Shell Scripting Assignments Part-01

Sakib

NIT Kurukshetra,Haryana

B.Tech in IT (5th Semester)

ikasakib1@gmail.com

Assignment 1 : Create readfile.sh in which you can read the information of PWD like size, permission, date time etc.

```
#!/bin/bash

echo " Here is the date --> $(date) "
echo
echo " This is your current working directory--> $(pwd) "
echo
echo " These are the files and directories under your current directory."
echo $(ls -lh)
echo
echo "These are the permission of each file and dir."
echo $(ls -lh | awk '{print $1}')
echo
echo "These are the size of each file and dir."
echo $(ls -lh | awk '{print $5}')
~
```

```
sakib@bangbang:~/scripts$ ls -l | grep readfile.sh
-rwxrwxrwx 1 sakib sakib  389 Jun 11 15:20 readfile.sh
sakib@bangbang:~/scripts$ vim readfile.sh
sakib@bangbang:~/scripts$ ./readfile.sh
 Here is the date --> Sun Jun 11 03:21:15 PM +06 2023

 This is your current working directory--> /home/sakib/scripts

 These are the files and directories under your current directory.
total 40K -rwxrwxrwx 1 sakib sakib 414 Jun 11 09:38 assignment3.sh -rwxrwxrwx 1 sakib sakib 565 Jun 11 10:02 assignment4.sh -rwxrwxrwx 1 sakib sakib 435 Jun 1
1 12:15 assignment5.sh -rw-rw-r-- 1 sakib sakib 450 Jun 11 11:30 awk_practice.txt -rwxrwxrwx 1 sakib sakib 376 Jun 9 22:55 Case_statement.sh -rwxrwxrwx 1 saki
b sakib 294 Jun 9 11:08 condtional.sh -rwxrwxrwx 1 sakib sakib 1.6K Jun 10 11:21 devops_ss_class2.sh -rwxrwxrwx 1 sakib sakib 694 Jun 10 12:13 Password_genera
tor.sh -rwxrwxrwx 1 sakib sakib 1.8K Jun 10 11:24 Practie.sh -rwxrwxrwx 1 sakib sakib 389 Jun 11 15:20 readfile.sh

These are the permission of each file and dir.
total -rwxrwxrwx -rwxrwxrwx -rwxrwxrwx -rw-rw-r-- -rwxrwxrwx -rwxrwxrwx -rwxrwxrwx -rwxrwxrwx -rwxrwxrwx -rwxrwxrwx

These are the size of each file and dir.
414 565 435 450 376 294 1.6K 694 1.8K 389
sakib@bangbang:~/scripts$ _
```

Assignment 2: Take an input of name from user and print Have a great day ahead {name}

```
sakib@bangbang:~/scripts$ ls -l | grep assignment3.sh
-rwxrwxrwx 1 sakib sakib    0 Jun 11 09:30 assignment3.sh
sakib@bangbang:~/scripts$ echo "assignment3.sh has been created and it has been given permission to execute"
assignment3.sh has been created and it has been given permission to execute
sakib@bangbang:~/scripts$ vim assignment3.sh _
```

```
#!/bin/bash

# Take an input of name from user and print Have a great day ahead {name}

echo "Enter your name: "
read -r name
echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"
echo "~~~~~~~~~~~~~~~~~~~~          Have a great day ahead {$name}       ~~~~~~~~~~~~~~~~~~~~~~~"
echo "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"
~
~
```

```
sakib@bangbang:~/scripts$ ls -l | grep assignment3.sh
-rwxrwxrwx 1 sakib sakib    0 Jun 11 09:30 assignment3.sh
sakib@bangbang:~/scripts$ echo "assignment3.sh has been created and it has been given permission to execute"
assignment3.sh has been created and it has been given permission to execute
sakib@bangbang:~/scripts$ vim assignment3.sh
sakib@bangbang:~/scripts$ ./assignment3.sh
Enter your name:
Sakib
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~          Have a great day ahead {Sakib}       ~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
sakib@bangbang:~/scripts$ _
```

Assignment 3 : Let's take a scenario of fintech app 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:

Use "if" condition to check if the balance is less than zero. If this condition evaluates to true, display
the message using the echo command: "Balance is less than zero, Please add more funds else you will be charged penalty".
If the above condition does not match, then use "elif" condition to check if the balance is equal to

zero. If it evaluates to true, display the message: Balance is zero, please add funds
If none of the above condition matches, use the “else” condition to display the: Your balance is above zero.

```
sakib@bangbang:~/scripts$ ls -l | grep assignment4.sh
-rw-rw-r-- 1 sakib sakib    0 Jun 11 09:42 assignment4.sh
sakib@bangbang:~/scripts$ sudo chmod 777 assignment4.sh
sudo: unable to resolve host bangbang: Name or service not known
sakib@bangbang:~/scripts$ ls -l | grep assignment4.sh
-rwxrwxrwx 1 sakib sakib    0 Jun 11 09:42 assignment4.sh
sakib@bangbang:~/scripts$ echo "assignment4.sh has been created and it has been given permission to execute"
assignment4.sh has been created and it has been given permission to execute
sakib@bangbang:~/scripts$ vim assignment4.sh _
```

```
#!/bin/bash

# Assignment 4 ------> FinTech: implementing the if,elif,else conditionals.

echo "Welcome to our system"
echo
echo "Enter your current account balance:"
read -r balance

if [ $balance -lt 0 ]; then       # The balance is less than zero
  echo "Balance is less than zero. Please add more funds, or you will be charged a penalty."
elif [ $balance -eq 0 ]; then     # The balance is equal to zero
  echo "Balance is zero. Please add funds."
else                              # The balance is greater than zero
  echo "Your balance is above zero."
fi
```

```
sakib@bangbang:~/scripts$ vim assignment4.sh
sakib@bangbang:~/scripts$ ./assignment4.sh
Welcome to our system

Enter your current account balance:
0
Balance is zero. Please add funds.
sakib@bangbang:~/scripts$ ./assignment4.sh
Welcome to our system

Enter your current account balance:
20
Your balance is above zero.
sakib@bangbang:~/scripts$ ./assignment4.sh
Welcome to our system

Enter your current account balance:
-100
Balance is less than zero. Please add more funds, or you will be charged a penalty.
sakib@bangbang:~/scripts$ _
```

