DOS LAB ASSIGNMENT- 01

1. Write the commands to create the following directory hirarchy: -> DOS_Regdno ->DOSass1->dir1

```
shubhi@Shubhangini:~$ mkdir DOS_9442
shubhi@Shubhangini:~$ cd DOS_9442
shubhi@Shubhangini:~/DOS_9442$ mkdir DOSass1
shubhi@Shubhangini:~/DOS_9442$ dc DOSass1
Command 'dc' not found, but can be installed with:
sudo apt install dc
shubhi@Shubhangini:~/DOS_9442$ cd DOSass1
shubhi@Shubhangini:~/DOS_9442/DOSass1$ mkdir dir1
shubhi@Shubhangini:~/DOS_9442/DOSass1$ cd dir1
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$
```

2. Write the commands to create another directory with name dir2 in directory DOSass1 and make dir2 as the current working directo

```
shubhi@Shubhangini:~$ cd DOS_9442
shubhi@Shubhangini:~/DOS_9442$ cd DOSass1
shubhi@Shubhangini:~/DOS_9442/DOSass1$ cd dir1
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cd -
/home/shubhi/DOS_9442/DOSass1
shubhi@Shubhangini:~/DOS_9442/DOSass1$ mkdir dir2
shubhi@Shubhangini:~/DOS_9442/DOSass1$ cd mkdir
-bash: cd: mkdir: No such file or directory
shubhi@Shubhangini:~/DOS_9442/DOSass1$ cd dir2
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir2$
```

3.Write the command to delete the directory dir2 , when DOS_Regdno will be the current working directory.

```
shubhi@Shubhangini:~/DOS_9442$ cd DOSass1
shubhi@Shubhangini:~/DOS_9442/DOSass1$ cd dir2
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir2$ cd ../
shubhi@Shubhangini:~/DOS_9442/DOSass1$ cd ../
shubhi@Shubhangini:~/DOS_9442$
```

4.Write the command to create a file named as file1 using cat command inside dir1. Write your name, regdno, branch, semester and section in file1. Then display the content of the file.

```
shubhi@Shubhangini:~/DOS_9442$ cat> file1.txt
Q5>Name:Shubhangini
RegNo.: 2241019442
Branch: CSE
Sem: 5th
Sec: 31
^C
shubhi@Shubhangini:~/DOS_9442$ cat file1.txt
Q5>Name:Shubhangini
RegNo.: 2241019442
Branch: CSE
Sem: 5th
Sec: 31
shubhi@Shubhangini:~/DOS_9442$
```

5. Write the command to create a file named as file2 using cat command inside dir1. Write your semester wise SGPA in file2.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat> file2.txt
Q5)Previous was Q4
Sem1: 8
Sem2: 9
Sem3: 9.48
Sem4: 9.66
^C
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat file2.txt
Q5)Previous was Q4
Sem1: 8
Sem2: 9
Sem3: 9.48
Sem4: 9.66
```

6.Create a file named as file3 storing content of file1 merged with content of file2.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat>file1.txt
This is my File1.
^C
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat>file2.txt
This is my File2.
^C
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat file1.txt file2.txt>file3.txt
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat file3.txt
This is my File1.
This is my File2.
```

7. Write the command to rename file2 as markinfo. (mv – to rename)

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ mv file2.txt markinfo
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls
file1.txt  file3.txt  markinfo
```

8. Write the command to copy the content of file1 to reginfo. (cp- Copy)

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cp file1.txt regInfo
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat regInfo
This is my File1.
```

9.Write the command to display the inode (index Node)values of file1,markinfo, reginfo.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls -i
46446 file1.txt  46447 file3.txt  46445 markinfo  46448 regInfo
```

10. Write the command to delete file1. (rm – to remove/delete)

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ rm file1.txt
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls
file3.txt  markinfo  regInfo
```

11.Write the command to count the number of lines, words, characters in markinfo.(wc - wordcont)

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ wc -l markinfo
1 markinfo
```

12. Write the command to create a file named as Personalinfo inside dir1. Write your name, regdno, address in the file.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat>PersonalInfo.txt
Name: Shubhangini
Reg : 2241019442
Add : Jagamara
^C
```

13. Write the command to display the content of markinfo in reverse order.

Command : sort –r filename.txt

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat>markInfo.txt
1
2
3
4
5
^C
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ sort -r markInfo.txt
5
4
3
2
1
```

14. Check the output of the following command: cmp reginfo personalinfo diff reginfo personalinfo

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat>regInfo.txt
This is RegInfo
Thank You.^C
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat>PerInfo.txt
This is Personal Info
Thank You.^C
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cmp regInfo PerInfo
cmp: PerInfo: No such file or directory
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cmp regInfo.txt PerInfo.txt
regInfo.txt PerInfo.txt differ: byte 9, line 1
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ diff regInfo.txt PerInfo.txt
1c1
< This is RegInfo
---
> This is Personal Info
```

15) Write a command to count the number of files in the current working directory and display that number.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls | wc -l
7
```

16) Write a command to include all the file names present in a current working directory in a file named as filelist without causing filelist to be included in the names.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls | grep -v "FileList">FileList
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ cat FileList
PerInfo.txt
PersonalInfo.txt
file3.txt
markInfo.txt
markinfo
regInfo
regInfo.txt
```

