Date 15/11/21

Expt. No. 11

Page No. 11

Program 11

Shell script to find the power of a number

```
echo "enter the number"
read number
echo "enter the power"
read power
val=1
for ((i=1; i<=$power; i++)); do
    val=`expr $val \* $number`
done
echo "value is: $val"
```

o/p

enter the number

5

enter the power

2

value is 25

enter the number

2

enter the power

3

value is 8

Date 15/11/21

Expt. No. 12 Page No. 12

Shell script to find sum of n natural numbers

```
echo "enter number:"
read number
sum=0
for (( i=1; i<=$number; i++)); do
    sum='expr $sum + $i'
done
echo "sum is $sum"
```

O/P

enter number:

10

sum is 55

enter number:

100

sum is 5050

Date

Expt. No. 13 Page No. 13

Shell script to display pass class of a student

```
#!/bin/sh

echo "enter number of subjects"
read subjects
i=1
while [ $i -le $subjects ]; do
    echo "subject : " $i
    echo "enter cie marks"
    read cie
    echo "enter see marks"
    read see
see = `expr $see / 2`
total = `expr $cie + $see`
    echo "\n total in subject $i : $total"
    if [ $total -gt 90 ]; then
        echo "s grade\n"
    elif [ $total -gt 80 ]; then
        echo " A grade\n"
    elif [ $total -gt 70 ]; then
        echo " B grade\n"
    elif [ $total -gt 60 ]; then
        echo " C grade\n"
    else
        echo " you have failed in this subject\n"
    fi
    i = `expr $i + 1`
done
```

Teacher's Signature : _____

Date

Expt. No. 14 Page No. 14

shell script to find fibonacci series upto n.

```
#!/bin/sh

echo "enter value of n:"
read n

first=0
second=1
next=`expr $first + $second`
echo $first
echo $second

while [ $next -le $n ]; do
    echo $next
    first=`expr $second`
    second=`expr $next`
    next=`expr $first + $second`
done
```

o/p

```
enter value of n:
5
0
1
1
2
3
5
```

Date

Expt. No. 15 Page No. 15

Shell script to find no. of vowels in a string

```
#!/bin/bash

echo "enter string"
read string

count=0
for (( i=0; i<${#string}; i++)); do
   char=${string:$i:1}
   if [ $char = 'a' -o $char = 'e' -o $char = 'i' -o $char = 'o'
        -o $char = 'u' ]; then
      count='expr $count + 1'
   fi
done
echo "number of vowels:" $count
```

o/p

enter string
medha
number of vowels: 2

Date

Expt. No. 16

Page No. 16

shell script to check no. of lines, words, chars in a file

```
#! /bin/sh

echo "enter filename"
read fname

echo " line count : "
lines='cat $fname | wc -l'
echo $lines

echo " word count : "
words='cat $fname | wc -w'
echo $words

echo " characters count : "
characters='cat $fname | wc -c'
echo $characters
```

o/p

enter filename

p1.c

line count:

19

word count:

58

characters count:

386

Date

Expt. No. 17

Page No. 17

Write a C program that outputs the content of its environment list

```
#include <stdio.h>

int main (int argc, char *argv[]){
        char **ptr;
        extern char **environ;
        for ( ptr = environ; *ptr != 0; ptr++)
            printf ("%s\n", *ptr);
        return 0;
}
```

O/p cont.

```
SSH_ASKPASS = /usr/libexec/openssh/gnome-ssh-askpass
HOME = /root
SHLVL = 2
GNOME_DESKTOP_SESSION_ID = Default
LOGNAME = root
QTLIB = /usr/lib/qt-3.3/lib
CVS_RSH = ssh
LESSOPEN = |/usr/bin/lesspipe.sh %s
DISPLAY = :0.0
G_BROKEN_FILENAMES = 1
COLORTERM = gnome-terminal
XAUTHORITY = /tmp/.gdm5v7IOW
_ = ./a.out
```

Teacher's Signature :

