Date 31/10/22
Page

Basic Unix Commands

- cal - shows a calendar of the current month
- date - shows the current date & time
- echo - outputs the string that are passed to it as args
- bc - Used for command line calculator
- man - Used to display the user manual of any commands
- ls - Used to list the names of files in particular
- mkdir - Allows users to create or make new directories
- cd - Allows users to change directory
- cat - Reads data from the file & gives their content as output
- pwd - Writes the full pathname of current working directory
- tty - Prints the filename of the terminal connected to standard output.
- Uname - Prints the name of the operating system that you are using
- who - Displays the list of users who are currently logged in.
- echo $path - list of directories where executable files are stored.
- echo $SHELL -
- wc - Used for printing new line, word & byte count for files
- rmdir - removes directory
- rm - removes files
- stty - Displays or changes characteristics of terminals

- ps - Displays the list of process currently running
- type describes how its arguments would be interpreted
- a - Prints all current settings in human readable form

Lab 1
Date: 7/11/22

- Input : tty
- Output : /dev/pts/10

- Input : stty
- Output : speed 38400 baud, line = 0;
  -brkint -imaxbel iutf8

- Input : stty -a
- Output : speed 38400 baud; rows 38; columns 142;
  line = 0;
  intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^
  . . . . . .

- Input : stty intr ^a

- Input : stty -a
- Output : speed 38400 baud; rows 38; columns 142;
  line = 0;
  intr = ^A; quit = ^\; erase = ^?; kill = ^U eof

- Input : script
  Output : Script started, file is typescript

classmate
Date 7/11/22
Page

① # Nano editor

```
#! bin/bash
echo " Hello World"
```

Unix terminal

```
chmod a+x test.sh
sh test.sh
```

② # Nano Editor

```
#! bin/bash
echo "Printing text with
  newline"
echo -e "Enter the file \n"
echo -n "Printing text
  without newline"
echo -e "Name \t USN"
```

Unix Commands

```
chmod a+x test1.sh
sh test1.sh
```

Nano Editor

```
#! bin/bash
  x = 5
echo $x
```

Unix terminal

```
chmod a+x test2.sh
sh test2.sh
```

④ # Nano Editor

```
#! /bin/bash
  var1 = 10
  var2 = 20
  sum = $ (( $var1 + $var2 ))
echo  $sum
```

Unix terminal

```
chmod a+x test3.sh
sh test3.sh
```

⑤ # Nano Editor

```
#! bin/bash
echo "Enter the value"
read -n
echo $n
```

Unix terminal

```
chmod a+x test4.sh
sh test4.sh
```

LAB-2

Que) Area of a circle
Que) Copy a set of files to a particular directory. Use copy, read, make dir command

→ Nano Editor

Ans)
```
#!/bin/bash
echo "Enter the radius of the circle"
read r
echo "Area is equal to :"
echo "3.14 * $r * $r " | bc
```

O/p
10
314

→ Unix Terminal

```
chmod a+x area.sh
sh area.sh
```

Ans)
```
#! bin/bash
echo " Enter the first file"
read x
echo " Enter the second file"
read y
echo " Creating a new directory"
mkdir newdirpg2
echo " Copying files"
cp $x newdirpg
cp $y newdirpg
```

Que Create 2 files with names flower1 & flow
In the first directory

Flower1
Rose
Daisy
Marigold Lily Daffodil

Flower2
Rose
Jasmine
Hibiscus Lily Dais

Terminal -> nano flower1 , nano flower2

Nano Editor

Flower 1 -> Rose
Daisy
Marigold
Lily
Daffodil

Flower 2 -> Rose
Jasmine
Hibiscus
Lily
Daisy

Unix terminal

cmp flower1 flower2

O/p -> flower1 flower2 differ : byte 6, line 2

diff flower1 flower2

O/p 2, 3c2, 3
< Daisy
< Marigold

> Jasmine
> Hibiscus
5c5
< Daffodil

> Daisy

comm flower1 flower2

O/p Rose
Daisy
Jasmine
Marigold
Comm -> file 1 is not in sorted order Lily Daffodil

LAB-3

① Write a program to check for leap year for all 3 conditions like divisible by 4, 400 & not divisible by 100.

```
#!/bin/bash
echo "Enter the year"
read y
a = `expr $y %4`
b = `expr $y %100`
c = `expr $y %400`

if [$a -eq 0 -a $b -ne 0 $c -eq 0]
then
echo "$y is a leap year"
else
echo "$y is not a leap year
```