17. Write a command to give write permission to all the users of file reginfo.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ chmod a+w regInfo.txt
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls -l
total 32
-rw-r--r-- 1 shubhi shubhi 81 Jan  7 18:22 FileList
-rw-r--r-- 1 shubhi shubhi 22 Jan  7 18:15 PerInfo.txt
-rw-r--r-- 1 shubhi shubhi 50 Jan  7 17:59 PersonalInfo.txt
-rw-r--r-- 1 shubhi shubhi 36 Jan  7 17:49 file3.txt
-rw-r--r-- 1 shubhi shubhi 10 Jan  7 18:11 markInfo.txt
-rw-r--r-- 1 shubhi shubhi 18 Jan  7 17:49 markinfo
-rw-r--r-- 1 shubhi shubhi 18 Jan  7 17:52 regInfo
-rw-rw-rw- 1 shubhi shubhi 16 Jan  7 18:14 regInfo.txt
```

18.Write a command to discard write permission from group users group users of file reginfo.

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ chmod g-w regInfo.txt
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls -l
total 32
-rw-r--r-- 1 shubhi shubhi 81 Jan  7 18:22 FileList
-rw-r--r-- 1 shubhi shubhi 22 Jan  7 18:15 PerInfo.txt
-rw-r--r-- 1 shubhi shubhi 50 Jan  7 17:59 PersonalInfo.txt
-rw-r--r-- 1 shubhi shubhi 36 Jan  7 17:49 file3.txt
-rw-r--r-- 1 shubhi shubhi 10 Jan  7 18:11 markInfo.txt
-rw-r--r-- 1 shubhi shubhi 18 Jan  7 17:49 markinfo
-rw-r--r-- 1 shubhi shubhi 18 Jan  7 17:52 regInfo
-rw-r--rw- 1 shubhi shubhi 16 Jan  7 18:14 regInfo.txt
```

19. Write the command to set rwx permissions for all the users of file reginfo.

- 7 for the owner (read, write, execute).
- 7 for the group (read, write, execute).
- 7 for others (read, write, execute).

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ chmod 777 regInfo.txt
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ ls -l
total 32
-rw-r--r-- 1 shubhi shubhi 81 Jan  7 18:22 FileList
-rw-r--r-- 1 shubhi shubhi 22 Jan  7 18:15 PerInfo.txt
-rw-r--r-- 1 shubhi shubhi 50 Jan  7 17:59 PersonalInfo.txt
-rw-r--r-- 1 shubhi shubhi 36 Jan  7 17:49 file3.txt
-rw-r--r-- 1 shubhi shubhi 10 Jan  7 18:11 markInfo.txt
-rw-r--r-- 1 shubhi shubhi 18 Jan  7 17:49 markinfo
-rw-r--r-- 1 shubhi shubhi 18 Jan  7 17:52 regInfo
-rwxrwxrwx 1 shubhi shubhi 16 Jan  7 18:14 regInfo.txt
```

20. Differentiate between following commands:

Command	Execution Style	Output
`date; pwd`	Independent execution	Displays the date, then the current directory
`date; pwd	wc -l`	Counts lines only from `pwd`
`(date; pwd)	wc -l`	Groups and counts both commands

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ date;pwd
Tue Jan  7 18:29:58 UTC 2025
/home/shubhi/DOS_9442/DOSass1/dir1
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ date;pwd|wc -l
Tue Jan  7 18:30:12 UTC 2025
1
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ (date;pwd)|wc -l
2
```

21. Interpret the output of the following commmands:

```
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo *
FileList PerInfo.txt PersonalInfo.txt file3.txt markInfo.txt markinfo
 regInfo regInfo.txt
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo ***
FileList PerInfo.txt PersonalInfo.txt file3.txt markInfo.txt markinfo
 regInfo regInfo.txt
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo '***'
***
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo \***
***
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo \**\*
***
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo \*\*\*
***
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo */*
*/*
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo Don't do this
> Hey its Shubhangini
> ^C
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo Hello # WOrld
Hello
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo "Hello # WOrld"
Hello # WOrld
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo date
date
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo 'date'
date
shubhi@Shubhangini:~/DOS_9442/DOSass1/dir1$ echo `date`
Tue Jan 7 18:35:20 UTC 2025
```

DOS LAB ASSIGNMENT- 02

1.Write a shell script named as prog for merge the content of files a.txt, b.txt, and c.txt sort them and save the result in a file called result and display the sorted output on the screen.

```
shubhi@Shubhangini:~/DOS_9442$ cat > a.txt
Kumari
^C
shubhi@Shubhangini:~/DOS_9442$ cat > b.txt
Shubhangini
^C
shubhi@Shubhangini:~/DOS_9442$ cat > c.txt
SuperWomen
^C
shubhi@Shubhangini:~/DOS_9442$ sort -n a.txt b.txt c.txt>result
shubhi@Shubhangini:~/DOS_9442$ cat result
Kumari
Shubhangini
SuperWomen
```

2. Write a shell script named as systeminfo that will display the information about the login name of the user, name of the Unix system used by the user, type of the SHELL, Path of current working directory of the user and list of file contain in current working directory.

```
shubhi@Shubhangini:~/DOS_9442$ nano systeminfo.sh
shubhi@Shubhangini:~/DOS_9442$ chmod a+x systeminfo.sh
shubhi@Shubhangini:~/DOS_9442$ ./systeminfo.sh
Login User : shubhi
Unix System: GNU/Linux
Shell type: /bin/bash
PWD: /home/shubhi/DOS_9442
No. of file in thr directory :
total 28
drwxr-xr-x 4 shubhi shubhi 4096 Jan  7 17:29 DOSass1
-rw-r--r-- 1 shubhi shubhi    8 Jan  7 19:12 a.txt
-rw-r--r-- 1 shubhi shubhi   12 Jan  7 19:12 b.txt
-rw-r--r-- 1 shubhi shubhi   11 Jan  7 19:12 c.txt
-rw-r--r-- 1 shubhi shubhi   68 Jan  7 17:42 file1.txt
-rw-r--r-- 1 shubhi shubhi   31 Jan  7 19:13 result
-rwxr-xr-x 1 shubhi shubhi  146 Jan  7 19:35 systeminfo.sh
```

```
 GNU nano 7.2                    systeminfo.sh
echo "Login User : $USER"
echo "Unix System: $(uname -o)"
echo "Shell type: $SHELL"
echo "PWD: $PWD"
echo "No. of file in thr directory :"
ls -l
```

3. Write a shell script named as dtcal for displaying both the system date and calendar for specific month, say march 2022, in the given format:-Date : specific date Calender : current calender.

```
shubhi@Shubhangini:~/DOS_9442$ nano dtcl.sh
shubhi@Shubhangini:~/DOS_9442$ chmod a+x dtcl.sh
shubhi@Shubhangini:~/DOS_9442$ ./dtcl.sh
Date: Tue Jan  7 19:50:47 UTC 2025
Calender:       March 2022
Su Mo Tu We Th Fr Sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

