ASSIGNMENT - 1

1. Change your password to a password you would like to use for the remainder of the semester.

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % sudo passwd tanmaykadam
Changing password for tanmaykadam.
[Old password:
[New password:
[Retype new password:
```

2. Display the system's date.

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % date
Wed Sep 14 10:37:02 IST 2022
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

3. Count the number of lines in the /etc/passwd file.

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % wc /etc/passwd
     123     329    7868 /etc/passwd
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

4. Find out who else is on the system.

```
tanmaykadam@Tanmays-MacBook-Pro ~ % w
10:39  up 2 days,  1:01, 2 users, load averages: 2.48 3.03 2.66
USER     TTY      FROM              LOGIN@  IDLE WHAT
tanmaykadam console  -                 Mon09   2days -
tanmaykadam s000     -                 10:33      - w
```

5. Direct the output of the man pages for the date command to a file named *mydate*.

```
tanmaykadam@Tanmays-MacBook-Pro ~ % man date>mydate.txt
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

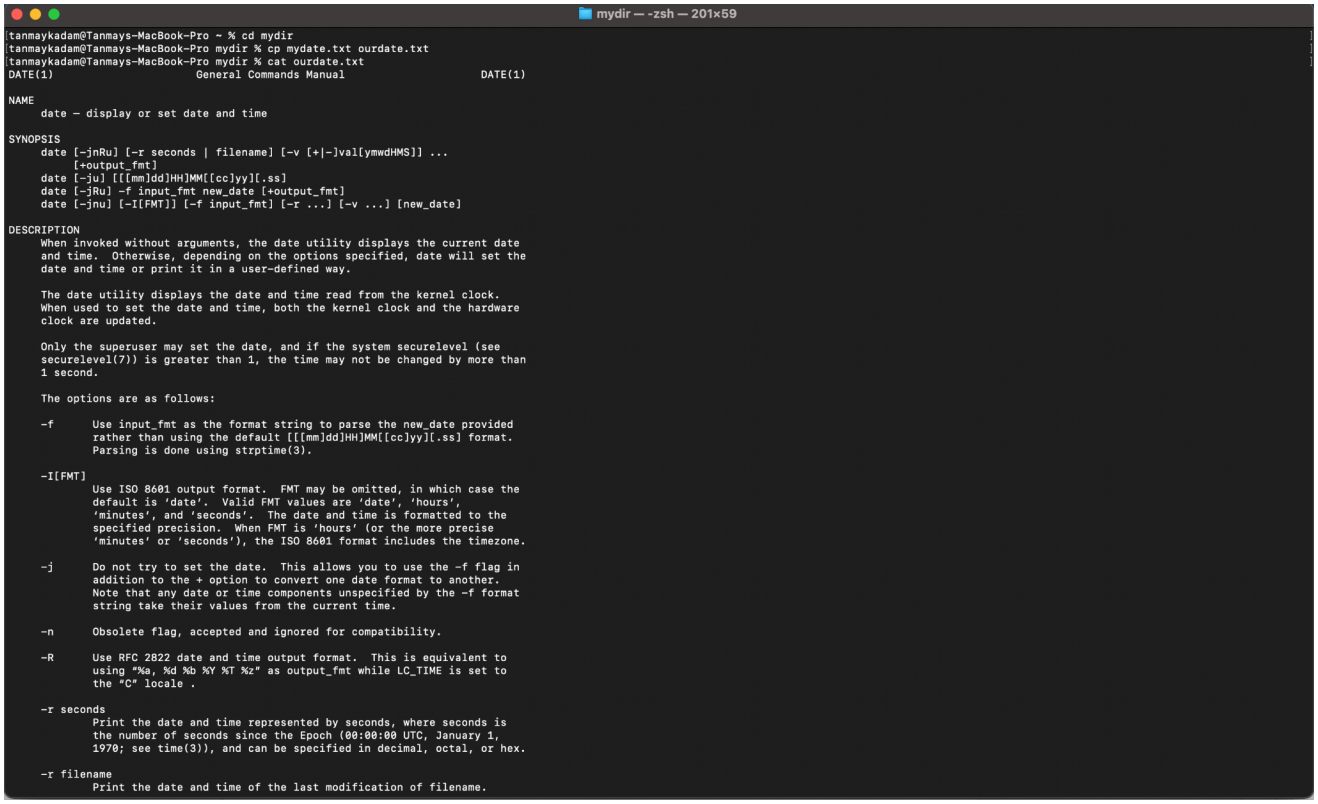
6. Create a subdirectory called *mydir*.

```
tanmaykadam@Tanmays-MacBook-Pro ~ % mkdir mydir
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

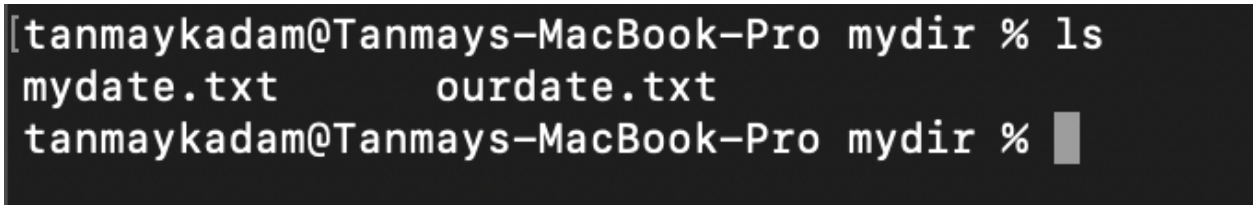
7. Move the file *mydate* into the new subdirectory.

