Experiment No. 5

Title: Shell scripting in Linux

Batch: B2 **Roll No: 16010420117** **Experiment No: 5**

Aim: Introduction to Shell Scripting.

Resources needed: Any open source OS / Cocalc online editor

Theory:
Pre lab/Prior concepts:

Create a script

As discussed earlier shell scripts stored in plain text file, generally one command per line. You can use text editor such as vi or emacs. I recommend using vim (Vi Improved) as it is equipped with features such as syntax highlighting and indenting. Most modern Linux (or *BSD) distribution comes with vim. You can start vi or vim from shell prompt by typing any one of the following command:

```
$ vi  myscript.bash
$ vi  myscript.sh
$ vim  myscript.bash
```

Make sure you use .bash or .sh file extension for each script. This ensures easy identification of shell script.

Setup executable permission

Once script is created, you need to setup executable permission on a script. Why to setup executable permission, might be next question in your mind, right?

Simple,

- Without executable permission, running a script is almost impossible
- Besides executable permission, script must have a read permission

Syntax to setup executable permission:

chmod permission your-script-name

Examples:

```
$ chmod +x your-script-name
$ chmod 755 your-script-name
```

Run a script (execute a script)

Now your script is ready with proper executable permission on it. Next, test script by running it.

Syntax:

bash your-script-name
sh your-script-name
./your-script-name
Examples:

$ bash bar
$ sh bar
$ ./bar
In last example, you are using . (dot) command which read and execute commands from filename in the current shell. If filename does not contain a slash, file names in PATH are used to find the directory containing filename. For example:

./bar
bar script executed from current directory. The specialty of dot (.) command is you do not have to setup an executable permission on script.

Debug a script if required

While programming shell sometimes you need to find out errors (bugs) in shell script and correct all errors (remove errors i.e. debug script). For this purpose you can pass -v and -x option to sh/bash command to debug the shell script. General syntax is as follows: Syntax:

sh option { shell-script-name }
OR

bash option { shell-script-name }
Where,

- **-v**: print shell input lines as they are read
- **-x**: after expanding each simple-command, bash displays the expanded value of PS4 system variable, followed by the command and its expanded arguments.

Debugging discussed later in depth. Now you are ready to write first shell script that will print "Knowledge is Power" on screen. See the common vi command list , if you are new to vi.

```
$ vi first#
# My first shell script#
clear
echo "Knowledge is Power"
```

After saving the above script, you can run the script as follows:
$ ./first

This will not run script since we have not set execute permission for our script first; to do this type command
$ chmod 755 first
$ ./first

First screen will be clear, then Knowledge is Power is printed on screen.

Script Command(s)	Meaning
$ vi first	Start vi editor
# # My first shell script #	# followed by any text is considered as comment. Comment gives more information about script, logical explanation about shell script. *Syntax:* # comment-text
clear	clear the screen

Tip: For shell script file try to give file extension such as .sh, which can be easily identified by you as shell script.

Variables in Shell

To process our data/information, data must be kept in computers RAM memory. RAM memory is divided into small locations, and each location had unique number called memory location/address, which is used to hold our data. Programmer can give a unique name to this memory location/address called memory variable or variable (Its a named storage location that may take different values, but only one at a time).

In Linux (Shell), there are two types of variable:
(1) System variables - Created and maintained by Linux itself. This type of variable defined in CAPITAL LETTERS.
(2) User defined variables (UDV) - Created and maintained by user. This type of variable defined in lower letters.

You can see system variables by giving command like **$ set**, some of the important System variables are:

System Variable	Meaning
BASH=/bin/bash	Our shell name
BASH_VERSION=1.14.7(1)	Our shell version name
COLUMNS=80	No. of columns for our screen
HOME=/home/vivek	Our home directory
LINES=25	No. of columns for our screen
LOGNAME=students	students Our logging name
OSTYPE=Linux	Our Os type
PATH=/usr/bin:/sbin:/bin:/usr/sbin	Our path settings
PS1=[\u@\h \W]\$	Our prompt settings
PWD=/home/students/Common	Our current working directory
SHELL=/bin/bash	Our shell name
USERNAME=vivek	User name who is currently login to this PC

NOTE that Some of the above settings can be different in your PC/Linux environment. You can print any of the above variables contains as follows:

$ echo $USERNAME
$ echo $HOME

How to define User defined variables (UDV)

To define UDV use following syntax
Syntax:
variable name=value

'value' is assigned to given 'variable name' and Value must be on right side = sign.