```
echo "Date: $(date)"
echo "Calender: $(cal 03 2022)"
ls -l
```

4. Write a shell script named as nvwc which will display the filename and linecount,wordcount and char count of the file dtcal in the following format: Filename: dtcal Line count: - Word count: - Charcout: -

```
File="dtcl"
lc=$(wc -l < "$File")
wc=$(wc -w < "$File")
cc=$(wc -c < "$File")

echo "File Name: $File"
echo "Line Count: $lc"
echo "Word Count: $wc"
echo "Char count: $cc"
```

```
shubhi@Shubhangini:~/DOS_9442$ nano nvwc.sh
shubhi@Shubhangini:~/DOS_9442$ chmod a+x nvwc.sh
shubhi@Shubhangini:~/DOS_9442$ ./nvwc.sh
./nvwc.sh: line 2: dtcl: No such file or director
./nvwc.sh: line 3: dtcl: No such file or director
./nvwc.sh: line 4: dtcl: No such file or director
File Name: dtcl
Line Count:
Word Count:
Char count:
```

5. Write a shell script named as nvwc2 which will display the filename and linecount, word count and char count of any file given as argument to nvwc2 in the following format: filename linecount , wordcount , charcount file1 - - -

```
File="$1"
lc=$(wc -l < "$File")
wc=$(wc -w < "$File")
cc=$(wc -c < "$File")

echo "File Name: $File"
echo "Line Count: $lc"
echo "Word Count: $wc"
echo "Char Count: $cc"
```

Same as Q4

6.Write a shell script named as darg to display the total number of command line arguments along with the first two arguments. -Modify the script to display all the arguments.

```
  GNU nano 7.2                    d
echo "Total Number of args: $#"
echo "1st arg: ${1: -None}"
echo "2nd arg: ${2: -None}"
echo "All Args: $@"
```

```
shubhi@Shubhangini:~/DOS_9442$ nano darg.sh
shubhi@Shubhangini:~/DOS_9442$ chmod a+x darg.sh
shubhi@Shubhangini:~/DOS_9442$ ./darg.sh arg1 arg2 arg3
Total Number of args: 3
1st arg: arg1
2nd arg: arg2
All Args: arg1 arg2 arg3
```

7. Write a shell script named as ndisp that will take three command line arguments specifying the value of n, m and a filename and display the first n number of lines and last m number of lines of the file given as argument.

```
  GNU nano 7.2                    n
n="$1"
m="$2"
file="$3"
echo "First $n lines of $file:"
head -n "$n" "$file"
echo "Last $m lines of $file:"
tail -n "$m" "$file"
```

```
shubhi@Shubhangini:~/DOS_9442$ nano ndisp.sh
shubhi@Shubhangini:~/DOS_9442$ chmod a+x ndisp.sh
shubhi@Shubhangini:~/DOS_9442$ cat>myFile
1
2
3
4
Hey
Shubhi
5
^C
shubhi@Shubhangini:~/DOS_9442$ ./ndisp.sh 2 4 myFile
First 2 lines of myFile:
1
2
Last 4 lines of myFile:
4
Hey
Shubhi
5
```

Laboratory Assignments 3
Subject: Design Principles of Operating Systems
Subject code: CSE 3249

Assignment 3: Shell Programming using user defined variables, arithmetic operators, conditional statements.

1. Write a shell script **iaop** to perform integer arithmetic on two numbers, where the value of the two numbers will be given during runtime.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano iaop.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x iaop.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./iaop.sh
enter two numbers
5
6
Sum: 11
Difference: -1
Product: 30
Quotient: 0
Remainder: 5
```

```
 GNU nano 8.1                                    iaop.sh *
echo "enter two numbers"
read num1
read num2
sum=$(($num1 + $num2))
echo "Sum: $sum"
diff=$(( $num1 - $num2))
echo "Difference: $diff"
prod=$(($num1 * $num2))
echo "Product: $prod"
quo=$(($num1 / $num2))
echo "Quotient: $quo"
rem=$(($num1 % $num2))
echo "Remainder: $rem"
```

2. Write a shell script **faop** to perform floating point arithmetic on two numbers, where the value of the two numbers will be given during runtime.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano faop.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x faop.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./faop.sh
enter two numbers
5.6
2.4
Sum: 8.0
Difference: 3.2
Product: 13.44
Quotient: 2.33
Remainder: .008
```

```
 GNU nano 8.1                                    faop.sh
echo "enter two numbers"
read num1
read num2

sum=$(echo "scale=2; $num1 + $num2"| bc )
echo "Sum: $sum"
diff=$(echo "scale=2; $num1 - $num2"| bc)
echo "Difference: $diff"
prod=$(echo "scale=2; $num1 * $num2"| bc)
echo "Product: $prod"
quo=$(echo "scale=2; $num1 / $num2"| bc)
echo "Quotient: $quo"
rem=$(echo "scale=2; $num1 % $num2"| bc)
echo "Remainder: $rem"
```

3. Ramesh's basic salary is input through the keyboard. His dearness allowance is 40% of basic salary, and house rent allowance is 20% of basic salary. Write a program to calculate his gross salary.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano salary.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x salary.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./salary.sh
enter ramesh's basic salary
20000
Your dearness allowance is: 8000.00
Your house rent allowance is: 4000.00
Your gross salary is: 32000.00
```

```
 GNU nano 8.1                                  salary.sh *
echo "enter ramesh's basic salary"
read bas_sal
DA=$(echo "scale=2; 0.40 * $bas_sal"| bc)
echo "Your dearness allowance is: $DA"
HRA=$(echo "scale=2; 0.20 * $bas_sal"| bc)
echo "Your house rent allowance is: $HRA"
gross=$(echo "scale=2; $bas_sal + $DA + $HRA" | bc)
echo "Your gross salary is: $gross"
```

4. If a five digit number is input given through the keyboard during runtime, write a program to calculate the sum of its digits.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x sumofdigits.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano sumofdigits.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x sumofdigits.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./sumofdigits.sh
Enter a five digit number
12345
Sum : 15
```

