vi Editor Commands

Command	Description	Example
vi	To create a shell script through vi editor	vi first.sh
sh	To execute the shell script	sh first.sh
bash	To execute the shell script	bash first.sh
./	To execute the shell script	./ first.sh
i	To insert text into vi editor	Esc i
wq or x	To save & exit from editor to terminal	Esc :wq or :x
q	To quit editing mode with no changes	Esc :q
!q	To quit without saving.	Esc :!q
w	To save & remain in editing mode.	Esc :w
dd	To delete entire line.	dd
Operator	**Type**	**switch**
!	Logical Not	!
Less than	Relational	-lt
Less than equal to	Relational	-le
Greater than	Relational	-gt
Greater than equal to	Relational	-ge
Equal to	Relational	-eq
Not equal to	Relational	-ne
AND	Logical and (True if all conditions are true)	-a
OR	Logical or (False if all conditions are false)	-o
File Operators	True if file exists and has size > zero	-s filename
	True if file exists and is not a directory	-f filename
	True if a directory exists	-d dirname
	True if file exists and has read permission	-r filename
	True if file exists and has write permission	-w filename
	True if file exists and has execute permission	-x filename

Note:- 1) To type script type in command prompt → vi filename For example:- vi s1
Press Esc + i (For insert mode)
Type script
Press Esc + : (colon sign) +wq or x (for last line mode)
2) To show the script type in command prompt → sh filename For example :- sh s1

1) **Script to welcome the user who logged into the system.**

```
clear
echo “Hello”
echo  “`who am i |  cut –d “ “ –f 1` “
echo “Welcome to Ubuntu Linux”
```

2) **Script to welcome the user and display login date and time.**

```
clear
echo “Hello”
echo “`who am i   | cut –d “ “ –f   1` “
echo “Welcome to Ubuntu Linux”
echo “The current date and time “
date
```

3) **Shell script to show use of variables.**

```
clear
name=”Sonu”
subject=”Computer science”
echo “My name is $name”
echo “I like to study $subject.”
```

4) **Shell script to accept value of variables form user.**

```
clear
echo –n  “Enter your name”
read name
echo –n “Enter your favourite subject”
read subject
echo “ I am $name and I like to study $subject”
```

5) **Shell script to display total number lines in a file.**

```
clear
echo  “ Enter a file name”
read    fname
echo    “There are `cat   $fname   |  wc   –l` lines in  $fname  file.”
```

6) **Shell script to find the size of   a file.**

```
clear
echo  “ Enter a file name”
read fname
echo “The size of $fname file  is `cat     $fname     |    wc   –c` Bytes.”
```

7) Script to addition of two numbers

```
clear
echo –n “Enter first number”
read num1
echo –n “Enter second number”
read num2
sum=`expr $num1 +  $num2`
echo “The addition of $num1 and $num2 is $sum”
```

Note:- Use other mathematical operators like Subtraction (-) , division (/), multiplication (*) and for remainder (%).

8) Script to calculate age of user in years.

```
clear
echo  “Enter year of your birth : “
read byear
cyear=`date  | tr  –s  ‘ ‘ | cut  –d  “ “  –f   6`
age=`expr  $cyear  -  $byear`
echo “You are $age years old as of today.”
```

Note :- You can also use cyear=`date +”%Y” `

9) To display the date and time in the given format:-

“Today is December 15, 2019 and current time is 12:10:23”

```
clear
echo “Today  is `date +”%B %d, %Y” ` and current time is `date +”%T” `. “
```

10) To display the login details of current user in the following format.