```
tanmaykadam@Tanmays-MacBook-Pro ~ % mv mydate.txt mydir
tanmaykadam@Tanmays-MacBook-Pro ~ % ls
Applications            Downloads               Music                   VIT
Courses                 Food Menu Project       Pictures                mydir
Desktop                 Library                 Public
Documents               Movies                  PycharmProjects
tanmaykadam@Tanmays-MacBook-Pro ~ % cd mydir
tanmaykadam@Tanmays-MacBook-Pro mydir % ls
mydate.txt
tanmaykadam@Tanmays-MacBook-Pro mydir %
```

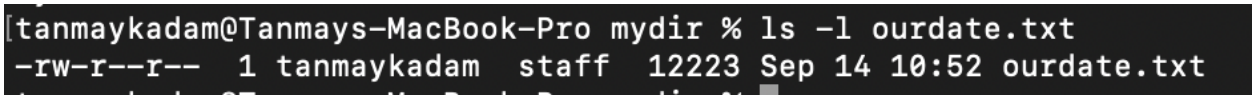
8. Go to the subdirectory *mydir* and copy the file *mydate* to a new file called *ourdate*



9. List the contents of *mydir*.



10. Do a long listing on the file *ourdate* and note the permissions.



11. Display the name of the current directory starting from the root.

```
tanmaykadam@Tanmays-MacBook-Pro mydir % pwd
/Users/tanmaykadam/mydir
```

12. Move the files in the directory *mydir* back to the HOME directory.

```
tanmaykadam@Tanmays-MacBook-Pro ~ % mv mydir home
tanmaykadam@Tanmays-MacBook-Pro ~ % ls
Applications            Food Menu Project       Public
Courses                 Library                 PycharmProjects
Desktop                 Movies                  VIT
Documents               Music                   home
Downloads               Pictures
tanmaykadam@Tanmays-MacBook-Pro ~ % cd home
tanmaykadam@Tanmays-MacBook-Pro home % ls
mydate.txt      ourdate.txt
tanmaykadam@Tanmays-MacBook-Pro home %
```

13. List all the files in your HOME directory.