```
 GNU nano 8.1                                  sumofdigits.sh

echo "Enter a five digit number"
read n
sum=0
while [ $n -gt 0 ]; do
r=$(($n%10))
sum=$(($sum + $r))
n=$(($n/10))
done
echo "Sum : $sum"
```

5. If cost price and selling price of an item is input through the keyboard, write a program to determine whether the seller has made profit or incurred loss. Also determine how much profit was made or loss incurred.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano profitloss.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x profitloss.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./profitloss.sh
enter cost price
200
enter selling price
300
The seller has made a profit of : 100
profit percentage: 50.00 %
```

```
 GNU nano 8.1                                profitloss.sh
echo "enter cost price"
read CP
echo "enter selling price"
read SP
profit=$((SP-CP))
loss=$((CP-SP))
if [ $profit -gt 0 ]; then
echo "The seller has made a profit of : $profit"
profit_per=$(echo "scale=2; $profit * 100 / $CP" | bc )
echo "profit percentage: $profit_per %"
elif [ $loss -gt 0 ]; then
echo "The seller has made a loss"
loss_per=$(echo "scale=2; $loss * 100 / $CP" | bc )
echo "Loss percentage: $loss_per %"
else
echo "neither profit nor loss"
fi
```

6. Write a shell script which receives any year from the keyboard and determines, whether the year is a leap year or not. If no argument is supplied the current year should be assumed.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano leapyear.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x leapyear.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./leapyear.sh
Enter a year :
2022
2022 is not a leap year .
```

```
 GNU nano 8.1                                    leapyear.sh
#!/bin/bash
echo "Enter a year : "
read year
if (( ($year % 4 == 0 && $year % 100 != 0) || $year % 400 == 0 )); then
        echo "$year is a leap year . "
else
        echo "$year is not a leap year . "
fi
```

7. Write a shell script **allow** that will display a message to enter internal mark and percentage in attendance, if the entered mark is greater than equal to 20 and entered percentage in attendance is greater that equal to 75 then display the message Allowed for Semester otherwise display the message Not allowed.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano allow.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x allow.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./allow.sh
Enter marks :
23
Enter Attendance in Percentage :
75
Allowed
```

```
 GNU nano 8.1                                    allow.sh
echo "Enter marks : "
read marks
echo "Enter Attendance in Percentage : "
read attendance
if [ $marks -ge 20 ] && [ $attendance -ge 75 ]; then
        echo "Allowed "
else
        echo "Not Allowed "
fi
```

8. Write a shell script **small3** that will compare three numbers passed as command line arguments and display the smallest one.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano small3.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x small3.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./small3.sh
Enter Number :
5
Enter Number :
4
Enter Number :
5
Smallest : 4
```

```
 GNU nano 8.1                                                small3.sh
echo "Enter Number : "
read num1
echo "Enter Number : "
read num2
echo "Enter Number : "
read num3
if [ $num1 -le $num2 ] && [ $num1 -le $num3 ]; then
    echo "Smallest : $num1 "
elif [ $num2 -le $num1 ] && [ $num2 -le $num3 ]; then
    echo "Smallest : $num2 "
else
    echo "Smallest : $num3 "
fi
```

9. Write a shell script **check_char** which will display one message to enter a character and according to the character entered it will display appropriate message from the following options:
    - You entered a lower case alphabet
    - You entered an upper case alphabet.
    - You have entered a digit.
    - You have entered a special symbol.
    - You have entered more than one character.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano check_char.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x check_char.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./check_char.sh
Enter A Character:
A
It is of Upper Case
```

```
 GNU nano 8.1                                  check_char.sh
#!/bin/bash
echo "Enter A Character: "
read char

if [[ ${#char} -gt 1 ]]; then
    echo "You Entered more than one character"
elif [[ "$char" =~ [a-z] ]]; then
    echo "It is of Lower case"
elif [[ "$char" =~ [A-Z] ]]; then
    echo "It is of Upper Case"
elif [[ "$char" =~ [0-9] ]]; then
    echo "It is a Number"
else
    echo "Special Character"
fi
```

10. Write a shell script **class_time** which will display one message to enter a day and according to the day entered it will display the DOS class time along with the room information or the message "No class on day_name" or "Holiday" for Sunday.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano class_time.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x class_time.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./class_time.sh
Enter A Day:
monday
DOS class at Room No. 109 at 11:30
```

```
 GNU nano 8.1                                  class_time.sh
echo "Enter A Day: "
read day
case "$day" in
    "monday" | "MONDAY")
        echo "DOS class at Room No. 109 at 11:30"
        ;;
    "tuesday" | "TUESDAY")
        echo "DOS class at Room No. 109 at 10:00"
        ;;
    "wednesday" | "WEDNESDAY")
        echo "DOS class at Room No. 109 at 12:00"
        ;;
    "thursday" | "THURSDAY")
        echo "DOS class at Room No. 108 at 09:00"
        ;;
    "friday" | "FRIDAY")
        echo "DOS class at Room No. 110 at 09:30"
        ;;
    "saturday" | "SATURDAY")
        echo "DOS class at Room No. 110 at 03:00"
        ;;
    "sunday" | "SUNDAY")
        echo "Holiday"
        ;;
    *)
        echo "Invalid Day"
        ;;
esac
```

11. Write a shell script **filechk** that will take two file names as command line arguments, and check whether the content of two files are same or not . If contents of two files are same, then it will display the message: Files filename1 and filename2 have same content.
then delete the second file
and display the message: So filename2 is deleted.
Otherwise display the message: Files filename1 and filename2 have different content.

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano filechk.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x filechk.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./filechk.sh leapyear.sh check_char.sh
leapyear.sh check_char.sh differ: byte 25, line 2
Files leapyear.sh and check_char.sh have different contents
```

```
 GNU nano 8.1                                                filechk.sh
#!/bin/bash
if [ $# -ne 2 ]; then
        echo "please provide exactly two filenames."
        exit 1
fi
if cmp "$1" "$2"; then
        echo "Files $1 and $2 have the same content "
        rm "$2"
        echo "$2 is deleted "
else
        echo "Files $1 and $2 have different contents "
fi
```

12. Write a shell script **calculator** that will take three command line arguments, where the first argument will specify the first operand, second argument will specify the operator and the third argument will specify the second operand and display the output of the arithmetic operation specified in the following format: op1 operator op2 = result .

