

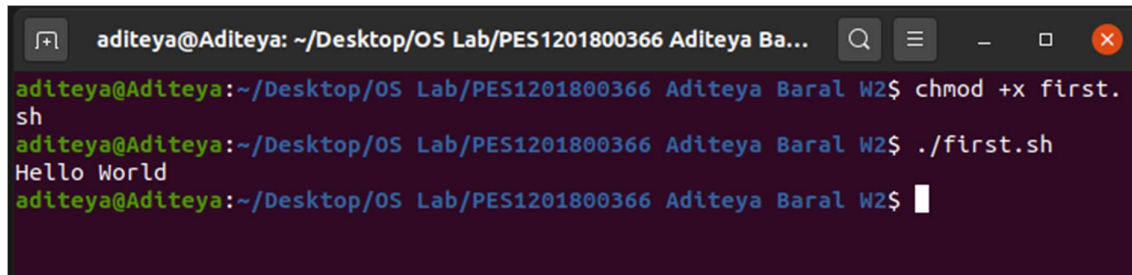
OS Lab Report – Week 2

PES1201800366

Aditeya Baral

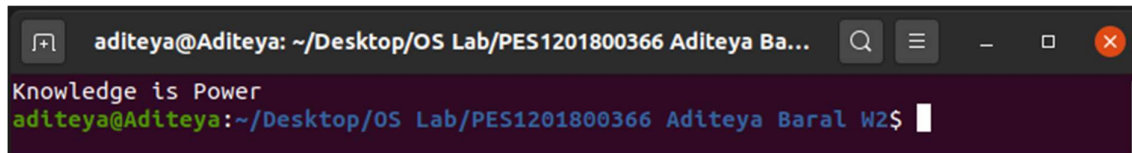
1. First.sh

```
1. #!/bin/sh
2. # This is a comment!
3. echo Hello World      # This is a comment, too!
```

A terminal window titled 'aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Ba...' shows the execution of a script. The user runs 'chmod +x first.sh' and then './first.sh'. The output is 'Hello World'.

2. Second.sh

```
1. #!/bin/sh
2. clear
3. echo "Knowledge is Power"
```

A terminal window titled 'aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Ba...' shows the execution of a script. The user runs './Second.sh' and the output is 'Knowledge is Power'.

3. Exercise 1

```
1. #!/bin/sh
2. # Script to print user information who currently login, current date
  & time
3. # Enter the following commands in a file
4. echo "Hello $USER"
5. echo "Today is ";date
6. echo "Number of user login : " ; who | wc -l
7. echo "Calendar"
8. cal
9. exit 0
```

3.1 Output of Code Snippet

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ chmod +x exercise_1.sh
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./exercise_1.sh
Hello aditeya
Today is
Monday 07 September 2020 11:28:45 AM IST
Number of user login :
1
Calendar
September 2020
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
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

3.2 Reason behind `exit 0`

Every Linux or Unix command executed by the shell script or user has an exit status which is an integer number. `exit 0` status means the command was successful without any errors. A non-zero (1-255 values) exit status means the command was a failure.

3.3 Significance of `$`

In a shell script, the `$` symbol is used to access the value stored in an identifier or variable. If the `$` symbol is not used as a prefix before the variable name, the shell will just display the name of a variable.

4. System Variables

1. `echo $USERNAME`
2. `echo $USER`
3. `echo $HOME`
4. `echo $BASH`

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo $USERNAME
aditeya
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo $USER
aditeya
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo $HOME
/home/aditeya
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo $BASH
/usr/bin/bash
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

5. Exercise 2

```
1. echo $HOME
```

```
1. echo HOME
```

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo $HOME
/home/aditeya
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo HOME
HOME
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

Hence, `echo $HOME` is the right command to display the home directory, since `HOME` is a system variable, and values stored in variables in a shell script can only be accessed using a `$` symbol as a prefix.