Example:
$ no=10# this is ok
$ 10=no# Error, NOT Ok, Value must be on right side of = sign.
To define variable called 'vech' having value Bus
$ vech=Bus
To define variable called n having value 10
$ n=10

Rules for Naming variable name (Both UDV and System Variable)

(1) Variable name must begin with Alphanumeric character or underscore character (_), followed by one or more Alphanumeric character. For e.g. Valid shell variable are as follows

HOME
SYSTEM_VERSION
vech
no

(2) Don't put spaces on either side of the equal sign when assigning value to variable. For e.g. In following variable declaration there will be no error
$ no=10
But there will be problem for any of the following variable declaration:
$ no =10
$ no= 10
$ no = 10

(3) Variables are case-sensitive, just like filename in Linux. For e.g.
$ no=10
$ No=11
$ NO=20
$ nO=2
Above all are different variable name, so to print value 20 we have to use $ echo $NO and not any of the following
$ echo $no # will print 10 but not 20
$ echo $No# will print 11 but not 20
$ echo $nO# will print 2 but not 20

(4) You can define NULL variable as follows (NULL variable is variable which has no value at the time of definition) For e.g.
$ vech=
$ vech=""
Try to print it's value by issuing following command
$ echo $vech
Nothing will be shown because variable has no value i.e. NULL variable.

(5) Do not use ?,* etc, to name your variable names.

How to print or access value of UDV (User defined variables)

To print or access UDV use following syntax
Syntax:
$variablename

Define variable vech and n as follows:
$ vech=Bus

$ n=10
To print contains of variable 'vech' type
$ echo $vech
It will print 'Bus',To print contains of variable 'n' type command as follows
$ echo $n

Caution: Do not try $ echo vech, as it will print vech instead its value 'Bus' and $ echo n, as it will print n instead its value '10', You must use $ followed by variable name.

echo Command

Use echo command to display text or value of variable.

echo [options] [string, variables...]
Displays text or variables value on screen.
Options
-n Do not output the trailing new line.
-e Enable interpretation of the following backslash escaped characters in the strings:
\a alert (bell)
\b backspace
\c suppress trailing new line
\n new line
\r carriage return
\t horizontal tab
\\ backslash

For e.g. **$ echo -e "An apple a day keeps away \a\t\tdoctor\n**

Activities:

Q.1.How to Define variable x with value 10 and print it on screen.

Q.2.How to Define variable xn with value Rani and print it on screen

Q.3.Write a shell script to display your name(stored in UDV), name of the shell you are using, type of the operating system you are using, today's date and time, current month in calendar. Q.4. Write a shell script to store three student's data (name, rollno and totalmarks) in separate variables and print this data each on new line with proper spacing in between fields(use echo with - e option and escape characters \n,\t etc)

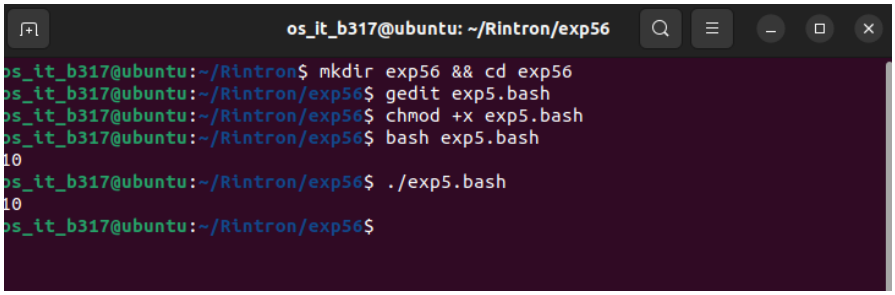
Results: Perform the activity.

The assignment submitted should be e- media saved as <Roll No_Batch No_Date>

Q.1.How to Define variable x with value 10 and print it on screen.