If the arguments will be passed in any other sequence, it will display the message:

"Invalid input "

Enter input in following format: op1 operator op2

The symbols to be used for different operators are as follows:

Addition:	+	Subtraction:	-
Multiplication:	x	Division:	/
Modulo:	%	Exponent:	^

```
┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ nano calculator.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ chmod +x calculator.sh

┌──(munu㉿kali)-[~/Dos/assgn1]
└─$ ./calculator.sh 2 + 4
2 + 4 = 6
```

```
 GNU nano 8.1                                  calculator.sh *
#!/bin/bash
op1=$1
operator=$2
op2=$3
case "$operator" in
    +)
        result=$(echo "$op1 + $op2" | bc -l)
        ;;
    -)
        result=$(echo "$op1 - $op2" | bc -l)
        ;;
    x)
        result=$(echo "$op1 * $op2" | bc -l)
        ;;
    /)
        if [ "$op2" == "0" ]; then
            echo "Error: Division by zero is not allowed"
            exit 1
        fi
        result=$(echo "$op1 / $op2" | bc -l)
        ;;
    %)
        result=$(echo "$op1 % $op2" | bc)
        ;;
    ^)
        result=$(echo "$op1 ^ $op2" | bc -l)
        ;;
    *)
        echo "Invalid input: Unsupported operator"
        echo "Supported operators are: +, -, x, /, %, ^"
        exit 1
        ;;
esac
echo "$op1 $operator $op2 = $result"
```

Laboratory Assignments 4
Subject: Design Principles of Operating Systems
Subject code: CSE 3249

Assignment 4: Familiarization with Process Management in Linux environment.

Objective of this Assignment:

- To trace the different states of a process during its execution.
- To learn the use of different system calls such as (fork(),vfork(),wait(),execl()) for process handling in Unix/Linux environment.

1. Write a C program to create a child process using fork() system call. The child process will print the message "Child" with its process identifier and then continue in an indefinite loop. The parent process will print the message "Parent" with its process identifier and then continue in an indefinite loop.

   a) Run the program and trace the state of both processes.

   b) Terminate the child process. Then trace the state of processes.

   c) Run the program and trace the state of both processes. Terminate the parent process. Then trace the state of processes.

   d) Modify the program so that the parent process after displaying the message will wait for child process to complete its task. Again run the program and trace the state of both processes.

   e) Terminate the child process. Then trace the state of processes.

2. Trace the output of the following codes:

a)
```
int main( )
 {
 if(fork()==0)
 printf("1");
 else
 printf("2");
 printf("3");
return 0;
 }
```

b)
```
int main( )
 {
 if(vfork()==0)
 {
 printf("1");
 _exit(0);
 }
 else
 printf("2");
 printf("3");
 }
```


c)
```
int main( )
{
pid_t pid;
int i=5;
pid=fork();
i=i+1;
if(pid= =0)
{
printf("Child: %d",i);
}
else

{
wait(NULL);
printf("Parent: %d",i);
}
return 0;
}
```

d)
```
int main( )
{
pid_t pid;
int i=5;
pid=vfork();
i=i+1;
if(pid==0)
{
printf("Child: %d",i);
_exit(0);
}

else
{
printf("Parent: %d",i);
}
return 0;
}
```

e)
```
int main( )
{
pid_t pid;
int i=5;
pid=fork();
if(pid= =0)
{
i=i+1;
printf("Child: %d",i);
}
else
{
wait(NULL);
printf("Parent: %d",i);
}
return 0;
}
```

f)
```
int main( )
{
pid_t pid;
int i=5;
pid=vfork();
if(pid==0)
{
i=i+1;
printf("Child: %d",i);
_exit(0);
}
else
{
printf("Parent: %d",i);
}
return 0;
}
```

g)
```
int main( )
{
int i=5;
if(fork( )==0)
{
printf("Child: %d",i);
}
else
{
printf("Parent: %d",i);
}
return 0;
}
```

h)
```
int main( )
{
int i=5;
if(vfork( )==0)
{
printf("Child: %d",i);
_exit(0);
}
else
{
printf("Parent: %d",i);
}
return 0;
}
```

i)
```
int main( )
{
if(fork( )==0)
{
printf("1");
}
else
{
wait(NULL);
printf("2");
printf("3");
}
return 0;
}
```

j)
```
int main( )
{
if(vfork( )==0)
{
printf("1");
_exit(0);
}
else
{
printf("2");
printf("3");
}
return 0;
}
```

k)
```
int main( )
{
pid_t c1;
int n=10;
c1=fork( );
if(c1==0)
{
printf(" Child\n");
n=20;
printf("n=%d \n",n);
}
else
{
wait(NULL);
printf("Parent\n");
printf("n=%d \n",n);
}
return 0;
}
```

l)
```
int main( )
{
pid_t c1;
int n=10;
c1=vfork( );
if(c1==0)
{
printf(" Child\n");
n=20;
printf("n=%d \n",n);
_exit(0);
}
else
{
printf("Parent\n");
printf("n=%d \n",n);
}
return 0;
}
```

m)
```
int main( )
{
int i=5;
fork();
i=i+1;
fork();
printf ( "% d",i);
return 0;
}
```

n)
```
int main( )
{
pid_t pid;
int i=5;
pid=vfork();
if(pid==0)
{
printf("Child: %d",i);
_exit(0);
}
else
{
i=i+1;
printf("Parent: %d",i);
}
return 0;
}
```

o)
```
int main( )
{
int i=5;
if(fork()==0)
i=i+1;
else
i=i-1;
printf("%d",i);
return 0;
}
```

p)
```
int main( )
{
int i=5;
if(vfork()==0)
{
i=i+1;
_exit(0);
}
else
i=i-1;
fprintf(stderr,"%d",i);
return 0;
}
```

q)
```
int main( )
{
int j,i=5;
for(j=1;j<3;j++)
{
if(fork()==0)
{
i=i+1;
break;
}
else
wait(NULL);
}
printf("%d",i);
return 0;
}
```

r)
```
int main( )
{
int j,i=5;
for(j=1;j<3;j++)
{
if(fork()!=0)
{
i=i-1;
break;
}
}
fprintf(stderr,"%d",i);
return 0;
}
```

s)
```
int main( )
{
if(fork() == 0)
if(fork())
printf("1\n");
return 0;
}
```

t)
```
void fun1(){
fork();
fork();
printf("1\n");
}

int main() {
fun1();
printf("1\n");
return 0;
}
```

3. Write a C program that will create three child process to perform the following operations respectively:
   - First child will copy the content of file1 to file2

- Second child will display the content of file2
- Third child will display the sorted content of file2 in reverse order.
- Each child process being created will display its id and its parent process id with appropriate message.
- The parent process will be delayed for 1 second after creation of each child process. It will display appropriate message with its id after completion of all the child processes.