Name of the user :
Login date:
Login time :

```
clear
echo “Name of the user   : `who am i  | cut  –d  “  “ –f   1`. “
echo “Login date           : `who am i  | tr –s  ‘  ‘ | cut  –d  “ “ –f   4`. “
echo “Login time            : `who am i  | tr  -s  ‘  ‘  | cut  –d  “ “ –f   5`. “
```

11) To display the date, time, user name and current directory.

```
clear
echo “Current date is `date +”%d” ` ”
echo “Current time is `date +”%T” ` “
echo “User name is `who am i  | cut –d  “ “ –f  1`  “
echo  “Current directory is `pwd` “
```

12) To accept two file names from the user and creates new file containing the contents of both the files provided as input.

```
clear
echo  “Enter two file names”
read  a   b
echo “Enter new file name “
read  c
cat $a   $b  > $c
echo “$c file created successfully”
echo “to display the contents of $c file is ”
cat $c
```

13) To accept two file names from the user and compare them. (without if condition)

```
clear
echo “ Enter two file names”
read   a  b
cmp    $a    $b
```

14) Script to find out how many processes user is running.

```
clear
echo –n  “Enter user name”
read username
cnt=`ps –ef  |  cut  -d “  “ –f  1 |  grep –o $username  | wc –w`
echo “User  $username is running $cnt processes”
```

15) Script to find out how may terminals a user has opened.

```
clear
user=”user11”
cnt=`who | cut –d  “  “ –f   1  | grep –o  $user  | wc –w`
echo “ User  $user  has opened $cnt terminals”
```

OR

With the user of shell variable $1.

```
clear
cnt=`who | cut –d  “  “ –f   1  | grep –o  $1  | wc –w`
echo “ User  $1 has opened $cnt terminals”
```

For output :- sh filename user11

16) Script to display welcome message to the user.

```
clear
hour=`date   +”%H” `
username=`who am i  |   cut  -d   “  “   -f   1`
if  [  $hour   -ge    0    -a    $hour   -lt   12   ]
then
echo   “Good Morning $username, welcome to Ubuntu Linux Session”
elif  [  $hour   -ge    12    -a    $hour   -lt   18   ]
then
echo   “Good Afternoon  $username, welcome to Ubuntu Linux Session”
else
echo   “Good Evenin $username, welcome to Ubuntu Linux Session”
fi
```

17) Script to create a directory with appropriate message.

```
clear
echo –n  “Enter directory name : “
read dir
if  [ -d  $dir  -o –f  $dir ]
then
echo “A file or directory with the name $dir already exists.”
else
mkdir $dir
echo   “Directory with name $dir created successfully”
fi
```

18) Script to create a new file.

```
clear
echo  “Enter a file name”
read fn
if  [  -r   $fn   ]
then
echo   “$fn file already exists”
else
cat > $fn
echo   “$fn file created successfully”
fi
```

Note :- If the file does not exists than type some contents and press ctrl + d. After this file created successfully message to be displayed on screen.

19) Script to copy file into another file.

```
clear
echo “Enter your source file name”
read sf
echo “Enter destination file name”
read df
if [  -r    $sf   ]
cp  $sf   $df
then
echo “$sf file copied successfully”
else
echo “$sf file does not exists”
fi
```

20) Script to rename a file.

```
clear
echo  -n  “Enter your file name  “
read   of
echo  -n   “Enter new file name”
read   nf
if [  -r  $of   ]
then
mv  $of   $nf
echo   “$of file renamed successfully”
else
echo   “$of file does not exists”
fi
```

21) Script to remove a file.

```
clear
echo    -n    “Enter file name”
read   fn
if  [  -r    $fn  ]
then
rm   $fn
echo  “$fn file removed successfully”
else
echo   “$fn file does not exists”
fi
```

22) Script to check both strings are equal or not.

```
clear
echo  “Enter first string”
read  str1
echo  “Enter second string”
read str2
if [ $str1  =  $str2 ]
then
echo “Both strings are equal”
else
echo  “Both strings are not equal”
fi
```

23) Print 1 to 10 numbers using for loop.

```
clear
for a in 1 2 3 4 5 6 7 8 9 10
do
echo -n  “$a “
done
```

Note:- If user uses -n keyword with echo command than print 1 to 10 numbers horizontally. If user does not use –n keyword with echo command than print 1 to 10 numbers vertically.