Ans:

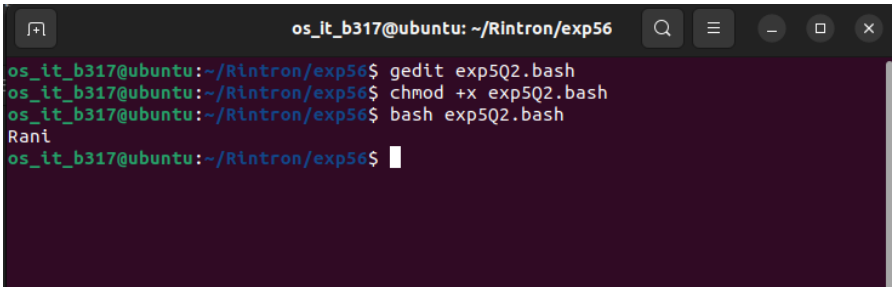
```
os_it_b317@ubuntu:~/Rintron$ mkdir exp56 && cd exp56
os_it_b317@ubuntu:~/Rintron/exp56$ gedit exp5.bash
os_it_b317@ubuntu:~/Rintron/exp56$ chmod +x exp5.bash
os_it_b317@ubuntu:~/Rintron/exp56$ bash exp5.bash
10
os_it_b317@ubuntu:~/Rintron/exp56$ ./exp5.bash
10
os_it_b317@ubuntu:~/Rintron/exp56$
```



Q.2.How to Define variable xn with value Rani and print it on screen

Ans:

```
os_it_b317@ubuntu:~/Rintron/exp56$ gedit exp5Q2.bash
os_it_b317@ubuntu:~/Rintron/exp56$ chmod +x exp5Q2.bash
os_it_b317@ubuntu:~/Rintron/exp56$ bash exp5Q2.bash
Rani
os_it_b317@ubuntu:~/Rintron/exp56$
```



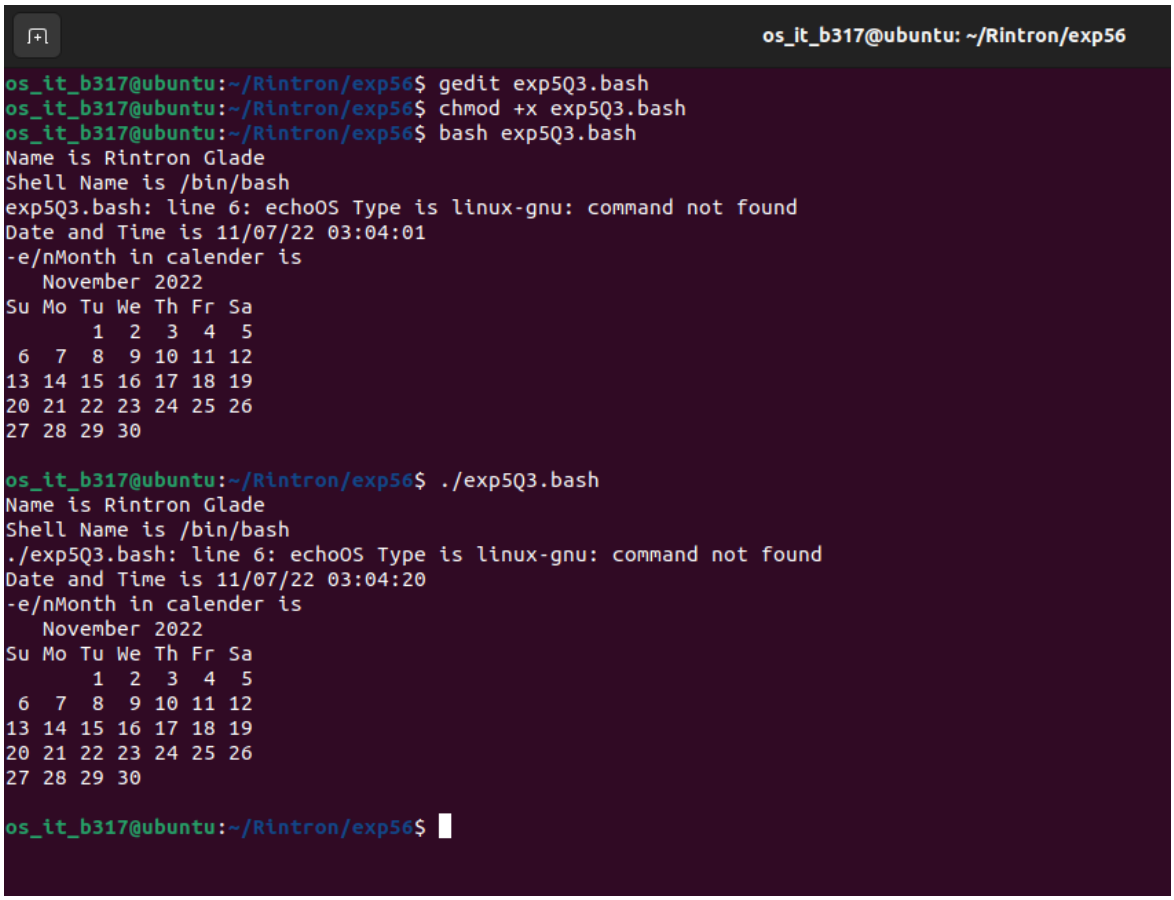
Q.3.Write a shell script to display your name(stored in UDV), name of the shell you are using, type of the operating system you are using, today's date and time, current month in calendar.