```
tanmaykadam@Tanmays-MacBook-Pro home % cd ~
tanmaykadam@Tanmays-MacBook-Pro ~ % ls
Applications            Food Menu Project       Public
Courses                 Library                 PycharmProjects
Desktop                 Movies                  VIT
Documents               Music                   home
Downloads               Pictures
```

14. Display the first 5 lines of *mydate*.

```
tanmaykadam@Tanmays-MacBook-Pro home % head mydate.txt 5
==> mydate.txt <==
DATE(1)                     General Commands Manual                     DATE(1)

NAME
     date – display or set date and time

SYNOPSIS
     date [-jnRu] [-r seconds | filename] [-v [+|-]val[ymwdHMS]] ...
          [+output_fmt]
     date [-ju] [[[mm]dd]HH]MM[[cc]yy][.ss]
     date [-jRu] -f input_fmt new_date [+output_fmt]
head: 5: No such file or directory
```

15. Display the last 8 lines of *mydate*.

```
tanmaykadam@Tanmays-MacBook-Pro home % tail mydate.txt 8
==> mydate.txt <==
     to the standard.

     The format selected by the -I flag is compatible with ISO 8601.

HISTORY
     A date command appeared in Version 1 AT&T UNIX.

     The -I flag was added in FreeBSD 12.0.

macOS 12.5                      August 25, 2020                     macOS 12.5
tail: 8: No such file or directory
```

16. Remove the directory mydir.

```
tanmaykadam@Tanmays-MacBook-Pro ~ % rm -r home
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

17. Redirect the output of the long listing of files to a file named list.

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % cd home
[tanmaykadam@Tanmays-MacBook-Pro home % ls -l mydate.txt ourdate.txt | tee list.txt
 -rw-r--r--  1 tanmaykadam  staff  12223 Sep 14 11:18 mydate.txt
 -rw-r--r--  1 tanmaykadam  staff  12223 Sep 14 11:19 ourdate.txt
[tanmaykadam@Tanmays-MacBook-Pro home % cat list list.txt
 cat: list: No such file or directory
 -rw-r--r--  1 tanmaykadam  staff  12223 Sep 14 11:18 mydate.txt
 -rw-r--r--  1 tanmaykadam  staff  12223 Sep 14 11:19 ourdate.txt
 tanmaykadam@Tanmays-MacBook-Pro home %
```

Lab Assignment 2

Advance Linux Command

Problems to be solved in the lab:

1. Select any 5 capitals of states in India and enter them in a file named *capitals1*. Choose 5 more capitals and enter them in a file named *capitals2*. Choose 5 more capitals and enter them in a file named *capitals3*. Concatenate all 3 files and redirect the output to a file named *capitals*.

```
root@LAPTOP-KA152M9D:~# cat capitals1.txt capitals2.txt capitals3.txt > capital.txt
root@LAPTOP-KA152M9D:~# cat capital.txt
Mumbai
Chennai
Delhi
Bhopal
Gangtok
Itanagar
Imphal
Hyderabad
Dispur
Chandigarh
Agartala
Aizawl
Bengluru
Panaji
Jaipur
root@LAPTOP-KA152M9D:~#
```

2. Concatenate the file *capitals2* at the end of file *capitals*.

```
root@LAPTOP-KA152M9D:~# cat capitals2.txt capital.txt
Itanagar
Imphal
Hyderabad
Dispur
Chandigarh
Mumbai
Chennai
Delhi
Bhopal
Gangtok
Itanagar
Imphal
Hyderabad
Dispur
Chandigarh
Agartala
Aizawl
Bengluru
Panaji
Jaipur
root@LAPTOP-KA152M9D:~#
```

3. Redirect the file *capitals* as an input to the command "wc –l".

