OPERATING SYSTEM LAB 4

S ABHISHEK

AM.EN.U4CSE19147

1. Write a shell script to generate emails in the given format and write it into a file. Your script should accept sender and recipient email id's and subject as command line arguments.

From: abc@domain1.com To: xx@domain.com Cc: yy@domain.com

Subject: Subject 1

This email is generated by my shell script.

Thanks and regards

S4 CSE student

Amritapuri

```
#!/bin/bash
echo -e "\nFrom : $1    To : $2    cc : abc@domain.com"
echo -e "\nSubject : $3"
echo -e "\n            This email is generated by my shell script."
echo -e "\nThanks and regards"
echo "S4 CSE student"
echo -e "Amritapuri\n"
```

```
sabhishek@S:~/Downloads$ ./email.sh abhishek@gmail.com abhi@gmail.com 'This is the Exa
mple'

From : abhishek@gmail.com     To : abhi@gmail.com     cc : abc@domain.com

Subject : This is the Example

                This email is generated by my shell script.

Thanks and regards
S4 CSE student
Amritapuri
```

2. Modify Question 1 to allow user to enter text at the beginning of email content, by passing it as a command line argument.

```
#!/bin/bash

echo -e "\n$1"
echo -e "\nFrom : $2    To : $3    cc : abc@domain.com"
echo -e "\nSubject : $4"
echo -e "\n                This email is generated by my shell script."
echo -e "\nThanks and regards"
echo "S4 CSE student"
echo -e "Amritapuri\n"
```

```
sabhishek@S:~/Downloads$ ./Email_Modified.sh 'This is a Exmaple Email' abhishek@gmail.
com abhi@gmail.com 'This is the Example'

This is a Exmaple Email

From : abhishek@gmail.com     To : abhi@gmail.com     cc : abc@domain.com

Subject : This is the Example

                This email is generated by my shell script.

Thanks and regards
S4 CSE student
Amritapuri
```

3. Write a shell script to print all the primes below a given number.

```
#!/bin/bash
prime()
{
for ((i=2; i<=$1; i++ ))
do
f=0
if [ $i -lt 1 ]
then
continue
fi
for ((j=2; j<$i; j++ ))
do
if [ $(($i % $j)) == 0 ]
        then
                f=1
                break
        fi
done
if [ $f -eq 0 ]
```

```
then
        echo -n -e "$i "
fi
done
echo ""
}

prime $1
```

```
sabhishek@S:~/Downloads$ ./Prime.sh 30
2 3 5 7 11 13 17 19 23 29
sabhishek@S:~/Downloads$ ./Prime.sh 20
2 3 5 7 11 13 17 19
sabhishek@S:~/Downloads$ ./Prime.sh 10
2 3 5 7
```

4. Write a shell script to print the first n Fibonacci numbers.

```
#!/bin/bash
fibonacci()
{
s1=0
s2=1
```

```
s3=0
echo -n -e "Fibonnaci Series : "
for (( i=1;i <= $1;i++))
do
echo -n -e "$s1 "
s3=$((s1+s2))
s1=$s2
s2=$s3
done
echo ""
}
fibonacci $1
```

```
sabhishek@S:~/Downloads$ ./Fibonacci.sh 10
Fibonnaci Series : 0 1 1 2 3 5 8 13 21 34
sabhishek@S:~/Downloads$ ./Fibonacci.sh 20
Fibonnaci Series : 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
```

5. Write a shell script to generate a multiplication table.

a. Interactive version: The program should accept an integer n given by the user and should print the multiplication table of that n.

b. Command line arguments version: The program should take the value of n from the arguments followed by the command.

c. Redirection version: The value of n must be taken from a file using input redirection.

```
#!/bin/bash
table()
{
for((i=1;i<=$1;i++))
do
echo "$i * $1 = $((i*$1))"
done
}
 read -p "Enter the Number : " num
table $num
```

```
sabhishek@S:~/Downloads$ ./Mul_Int.sh
Enter the Number : 10
1 * 10 = 10
2 * 10 = 20
3 * 10 = 30
4 * 10 = 40
5 * 10 = 50
6 * 10 = 60
7 * 10 = 70
8 * 10 = 80
9 * 10 = 90
10 * 10 = 100
```

```
sabhishek@S:~/Downloads$ ./Mul_Int.sh 20
Enter the Number : 20
1 * 20 = 20
2 * 20 = 40
3 * 20 = 60
4 * 20 = 80
5 * 20 = 100
6 * 20 = 120
7 * 20 = 140
8 * 20 = 160
9 * 20 = 180
10 * 20 = 200
11 * 20 = 220
12 * 20 = 240
13 * 20 = 260
14 * 20 = 280
15 * 20 = 300
16 * 20 = 320
17 * 20 = 340
18 * 20 = 360
19 * 20 = 380
20 * 20 = 400
```

```
#!/bin/bash

table()

{

for((i=1;i<=$1;i++))

do

echo "$i * $1 = $((i*$1))"

done

}
```