Terminal Commands

```
chmod a+x leapyr.sh
sh leapyr.sh
```

Output

```
Enter the year
2024
2024 is a leap year
```

② Write a shell program to check whether the no. is +ve, -ve or 0

```
#!bin/bash
echo "Enter a number"
read num

if [$num -lt 0]
then
```

```
echo "Negative"
elif [ $num -gt 0 ]
then
 echo "Positive"
else
echo "The number is equal to zero"
fi
```

```
# Unix terminal
Chmod a+x test2.sh
sh test2.sh
```

```
# Output
Enter a number
12
Positive
```

(3) Write a shell program to find greatest of three numbers

```
#! bin/bash
echo "Enter num1"
read num1
echo "Enter num2"
read num2
echo "Enter num3"
read num3

if [ $num1 -gt $num2 ] && [$num1 -gt $num
then
echo $num1
elif [ $num2 -gt $num1] && [$num2 -gt $ nu
then
echo $num2
else
echo $num3
fi
```

```
# Terminal Commands
Enter num1
10
Enter num2
20
Enter num3
30
```

Extra Program

① Write a program to check whether its even o

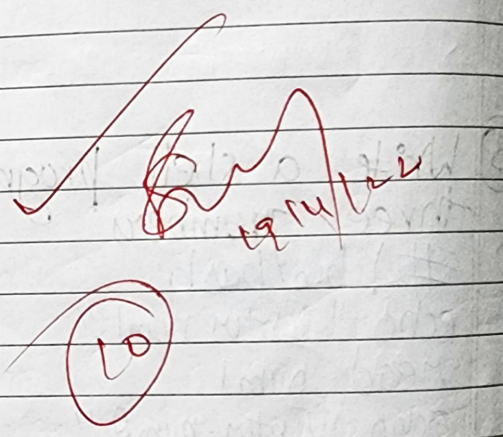
```
#!/bin/bash
echo "Enter a number"
read n
if [ `expr $n % 2` -eq 0 ]
then
echo "$n is even"
else
echo "$n is odd"
fi
```

Output

Enter a number

4

Number is even



10

LAB-4

① Shell script program to find the factorial of a number.

```
#!/bin/bash
echo "Enter a number"
read num
fact=1
while [ $num -gt 1 ]
do
fact = $((fact*num))
num = $((num-1))
```

```
# Terminal Commands
chmod a+x fact.sh
sh. fact.sh
```

```
# Output
Enter a number
4
24
```

② Shell script to compute the gross salary of an employee (given basic, hra 20% of basic, da 10% of basic)

```
#!/bin/bash
echo "Enter the salary"
read sal
grosal=$(echo "$sal+((20/100)*$sal)+((10/100)*$sal)" | bc -l)
echo "The gross salary is : $grosal"
```

Output

Enter the salary
100
The gross salary is : 130.0000

③ Shell script to convert the temperature fahrenheit to Celsius

```
#!/bin/bash
echo "Enter temperature in Fahrenheit"
read tf
tc=$(echo "scale = (5/9)*($tf-32)" | bc -l)
echo "$tf = tc"
```

Output: Enter the temperature in fahrenheit
100
100 fahrenheit = 37.4 Cel

④ Shell script to perform arithmetic operations on given 2 nos. (using case command)

```
#!/bin/bash
echo "Enter two numbers"
read a b
echo "Select the arithmetic operation"
echo "1) Sum"
echo "2) Difference"
echo "3) Product"
echo "4) Quotient"
echo "5) Remainder"
echo "Enter your choice"
read n
case "$ n" in
 1) echo " The Sum of $a and $b is 'expr $at$b
2) echo " The Difference of $a and $b is 'expr $a-$b
3) echo " The Product of $a and $b is 'expr' $a*$
4) echo " The Quotient of $a and $b is 'expr $a/$
5) echo " The Remainder of $a and $b is 'expr $a%$
esac
```

Output

```
Enter two numbers
10 20
Select the arithmetic operation
1) Sum
2) Difference
3) Product
4) Quotient
5) Remainder
Enter your choice
1
The Sum of 10 and 20 is 30
```

Shell Script to find the sum of even numbers upto n

```
#!/bin/bash
echo "Enter a number"
read n
$i=2    sum=0
while [$i -lt $n]
do
expr '$sum=$sum+$i'
expr '$i=$i+2'
done
echo "Sum is : $sum"
```

