done
```

Output.

```
2 4 6 8 10
```

14 Use pipe to demonstrate filter and grep

Problem 14 *Use pipe to demonstrate filter and grep*

Code.

```
read -p "Enter file name: " file
read -p "Enter the key: " key
cat $file | grep -i "$key"
```

Output.

```
Enter file name: inp.txt
Enter the key: R
Rudra doesn't like Tokon
Rohit a doesn't like tokon
Rahul doesn't like tokon
Sumitra doesn't like tokon
```

15 Fork

Problem 15 *Write a program to demonstrate forking in C*

Code.

```
#include <unistd.h>
#include <stdio.h>
int main()
{
        pid_t t;
        t = fork();
        if (t > 0) {
```

```
                printf("I am parent\n");
        } else if (t == 0) {
        printf("CHILD\n");
        } else {
                printf("ERROR\n");
        }
        printf("This line is common\n");
}
```

Output.

```
I am parent
This line is common
CHILD
This line is common
```

16 PID

Problem 16 *Write a program to create a child process in C and print the PID of parent and child*

Code.

```
#include <unistd.h>
#include <stdio.h>
int main()
{
        pid_t t;
        t = fork();
        if (t > 0) {
                printf("I am parent. My pid is: %d\n",getpid());
        } else if (t == 0) {
                printf("CHILD. My ppid is %d\n",getppid());
        } else {
                printf("ERROR\n");
        }
        printf("This line is common\n");
}
```

Output.

```
I am parent. My pid is: 7052
This line is common
CHILD. My ppid is 7052
This line is common
```

17 Process Hierarchy and Zombie processes

Problem 17 *Write a program to demonstrate zombie processes*

Code.

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main()
{
        pid_t t = vfork();
        if (t > 0) {
                // We are in A
                printf("In A. Pid=%d\n",getpid());
                pid_t te = vfork();
                if (te > 0) {
                        printf("In A. Pid: %d, Ppid=%d\n",
                        getpid(),getppid());
                        // We are in A
                } else if (te == 0) {
                        printf("In E. Pid= %d, PPid=%d\n",
                        getpid(),getppid());
                        // We are in E
                }
                exit(0);
        } else if (t == 0) {
                // We are in B
                printf("In B. Pid=%d, PPid=%d\n",getpid(),getppid());
                pid_t tee = vfork();
                if (tee > 0) {
                        printf("IN B,  Pid= %d, PPid=%d\n",
                        getpid(),getppid());
                        // We are in B
                } else if (tee == 0) {
                        // We are in C
                        printf("In C  Pid= %d, PPid=%d\n",
                        getpid(),getppid());
                        pid_t teee = vfork();
                        if (teee > 0) {
                                printf("In B  Pid= %d, PPid=%d\n",
                                getpid(),getppid());
                                // We are in B
                        } else if (teee == 0) {
                                printf("In D  Pid= %d, PPid=%d\n",
                                getpid(),getppid());
                                // We are in D
                        }
                        sleep(10);
                        exit(0);
                }
                sleep(10);
                exit(0);
        }
        sleep(10);
```

```
        exit(0);
}
```

Output.