HARSH KASHYAP
CSE 4
101917088
hkashyap_be19@thapar.edu

A Practical activity Report submitted
for Practical Computing(UCS311)

PRACTICAL COMPUTING

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering
Patiala Campus
2020

Submitted to
Aashima Sharma

Assignment 3

Question 1

How can we pass arguments to a script in Linux? And how to access these arguments from within the script?

Solution -

Arguments can be passed to the script when it is executed, by writing them as a space-delimited list following the script file name. Inside the script, the $1 variable references the first argument in the command line, $2 the second argument and so forth. The variable $0 references to the current script.

```
harsh@harsh-VirtualBox:~/PCThapar$ cat > PC
#!/bin/bash
echo "First argument : $1"
echo "Second argument : $2"
```

To access the elements, write the argument while you are going to run the script.

```
harsh@harsh-VirtualBox:~/PCThapar$ chmod +x PC
harsh@harsh-VirtualBox:~/PCThapar$ ./PC Hello World
First argument : Hello
Second argument : World
```

Question 2

Write a shell script to get the current date, time, username and current working directory.

Solution

```
harsh@harsh-VirtualBox:~/PCThapar$ cat > PC
echo `date`
echo `time`
echo `uname`
echo `pwd`
```

```
harsh@harsh-VirtualBox:~/PCThapar$ ./PC
Tuesday 08 September 2020 03:17:46 PM IST

real    0m0.000s
user    0m0.000s
sys     0m0.000s

Linux
/home/harsh/PCThapar
```

Question 3

How to ask for input in a shell script from the terminal?

Solution

To ask for input , we can prompt user simply and then read the input given by the user.

```
harsh@harsh-VirtualBox:~/PCThapar$ cat >PC
echo "What's your name?"
read fname
echo "Hello, $fname"
```

```
harsh@harsh-VirtualBox:~/PCThapar$ ./PC
What's your name?
Harsh
Hello, Harsh
```

Question 4

How can we perform numeric comparisons in Linux?

Solution -

Performing comparisons is done as follows.
This is one the most common evaluation method i.e. comparing two or more numbers. We will now create a script for doing a numeric comparison, but before we do that we need to know the parameters that are used to compare numerical values .
Below mentioned is the list of parameters used for numeric comparisons

- num1 -eq num2 check if 1st number is equal to 2nd number
- num1 -ge num2 checks if 1st number is greater than or equal to 2nd number
- num1 -gt num2 checks if 1st number is greater than 2nd number
- num1 -le num2 checks if 1st number is less than or equal to 2nd number
- num1 -lt num2 checks if 1st number is less than 2nd number

- num1 -ne num2 checks if 1st number is not equal to 2nd number

```
harsh@harsh-VirtualBox:~/PCThapar$ cat PC
if test $1 -gt 0
then
echo "$1 is positive"
else
echo "$1 is not positive"
fi
harsh@harsh-VirtualBox:~/PCThapar$ ./PC 5
5 is positive
```

Question 5

Q5. What is the syntax for different types of loops available in shell scripting?

Solution -

Looping Statements in Shell Scripting: There are a total of 3 looping statements which can be used in bash programming while statement, for statement, until statement

Their descriptions and syntax are as follows:

while statement:

Here the command is evaluated and based on the result loop will executed, if command raise to false then loop will be terminated

Syntax

```
while command
do
```

```
  Statement to be executed

done
```

for statement.

The for loop operate on lists of items. It repeats a set of commands for every item in a list. Here var is the name of a variable and word1 to wordN are sequences of characters separated by spaces (words). Each time the for loop executes, the value of the variable var is set to the next word in the list of words, word1 to wordN.

Syntax

```
for var in word1 word2 ...wordn

do

  Statement to be executed

done
```

until loop

The until loop is executed as many as times the condition/command evaluates to false. The loop terminates when the condition/command becomes true.

Syntax

```
until command

do
```

```
  Statement to be executed until command is true

done
```

Question 6

How will you find the sum of all numbers in a file in Linux?

Solution -

Create a text file which contains integers.
Then in other file perform the following operations.

```
harsh@harsh-VirtualBox:~$ cat num.txt
1
2
3
4
5
6
7
harsh@harsh-VirtualBox:~$ cat count
SUM=0
for i in $(cat num.txt)
do
SUM=`expr $SUM + $i`
done
echo "The final value after adding is : $SUM"

harsh@harsh-VirtualBox:~$ ./count
The final value after adding is : 28
```

Question 7

Write a shell script to validate password strength. Here are a few assumptions for the password string.

- **Length – a minimum of 8 characters.**

- **Contain both alphabet and number.**
- **Include both the small and capital case letters.**

Solution -

```
harsh@harsh-VirtualBox:~$ cat pass
echo "Enter your password"
read pass
if [ ${#pass} -lt 8 ]
then
echo "Enter a length of password greater than equal to 8 "
exit 1
fi
echo "$pass" | grep -q [A-Z]
if test $? -eq 0
then
echo "$pass" | grep -q [a-z]
if test $? -eq 0
then
echo "$pass" | grep -q [0-9]
if test $? -eq 0
then
echo "Strong password"
else
echo "Weak password"
echo "No numbers present. Insert numbers"
fi
else
echo -e "Weak password\nNo lowercase alphabets present.\nInsert lowercase alphabet"
fi
else
echo -e "Weak password\nNo uppercase alphabets present.\nInsert uppercase alphabet"
fi
```

```
harsh@harsh-VirtualBox:~$ chmod +x pass
harsh@harsh-VirtualBox:~$ ./pass
Enter your password
harsh
Enter a length of password greater than equal to 8
harsh@harsh-VirtualBox:~$ ./pass
Enter your password
harsh123
Weak password
No uppercase alphabets present.
Insert uppercase alphabet
harsh@harsh-VirtualBox:~$ ./pass
Enter your password
Harsh123
Strong password
harsh@harsh-VirtualBox:~$ ./pass
Enter your password
HARSH!123
Weak password
No lowercase alphabets present.
Insert lowercase alphabet
```

Question 8

How to check if the previous command was run successfully?

