

Contents

1	Display Permission	2
2	Maximum	2
3	Directory checking	3
4	Directories in sorted order	3
5	GCD of two numbers	4
6	Fibonacci numbers	4
7	Factorial	5
8	Sort a list of n numbers	5
9	Change background and foreground color	6
10	Use different options of grep command	6
11	Fork	7
12	Process Hierarchy and Zombie processes	7

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--
-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 of 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
fibonacci()
{
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 ""
}
fibonacci $n
```

Output.

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

7 Factorial

Problem 7 Write a shell script 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 9 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 token
Rahul doesn't like token
Sumitra_doesn't like token
Rudra doesn't like Tokon
Rohit_a_doesn't like token
Debaa doesn't like token
Rahul_doesn't like token
Sumitra doesn't like token
Supi like token
LOLOLOLOLOL
Rohit_a_doesn't like token
Debaa doesn't like token
Rahul_doesn't like token
Sumitra doesn't like token
```

11 Fork

Problem 10 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
```

12 Process Hierarchy and Zombie processes

Problem 11 Write a program to demonstrate zombie processes

Code.

```
#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);
            //while(1);
            exit(0);
        }
        //while(1);
        sleep(10);
        exit(0);
    }
}

```

Output.