command: date

+% D,T to display

-s "date" to set

+% d,m,y,H,M,S for day month and year,hour,minute,seconds

+% a,A,b,B for abbreviated day,month

Commands for Managing Users and Groups in Linux (root user only)

useradd user -> for new or existing user profile

addgroup group -> add a group

adduser -> create a user account

adduser –ingroup sudo “groupname”

sudo usermod -a -G group1,group2 username

deluser abc

deluser –group example

deluser -removehome abc

passwd OR passwd abc

Commands for File Basics:

touch -t 06011102 file1
file HelloWorld.c
ln file1 link1
ln -s file1 slink1

Commands for displaying files
Tail , head , cat for displaying files

Commands for Copy, Move, Rename or Remove Files or Directory

cp -R <source_folder> <destination_folder>

cp -R /etc /etc_backup

cp -R <source_folder>/* <destination_folder> (only content)

mv file1.txt file.2.txt file3.txt folder

rm file1.txt

rename 's/\.txt/\.png/' *.tx

rename 's/file/document/' *.png

Commands for Directory Basics

pwd

mkdir -p dir1/subdir/subsubdir

ls /etc

*ls *.{htm,php,cgi}*

*ls [aeiou]**
‘$ ls file’ - list all the files in current directory starting with filename ‘file’.*
*‘$ ls *2.txt’ - list all the files in current directory ending with ‘2.txt’*
‘$ ls file.tx?’ - list all the files that begins with ‘file.tx’
‘$ ls rmt[12345]’ - list all the file that begins with ‘rmt’ and has a 1,2,3,4 or 5 after it.

Commands for File/Directory Permissions and Ownerships

R=4 w=2 x=1

chmod u=rwx,g=rx,o=r myfile == chmod 754 myfile

chown linuxize file1 dir1

chown -R user dir1

chown USER:GROUP FILE

OTHER USEFUL COMMANDS

grep [options] pattern [files]
grep -c "unix" geekfile.txt

Options Description
-c : This prints only a count of the lines that match a pattern
-h : Display the matched lines, but do not display the filenames.
-i : Ignores, case for matching
-l : Displays list of a filenames only.
-n : Display the matched lines and their line numbers.
-v : This prints out all the lines that do not matches the pattern
-e exp : Specifies expression with this option. Can use multiple times.
-f file : Takes patterns from file, one per line.
-E : Treats pattern as an extended regular expression (ERE)

-w : Match whole word
-o : Print only the matched parts of a matching line with each such part on a separate output line.

ps -A <processes with PID output>
alias => Renames a command alias l=’ls -al’

Shell scripting

```
#!/bin/bash => to show that file is a bash file
```

```
chmod +x test.sh => make the script executable
```

```
$./test.sh => to execute
```

Variables:

```
VAR1 = "Zara Ali"

VAR2 = 100
```

Reading values

```
Read x y
Echo “the value of x and y is $x $y”
Read -p “enter the values of x and y” x y
Echo “the value of x and y is $x $y”
```

Sending parameters in file:

```
./filename 1 2 3
```

Arguments in shell file:

There are different arguments that can be written in shell file

```
echo "File Name: $0"
echo "First Parameter : $1"
echo "Second Parameter : $2"
echo "Quoted Values: $@"
echo "Quoted Values: $*"
echo "Total Number of Parameters : $#"
```

and $? => exit status of last command

$$ => process number of shell

$! => process number of last command

Arthematic operations

Formula must be enclosed like ((formula))

```
x=8
y=2
echo "x=8, y=2"
echo "Addition of x & y"
num1 = $(( $x + $y ))
echo "Subtraction of x & y"
((num2 = $x - $y ))
echo "Multiplication of x & y"
echo $(( $x * $y ))
echo "Division of x by y"
echo $(( $x / $y ))
echo "Exponentiation of x,y"
echo $(( $x ** $y ))
echo "Modular Division of x,y"
echo $(( $x % $y ))
echo "Incrementing x by 5, then x= "
(( x += 5 ))
echo $x
echo "Decrementing x by 5, then x= "
(( x -= 5 ))
echo $x
echo "Multiply of x by 5, then x="
(( x *= 5 ))
echo $x
echo "Dividing x by 5, x= "
(( x /= 5 ))
echo $x
```

```
echo "Remainder of Dividing x by 5, x="
(( x %= 5 ))
echo $x
```

Conditional statements:

Format = [[a > b]] => space before and after command

1	-eq		Equal to	[[$count -eq 10]]
2	-ne	!=	Not equal to	[[$total -ne 1000]]
3	-lt	<	Less than	[[$balance -lt 0]]
4	-le	<=	Less than or equal to	[[$C -ie $B]]
5	-gt	>	Greater than	[[5 -gt 6]]
6	-ge	>=	Greater than or equal to	[[$total -ge $subtotal]]

S.NO	Operator	**Description**	**Example**
1	|	OR	[[$x = 0 | $y = 0]]
2	&&	AND	[[$A > $B && $B > $C]]
3	!	NOT	[[! $sum = 0]]

```
echo -n "Enter the first number: "
read VAR1
echo -n "Enter the second number: "
read VAR2
echo -n "Enter the third number: "
read VAR3
if [[ $VAR1 -ge $VAR2 ]];then
  if [[ $VAR1 -ge $VAR3 ]];then
    echo "$VAR1 is the largest number."
  else
    echo "$VAR3 is the largest number."
  fi
```

```
else
  if [[ $VAR2 -ge $VAR3 ]];then
    echo "$VAR2 is the largest number."
  else
    echo "$VAR3 is the largest number."
  fi
fi
```

Case Statement

Here is an example using the `case` statement in a bash script that will print the official language of a given country:

Example 1:

```
#!/bin/bash
echo "Enter the name of a country: "
read COUNTRY
echo "The official language of $COUNTRY is "
case $COUNTRY in
  [A-Z]|[a-z])
    echo "Lithuanian"
    ;;
  Romania | Moldova)
    echo "Romanian"
    ;;
  Italy | "San Marino" | Switzerland | "Vatican City")
    echo "Italian"
    ;;
  *)
    echo "unknown"
    ;;
Esac
```

Iteration in shell

While loop:

```
counter=1

while [ $counter -le 10 ]

do

echo $counter

((counter++))

done
```

For loop type 1:

```
#!/bin/bash

names='Stan Kyle Cartman'

for name in $names

do

echo $name

done
```

For loop type 2:

```
for value in {1..5..2}=>1 to 5 with 2 step

do

echo $value

done
```

For loop type 4:

```
for (( c=1; c<=5; c++ ))
do
   echo "Welcome $c times"
done
```

For loop type 5:

```
for i in $(seq 1..20)
do
   echo "Welcome $i times"
done
```

Functions:

```
Min() {
     if [[ $1 -lt $2 ]]; then
          Smallest=$1

else

          Smallest=$2

     fi

read -p “Enter two whole numbers, Separated by space: " N1 N2
Min $N1 $N2

echo “The smallest is: " $SmaIIest
```

Lab 4

```
#include<stdio.h>
#include <sys/types.h>
#include<unistd.h>
int main() {
pid_t pid;
pid = fork();
if(pid == 0) {
printf("I am child and my parent is %d and my own PID is %d\n", getppid(),
getpid());
}
else if(pid > 0) {
printf("I am a Parent and my pid is %d\n", getpid());
}
return 0;
```