6. User Defined Variables

```
1. #!/bin/sh
2. name="Aditeya Baral"
3. srn=PES1201800366
4. age=20
5. echo "Name\t$name"
6. echo "SRN\t$srn"
7. echo "Age\t$age"
```

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Ba...
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ chmod +x third.sh
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./third.sh
Name      Aditeya Baral
SRN       PES1201800366
Age       20
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

7. Shell Arithmetic

```
1. expr 1 - 2
2. expr 1 \* 2
```

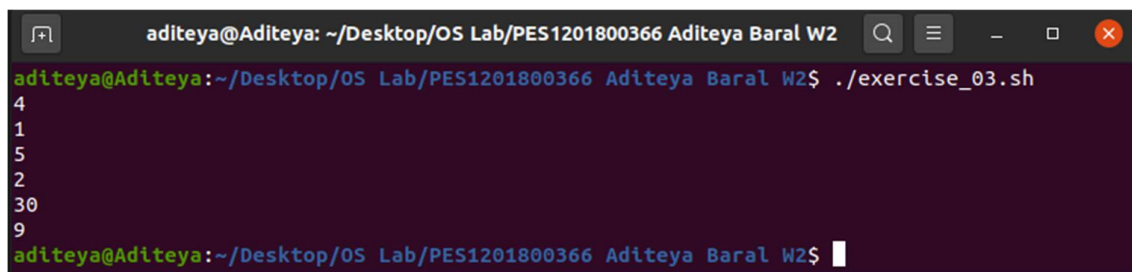
```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ expr 1 - 2
-1
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ expr 1 \* 2
2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ expr 1 + 2
3
```

8. Exercise 3

8.1 Code

```
1. #!/bin/sh
2. expr 1 + 3
3. expr 2 - 1
4. expr 10 / 2
5. expr 20 % 3
6. expr 10 \* 3
7. echo `expr 6 + 3`
```

8.2 Output

A terminal window titled 'aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2'. The prompt is 'aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2\$'. The command './exercise_03.sh' has been executed. The output is displayed on the following lines: 4, 1, 5, 2, 30, 9. The prompt is now 'aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2\$' with a cursor at the end.

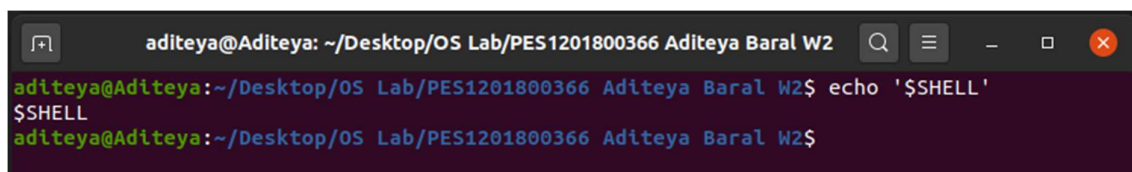
```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./exercise_03.sh
4
1
5
2
30
9
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

9. Exercise 4

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

Answer – The shell understands special characters (such as escape sequences) with special meanings. For example, **\$variable** is used to expand and obtain the value stored in variable. It also expands wildcards (such as * and ?). However, sometimes, we need to display them as is. In such cases, we can use the various quoting methods.

- **Single Quotes (‘):**
 - The single quote is used to remove the special meaning of **any character** enclosed within them i.e. the special characters are treated as ordinary strings
 - Variables, wildcards as well as command substitutions are disabled
 - Example: `echo ‘$SHELL’` will display `$SHELL`

A terminal window titled 'aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2'. The prompt is 'aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2\$'. The command 'echo '\$SHELL'' has been executed. The output is '\$SHELL'. The prompt is now 'aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2\$' with a cursor at the end.

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo '$SHELL'
$SHELL
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

- **Double Quotes (“):**

- The double quote is used to remove the special meaning of **most characters** enclosed within them i.e. most special characters are treated as ordinary strings
- Only wildcards are disabled
- This does not apply to \$, ‘, “, `, \\$, and \
- Example: `echo ‘$SHELL’` will display `/bin/bash`

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo "$SHELL"
/bin/bash
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

- **Back Quotes (`):**

- The back quote is used to **execute any command** enclosed within back quotes
- Example: `echo `expr 6 + 3`` will display 9

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ echo `expr 6 + 3`
9
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

10. Wildcards

1. `#!/bin/sh`
2. `ls *.sh`
3. `ls exercise_0?.sh`
4. `ls [ex]*.sh`

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ls *.sh
exercise_03.sh exercise_1.sh first.sh fourth.sh second.sh thirld.sh
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ls exercise_0?.sh
exercise_03.sh
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ls [ex]*.sh
exercise_03.sh exercise_1.sh
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

11. Redirection of Standard IO

11.1 Output Redirection

1. `ls > filenames.txt`
2. `cat filenames.txt`

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ls > filenames.txt
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ cat filenames.txt
exercise_03.sh
exercise_1.sh
filenames.txt
first.sh
fourth.sh
second.sh
third.sh
WEEK 2 - OS LAB manual for Shell Scripting.docx
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

11.2 Output Redirection with Redirector