Ans:

```
os_it_b317@ubuntu:~/Rintron/exp56$ gedit exp5Q3.bash
os_it_b317@ubuntu:~/Rintron/exp56$ chmod +x exp5Q3.bash
os_it_b317@ubuntu:~/Rintron/exp56$ bash exp5Q3.bash
Name is Rintron Glade
Shell Name is /bin/bash
exp5Q3.bash: line 6: echoOS Type is linux-gnu: command not found
Date and Time is 11/07/22 03:04:01
-e/nMonth in calender is
   November 2022
Su Mo Tu We Th Fr Sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

os_it_b317@ubuntu:~/Rintron/exp56$ ./exp5Q3.bash
Name is Rintron Glade
Shell Name is /bin/bash
./exp5Q3.bash: line 6: echoOS Type is linux-gnu: command not found
Date and Time is 11/07/22 03:04:20
-e/nMonth in calender is
   November 2022
Su Mo Tu We Th Fr Sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

os_it_b317@ubuntu:~/Rintron/exp56$
```


os_it_b317@ubuntu: ~/Rintron/exp56
os_it_b317@ubuntu:~/Rintron/exp56$ gedit exp5Q3.bash
os_it_b317@ubuntu:~/Rintron/exp56$ chmod +x exp5Q3.bash
os_it_b317@ubuntu:~/Rintron/exp56$ bash exp5Q3.bash
Name is Rintron Glade
Shell Name is /bin/bash
exp5Q3.bash: line 6: echoOS Type is linux-gnu: command not found
Date and Time is 11/07/22 03:04:01
-e/nMonth in calender is
November 2022
Su Mo Tu We Th Fr Sa
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
os_it_b317@ubuntu:~/Rintron/exp56$ ./exp5Q3.bash
Name is Rintron Glade
Shell Name is /bin/bash
./exp5Q3.bash: line 6: echoOS Type is linux-gnu: command not found
Date and Time is 11/07/22 03:04:20
-e/nMonth in calender is
November 2022
Su Mo Tu We Th Fr Sa
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
os_it_b317@ubuntu:~/Rintron/exp56$

Q.4. Write a shell script to store three student's data (name, rollno and totalmarks) in separate variables and print this data each on new line with proper spacing in between fields(use echo with - e option and escape characters \n,\t etc)

Ans:

```
os_it_b317@ubuntu:~/Rintron/exp56$ gedit exp5Q4.bash
os_it_b317@ubuntu:~/Rintron/exp56$ chmod +x exp5Q4.bash
os_it_b317@ubuntu:~/Rintron/exp56$ bash exp5Q4.bash
Name                    Roll No         TotalMarks
Rintron Glade           117             30
ThunderBolt OS          100             20
Magneto Maximoff        99              28
os_it_b317@ubuntu:~/Rintron/exp56$ ./exp5Q4.bash
Name                    Roll No         TotalMarks
Rintron Glade           117             30
ThunderBolt OS          100             20
Magneto Maximoff        99              28
os_it_b317@ubuntu:~/Rintron/exp56$
```

Outcomes: CO4: Demonstrate open source standards usage.

Conclusion:

<Students should write about the concluding remarks of the experiment themselves>.
Understood and Implemented successfully static shell scripting

Grade: AA/AB/BB/BC/CC/CD/DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

1. Richard Blum and Christine Bresnahan, “Linux Command Line & Shell Scripting”, II nd Edition edition, Wiley, 2012.