Date 10/1/23
Page

week 9: Shell Script Programs.

1. Write a shell script that accepts a path name creates all the components in that path name as directories.
For example, if the script is named mpc, then command mpc a/b/c/d should create directories a, a/b, a/b/c, a/.

→ nano pathname.sh.

```
→ #!/bin/bash
echo "enter the pathname"
read p
i=1
j=1
len = `echo $p | wc -c`
while [ $i -le $len ]
do
n = `echo $p | cut -d / -f $j`
namelength = `echo $n | wc -c`
mkdir $n
cd $n
pwd
 j = `expr $j + 1`
 i = `expr $ i + $namelength`
 done.
```

OUTPUT:

```
enter the pathname
g/h
/home/bsmsce/g
/home/bmsce/g/h.
```

g/h
g

2. Write a shell script that accepts two file names as arguments, checks if the permissions for these files are identical and if the permission are identical, output common permission and otherwise output each filename followed by its permissions.

→ name arg perm.sh.

```
→ #! / bin / bash.
if [ $ # -eq 0]
then
echo " arguments not entered sorry try again"
else
ls -l $1 > f1
x = ' cut -c 2-10 f1'
echo $x
ls - l $ 2 > f2
y =' cut -c 2-10 f2'
echo $y
echo " "
if [ $ x = $ y]
then
echo " permission of both files are same"
echo $x
else
echo " permissions are different "
echo $x
echo $y
fi
fi
```

OUTPUT:

```
sh arg perm.sh chap 1 chap 2.
rw-r --r --
rw - r - - r - -
permission of both files are same
rw-r --r--
```

3. Write a script which receivies three numeri arguments depicting the column number of the ls-l, output to be printed in the order we specify

Example

If we call the script with filename.sh 4,2,1, it means that we want to print columns 4,2 and 1.

```
→ name   numargs.sh
a #!/bin /bash.
if [$# -lt 1]
then
echo " minimum of one column should be specified as
        argument"
else
echo " enter the directory'
read dir
cd $dir
for val in $*
do
  echo "ls -l | cut -d " " -f $val`"
done
fi.
```

OUTPUT:

```
enter the directory
test dir

bms ce
4
1
total
-rw-r--r--
```

4. Write a shell script that accepts file name, starting and ending line numbers as arguments and displays all the lines between the given line numbers.

→ name b/w lines. sh.

```
⇒ #!/bin/sh
if [ $ # -ne 3 ]
then
    echo " Not enough parameters"
else
  last line = `wc -l < $ 1`
    if [ $ 2 -lt $ last line -a $ 3 -le $ last line ].
      then
    n line = "`tail +$ 2 $ 1 | head -$ n line `"
    else
      echo " Invalid range specification "
        fi
fi.
```

OUTPUT:

```
sh hello.sh hey 1 4
hi
hello
how
are
```

10/1/23