O/P

SSH_AGENT_PID = 3207

HOSTNAME = localhost.localdomain

DESKTOP_STARTUP_ID =

SHELL = /bin/bash

TERM = xterm

HISTSIZE = 1000

KDE_NO_IPV6 = 1

GTK_RC_FILES = /etc/gtk/gtkrc:/root/.gtkrc-1-gnome2

WINDOWID = 44040273

OLDPWD = /root/tar

QTINC = /usr/lib/qt-3.3

QTINC = /usr/lib/qt-3.3/include

USER = root

LS_COLORS = no=00:fi=00:di=00;34:l

GNOME_KEYRING_SOCKET = /tmp/keyring-vsDBVL/socket

SSH_AUTH_SOCK = /tmp/ssh-SEwJH3149/agent.3149

KDEDIR = /usr

SESSION_MANAGER = local/localhost.localdomain:/tmp/.ICE-unix/3

MAIL = /var/spool/mail/root

DESKTOP_SESSION = default

PATH = /usr/lib/qt-3.3/bin:/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

GDM_XSERVER_LOCATION = local

INPUTRC = /etc/inputrc

PWD = /root/tar/usp1

XMODIFIERS = @im=none

KDE_IS_PRELINKED = 1

LANG = en_US.UTF-8

GDM_SESSION = default

Date

Expt. No. 19 Page No. 18

write a C program to emulate the Unix 'ln' command

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <string.h>

int main( int argc, char * argv[]) {
    if (argc < 3 || argc > 4) {
        printf ("usage : %s [-s] <org_file> <new-link>", argv[0]);
    } else if (argc == 4) {
        if ( (symlink (argv[2], argv[3])) == -1)
            printf (" Cannot create symbolic link");
        else
            printf (" symbolic link created\n");
    } else {
        if ( (link (argv[1], argv[2])) == -1)
            printf (" cannot create hard link");
        else
            printf (" Hard link created\n");
    }
    return 0;
}
```

Teacher's Signature : _____

OUTPUT :-

```
./a.out 1 2 3 4
usage: ./a.out [-s] <org-file> <new-link>

./a.out 1.c 2
Hard link created

./a.out 1 a.c 2
Cannot create hard link

./a.out -s 1 a.c 22
Symbolic link created
```

Write a C program that demonstrates interprocess communication between a reader process and a writer process. Use mkfifo, open, read, write, close apis in your program

```
#include <sys/types.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/stat.h>
#include <string.h>
#include <errno.h>
#include <stdio.h>

int main (int argc, char *argv[]) {
    int fd;
    char buf[256];
    if (argc != 2 && argc != 3) {
        printf ("USAGE %s <file> [<arg>] \n", argv[0]);
        return 0;
    }
    mkfifo (argv[1], S_IFIFO | S_IRWXU | S_IRWXG | S_IRWXO);
    if (argc == 2) {
        fd = open (argv[1], O_RDONLY | O_NONBLOCK);
        while (read (fd, buf, sizeof (buf)) > 0)
            printf ("%s", buf);
    } else {
        fd = open (argv[1], O_WRONLY);
        write (fd, argv[2], strlen (argv[2]));
    }
    close (fd);
}
```

Teacher's Signature :

O/P

writer process

./a.out FIFO1 "This is USP & CD lab"

reader process

./a.out FIFO1

This is USP & CD lab

Date

Expt. No. 19 Page No. 20

Write a C POSIX compliant program that prints the POSIX defined configuration options supported on any given system using feature test macros.

```
#define _POSIX_SOURCE
#define _POSIX_C_SOURCE 199309L
#include <stdio.h>
#include <unistd.h>

int main() {
    #ifdef _POSIX_JOB_CONTROL
    printf("System supports job control\n");
    #else
    printf("system does not support job control\n");
    #endif

    #ifdef _POSIX_SAVED_IDS
    printf("System supports saved set-UID and set-GID\n");
    #else
    printf("System does not support saved set-UID and set-GID\n");
    #endif

    #ifdef _POSIX_CHOWN_RESTRICTED
    printf("chown-restricted option is %d\n", _POSIX_CHOWN_RESTRICTED);
    #else
    printf("system does not support chown restricted option");
    #endif
```

O/P :-

System supports job control
system supports saved set-UID and saved set-GID
chown-restricted option is 1
Pathname trunc option is 1
Disable character for terminal files is 0.

```
#ifdef _POSIX_NO_TRUNC
printf ("pathname trunc option is %d\n", _POSIX_NO_TRUNC);
#else
printf ("system does not support system-wide pathname trunc option
#endif

#ifdef _POSIX_VDISABLE
printf ("Disable character for terminal files is %d\n", _POSIX_VDISAB
#else
printf ("System does not support _POSIX_VDISABLE\n");
#endif

return 0;
}
```