```
root@LAPTOP-KA152M9D:~# cat capital.txt | tee wc-l
Mumbai
Chennai
Delhi
Bhopal
Gangtok
Itanagar
Imphal
Hyderabad
Dispur
Chandigarh
Agartala
Aizawl
Bengluru
Panaji
Jaipur
root@LAPTOP-KA152M9D:~#
```

4. Give read and write permissions to all users for the file *capitals*.

```
root@LAPTOP-KA152M9D:~# chmod 666 capital.txt
root@LAPTOP-KA152M9D:~# ls -l capital.txt
-rw-rw-rw- 1 root root 119 Sep 14 11:42 capital.txt
root@LAPTOP-KA152M9D:~#
```

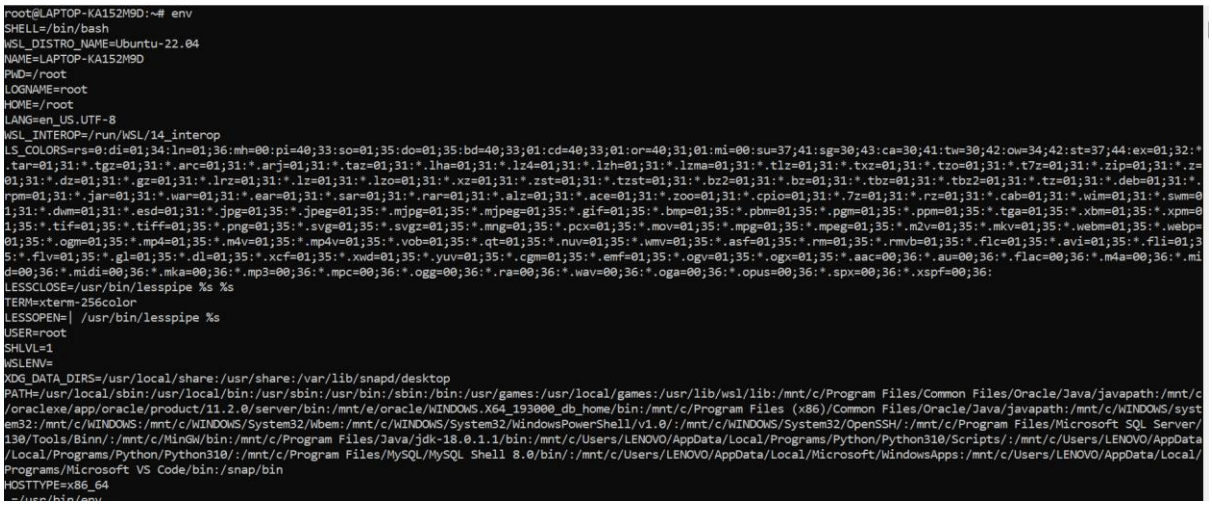
5. Give read permissions only to the owner of the file *capitals*. Open the file, make some changes and try to save it. What happens?

```
root@LAPTOP-KA152M9D:~# chmod +r capital.txt
root@LAPTOP-KA152M9D:~# cat capital.txt
Mumbai
Chennai
Delhi
Bhopal
Gangtok
Itanagar
Imphal
Hyderabad
Dispur
Chandigarh
Agartala
Aizawl
Bengluru
Panaji
Jaipur
root@LAPTOP-KA152M9D:~#
```

6. Create an alias to concatenate the 3 files *capitals1*, *capitals2*, *capitals3* and redirect the output to a file named *capitals*. Activate the alias and make it run.

```
root@LAPTOP-KA152M9D:~# alias concatfiles="cat capitals1.txt capitals2.txt capitals.txt > capitals.txt"
root@LAPTOP-KA152M9D:~# cat capitals.txt
Mumbai
Chennai
Delhi
Bhopal
Gangtok
Itanagar
Imphal
Hyderabad
Dispur
Chandigarh
root@LAPTOP-KA152M9D:~#
```

7. What are the environment variables PATH, HOME and TERM set to on your terminal?



8. Find out the number of times the string “the” appears in the file *mydate*.