#Output
Enter the number
10
Sum is : 30

Shell script to find power of a number

```
#!/bin/bash
echo "Enter the base number"
read x
echo "Enter the power"
read n
res=1
while [ $n -gt 0 ]
do
res=$((res*x))
n=$((n-1))
done
echo $res
```

#Output
Enter the base number
2
Enter the power
3
8

Shell script to find the sum of n natural nos.

```
#!/bin/bash
echo "Enter the number"
read n
sum=0
while [ $n -gt 0 ]
do
sum=$((sum+n))
n=$((n-1))
done
echo $sum
```

Output-
Enter the number
10
55

5/12/22 LAB-5

① Shell script to print the combinations of numbers 123

```
#!/bin/bash
for i in 1 2 3
do
for j in 1 2 3
do
for k in 1 2 3
do
 echo $i $j $k
done
done
done
```

#Output

111	221	323
112	222	331
113	223	332
121	231	333
122	232	
123	233	
131	311	
132	312	
133	313	
211	321	
212	322	
213		

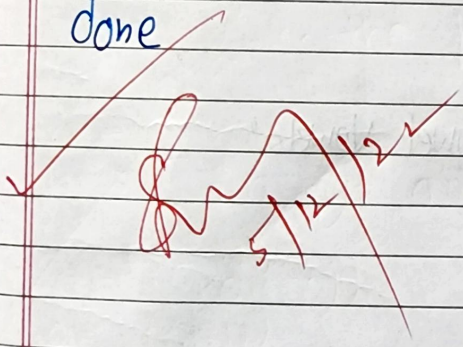
② Shell script to display the pass class of a student

```
#!/bin/bash
echo " Enter the marks "
read m
if [ $m -ge 40 && $m -lt 50 ]
then echo " Pass "
elif [ $m -ge 50 && $m -lt 60 ]
then echo " Second Class "
elif [ $m -ge 60 && $m -lt 70 ]
then echo " First Class "
    else
        echo " Distinction "
fi
```

#Output
Enter marks
50
SECOND CLASS

③ Shell script to print fibnocci series upto n

```
#!/bin/bash
echo "How many no. of terms to be generated"
read n
x=0
y=1
i=2
echo "Fibonacci Series upto $n terms:"
echo "$x"
echo "$y"
while[ "$i -lt $n ]
do
   i=`expr $i + 1`
   z=`expr $x + $y`
   echo "$z"
   x=$y
   y=$z
done
```



Output

```
Enter the number till the terms should be generated
10
Fibonacci Series:
0
1
1
2
3
5
8
13
21
34
```

12/12/22 LAB-6

① Shell script to count the number of vowels of a string

```
#!/bin/bash
echo "Enter the string"
read str
l=`expr length $str`
vowel=0
while [ $l -gt 0 ]
do
temp=`expr $str| cut -c $l`
case $temp
in
a|A) vowel=`expr $vowel+1`;;
e|E) vowel=`expr $vowel+1` ;;
i|I) vowel=`expr $vowel+1 ;;
o|O) vowel=`expr $vowel+1 ;;
esac
l=`expr $l-1
done
echo "The String has $vowel Vowels"
```

Output

```
Enter a String
Nadhya
The String has 4 vowels
```

(2) Shell script to check number of lines, words, characters in a file

```
#!/bin/bash
echo "Enter the file name"
read file
if [-f $file] then
 echo "file exists"
echo " Number of lines"
wc -l $file
echo "Number of characters"
wc -c $file
echo "Number of words"
wc -w $file
else
echo "File does not exist"
fi
```

Terminal Commands

```
cat > filename
File contents . . .
. . . . . ^C
```

Output

* Enter the file name
f2
File exists
Number of lines
2 f2
Number of characters
30 f2
Number of words
4 f2

* Enter the file name
vowel.sh
File exists
Number of lines
27 vowel.sh
Number of characters
362 vowel.sh
Number of words
69 vowel.sh

(3) Try commands set, shift trap:

```
#!/bin/bash
echo "Total no. of arguments $#"
echo "The argument are $*"
echo "The argument that is first is $1"
shift 2
echo "Now the first argument is : $1"
```