4. Write a C program that will create a child process to generate a Fibonacci series of specified length and store it in an array. The parent process will wait for the child to complete its task and then display the Fibonacci series and then display the prime Fibonacci number in the series along with its position with appropriate message.

1(a)

Open *q1.c ~/

```c
#include<stdio.h>
#include<unistd.h>
#include<sys/wait.h>
int main(){
pid_t C1 = fork();
if(C1 ==0)
{
fprintf(stderr,"Child Process id: %d",getpid());
while(1);
}
else
{
fprintf(stderr,"Parent Process id: %d",getpid());
while(1);
}
}
```

```
root@DESKTOP-8EIFV6T:~# gcc q1.c
root@DESKTOP-8EIFV6T:~# ./a.out
Parent Process id: 8226Child Process id: 8227
```

```
root@DESKTOP-8EIFV6T:~# ps -al
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 R     0    8226     354 99  80   0 -   661 -      pts/0    00:00:21 a.out
1 R     0    8227    8226 99  80   0 -   661 -      pts/0    00:00:21 a.out
4 R     0    8311    5528  0  80   0 -  1871 -      pts/2    00:00:00 ps
root@DESKTOP-8EIFV6T:~#
```

1(b)

```
root@DESKTOP-8EIFV6T:~# kill -9 8227
root@DESKTOP-8EIFV6T:~# ps -al
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 R     0    8226     354 99  80   0 -   661 -      pts/0    00:01:03 a.out
1 Z     0    8227    8226 88  80   0 -     0 -      pts/0    00:00:56 a.out <defunct>
4 R     0    8480    5528  0  80   0 -  1871 -      pts/2    00:00:00 ps
root@DESKTOP-8EIFV6T:~#
```

1(c)

```
root@DESKTOP-8EIFV6T:~# gcc q1.c
root@DESKTOP-8EIFV6T:~# ./a.out
Parent Process id: 8917Child Process id: 8918
```

```
root@DESKTOP-8EIFV6T:~# kill -9 8917
root@DESKTOP-8EIFV6T:~# ps -al
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
1 R     0    8918     352 99  80   0 -   661 -      pts/0    00:00:39 a.out
4 R     0    9077    5528  0  80   0 -  1871 -      pts/2    00:00:00 ps
root@DESKTOP-8EIFV6T:~#
```

1(d)

Open *q1.c ~/

```c
#include<stdio.h>
#include<unistd.h>
#include<sys/wait.h>
int main(){
pid_t C1 = fork();
if(C1 ==0)
{
fprintf(stderr,"Child Process id: %d",getpid());
while(1);
}
else
{
fprintf(stderr,"Parent Process id: %d",getpid());
wait(NULL);
while(1);
}
}
```

```
root@DESKTOP-8EIFV6T:~# gcc q1.c
root@DESKTOP-8EIFV6T:~# ./a.out
Parent Process id: 11533Child Process id: 11534
```

```
root@DESKTOP-8EIFV6T:~# ps -al
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S     0   11533     354  0  80   0 -   661 do_wai pts/0    00:00:00 a.out
1 R     0   11534   11533 99  80   0 -   661 -      pts/0    00:01:08 a.out
4 R     0   11803    5528  0  80   0 -  1871 -      pts/2    00:00:00 ps
```

1(e)

```
root@DESKTOP-8EIFV6T:~# kill -9 11534
root@DESKTOP-8EIFV6T:~# ps -al
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 R     0   11533     354  1  80   0 -   661 -      pts/0    00:00:01 a.out
4 R     0   12096    5528  0  80   0 -  1871 -      pts/2    00:00:00 ps
root@DESKTOP-8EIFV6T:~#
```

3

Open q3.c Save

```c
#include<stdio.h>
#include<sys/wait.h>
#include<unistd.h>
int main(){
 if(vfork()==0)
 {
  fprintf(stderr,"\nChild process id :%d  Parent process id :%d\n",getpid(),getppid());
  execl("/usr/bin/cp","cp","file1","file2",NULL);
  _exit(0);
 }
  sleep(1);
  if(vfork()==0)
 {
  fprintf(stderr,"\nChild process id :%d  Parent process id :%d\n",getpid(),getppid());
  execl("/usr/bin/cat","cat","file2",NULL);
  _exit(0);
  }
  sleep(1);
  if(vfork()==0)
 {
  fprintf(stderr,"\nChild process id :%d  Parent process id :%d\n",getpid(),getppid());
  execl("/usr/bin/sort","sort","-r","file2",NULL);
  _exit(0);
  }

   sleep(1);
   fprintf(stderr,"\nParent process id :%d",getpid());

}
```

```
┌──(munu㉿kali)-[~]
└─$ cat file1
1
2
3
4
5
6

┌──(munu㉿kali)-[~]
└─$ cat file2
1
2
3
4
5
6
```

```
┌──(munu㉿kali)-[~]
└─$ gedit q3.c
^C
┌──(munu㉿kali)-[~]
└─$ gcc q3.c
┌──(munu㉿kali)-[~]
└─$ ./a.out
Child process id :7411  Parent process id :7410

Child process id :7420  Parent process id :7410
1
2
3
4
5
6

Child process id :7429  Parent process id :7410
6
5
4
3
2
1

Parent process id :7410
```

