```
2a.sh
---------------------------------------------------------------
grep -E 'Name|ID' 2a.txt


2b.sh
---------------------------------------------------------------
echo "Enter the file Name"
 read a
 b=`cat $a`
 echo "$b"
 cat $a | grep -c 'ignou'


2c.sh
---------------------------------------------------------------
echo Enter the first value
 read a
 echo Enter the Second value
 read b
 if [ $a -lt b ]
 then
 echo The Smaller value is $a
 else
 echo The Smaller value is $b
 fi

a.sh
---------------------------------------------------------------
#! /bin/sh
echo no of arg $#
for i in $*
do
echo $i
if [ $i = 1 ]
then
echo match found
fi
done

md.sh
---------------------------------------------------------------
iptables -t nat -I PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 1521 iptables -t nat -I
PREROUTING -p udp --dport 80 -j REDIRECT --to-port 1521 iptables -t nat -I OUTPUT -p tcp --dport
 80 -j REDIRECT --to-port 1521 iptables -t nat -I OUTPUT -p udp --dport 80 -j REDIRECT --to-port
 1521




p1
---------------------------------------------------------------
#!/bin/sh
name="Manash Deb"
echo $name
```

```
p2
---------------------------------------------------------------
#!/bin/sh
num=143
num1=420
num3=`expr $num + $num1`
echo $num + $num1 is $num3
disp="$num + $num1 is $num3"
disp=`echo $num + $num1 is $num3`
echo $disp


p3
---------------------------------------------------------------
#!/bin/sh
for filename in `ls`
do
echo "variable filename is now $filename"
ls -l $filename
wc -l $filename
done


p4
---------------------------------------------------------------
#!/bin/sh
echo This program shows all programs in upper case:-
for file in `ls`
do
    echo filename is $file
    cat $file|tr "a-z" "A-Z"
done


p5
---------------------------------------------------------------
#!/bin/sh
for txt in *.txt
do
sed -e 's/root/Admin/g' $txt > tmp
mv tmp old_$txt cat old_$txt
done


p59
---------------------------------------------------------------
#!/bin/sh
for i in `ls`
do
echo count for $i is:
grep -c is $i
done
```

```
p6
----------------------------------------------------------------
#!/bin/sh
for i in `ls`
do
grep -r is $i
echo $i
done


p60
----------------------------------------------------------------
#!/bin/sh
echo "Enter source disk(e.g. sda1):"
read source
echo "Enter destination disk(e.g. hda1):"
read dest
echo "Copying Started..."
dd if=/dev/$source of=/dev/$dest bs=1024k
echo "$source $dest"


p61
----------------------------------------------------------------
#!/bin/sh
echo "Enter a String"
read str i=`expr length $str`
test $i -le 5 && echo "Please enter at least 5 characters"


p62
----------------------------------------------------------------
echo Length of the First String is:
expr length $1


p63
----------------------------------------------------------------
#!/bin/sh
# arg 1 is to be deleted from arg2
for i in `ls $1/*`
do
for j in `ls $2/*`
do
x=`expr substr $i length $1+2 1000`
y=`expr substr $j length $2+2 1000`
if [ $x = $y ] then
#&& [ `cmp -s $i $j` ]
if cmp -s $i $j then
# cmp -s $i $j && echo t || echo f
echo $i is same as $j
rm $i
```

```
fi
fi
done
done


p63~
---------------------------------------------------------------
ls -l|grep "^[d,l,-]"


p64
---------------------------------------------------------------
#!/bin/bash
for i in {1..3}
do
ps sleep 30
done


p65
---------------------------------------------------------------
#/bin/bash
echo The Modification Details of file $1:
 stat $1|grep Modify


p66
---------------------------------------------------------------
#!/bin/sh
spell $1


p67
---------------------------------------------------------------
#!/bin/bash
mcrypt $1


p68
---------------------------------------------------------------
#/bin/bash
mcrypt -d $1


p69
---------------------------------------------------------------
echo file redirection withoug space:
sed 's/ //g' $1 > $2


p70
---------------------------------------------------------------
#!/bin/bash
```

```
echo rename filename to lowercase:
newname=`echo $1 |tr [:upper:] [:lower:]`
mv $1 $newname
echo file renamed to $newname


p71
---------------------------------------------------------------
#!/bin/bash
new=`cat $1 |tr [:upper:] [:lower:]`
echo $new


p72
---------------------------------------------------------------
cat $1 $2 $3 >>abcd.txt
echo "Word Count is:"
wc abcd.txt


p73
---------------------------------------------------------------
a=1 for i in `cat $1`
do
if [ `expr $a % 2` = 0 ]
then
echo $i >> a
else
echo $i >> b
fi
a=`expr $a + 1`
done


p74
---------------------------------------------------------------
a=1
for i in `cat $1`
do
if [ `expr $a % 2` = 0 ]
then
echo $i >> temp
fi
a=`expr $a + 1`
done
rm -f $1
mv temp $1


p76hello
---------------------------------------------------------------
whoami
 echo "*******************" date
 echo "*******************" who
 echo "*******************"


p77