```
1. date >> filenames.txt
2. cat filenames.txt
```

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ date >> filenames.txt
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ cat filenames.txt
exercise_03.sh
exercise_1.sh
filenames.txt
first.sh
fourth.sh
second.sh
third.sh
WEEK 2 - OS LAB manual for Shell Scripting.docx
Monday 07 September 2020 08:24:28 PM IST
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

11.3 Input Redirection

```
1. wc -l filenames.txt
```

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ wc -l < filenames.txt
9
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

12. Exercise 5

Question – What does the following command do?

```
1. sort < myfile > sorted_file
```

Answer – The given command will first use input redirection to obtain the contents (lines) of myfile, which is sent to the sort command. After sorting the lines in alphabetical order, the output is redirected to sorted_file and stored.


```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ sort filenames.txt > sorted_filenames.txt
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ cat sorted_filenames.txt
exercise_1.sh
exercise_3.sh
filenames.txt
first.sh
fourth.sh
Monday 07 September 2020 08:35:33 PM IST
second.sh
sorted_filenames.txt
third.sh
WEEK 2 - OS LAB manual for Shell Scripting.docx
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

13. Pipes

1. `ls | more`
2. `who | sort`
3. `who | sort > user_list.txt`
4. `who | wc -l`
5. `ls -l | wc -l`
6. `who | grep aditeya`

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ls | more
exercise_1.sh
exercise_3.sh
filenames.txt
hello_world.sh
knowledge_is_power.sh
op
pipe.sh
sorted_filenames.txt
user_list
user_list.txt
variables.sh
WEEK 2 - OS LAB manual for Shell Scripting.docx
wildcard.sh
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ who | sort
aditeya  :0                2020-09-07 19:24 (:0)
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ who | sort > user_list
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ who | wc -l
1
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ls -l | wc -l
14
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ who | grep aditeya
aditeya  :0                2020-09-07 19:24 (:0)
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

14. if-else-fi Construct

14.1 Code

1. `#!/bin/sh`

```

2. osch=0
3.
4. echo "1. Unix (Sun OS)"
5. echo "2. Linux (Red Hat)"
6. echo -n "Select your os choice [1 or 2]? "
7. read osch
8.
9. if [ $osch -eq 1 ]
10. then
11.     echo "You selected Unix (Sun OS)"
12.
13. else
14.     if [ $osch -eq 2 ]
15.     then
16.         echo "You selected Linux (Red Hat)"
17.     else
18.         echo "You don't like Unix/Linux OS."
19.     fi
20. fi

```

14.2 Output

```

aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./ifleese.sh
1. Unix (Sun OS)
2. Linux (Red Hat)
Select your os choice [1 or 2]? 1
You selected Unix (Sun OS)
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./ifleese.sh
1. Unix (Sun OS)
2. Linux (Red Hat)
Select your os choice [1 or 2]? 2
You selected Linux (Red Hat)
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./ifleese.sh
1. Unix (Sun OS)
2. Linux (Red Hat)
Select your os choice [1 or 2]? 3
You don't like Unix/Linux OS.
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$