```
root@LAPTOP-KA152M9D:~# grep the mydate.txt | wc -w
198
```

9. Find out the line numbers on which the string “date” exists in *mydate*.

```
root@LAPTOP-KA152M9D:~# grep date mydate.txt | wc -l
27
```

10. Print all lines of *mydate* except those that have the letter “i” in them.

```
root@LAPTOP-KA152M9D:~# cat mydate.txt | grep -v "i" | grep '.'
DATE(1)                          User Commands                          DATE(1)
NAME
SYNOPSIS
       date [OPTION]... [+FORMAT]
DESCRIPTION
       -d, --date=STRING
       --debug
              stderr
              02:34:56 -0600
       --rfc-3339=FMT
              2006-08-14 02:34:56-06:00
       -r, --reference=FILE
       -s, --set=STRING
       FORMAT controls the output.  Interpreted sequences are:
       %A     locale's full weekday name (e.g., Sunday)
       %B     locale's full month name (e.g., January)
       %d     day of month (e.g., 01)
       %D     date; same as %m/%d/%y
       %e     day of month, space padded; same as %_d
       %h     same as %b
       %H     hour (00..23)
       %I     hour (01..12)
       %j     day of year (001..366)
       %k     hour, space padded ( 0..23); same as %_H
       %l     hour, space padded ( 1..12); same as %_I
       %m     month (01..12)
       %N     nanoseconds (000000000..999999999)
       %q     quarter of year (1..4)
       %S     second (00..60)
       %t     a tab
       %Y     year
       flags may follow '%':
EXAMPLES
              $ date --date='@2147483647'
DATE STRING
       as "Sun, 29 Feb 2004 16:21:42 -0800"  or  "2004-02-29  16:21:42"  or  even
```

11. Create the file *monotonic* as follows:

    ^a?b?b?c?……..x?y?z$

    Run the egrep command for *monotonic* against /usr/dict/words and search for all 4 letter words.

    echo "^a?b?b?c?……..x?y?z$" > monotonic

12. List 5 states in north east India in a file *mystates*. List their corresponding capitals in a file *mycapitals*. Use the *paste* command to join the 2 files.

```
root@LAPTOP-KA152M9D:~# paste mystates.txt mycapitals.txt
Jammu Punjab Goa Hariyana Delhi Bihar
        Andhrapradesh
        Rajsthan
        Madhyapradesh
        Keral
```

13. Use the *cut* command to print the 1st and 3rd columns of the /etc/passwd file for all students in this class.

```
root@LAPTOP-KA152M9D:~# cut -b 1,3 /etc/passwd | head
ro
de
bn
ss
sn
gm
mn
l:
mi
nw
```

14. Count the number of people logged in and also trap the users in a file using the *tee* command.

```
root@LAPTOP-KA152M9D:~# getent passwd | wc -l | tee file1.txt
31
root@LAPTOP-KA152M9D:~# cat file1.txt
31
```

TUT - 2

1. **Write shell script to find out length of given string.**

```
echo "Enter String : "$str1
read str1;

len_str1=${#str1}
echo "Length of String is: "$len_str1
```

```
tanmaykadam@Tanmays-MacBook-Pro Tutorials % bash Tut-2.sh
Enter String :
Tanmay KAdam
Length of String is: 12
tanmaykadam@Tanmays-MacBook-Pro Tutorials %
```

2. **Write a shell script to convert the original string into reverse string.**

```
echo "Enter String to be Reversed : "
read str

reversed_string=""

len=${#str}

for (( i=$len-1; i>=0; i-- ))
do
  reversed_string="$reversed_string${str:$i:1}"
done

echo "Reversed String is : $reversed_string"
```

```
tanmaykadam@Tanmays-MacBook-Pro Tutorials % bash Tut-2.sh
Enter String to be Reversed :
Tanmay Kadam
Reversed String is : madaK yamnaT
tanmaykadam@Tanmays-MacBook-Pro Tutorials %
```

3. **Write a shell script to find out if the given string is palindrome or not.**