Terminal comands

```
set -x
```

LAB-7

① Write a shell script for GCD & LCM

```
#!/bin/bash
echo "Enter two integers"
read m n
echo "To find GCD & LCM"
echo "Given two numbers are"
echo "m = $m and n = $n
temp = `expr $m \* and $n = $n`
While [$m != $n]
do
if [$m -gt $n]
then
m = `expr $m - $n`
else
n = `expr $n - $m`
fi
done
echo GCD = $n
lcm = `expr $temp / $n`
echo LCM = $lcm
```

```
# Output
Enter two integers
10 15
To find GCD and LCM:
Given two numbers are
m=10 and n=15
GCD = 5
LCM = 30
```

② Write a shell script to print the below pattern

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

```
#!/bin/bash
number=1
rows=5
for((i=1; i<=rows; i++))
do
for((j=1; j<=i; j++))
do
 echo -n "$number"
number=$((number+1))
done
number=1
echo
done
```

#Output

```
1
12
123
1234
12345
```

(3) Write a shell script for creating Hard & soft link files.

```
#!/bin/bash
echo "Enter directory name"
read dir
echo "Enter soft link name"
read sl
if [-f $dir ]
then
echo "Creating softlink"
ln -s $dir $sl
ls -ltr
else
echo "File doesint exist"
fi
```

#Output

```
Enter directory name
marh.sh
Enter soft link name
Roadify.a
Creating Softlink total 85028
```

8Dec 19 11:8 Roadify.sh

Write a shell for hardlink

```
#!/bin/bash
echo "Enter directory name"
read dir
echo "Enter hardlink name"
read hl
if [ -f $dir ]
then
echo "Creating hardlink"
ln $dir $hl
ls -ltr
else
echo "File doesn't exist"
fi
```

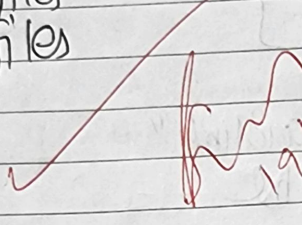
#Output

```
Enter directory name
vowels.sh
Enter hardlink name
Aaditya 123
Creating hardlink
total 85036
Aaditya -> vowels.sh
```

Write a shell script to locate & print from your home directory tree
All files having .html extensions
Directories having 666 permissions

```
#!/bin/bash
echo "Enter the directory name in which you want to find the file"
read dir
echo "Enter the type of file you want to find"
echo "1) HTML files
echo "2) 666 files
read type
case $type in
1) find $dir -name "*.html"
;;
2) find $dir -perm 666
;;
*) echo "Incorrect type"
esac
```

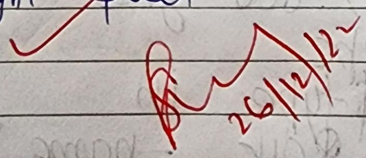
#Output
Enter the directory name in which you want to find the file
*
Enter the type of file you want to find
1) HTML file
2) 666 files



Lab-8

Based on the log-in time of the user display the message "Good morning" "Good Afternoon" & Good evening & Good night

```
#!/bin/bash
t=$(date |cut -c 25-26) user=$(who | cut -c 1)
if [ $t -gt 5 -a $t -le 12]
then
echo "Good Morning $user"
elif [ $t -gt 12 -a $t -le 16 ]
echo "Good Afternoon" $user"
elif [ $t -gt 16 -a $t -le 20 ]
then echo "Good Evening $user"
else
echo " Good Night $user "
fi
```



LAB-9

Create a database for employee details & perform followings operations on it

1. to display header & footer of a file
2. to print first 5 lines & last five lines of file
3. grep cmd with any 5 basic regular expression operations on i/p file
4. demo of cut cmd column & field wise both

Emp_id	Emp_name	Emp_sal
1	A	1000
2	B	2000
3	C	3000
4	D	4000
5	E	5000
6	F	6000
7	G	7000
8	H	8000
9	I	9000
10	J	10000

1. head -n 5 emp.txt → first 5 lines
2. tail -n 5 emp.txt → last 5 lines
3. grep "[a]" emp.txt

Emp_id	Emp_name	Emp_sal
1	A	1000

4. grep "^em" emp.txt

Emp_id	Emp_name	Emp_sal

5. grep "[e-p]" emp.txt

Emp_id	Emp_name	Emp_sal

```
grep '00$' emp.txt
1       A       1000
2       B       2000
3       C       3000
4       D       4000
5       E       5000
6       F       6000
7       G       7000
8       H       8000
9       I       9000
10      J       10000
```

```
grep "[c-p]" emp.txt
Emp_id  Emp_name  Emp_sal
1       A       1000
2       B       2000
3       C       3000
4       D       4000
5       E       5000
6       F       6000
7       G       7000
8       H       8000
9       I       9000
10      J       10000
```

```
pr -h "header" emp.txt
2023-01-02- 12:06 header Page1
```