```

15. Loops

15.1 For Loop

```

1. #!/bin/sh
2. for ((i = 0;i <= 4;i++))
3. do
4.     echo "Welcome $i times"
5. done

```



```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Ba...
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./for.sh
Welcome 0 times
Welcome 1 times
Welcome 2 times
Welcome 3 times
Welcome 4 times
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

15.2 While Loop

```
1. #!/bin/sh
2. a=0
3. while [ $a -lt 10 ]
4. do
5.     echo $a
6.     a=`expr $a + 1`
7. done
```

```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Ba...
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./while.sh
0
1
2
3
4
5
6
7
8
9
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$
```

15.3 Until Loop

```
1. #!/bin/sh
2. a=0
3. until [ ! $a -lt 10 ]
4. do
5.     echo $a
6.     a=`expr $a + 1`
7. done
```

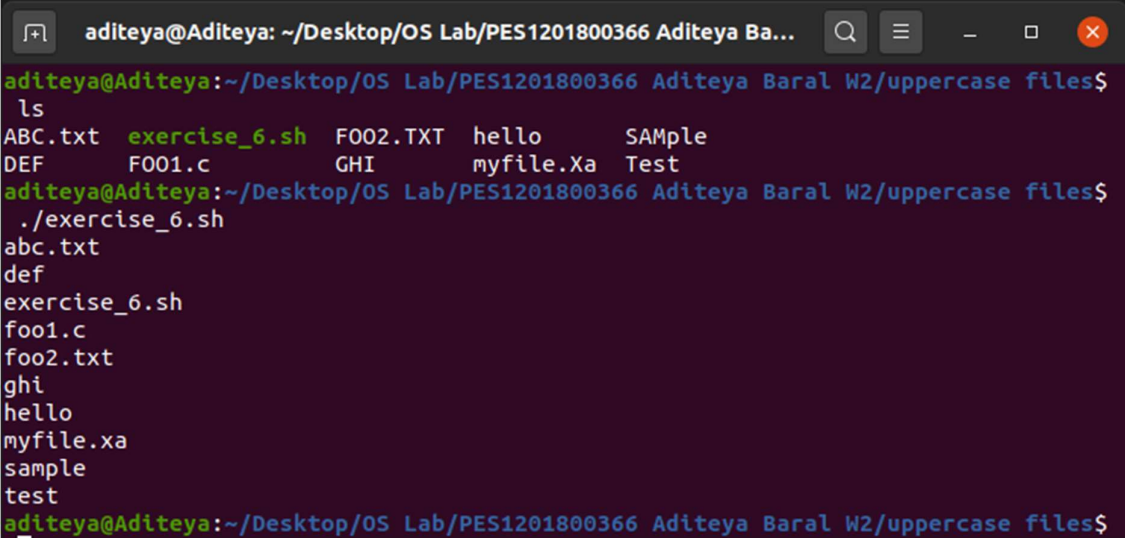
```
aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Ba...
aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2$ ./until.sh
0
1
2
3
4
5
6
7
8
9
```

16. Exercise 6

16.1 Code

```
1. #!/bin/sh
2. ls | tr '[:upper:]' '[:lower:]'
```

16.2 Output

A terminal window titled 'aditeya@Aditeya: ~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2/uppercase files\$'. The prompt is 'aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2/uppercase files\$'. The user enters 'ls', and the output is: 'ABC.txt exercise_6.sh F002.TXT hello Sample', 'DEF F001.c GHI myfile.Xa Test'. The user then enters './exercise_6.sh', and the output is: 'abc.txt', 'def', 'exercise_6.sh', 'foo1.c', 'foo2.txt', 'ghi', 'hello', 'myfile.xa', 'sample', 'test'. The prompt returns to 'aditeya@Aditeya:~/Desktop/OS Lab/PES1201800366 Aditeya Baral W2/uppercase files\$'.