```
echo "Enter a String"
read input
reverse=""

len=${#input}
for (( i=$len-1; i>=0; i-- ))
do
        reverse="$reverse${input:$i:1}"
done
if [ $input == $reverse ]
then
  echo "$input is palindrome"
else
  echo "$input is not palindrome"
fi
```

```
[tanmaykadam@Tanmays-MacBook-Pro Tutorials % bash Tut-2.sh
 Enter a String
 Tanmay
 Tanmay is not palindrome
[tanmaykadam@Tanmays-MacBook-Pro Tutorials % bash Tut-2.sh
 Enter a String
 IOI
 IOI is palindrome
 tanmaykadam@Tanmays-MacBook-Pro Tutorials %
```

ASSIGNMENT - 3

1. Write a shell script program to execute ls, date and echo commands repeatedly.

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % touch script.sh
[tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
[tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh
Applications            Food Menu Project       Public
Courses                 Library                 PycharmProjects
Desktop                 Movies                  VIT
Documents               Music                   home
Downloads               Pictures                script.sh
Wed Sep 14 21:43:29 IST 2022
Tanmay Kadam
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

2. Write a shell script program to show the details related to shell, path and home directories of the user.

```
  UW PICO 5.09                                    File: script.sh

echo "SHELL:$SHELL"
echo "PATH:$PATH"
echo "HOME:$HOME"
echo "PWD:$PWD"
echo "LOGNAME:$LOGNAME"
```

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh                                                                ]
[tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh                                                                ]
SHELL:/bin/zsh
PATH:/Library/Frameworks/Python.framework/Versions/3.10/bin:/opt/homebrew/bin:/opt/homebrew/sbin:/usr/local/bin:
/usr/bin:/bin:/usr/sbin:/sbin:/opt/homebrew/bin/
HOME:/Users/tanmaykadam
PWD:/Users/tanmaykadam
LOGNAME:tanmaykadam
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

3. **Write a shell script program to create two files. The name of the files will be passed by user using read function.**

```
  UW PICO 5.09

echo "Enter a First Name"
read filename1
touch $filename1
echo "First File Created Sucessfully !!"
echo "Enter second File name"
read filename2
echo "Second File Created Sucessfully !!"
```

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
[tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh
Enter a First Name
abc.txt
First File Created Sucessfully !!
Enter second File name
pqr.txt
Second File Created Sucessfully !!
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

4. **Write a shell script program to create two files. The name of the files will be passed by user using command line arguments.**

```
touch $1
echo "First File created Sucessfully !!"
touch $2
echo "Second File created Sucessfully !!"
```

```
tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh s1.txt s2.txt
First File created Sucessfully !!
Second File created Sucessfully !!
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

5. **Write a shell script program to create two directories. The name of the files will be passed by user using read function.**

```
echo "Enter First Directory Name"
read dir1
mkdir $dir1
echo "First Directory Created Sucessfully"
echo "Enter Second directory Name"
read dir2
mkdir $dir2
echo "Second directory Created Sucessfully"
```

```
tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh
Enter First Directory Name
dir1
First Directory Created Sucessfully
Enter Second directory Name
dir2
Second directory Created Sucessfully
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

6. **Write a shell script program to create two directories. The name of the files will be passed by user using command line arguments.**

```
mkdir $1
echo "First Directory Created Successfuly"

mkdir $2
echo "Second Directory Created Sucessfully"
```

```
tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh mydir1 mydir2
First Directory Created Successfuly
Second Directory Created Sucessfully
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