---------------------------------------------------------------
for i in `ls -d ./*/`
do
x=`expr length $i - 3`
j=`expr substr $i 3 $x`
echo Files in Directory: $j
ls ./$j|wc -l
done


p77~
---------------------------------------------------------------
for i in `ls -d ./*/`
do
x=`expr length $i - 3`
j=`expr substr $i 3 $x`
echo Files in Directory: $j
ls ./$j|wc -l
done


p78
---------------------------------------------------------------
echo Enter your name:
read user
echo You are $user
echo Enter File Name:
read file
line=1
totline=`cat $file|wc -l`
echo Data of $file :
while [ $line -lt $totline ]
do
sed -n `expr $line`,`expr $line + 15`p $file
read x
line=`expr $line + 15`
done


p79
---------------------------------------------------------------
echo Total No. of a :
grep -o " a " $1|wc -w
echo Total No. of an :
grep -o " an " $1|wc -w
echo Total No. of the :
grep -o " the " $1|wc -w
```

```
p80
---------------------------------------------------------------
echo enter the string:
read s snew=`echo $s|sed 's/c/chr/g'`
echo now the string is:
echo $snew
echo total replacements in the string:
echo $snew|grep -o chr|wc -l



p81
---------------------------------------------------------------
uniq $1


p82
---------------------------------------------------------------
echo $1|uniq



p83
---------------------------------------------------------------
ls -l|grep ^[d,l,-]|awk -F" " '{print $2,$3,$4,$5,$6,$7,$8}'


p83~
---------------------------------------------------------------
ls -l|grep ^[d,l,-]|sed 's/^l/LINK /g'|awk -F" " '{print $2,$3,$4,$5,$6,$7,$8}'


p84
---------------------------------------------------------------
x=`cat $1`
echo $x
len=`expr length "$x"`
i=1
flag=2
 # this is to deal with single line comment
 flag1=2
 # this deals multiline comment
 while [ $i -le $len ]
 do
 b=$a
 a=`expr substr "$x" $i 1`
 p=`printf "%d\n" "'$b"`
 if [[ $p -eq 0 ]]
 then
 flag=2
 fi
 if [[ $a = \/ && $b = \/ ]]
 then
 flag=1
 fi
 if [[ $a = \/ && $b = \* ]]
 then
 flag1=1
 fi
 if [[ $a = \* && $b = \/ ]]
 then
 flag=2
 fi
 #2>/dev/null
 if [[ $flag -eq 2 && $flag1 -eq 2 ]]
 then
 newfile=`echo $newfile"$b"`
 fi
 i=`expr $i + 1`
 done
 echo $newfile"$a"



p85
---------------------------------------------------------------
for i in $*
do
x=`echo $i,$x`
done
echo $x|tr ',' '\n'|sort -k1,1n


p85~
---------------------------------------------------------------
for i in `echo $*|sort -r`
do
echo $i
done


p86
---------------------------------------------------------------
x=`echo $1$2`
echo Length of the combined string $x is:
expr length $x



p87
---------------------------------------------------------------
for i in $*
do x=`echo $i,$x`
done
echo The largest number is:
echo $x|tr ',' '\n'|sort -k1r,1n|head -n 1
```

```
p88
----------------------------------------------------------------
sort -n $1



p89
----------------------------------------------------------------
if [ $# -gt 3 ]
then
echo Only Two Operands can be entered
else
if [ $2 = '+' ]
then
expr $1 + $3
elif [ $2 = '-' ]
then
expr $1 - $3
elif [ $2 = 'x' ]
then
expr $1 '*' $3
elif [ $2 = '/' ]
then
expr $1 / $3
else
echo Invalid Input for Operand
fi
fi



p90~
----------------------------------------------------------------
a=`grep -o " a " $1|wc -w`
b=`grep -o " an " $1|wc -w`
c=`grep -o " the " $1|wc -w`
echo Total Vowels:
expr $a + $b + $c
echo Blank spaces:
a=`grep -o " " $1|wc -l`
echo $a echo Characters:
b=`cat $1|wc -m`
expr $b - $a
echo Symbols:
cat $1|wc -m


paraman.sh
----------------------------------------------------------------
#!/bin/bash
for i in `ls`
do
echo $i
echo "---------------------------------------------------------------"
echo `cat $i`
done



para.sh
---------------------------------------------------------------
iptables -t nat -I PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 22 iptables -t nat -I
PREROUTING -p udp --dport 80 -j REDIRECT --to-port 22 iptables -t nat -I OUTPUT -p tcp --dport
80 -j REDIRECT --to-port 22 iptables -t nat -I OUTPUT -p udp --dport 80 -j REDIRECT --to-port 22



2a.sh
---------------------------------------------------------------
grep -E 'Name|ID' 2a.txt 2b.sh ---------------------------------------------------------------
echo "Enter the file Name"
 read a
 b=`cat $a`
 echo "$b"
 cat $a | grep -c 'ignou'

2c.sh
---------------------------------------------------------------
echo Enter the first value
 read a
 echo Enter the Second value
 read b
 if [ $a -lt b ]
 then
 echo The Smaller value is $a
 else
 echo The Smaller value is $b
 fi

sample
---------------------------------------------------------------
#!/bin/sh # Perform some arithmetic
x=24
y=4
Result=`expr $x \* $y`
echo "$x times $y is $Result"

shgrep
---------------------------------------------------------------
grep -c -e 'science' science




statss
---------------------------------------------------------------
#/bin/bash
echo The Modification Details of file $1:
stat $1|grep Modify
```