4

Open q4.c

```c
#include<stdio.h>
#include<unistd.h>
#include<stdbool.h>
int main() {
    int length=10;
    printf("Enter the length : ");
    scanf("%d", &length);
    int fibArray[length];
    if(vfork()==0){
        fibArray[0] = 0;
        fibArray[1] = 1;
        for (int i = 2; i < length; i++) {
            fibArray[i]=fibArray[i-1]+fibArray[i-2];
        }
        _exit(0);
    }
      printf("\n");
        printf("Fibonacci Series:\n");
        for (int i = 0; i < length; i++) {
                printf("%d ", fibArray[i]);
        }
        printf("\n");
        printf("Prime Fibonacci numbers:\n");
        for (int i = 3; i < length; i++) {
        int isPrime = 1;
        for (int j = 2; j * j <= fibArray[i]; j++) {
        if (fibArray[i] % j == 0)
            isPrime = 0;
        }
        if (isPrime)
            printf("%d at index : %d \n", fibArray[i],i);
    }
    return 0;
}
```

```
┌──(munu㉿kali)-[~]
└─$ gcc q4.c

┌──(munu㉿kali)-[~]
└─$ ./a.out
Enter the length : 10

Fibonacci Series:
0 1 1 2 3 5 8 13 21 34
Prime Fibonacci numbers:
2 at index : 3
3 at index : 4
5 at index : 5
13 at index : 7
```

```c
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#define BUFFER_SIZE 10
int buffer[BUFFER_SIZE];
int count = 0;
sem_t empty, full;
pthread_mutex_t mutex;
void *producer(void *arg) {
for (int i = 1; i <= 100; i++) {
sem_wait(&empty);
pthread_mutex_lock(&mutex);
buffer[count++] = i;
printf("Produced: %d\n", i);
pthread_mutex_unlock(&mutex);
sem_post(&full);
}
return NULL;
}
void *consumer(void *arg) {
for (int i = 1; i <= 100; i++) {
sem_wait(&full);
pthread_mutex_lock(&mutex);
int item = buffer[--count];
printf("Consumed: %d\n", item);
pthread_mutex_unlock(&mutex);
sem_post(&empty);
}
return NULL;
}
int main() {
pthread_t prod, cons;
sem_init(&empty, 0, BUFFER_SIZE);
sem_init(&full, 0, 0);
pthread_mutex_init(&mutex, NULL);
pthread_create(&prod, NULL, producer, NULL);
pthread_create(&cons, NULL, consumer, NULL);
pthread_join(prod, NULL);
pthread_join(cons, NULL);
sem_destroy(&empty);
sem_destroy(&full);
pthread_mutex_destroy(&mutex);
return 0;
}
```

```
┌──(munu㉿kali)-[~]
└─$ gedit producer_consumer.c

(gedit:5456): Gtk-WARNING **: 19:1
^C

┌──(munu㉿kali)-[~]
└─$ chmod +x producer_consumer.c

┌──(munu㉿kali)-[~]
└─$ gcc producer_consumer.c

┌──(munu㉿kali)-[~]
└─$ ./a.out
Produced: 1
Produced: 2
Produced: 3
Produced: 4
Produced: 5
Produced: 6
Produced: 7
Produced: 8
Produced: 9
Produced: 10
Consumed: 10
Consumed: 9
Consumed: 8
Consumed: 7
Consumed: 6
Consumed: 5
Consumed: 4
Consumed: 3
Consumed: 2
Consumed: 1
Produced: 11
Produced: 12
Produced: 13
Produced: 14
Produced: 15
Produced: 16
Produced: 17
Produced: 18
Produced: 19
Consumed: 18
Consumed: 17
Consumed: 16
Consumed: 15
Consumed: 14
Consumed: 13
Consumed: 12
Consumed: 11
Produced: 21
Produced: 22
Produced: 23
Produced: 24
Produced: 25
Produced: 26
Produced: 27
Produced: 28
Produced: 29
Produced: 30
Consumed: 30
Consumed: 29
Consumed: 28
Consumed: 27
Consumed: 26
Consumed: 25
Consumed: 24
Consumed: 23
Consumed: 22
Consumed: 21
Produced: 31
Produced: 32
Produced: 33
Produced: 34
Produced: 35
Produced: 36
Produced: 37
Produced: 38
Produced: 39
Produced: 40
Consumed: 40
Consumed: 39
Consumed: 38
Consumed: 37
Consumed: 36
Consumed: 35
Consumed: 34
Consumed: 33
Consumed: 32
Consumed: 31
Produced: 41
Produced: 42
Produced: 43
Produced: 44
Produced: 45
Produced: 46
Produced: 47
Produced: 48
Produced: 49
Produced: 50
Consumed: 50
Consumed: 49
Consumed: 48
Consumed: 47
Consumed: 46
Consumed: 45
Consumed: 44
Consumed: 43
Consumed: 42
Consumed: 41
Produced: 51
Produced: 52
Produced: 53
Produced: 54
Produced: 55
Produced: 56
Produced: 57
Produced: 58
Produced: 59
Produced: 60
Consumed: 60
Consumed: 59
Consumed: 58
Consumed: 57
Consumed: 56
Consumed: 55
Consumed: 55
Consumed: 54
Consumed: 53
Consumed: 52
Consumed: 51
Produced: 61
Produced: 62
Produced: 63
Produced: 64
Produced: 65
Produced: 66
Produced: 67
Produced: 68
Produced: 69
Produced: 70
Consumed: 70
Consumed: 69
Consumed: 68
Consumed: 67
Consumed: 66
Consumed: 65
Consumed: 64
Consumed: 63
Consumed: 62
Consumed: 61
Produced: 71
Produced: 72
Produced: 73
Produced: 74
Produced: 75
Produced: 76
Produced: 77
Produced: 78
Produced: 79
Produced: 80
Consumed: 80
Consumed: 79
Consumed: 78
Consumed: 77
Consumed: 76
Consumed: 75
Consumed: 74
Consumed: 73
Consumed: 73
Consumed: 72
Consumed: 71
Produced: 81
Produced: 82
Produced: 83
Produced: 84
Produced: 85
Produced: 86
Produced: 87
Produced: 88
Produced: 89
Produced: 90
Consumed: 90
Consumed: 89
Consumed: 88
Consumed: 87
Consumed: 86
Consumed: 85
Consumed: 84
Consumed: 83
Consumed: 82
Consumed: 81
Produced: 91
Produced: 92
Produced: 93
Produced: 94
Produced: 95
Produced: 96
Produced: 97
Produced: 98
Produced: 99
Produced: 100
Consumed: 100
Consumed: 99
Consumed: 98
Consumed: 97
Consumed: 96
Consumed: 95
Consumed: 94
Consumed: 93
Consumed: 92
Consumed: 91
```

```c
C code:
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
sem_t sem_odd, sem_even;
void *print_odd(void *arg) {
for (int i = 1; i <= 20; i += 2) {
sem_wait(&sem_odd);
printf("%d ", i);
sem_post(&sem_even);
}
return NULL;
}
void *print_even(void *arg) {
for (int i = 2; i <= 20; i += 2) {
sem_wait(&sem_even);
printf("%d ", i);
sem_post(&sem_odd);
}
return NULL;
}
int main() {
pthread_t odd, even;
sem_init(&sem_odd, 0, 1);
sem_init(&sem_even, 0, 0);
pthread_create(&odd, NULL, print_odd, NULL);
pthread_create(&even, NULL, print_even, NULL);
pthread_join(odd, NULL);
pthread_join(even, NULL);
sem_destroy(&sem_odd);
sem_destroy(&sem_even);
return 0;
}
```

```
┌──(munu㉿kali)-[~]
└─$ gedit altertwothread.c

(gedit:13920): Gtk-WARNING **: 19:34:05.558: Calling org.x
^C

┌──(munu㉿kali)-[~]
└─$ gcc altertwothread.c

┌──(munu㉿kali)-[~]
└─$ ./a.out
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