LAB 10

1. Shell program on paste cmd
2. Shell program for sort with 3 options
3. Demo of Uniq & tr cmd
4. C-program that outputs the contents of its environment list

1.
```
#!/bin/bash
echo "Apple\n Orange\n Banana\n Pear" > fruit1.txt
echo "file1"
cat fruit1.txt
echo "----"
echo "Red\n Orange\n Yellow\n Green" > fruit2.txt
echo "file2"
cat fruit2.txt
echo "----"
echo "After paste command"
paste -d "\t" fruit1.txt fruit2.txt
```

O/P
```
file1
Apple
Orange
Banana
Pear
----
file2
Red
Orange
Yellow
Green
----
After Paste Command
Apple       Red
Orange      Orange
Banana      Yellow
Pear        Green
```

② #!/bin/bash
```
echo "Cherry\n Orange\n Apple\n Banana\n Pear"
     > fruit.txt
echo "Fruit file"
cat fruit.txt
echo " "
echo "Sorted in reverse order."
sort -r fruit.txt
echo " "
echo "Sorted in alphabetical order"
sort -f fruit.txt
echo "----"
echo "3\n1\n2" > numbers.txt
echo "Numbers file"
cat numbers.txt
echo " "
echo "Sort in numerical order"
sort -n numbers.txt
```

Output

fruit file
Cherry
Orange
Apple
Banana
Pear

Sorted in reverse order
Pear
Orange
Cherry
Banana
Apple

Sorted in alphabetical order
Apple
Banana
Cherry
Orange
Pear

Numbers file
3
1
2

Sorted in numerical order
1
2
3

```
#include<stdio.h>
int main(int argc, char* argv[])
{
int i;
char **ptr;
extern char **environ;
for(ptr = environ; *ptr != 0; ptr++)
printf("%s\n", *ptr);
return 0;
}
```

```
#!/bin/hash
echo "Apple\n Orange\n Banana\n Pear" > fruit.txt
echo "Orignal File"
cat fruit.txt
echo ""
echo "After removing duplicate"
Uniq fruit.txt
echo ""
echo "Changing case"
tr '[:lower:]' '[upper:]' < fruit.txt
   a-z          A-Z
```

Output cat unique

Apple
Apple
Chevry
Orange
Pear

Uniq Unique

Apple
Chevry
Orange
Pear

LAB - 11

① Write a C/C++ prog 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 || (argc==4 && strcmp
                        (argv[1], "-s")))
{
printf ("Usage: ./a.out [-s] <org_file> <new_link>");
return 1;
}
if (argc==4){
if ((symlink (argv[2], argv[3])) ==-1)
printf ("Cannot create symbolic link\n");
else
printf ("Symbolic link created\n");
}
else{
if ((link (argv[1], argv[2])) ==-1)
printf ("Cannot create hard link\n");
else{
printf ("Hard link created");
}
return 0;
}
```

(2) Write a C/C++ POSIX compliant prog. that prints the POSIX defined

```
#define _POSIX_SOURCE
#define _POSIX_C_SOURCE
#include <stdio.h>
#include <unistd.h>
int main () {
#ifdef _POSIX_JOB_CONTROL
printf ("System supports job control \n");
#else
printf ("System doesn't support job control \n");
#endif
#ifdef _POSIX_SAVED_IDS
printf ("System supports saved set-UIDs and saved set-GIDs\n");
#else
printf ("System doesn't not support saved set-UIDs and saved set-GIDs\n");
#endif
#ifdef _POSIX_CHOWN_RESTRICTED
printf ("Chown restricted option is %d\n", _POSIX_CHOWN_RESTRICTED);
#else
printf ("System does not support saved chown restricted option\n");
#endif
#ifdef _POSIX_NO_TRUNC
printf ("Pathname trunc option is %d\n", _POSIX_NO_TRUNC);
#else
printf ("System does not support pathname trunc option\n");
#endif
#ifdef _POSIX_VDISABLE
printf ("Disable character for terminal files is option is
        %d\n", _POSIX_VDISABLE);
#else
printf ("Syste does not support disable character
        for terminal files\n");
#endif
return 0; }
```

(3) Write a C/C++ prog which demonstrates interprocess communication b/w a reader process & a write process

```
#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);
}
```