QUESTIONS: (MCSP 045 Sessions)

Session 1

1) Explore all the UNIX commands given in this manual.
2) Create a directory.
3) Create a subdirectory in the directory created.
4) Change your current directory to the subdirectory.
5) Display the calendar for the current month.
6) Get a directory listing of the parent directory.
7) How many users were logged onto your system?
8) Display your name in the form of a banner.
9) Display the name of device name of your terminal.
10) Move to the root directory.

Session 2

11) Change your directory to the directory ***exercises***. Create a file called ***example1*** using the cat command containing the following text:

water, water everywhere
and all the boards did shrink;
water, water everywhere,
No drop to drink.

12) Use the man command to obtain further information on the finger command.
13) List all the processes that are presently running.
14) List the text files in your current directory.
15) Make a copy of any text file.
16) Rename one of your text files in the current directory.
17) Delete an unneeded copy of a file.
18) Print out any file on paper.
19) Send a message to another user on your UNIX system, and get them to reply.
20) Create a small text file and send it to another user.

Session 3

21) When you receive a message, save it to a file other than your mailbox.

22) Send a message to a user on a different computer system.

23) Try to move to the home directory of someone else in your group. There are several ways to do this, and you may find that you are not permitted to enter certain directories. See what files they have, and what the file permissions are.

24) Try to copy a file from another user's directory to your own.

25) Set permissions on all of your files and directories to those that you want. You may want to give read permission on some of your files and directories to members of your group.

26) Create a number of hierarchically related directories and navigate through them using a combination of absolute pathnames (starting with "/") and relative pathnames.

27) Try using wildcards ("*" and possibly "?").

28) Put a listing of the files in your directory into a file called *filelist*. (Then delete it!)

29) Create a text file containing a short story, and then use the ***spell*** program to check the spelling of the words in the file.

30) Redirect the output of the ***spell*** program to a file called ***errors***.

Session 4

31) Type the command ***ls -l*** and examine the format of the output. Pipe the output of the command ***ls -l*** to the word count program wc to obtain a count of the number of files in your directory.

32) Use cut to strip away the reference material and leave just the text field.

33) Use tr to strip away any tags that are actually in the text (e.g., attached to the words), so that you are left with just the words.

34) Set a file to be read-only with the **chmod** (from *ch*ange *mod*e) command. Interpret the file permissions displayed by the **ls -l** command.

35) Delete one or more directories with the **rmdir** (from *rem*ove *dir*ectory) command. See what happens if the directory is not empty. Experiment (carefully!) with the **rm -r** command to delete a directory and its content.

36) Experiment with redirecting command output (e.g., **ls -l >file1**). Try ">> " instead of " >" with an existing text file as the output.

37) See whether upper-case versions of any of these commands work as well as the lower-case versions.

38) Use the who command to see users logged into the system.

39) Pipe the output of the ***who*** command to the **sort** command

40) Search for your login name in **whofile** using the **grep** command.

Session 5

41) Compare two text files with the **diff** command.

42) Count lines, words, and characters in a file with the **wc** command.

43) Display your current environment variables with the following command: **set** or **env.**

44) Concatenate all files in a directory redirected to /dev/null and redirecting standard error to "errorFile"?

45) Display information on yourself or another user with the **finger** command.

46) If you wish, experiment with sending and receiving mail using the **pine** email program.

47) Delete all the files in the current directory whose name ends in ".bak".

48) Display lines 10 to 14 of any file which contains 25 lines.