table $1

```
sabhishek@S:~/Downloads$ ./Mul_Com.sh 20
1 * 20 = 20
2 * 20 = 40
3 * 20 = 60
4 * 20 = 80
5 * 20 = 100
6 * 20 = 120
7 * 20 = 140
8 * 20 = 160
9 * 20 = 180
10 * 20 = 200
11 * 20 = 220
12 * 20 = 240
13 * 20 = 260
14 * 20 = 280
15 * 20 = 300
16 * 20 = 320
17 * 20 = 340
18 * 20 = 360
19 * 20 = 380
20 * 20 = 400
```

```
sabhishek@S:~/Downloads$ ./Mul_Com.sh 10
1 * 10 = 10
2 * 10 = 20
3 * 10 = 30
4 * 10 = 40
5 * 10 = 50
6 * 10 = 60
7 * 10 = 70
8 * 10 = 80
9 * 10 = 90
10 * 10 = 100
```

```
#!/bin/bash


table()

{

for((i=1;i<=$1;i++))

do

echo "$i * $1 = $((i*$1))"

done

}


read num

table $num
```

```
sabhishek@S:~/Downloads$ cat > 1.txt <<eof
> 10
> eof
sabhishek@S:~/Downloads$ ./Mul_File.sh < 1.txt
1 * 10 = 10
2 * 10 = 20
3 * 10 = 30
4 * 10 = 40
5 * 10 = 50
6 * 10 = 60
7 * 10 = 70
8 * 10 = 80
9 * 10 = 90
10 * 10 = 100
sabhishek@S:~/Downloads$ cat 1.txt
10
```

```
sabhishek@S:~/Downloads$ ./Mul_File.sh < 1.txt
1 * 20 = 20
2 * 20 = 40
3 * 20 = 60
4 * 20 = 80
5 * 20 = 100
6 * 20 = 120
7 * 20 = 140
8 * 20 = 160
9 * 20 = 180
10 * 20 = 200
11 * 20 = 220
12 * 20 = 240
13 * 20 = 260
14 * 20 = 280
15 * 20 = 300
16 * 20 = 320
17 * 20 = 340
18 * 20 = 360
19 * 20 = 380
20 * 20 = 400
sabhishek@S:~/Downloads$ cat 1.txt
20
```

6. Using function write a shell script to find gcd of two numbers.

```
#!/bin/bash
gcd()
{
gcd=0
for((i=1;i<=$1 && i<=$2;i++))
do
if [ $(($1 % i)) == 0 ]
```

```
then
if [ $(($2 % i)) == 0 ]
then
gcd=$i
fi
fi
done
echo "GCD = $gcd"
}
gcd $1 $2
```

```
sabhishek@S:~/Downloads$ ./Gcd.sh 24 36
GCD = 12
sabhishek@S:~/Downloads$ ./Gcd.sh 81 153
GCD = 9
sabhishek@S:~/Downloads$ ./Gcd.sh 2 9
GCD = 1
sabhishek@S:~/Downloads$ ./Gcd.sh 18 36
GCD = 18
sabhishek@S:~/Downloads$ ./Gcd.sh 18 30
GCD = 6
```

7. Using Recursion find factorial of a number

```
#!/bin/bash
factorial()
{
if (( $1 <= 1 ))
then
        echo 1
        else
        echo $(( $1 * $(factorial $(( $1 - 1 )))))
        fi
}

factorial $1
```

```
sabhishek@S:~/Downloads$ ./Factorial.sh 5
120
sabhishek@S:~/Downloads$ ./Factorial.sh 4
24
sabhishek@S:~/Downloads$ ./Factorial.sh 3
6
sabhishek@S:~/Downloads$ ./Factorial.sh 2
2
sabhishek@S:~/Downloads$ ./Factorial.sh 1
1
sabhishek@S:~/Downloads$ ./Factorial.sh 6
720
```

8. Write shell script to show various system configuration like:

a. Currently logged user and his long name

b. Current shell

c. Home directory

d. Operating system type

e. Current path setting

f. Current working directory

g. All available shells

```
#!/bin/bash


echo ""
echo "################ Currently logged user
################"
whoami
echo ""
echo "################ Current shell ################"
echo "$SHELL"


echo ""
echo "################ Home Directory ################"
cd ~ | ls
```

```
echo ""

echo "################ Operating system type ################"

egrep '^(VERSION|NAME)=' /etc/os-release


echo ""

echo "################ Current path setting ################"

echo $PATH


echo ""

echo "################ Current working directory ################"

pwd

echo ""

echo "################ All available shells ################"

cat /etc/shells
```

```
sabhishek@S:~/Downloads$ ./Details.sh

############### Currently logged user ###############
sabhishek

############### Current shell ###############
/bin/bash

############### Home Directory ###############
Details.sh          email.sh       Fibonacci.sh  Mul_Com.sh    Mul_Int.sh  Prime.sh
Email_Modified.sh   Factorial.sh   Gcd.sh        Mul_File.sh   OS

############### Operating system type ###############
NAME="Ubuntu"
VERSION="20.10 (Groovy Gorilla)"

############### Current path setting ###############
/home/sabhishek/.local/bin:/home/sabhishek/bin:/usr/local/sbin:/usr/local/bin:/usr/sbi
n:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin

############### Current working directory ###############
/home/sabhishek/Downloads
```

```
############### All available shells ###############
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/usr/bin/bash
/bin/rbash
/usr/bin/rbash
/bin/dash
/usr/bin/dash
```

Thankyou!!