```c
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
sem_t sem_A, sem_B;
void *print_A(void *arg) {
for (int i = 0; i < 10; i++) {
sem_wait(&sem_A);
printf("A");
sem_post(&sem_B);
}
return NULL;
}
void *print_B(void *arg) {
for (int i = 0; i < 10; i++) {
sem_wait(&sem_B);
printf("B");
sem_post(&sem_A);
}
return NULL;
}
int main() {
pthread_t thread_A, thread_B;
sem_init(&sem_A, 0, 1);
sem_init(&sem_B, 0, 0);
pthread_create(&thread_A, NULL, print_A, NULL);
pthread_create(&thread_B, NULL, print_B, NULL);
pthread_join(thread_A, NULL);
pthread_join(thread_B, NULL);
sem_destroy(&sem_A);
sem_destroy(&sem_B);
printf("\n");
return 0;
}
```

```
┌──(munu㉿kali)-[~]
└─$ gedit alterchar.c

(gedit:15096): Gtk-WARNING **:
^C
┌──(munu㉿kali)-[~]
└─$ chmod +x alterchar.c

┌──(munu㉿kali)-[~]
└─$ gcc alterchar.c

┌──(munu㉿kali)-[~]
└─$ ./a.out
ABABABABABABABABABAB
```

```c
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
sem_t sem_countdown, sem_countup;
void *countdown(void *arg) {
for (int i = 10; i >= 1; i--) {
sem_wait(&sem_countdown);
printf("Countdown: %d\n", i);
sem_post(&sem_countup);
}
return NULL;
}
void *countup(void *arg) {
for (int i = 1; i <= 10; i++) {
sem_wait(&sem_countup);
printf("Countup: %d\n", i);
sem_post(&sem_countdown);
}
return NULL;
}
int main() {
pthread_t thread_countdown, thread_countup;
sem_init(&sem_countdown, 0, 1);
sem_init(&sem_countup, 0, 0);
pthread_create(&thread_countdown, NULL, countdown, NULL);
pthread_create(&thread_countup, NULL, countup, NULL);
pthread_join(thread_countdown, NULL);
pthread_join(thread_countup, NULL);
sem_destroy(&sem_countdown);
sem_destroy(&sem_countup);
return 0;
}
```

```
┌──(munu㉿kali)-[~]
└─$ gedit count.c

(gedit:17029): Gtk-WARNING
^C
┌──(munu㉿kali)-[~]
└─$ chmod +x count.c
┌──(munu㉿kali)-[~]
└─$ gcc count.c
┌──(munu㉿kali)-[~]
└─$ ./a.out
Countdown: 10
Countup: 1
Countdown: 9
Countup: 2
Countdown: 8
Countup: 3
Countdown: 7
Countup: 4
Countdown: 6
Countup: 5
Countdown: 5
Countup: 6
Countdown: 4
Countup: 7
Countdown: 3
Countup: 8
Countdown: 2
Countup: 9
Countdown: 1
Countup: 10
```

```c
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
sem_t sem_A, sem_B, sem_C;
void *print_A(void *arg) {
for (int i = 1; i <= 20; i += 3) {
sem_wait(&sem_A);
printf("A%d ", i);
sem_post(&sem_B);
}
return NULL;
}
void *print_B(void *arg) {
for (int i = 2; i <= 20; i += 3) {
sem_wait(&sem_B);
printf("B%d ", i);
sem_post(&sem_C);
}
return NULL;
}
void *print_C(void *arg) {
for (int i = 3; i <= 20; i += 3) {
sem_wait(&sem_C);
printf("C%d ", i);
sem_post(&sem_A);
}
return NULL;
}
int main() {
pthread_t thread_A, thread_B, thread_C;
sem_init(&sem_A, 0, 1);
sem_init(&sem_B, 0, 0);
sem_init(&sem_C, 0, 0);
pthread_create(&thread_A, NULL, print_A, NULL);
pthread_create(&thread_B, NULL, print_B, NULL);
pthread_create(&thread_C, NULL, print_C, NULL);
pthread_join(thread_A, NULL);
pthread_join(thread_B, NULL);
pthread_join(thread_C, NULL);
sem_destroy(&sem_A);
sem_destroy(&sem_B);
sem_destroy(&sem_C);
printf("\n");
return 0;
}
```

```
┌──(munu㉿kali)-[~]
└─$ gedit seqprint.c

(gedit:19488): Gtk-WARNING **: 19:45:06.309: Calling org.xfce.Session.Ma
^C
┌──(munu㉿kali)-[~]
└─$ chmod +x seqprint.c
┌──(munu㉿kali)-[~]
└─$ gcc seqprint.c
┌──(munu㉿kali)-[~]
└─$ ./a.out
A1 B2 C3 A4 B5 C6 A7 B8 C9 A10 B11 C12 A13 B14 C15 A16 B17 C18 A19 B20
```