

NAME: VARSHA D KULKARNI

SRN: PES1UG19EC339

OS-LAB-WEEK-3

EXERCISE-1:

Write the following shell script, save it, execute it and note down the output.

```
# Script to print user information who currently login, current date & time
# Enter the following commands in a file
#
clear
echo "Hello $USER"
echo "Today is "; `date`
echo "Number of user login : " ; `who | wc -l`
echo "Calendar"
cal
exit 0
```

At the end, why statement exit 0 is used? What is the meaning of \$?

```
clear
echo "hello $USER"
echo "Today is: "; `date`
echo "number of user login: " ; `who | wc -l`
echo "calender"
cal
exit 0
```

OUTPUT:

```
hello varsha
Today is:
Thu Jun 24 21:55:01 IST 2021
number of user login:
1
calender
      June 2021
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
```

- Here **exit 0** says commands are executed successfully.
- **\$?**- Gives exit status about the last command executed.

```
varsha@ubuntu:~/PES1UG19EC339/os$ echo welcome
welcome
varsha@ubuntu:~/PES1UG19EC339/os$ echo $?
0
```

(Gives 0 if the command is executed without errors, else it gives 1)

EXERCISE-2:

1) If you want to print your home directory location then you give command:

a) echo \$HOME OR

(b) echo HOME Which of the above command is correct & why?

```
varsha@ubuntu:~/PES1UG19EC339/os$ echo $HOME
/home/varsha
```

Here (a) is correct.

Because we want to print the contains of the command HOME not the word HOME. So \$ should be used before a specific command to get the contains of the command.

EXERCISE 3:

What is the output of the following expressions?

\$ expr 1 + 3

\$ expr 2 - 1

\$ expr 10 / 2

\$ expr 20 % 3

\$ expr 10 * 3

\$ echo `expr 6 + 3`

OUTPUT:

```
varsha@ubuntu:~/PES1UG19EC339/os$ expr 1 + 3
4
varsha@ubuntu:~/PES1UG19EC339/os$ expr 2 - 1
1
varsha@ubuntu:~/PES1UG19EC339/os$ expr 10 / 2
5
varsha@ubuntu:~/PES1UG19EC339/os$ expr 20 % 3
2
varsha@ubuntu:~/PES1UG19EC339/os$ expr 10 \* 3
30
varsha@ubuntu:~/PES1UG19EC339/os$ echo `expr 6 + 3`
expr 6 + 3
```

EXERCISE 4:

What is the meaning of Single quote (‘), Double quote (“) and Back quote (`) in shell?

Single Quote:

Enclosing characters in single quotation marks (‘) holds onto the literal value of each character within the quotes. In simpler words, the shell will interpret the enclosed text within single quotes literally and will not interpolate anything including variables, backticks, certain \ escapes, etc.

Double Quote:

Double quotes are similar to single quotes except that it allows the shell to interpret dollar sign (\$), backtick(`), backslash(\) and exclamation mark(!). The characters have special meaning when used with double quotes, and before display, they are evaluated.

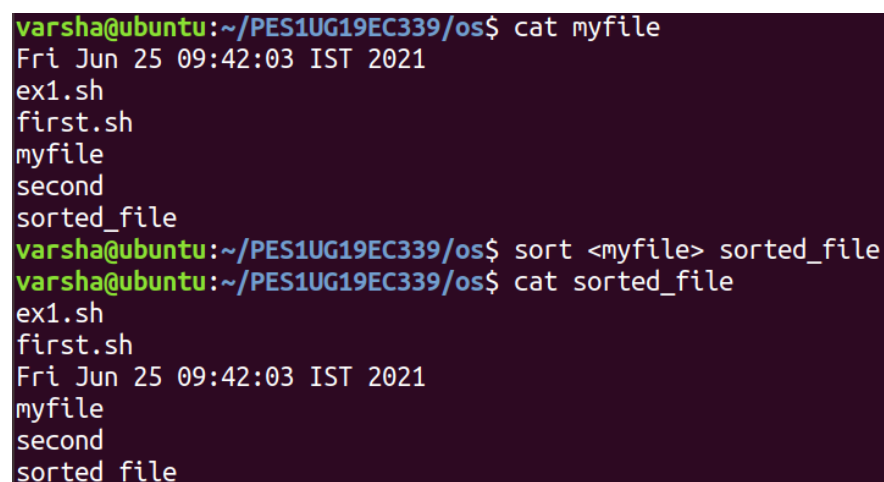
Back Quotes:

The back quote is the one to use when we want to assign output from system commands to variables. It tells the shell to take whatever is between the back quotes as a system command and execute its output.

EXERCISE 5:

What does the following command do?

\$ sort < myfile > sorted_file



```
varsha@ubuntu:~/PES1UG19EC339/os$ cat myfile
Fri Jun 25 09:42:03 IST 2021
ex1.sh
first.sh
myfile
second
sorted_file
varsha@ubuntu:~/PES1UG19EC339/os$ sort <myfile> sorted_file
varsha@ubuntu:~/PES1UG19EC339/os$ cat sorted_file
ex1.sh
first.sh
Fri Jun 25 09:42:03 IST 2021
myfile
second
sorted_file
```

(This command sort the content of the file myfile and the sorted output will get stored in a file sorted_file)

EXERCISE 6:

Create a shell script (using Bourne Shell or Bash) which converts all file names in the current directory to lowercase. You should execute the script and send a screenshot of the output.

Example: If you have these files in the current directory: ABC, foo1.c, Test, sample, foo2.TXT, myFile.Xa Expected output after running the script: abc, foo1.c, test, sample, foo2.txt, myfile.xa

low.sh

```
#!/bin/bash
a=`ls`
echo ${a,,}
```

OUTPUT:

```
varsha@ubuntu:~/PES1UG19EC339/os$ ls
a.out ex1.sh first.sh ifelse.sh low.sh myfile PJC.c PJS.c second SJF.c sorted_file
varsha@ubuntu:~/PES1UG19EC339/os$ ./low.sh
a.out ex1.sh first.sh ifelse.sh low.sh myfile pjc.c pjs.c second sjf.c sorted_file
```