24) Print A to H using for loop.

```
clear
for a in A B C D E F G H
do
echo -n  “$a “
done
```

25) Print Monday to Thursday using for loop with shell variables.

```
clear
for a in “$*”
do
echo  “$* “
done
```

Note :- Here $* means shell variables with any of the values that is pass in a command line. To execute this script type:- sh filename Moday Tuesday Wednesday Thursday

26) Print 1 to 10 numbers using while loop.

```
clear
a=1
while [ $a –le 10 ]
do
echo “$a  ”
a=`expr $a  + 1`
done
```

27) Print 10 to 1 numbers using while loop.

```
clear
a=10
while [ $a –ge 1 ]
do
echo “$a  ”
a=`expr $a  - 1`
done
```

28) Print odd number from 1 to 10 using while loop.

```
clear
a=1
while [ $a –le 10 ]
do
echo “$a  ”
a=`expr $a  + 2`
done
```

29) Print even numbers from 2 to 10 using while loop.

```
clear
a=2
while [ $a –le 10 ]
do
echo “$a  ”
a=`expr $a  + 2`
done
```

30) Print (2 * 1 = 2) till (2 * 10 = 20) using while loop.

```
clear
i=2
j=1
while [ $j –le 10 ]
do
k=`expr $i  \*  $j`
echo “$i   *   $j   =   $k”
j=`expr   $j    +   1`
done
```

31) Print the following patterns:-

```
1)   *                    2)   1                     3) 1
     * *                       2    2                   1   2
     *   *   *                 3    3    3              1   2   3
     *   *   *   *             4    4   4   4           1   2   3   4
     *   *   *   *   *         5    5    5   5   5      1   2   3   4   5
```

```
clear
i=1
while [ $i –le 5 ]
do
        j=1
        while [ $j –le $i  ]
        do
        echo   -n   “*   ”
        j=`expr $j  + 1`
        done
echo   “  ”
i=`expr $i  + 1`
done
```

Note:- If user print **echo –n “$i “** than 2nd pattern to be print.
If user print **echo –n “$j “** than 3rd pattern to be print.

32) Print 21 to 30 number using until loop.

```
clear
a=21
until  [ $a –gt  30 ]
do
echo “$a  ”
a=`expr $a  + 1`
done
```

Note:- The **while loop** executes till the condition is **true**.
The **until loop** executes till the condition is **false**.

33) Print user’s choice table.

```
clear
echo  –n “Enter no. to print the table  : ”
read  i
j=1
until [ $j  -gt  10 ]
do
k=`expr  $i  \*  $j`
echo   “$i   *  $j   =  $k”
j=`expr $j  +  1`
done
```

34) Print Series of 1 to user’s choice.

```
clear
echo –n “Enter the end number.”
read    n
i=1
while  [  $i  -le  $n  ]
do
echo  “ $i  ”
i=`expr  $i  +  1`
done
```

35) Sum of N numbers with while loop:-

```
clear
echo –n “Enter the end number”
read n  or n=$1
i=1
sum=0
            while [ $i  -lt  $n  ]
            do
            sum=`expr   $sum  +  $i`
            i=`expr  $i   +  1`
            done
echo “The sum of $n numbers are $sum”
```

36) Example of shell variable or command line variable.

```
clear
echo   “The name of the file is     :  $0”
echo   “The first parameter is       :  $1”
echo   “The second parameter is  :   $2”
echo   “The quoted value is          :  $@”
echo  “The quoted value is           : $*”
echo  “The value supplied to the shell script are : $#”
echo   “The process number of the current shell is : $$”
echo   “Exit  :  $?”
```

Note: When user want to execute the above script type:- sh filename mon tue wed thurs fri satur

37) Enter any three files and find the file with maximum size.

```
clear
echo  “Enter any three file names which is already created”
read   f1    f2   f3

s1=`wc  -c  $f1   | cut   -d   “   “   -f   1`
s2=`wc  -c  $f2   | cut   -d   “   “   -f   1`
s3=`wc  -c  $f3   | cut   -d   “   “   -f   1`

echo   “Size of $f1  =  $s1”
echo   “Size of $f2  =  $s2”
echo   “Size of $f3  =  $s3”

if  [   $s1   -gt     $s2    -a     $s1     -gt     $s3   ]
then
echo  “$f1 file has maximum size”
else   if  [   $s2    -gt     $s3    ]
then
echo   “$f2 file has maximum size”
else
echo   “$f3  file has maximum size”
fi
fi
```