7. **Write shell script to change the name of a file .Ask from user old filename and new filename.**

```
echo "Enter Old Filename"
read oldfilename
echo "Enter the New Filename"
read newfilename
mv $oldfilename $newfilename
```

```
[tanmaykadam@Tanmays-MacBook-Pro ~ % ls
Applications            Library                 VIT                     mydir2
Courses                 Movies                  abc.txt                 s1.txt
Desktop                 Music                   dir1                    s2.txt
Documents               Pictures                dir2                    script.sh
Downloads               Public                  home
Food Menu Project       PycharmProjects         mydir1
[tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh
Enter Old Filename
abc.txt
Enter the New Filename
pqr.txt
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

8. **Write a shell script to find the largest number among three numbers.**

```
echo "Enter Num 1"
read num1
echo "Enter Num 2"
read num2
echo "Enter Num 3"
read num3

if [ $num1 -eq $num2 ] && [ $num2 -eq $num3 ]
then
        echo "All Numbers are Equal"
elif [ $num1 -gt $num2 ] && [ $num1 -gt $num3 ]
then
        echo "Num 1 is Greater"
elif [ $num2 -gt $num1 ] && [ $num2 -gt $num3 ]
then
        echo "Num 2 is Greater"
else
        echo "Num 3 is Greater"
fi
```

```
tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh
Enter Num 1
77
Enter Num 2
99
Enter Num 3
1
Num 2 is Greater
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

9. **Write shell script to create three files f1, f2, f3 using for loop.**


```
tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh
tanmaykadam@Tanmays-MacBook-Pro ~ % ls
Applications            Movies                  dir2                    pqr.txt
Courses                 Music                   f1.txt                  s1.txt
Desktop                 Pictures                f2.txt                  s2.txt
Documents               Public                  f3.txt                  script.sh
Downloads               PycharmProjects         home
Food Menu Project       VIT                     mydir1
Library                 dir1                    mydir2
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

10. **Write a shell script to display date and time. Assume 1 sec delay.**

```
for i in {1..5}
do
        date
        sleep 1s
done
```

```
tanmaykadam@Tanmays-MacBook-Pro ~ % nano script.sh
tanmaykadam@Tanmays-MacBook-Pro ~ % bash script.sh
Wed Sep 14 22:38:02 IST 2022
usage: sleep seconds
Wed Sep 14 22:38:02 IST 2022
usage: sleep seconds
Wed Sep 14 22:38:02 IST 2022
usage: sleep seconds
Wed Sep 14 22:38:02 IST 2022
usage: sleep seconds
Wed Sep 14 22:38:02 IST 2022
usage: sleep seconds
tanmaykadam@Tanmays-MacBook-Pro ~ %
```

ASSIGNMENT - 4

1. **Write Shell script to find out positive and negative numbers from accepted array. Assume Array consists of 5 numbers.Also accept array from user.**

```
a=()

echo "Enter 5 Numbers"

 for((i=0;i<5;i++))
do
  read a[$i]
done

#Printing Positive Numbers
echo "Positive Numbers :"

for((i=0;i<5;i++))
do
  t=${a[$i]}

  if [ $t -gt 0 ]
  then
     echo $t
```

```
  fi
done


#Printing Negative Numbers
echo "Negative Numbers :"

for((i=0;i<5;i++))
do
  t=${a[$i]}

  if [ $t -lt 0 ]
  then
     echo $t
  fi
Done
```

```
tanmaykadam@Tanmays-MacBook-Pro Exp 3 % bash Assignment4.sh
Enter 5 Numbers
10
-20
-10
20
50
Positive Numbers :
10
20
50
Negative Numbers :
-20
-10
```

2. **Write Shell script to find out even and odd numbers from accepted array. Assume Array consists of 5 numbers. Also accept arrays from users.**