Assignment 4 : Debug and define briefly about the following program :-

#!/bin/bash

Print a message about disk useage.

space_free=$(df -h | awk '{ print $5 }' | sort -n | tail -n 1 | sed 's/%//')

case $space_free in
[1-5]*)echo Plenty of disk space available

[6-7]*)echo There could be a problem in the near future

8*)echo Maybe we should look at clearing out old files

9*)echo We could have a serious problem on our hands soon

*)echo Something is not quite right here;;
esac

My_Observatoin:

In the above mentioned script there is not provided “ “ after echo and moreover there is not ;; after each case statement after correction the script will look like below:

```
#!/bin/bash
# Print a message about disk usage.

space_free=$(df -h | awk 'NR>1 { print $5 }' | sort -n | tail -n 1 | sed 's/%//')

case $space_free in
[1-5]) echo "Plenty of disk space available";;
[6-7]) echo "There could be a problem in the near future";;
8) echo "Maybe we should look at clearing out old files";;
9) echo "We could have a serious problem on our hands soon";;
*) echo "Something is not quite right here";;
esac
```

Output:

```
sakib@bangbang:~/scripts$ vim assignment5.sh
sakib@bangbang:~/scripts$ ./assignment5.sh
Something is not quite right here
sakib@bangbang:~/scripts$ _
```

Step_wise brief explanation:

1)Disk Fragmentation in human readable format:

```
sakib@bangbang:~/scripts$ df -h
Filesystem              Size  Used Avail Use% Mounted on
tmpfs                   771M  2.4M  769M   1% /run
/dev/mapper/data-root   460G  162G  275G  37% /
tmpfs                   3.8G   60M  3.8G   2% /dev/shm
tmpfs                   5.0M     0  5.0M   0% /run/lock
/dev/nvme0n1p2          4.0G  3.0G  1.1G  73% /recovery
/dev/nvme0n1p1         1020M  409M  612M  41% /boot/efi
tmpfs                   771M  196K  771M   1% /run/user/1000
sakib@bangbang:~/scripts$ _
```

2)Printing the 5th field with awk

```
sakib@bangbang:~/scripts$ df -h | awk '{print $5}'
Use%
1%
38%
2%
0%
73%
41%
1%
sakib@bangbang:~/scripts$ _
```

3) NR>1 in awk command will print without Use → field header

```
sakib@bangbang:~/scripts$ df -h | awk 'NR>1 {print $5}'
1%
38%
2%
0%
73%
41%
1%
sakib@bangbang:~/scripts$ _
```

4) sort -n will sort the output in ascending order

```
sakib@bangbang:~/scripts$ df -h | awk 'NR>1 {print $5}' | sort -n
0%
1%
1%
2%
38%
41%
73%
sakib@bangbang:~/scripts$ _
```

5) The tail -n 1 will print only one line from the last of file

```
sakib@bangbang:~/scripts$ df -h | awk 'NR>1 {print $5}' | sort -n | tail -n 1
73%
sakib@bangbang:~/scripts$ _
```

6) sed 's/%//' the % will be replaced with blank—> that means the % will be removed

```
sakib@bangbang:~/scripts$ df -h | awk 'NR>1 {print $5}' | sort -n | tail -n 1 | sed 's/%//'
73
sakib@bangbang:~/scripts$ _
```

df -h | awk 'NR>1 {print $5}' | sort -n | tail -n 1 | sed 's/%//'
—> 73

space_free=$(df -h | awk 'NR>1 { print $5 }' | sort -n | tail -n 1 | sed 's/%//')

```
#!/bin/bash
# Print a message about disk usage.


space_free=$(df -h | awk 'NR>1 { print $5 }' | sort -n | tail -n 1 | sed 's/%//')
echo
echo "space_free = $space_free"
echo

#case $space_free in
#[1-5]) echo "Plenty of disk space available";;
#[6-7]) echo "There could be a problem in the near future";;
#8) echo "Maybe we should look at clearing out old files";;
#9) echo "We could have a serious problem on our hands soon";;
#*) echo "Something is not quite right here";;
#esac
```

Thus the space_free variable will be assigned with value 73.

```
sakib@bangbang:~/scripts$ vim assignment5.sh
sakib@bangbang:~/scripts$ ./assignment5.sh

space_free = 73

sakib@bangbang:~/scripts$ _
```

Hence the output will be *) case statement because the available value does not fall under the other case statement.

```
sakib@bangbang:~/scripts$ vim assignment5.sh
sakib@bangbang:~/scripts$ ./assignment5.sh
Something is not quite right here
sakib@bangbang:~/scripts$ _
```