38) Script to display day’s of a week using switch case.

```
clear
echo   “Enter your choice between 1 to 7”
read   n
case   $n   in
1)
            echo   “Monday”
;;

2)
            echo   “Tuesday”
;;

3)
            echo   “Wednesday”
;;

4)
            echo   “Thursday”
;;

5)
            echo   “Friday”
;;
```

```
6)
        echo  "Saturday"
;;

7)
        echo  "Sunday"
;;

*)
        echo  "Incorrect choice……Try again…."
;;
esac
```

39) Script to use user's choice mathematical operations using switch case.

```
Clear
echo  "choice 1- Addition"
echo  "choice 2- Subtraction"
echo  "choice 3- Division"
echo  "choice 4- Multiplication"
echo  "choice 5- Remainder"

echo  "Enter your choice between 1 to 5"
read  n

echo  -n   "Enter no. first  :- "
read  n1
echo  -n   "Enter no. second   :-"
read  n2

case  $n  in
1)
        echo  "sum = `expr  $n1  +  $n2` "
;;

2)
        echo  "sub = `expr  $n1  +   $n2` "
;;

3)
        echo  "div  = `expr  $n1  /   $n2` "
;;

4)
        echo  "Mul  = `expr  $n1  \*  $n2` "
;;

5)
        echo  "Rem  = `expr   $n1  %  $n2` "
;;
```

```
*)
        echo    "Incorrect choice ….Try again"
;;

esac
```

40) Script to user's choice and do the file operations using switch case.

```
clear
echo   "1 -  To create a file"
echo   "2 -  To copy a file"
echo   "3 -  To rename a file"
echo   "4 -  To remove a file"

echo   "Enter your choice [ 1 – 4 ]"
read   ch

case  $ch   in

1)
echo  "Enter a file name"
read fn
if  [  -r   $fn   ]
then
echo   "$fn file already exists"
else
cat > $fn
echo   "$fn file created successfully"
fi
;;

2)
echo "Enter your source file name"
read sf
echo "Enter destination file name"
read df
if [ -r   $sf  ]
cp  $sf   $df
then
echo "$sf file copied successfully"
else
echo "$sf file does not exists"
fi
;;

3)
echo  -n  "Enter your file name  "
read   of
echo  -n   "Enter new file name"
read   nf
```

```
if [  -r  $of   ]
then
mv  $of   $nf
echo   “$of file renamed successfully”
else
echo   “$of file does not exists”
fi
;;

4)
echo    -n    “Enter file name”
read   fn
if  [  -r    $fn  ]
then
rm   $fn
echo  “$fn file removed successfully”
else
echo   “$fn file does not exists”
fi
;;

*)
echo    “Incorrect choice ….Try again”
;;
esac
```

41) **shell script to convert file contents to lower or upper case as specified by user:**

```
clear
echo “Enter 1. for Upper case 2. for Lower case 3. for Exit”
read ch

echo “Enter file name”
read f

case  $ch  in

1)
cat  $f  |  tr  ‘[a-z]’  ‘[A-Z]’   > new
cat new
;;
2)
cat   $f   |  tr  ‘[A-Z]’   ‘[a-z]’  >  new
cat  new
;;

3)
exit 1
;;
```

```
*)
echo  “Sorry ! incorrect choice, try again………..”
;;
esac
```

42) Shell script to convert file contents to upper case.

```
clear
echo   -n    “Enter any file name :- “
read fn
cat   $fn  |  tr    ‘[a-z]’     ‘[A-Z]’
```

43) User define functions in shell script.

1) Create a today() function to display system date in mm/dd/yy format.

```
today( )     #  today() is a user define function.
{
curdate=`date   +”%D”`
echo     “$curdate”
}

clear
today
```

2) Create a sum() function to calculate the sum of two numbers.

```
sum( )        # sum() is a user define function.
{
sum=`expr    $a  +  $b`
echo   “sum = $sum”
}

clear
echo   “Enter any two numbers”
read    a    b
sum
```