Solution -

Whenever a command runs, the return value of the command is stored in a specific bash variable.
Every command run in bash shell returns a value that's stored in the bash variable "$?". To get the value, run this command. $ echo $?
If a command succeeded successfully, the return value will be 0. If the return value is otherwise, then it didn't run as it's supposed to.

```
harsh@harsh-VirtualBox:~/PCThapar$ ls
PC  PC.txt  pyt3
harsh@harsh-VirtualBox:~/PCThapar$ echo $?
0
harsh@harsh-VirtualBox:~/PCThapar$
```

Question 9

How to get the last line from a file using just the terminal?

Solution -

We can use the tail command to see just the last line of the file.
tail [+-[number][lbcr]] [file]
To see last line,
tail -1 [file]

```
harsh@harsh-VirtualBox:~$ cat commands.txt
The commands used in linux are:-
ls
cd
whois
pwd
etc
few more commands are : cd, mkdir, rmdir, rm
harsh@harsh-VirtualBox:~$ tail -1 commands.txt
few more commands are : cd, mkdir, rmdir, rm
```

Question 10

How to get the first line from a file using just the terminal?

Solution -

head command is used to view the first line.
head -n 1 filename.
We put 1 to view the first line.

```
harsh@harsh-VirtualBox:~$ cat commands.txt
The commands used in linux are:-
ls
cd
whois
pwd
etc
few more commands are : cd, mkdir, rmdir, rm
harsh@harsh-VirtualBox:~$ head -n 1 commands.txt
The commands used in linux are:-
```

Question 11

How to print the first array element? How to print all array elements and their respective indexes?

Solution -

```
harsh@harsh-VirtualBox:~$ cat PC
i=0
array=(1 2 3 4 5)
echo "The first element is : ${array[0]}"
echo "Printing all the elemnts :"
while test $i -lt ${#array[*]}
do
echo "$i : ${array[i]}"
i=`expr $i + 1`
done
harsh@harsh-VirtualBox:~$ ./PC
The first element is : 1
Printing all the elemnts :
0 : 1
1 : 2
2 : 3
3 : 4
4 : 5
```

Question 12

Write a Shell Script that adds two numbers if provided as the command Line Argument and
if the two numbers are not entered throws an Error Message.

Solution -

```
harsh@harsh-VirtualBox:~$ cat PC

#!/bin/bash
# Calculate the sum of two integers with pre initialize values
# in a shell script
if test $# -ne 2
then
echo "Enter two integers to be added"
exit 1
fi


echo "The final value after adding $1 and $2 is : `expr $1 + $2`"

harsh@harsh-VirtualBox:~$ ./PC 1 3
The final value after adding 1 and 3 is : 4
harsh@harsh-VirtualBox:~$ ./PC 1
Enter two integers to be added
```

Question 13

For instance, we want to document the marks for a certain course. The total marks are 200 with 100 marks for Quizzes and 100 for assignments. We want to display the sum of assignments and quizzes while making sure the overall count does not exceed 200.

Solution -

```
harsh@harsh-VirtualBox:~$ cat PC

#!/bin/bash
# Calculate marks of assignment and quiz
# in a shell script
echo "Enter marks  obtained for quizzes out of 100"
read quiz

echo "Enter marks  obtained for assignment out of 100"
read ass

total=`expr $quiz + $ass`
if [ $quiz -gt 100 ] || [ $ass -gt 100 ]
then
echo "Invalid. Marks greater than 100"
else
echo -e "\tMarks Table\n1.Quizzes : $quiz \n2.Assignment :$ass "
echo "Total Marks : $total"
fi
```

```
harsh@harsh-VirtualBox:~$ ./PC
Enter marks  obtained for quizzes out of 100
11
Enter marks  obtained for assignment out of 100
33
        Marks Table
1.Quizzes : 11
2.Assignment :33
Total Marks : 44
harsh@harsh-VirtualBox:~$ ./PC
Enter marks  obtained for quizzes out of 100
122
Enter marks  obtained for assignment out of 100
34
Invalid. Marks greater than 100
```

Question 14

Let's take another example of a bank account program in which we want to have three separate outputs for 3 different situations:

- **The balance is less than zero**

- The balance is zero
- The balance is above zero

For instance, in the following program, use the if, elif, else statements to display different outputs in different scenarios

Solution -

```
harsh@harsh-VirtualBox:~$ cat PC

#!/bin/bash
# Calculate bank balance
echo "Enter your bank balance"
read bank

if [ $bank -gt 0 ]
then
echo "Bank balance is greater than 0"
elif [ $bank -lt 0 ]
then
echo "Bank balance is less than 0"
else
echo "Bank balance is 0"
fi
```

```
harsh@harsh-VirtualBox:~$ ./PC
Enter your bank balance
13
Bank balance is greater than 0
harsh@harsh-VirtualBox:~$ ./PC
Enter your bank balance
-98
Bank balance is less than 0
harsh@harsh-VirtualBox:~$ ./PC
Enter your bank balance
0
Bank balance is 0
```