```
a=()

echo "Enter 5 Numbers"

 for((i=0;i<5;i++))
do
  read a[$i]
done

echo "Even Numbers :"

for((i=0;i<5;i++))
do
  t=${a[$i]}
  t2=$(expr $t % 2)

  if [ $t2 -eq 0 ]
  then
    echo $t
  fi
done

echo "Odd Numbers :"
```

```
for((i=0;i<5;i++))
do
  t=${a[$i]}
  t2=$(expr $t % 2)

  if [ $t2 -ne 0 ]
  then
    echo $t
  fi
done
```

```
tanmaykadam@Tanmays-MacBook-Pro Exp 3 % bash Assignment4.sh
Enter 5 Numbers
10
13
20
15
17
Even Numbers :
10
20
Odd Numbers :
13
15
17
```

3. **Write Shell script to sort array numbers ascending and descending order. Assume Array consists of 5 numbers. Also accept arrays from users.**

```
a=()

echo "Enter 5 Numbers"

 for((i=0;i<5;i++))
do
   read a[$i]
done

#Ascending Order
echo ""
echo "Ascending Order : "


for((i=0;i<5;i++))
do
   for((j=0;j<5-$i-1;j++))
      do

         num1=${a[$i]}
         num2=${a[$j]}

         if [  ${a[j]} -gt ${a[$((j+1))]} ]
         then
            temp=${a[$j]}
            a[$j]=${a[$((j+1))]}
            a[$((j+1))]=$temp
         fi
```

```
    done
done
echo ${a[@]}


#Descending Order
echo ""

echo "Descending Order : "


for((i=0;i<5;i++))
do
  for((j=0;j<5-$i-1;j++))
    do

      if [ ${a[j]} -lt ${a[$((j+1))]} ]
      then
        temp=${a[$j]}
        a[$j]=${a[$((j+1))]}
        a[$((j+1))]=$temp
      fi
    done
done
echo ${a[@]}
```

```
[tanmaykadam@Tanmays-MacBook-Pro Exp 3 % bash Assignment4.sh
 Enter 5 Numbers
 40
 20
 10
 50
 30

 Ascending Order :
 10 20 30 40 50

 Descending Order :
 50 40 30 20 10
```

4. **Write Shell script to find out smallest number and largest number of given array. Assume Array consists of 5 numbers. Also accept arrays from users.**

```
a=()

echo "Enter 5 Numbers"

 for((i=0;i<5;i++))
 do
   read a[$i]
done

echo "Largest Numbers :"

num1=0
for((i=1;i<5;i++))
```

```
 do
   t=${a[$i]}

   if [ $t -gt $num1 ]
   then
      num1=${a[$i]}
   fi
done

echo $num1



echo "Smallest Numbers :"

num1=0
for((i=1;i<5;i++))
 do
   t=${a[$i]}

   if [ $t -lt $num1 ]
   then
      num1=${a[$i]}
   fi
done

echo $num1
```

```
[tanmaykadam@Tanmays-MacBook-Pro Exp 3 % bash Assignment4.sh
Enter 5 Numbers
10
50
-20
-40
50
Largest Numbers :
50
Smallest Numbers :
-40
```

5. **Write shell script to find out the reverse number of a given number.**

```
echo "Enter Number"
read num
echo "Reversed Number is :"
temp=0

while [ $num -gt 0 ]
do
  temp=$(expr $temp \* 10)
  k=$(expr $num % 10)
  temp=$(expr $temp + $k)
  num=$(expr $num / 10)
done
echo $temp
```

```
[tanmaykadam@Tanmays-MacBook-Pro Exp 3 % bash Assignment4.sh
Enter Number
123
Reversed Number is :
321
```

6. **Write a shell script to create a fibonacci series.**

```
echo "Enter Limit of Fibonacci Series"
read N

a=0
b=1

echo "The Fibonacci series is : "

for (( i=0; i<N; i++ ))
do
        echo -n "$a "
        fn=$((a + b))
        a=$b
        b=$fn
Done
```

```
tanmaykadam@Tanmays-MacBook-Pro Exp 3 % bash Assignment4.sh
Enter Limit of Fibonacci Series
10
The Fibonacci series is :
0 1 1 2 3 5 8 13 21 34 %
```