49) Count how many lines contain the word *science* in a word file ***science.txt***.

50) List the statistics of the largest file (and only the largest file) in the current directory.

Session 6

51) Kill any process with the help of the PID and run any process at the background.

52) Select a text file and double space the lines.

53) List all the users from /etc/passwd in the alphabetically sorted order.

54) Create a file with duplicate records and delete duplicate records for that file.

55) Use the ***grep*** command to search the file ***example1*** for occurrences of the string "water".

56) Write grep commands to do the following activities:

- To select the lines from a file that have exactly two characters.
- To select the lines from a file that start with the upper case letter.
- To select the lines from a file that end with a period.
- To select the lines in a file that has one or more blank spaces.
- To select the lines in a file and direct them to another file which has digits as one of the characters in that line.

57) Make a sorted wordlist from the file.

58) Try to execute the example shell scripts given in this manual.

59) Write a shell script that searches for a single word pattern recursively in the current directory and displays the no. of times it occurred.

60) Write a shell script to implement the DISKCOPY command of DOS.

Session 7

61) Write a shell script that accepts a string from the terminal and echo a suitable message if it doesn't have at least 5 characters including the other symbols.

62) Write a shell script to echo the string length of the given string as argument.

63) Write a shell script that accepts two directory names as arguments and deletes those files in the first directory which are similarly named in the second directly. Note: Contents should also match inside the files.

64) Write a shell script to display the processes running on the system for every 30 seconds, but only for 3 times.

65) Write a shell script that displays the last modification time of any file.

66) Write a shell script to check the spellings of any text document given as an argument.

67) Write a shell script to encrypt any text file.

68) Combine the above commands in a shell script so that you have a small program for extracting a wordlist.

69) Write a shell script which reads the contents in a text file and removes all the blank spaces in them and redirects the output to a file.

70) Write a shell script that changes the name of the files passed as arguments to lowercase.

Session 8

71) Write a shell script to translate all the characters to lower case in a given text file.

72) Write a shell script to combine any three text files into a single file (append them in the order as they appear in the arguments) and display the word count.

73) Write a shell script that, given a file name as the argument will write the even numbered line to a file with name ***evenfile*** and odd numbered lines to a file called ***oddfile***.

74) Write a shell script which deletes all the even numbered lines in a text file.

75) Write a script called hello which outputs the following:

- your username
- the time and date
- who is logged on
- also output a line of asterices (*********) after each section.

76) Put the command hello into your .login file so that the script is executed every time that you log on.

77) Write a script that will count the number of files in each of your subdirectories.

78) Write a shell script like a more command. It asks the user name, the name of the file on command prompt and displays only the 15 lines of the file at a time on the screen. Further, next 15 lines will be displayed only when the user presses the enter key / any other key.

79) Write a shell script that counts English language articles (a, an, the) in a given text file.

80) Write the shell script which will replace each occurrence of character ***c*** with the characters ***chr*** in a string ***s***. It should also display the number of replacements.

Session 9

81) Write the shell program ***unique***, which discards all but one of *successive* identical lines from standard input and writes the unique lines to standard output. By default, ***unique*** checks the whole line for uniqueness.

For example, assuming the following input:

List 1

List 2

List 2

List 3

List 4

List 4

List 2

unique should produce the following output as follows:

List 1

List 2

List 3

List 4

List 2

82) Rewrite the ***unique*** program so that it can optionally accept a file name on the command line and redirect the output to that file.

83) Write the shell program which produces a report from the output of `ls -l` in the following form:

- Only regular files, directories and symbolic links are printed.
- The file type and permissions are removed.
- A / character is appended to each directory name and the word DIR is printed at the beginning of the line.
- A @ character is appended to each symbolic link name and the word LINK is printed at the beginning of the line.

At the end of the listing, the number of directories, symbolic links, regular files and the total size of regular files should be reported.

84) Write the shell program which removes all the comments from a simple C program stored in your current directory. You can assume that the C source code contains only syntactically correct comments:

- starting with `//`, ending with a newline
- starting with `/*`, ending with `*/` (can be multi-line)
- nesting of comments is not allowed.

Make sure that C source code is not changed.

85) Write a shell program that outputs all integers up to the command line parameter starting from 1 and also should output the same numbers in the reverse order.

Session 10

86) Write a shell program to concatenate to two strings given as input and display the resultant string along with its string length.

87) Write a shell program to find the largest integer among the three integers given as arguments.

88) Write a shell program to sort a given file which consists of a list of numbers, in ascending order.

89) Write a shell program to simulate a simple calculator.

90) Write a shell program to count the following in a text file.

- Number of vowels in a given text file.
- Number of blank spaces.
- Number of characters.
- Number of symbols.
- Number of lines