ASSIGNMENT 1 by: SHRIRAJ PETHE

1.

Current folder has files: `abc.txt`, `def.txt`, `ghi.txt`, `jkl.txt`

You have to move these files to the folder like `abc.txt` in `abc/`

a) Create files in Current directory or any temporary directory
b) Print list of files to move.
c) Segregate basename and extension of a file.
d) Create folder using basename.
e) Move file to newly created folder.
f) Iterate above steps for all files.

```
Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ touch abc.txt def.txt ghi.txt jkl.txt

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ for file in *.txt; do
> fname=`echo "$file" | awk -F. '{print $1}' `
> mkdir -p "$fname"
> mv "$file" "$fname/$file"
> done

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ ls -R
.:
abc/        data.csv  ghi/  linux_chit_sheet.pdf     README.md
access.log  def/      jkl/  linux_problem_sheet.pdf

./abc:
abc.txt

./def:
def.txt

./ghi:
ghi.txt

./jkl:
jkl.txt

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$
```

Explanation:

1. `touch abc.txt def.txt ghi.txt jkl.txt`

This will create files that we want.

2. `for file in *.txt`

This will loop through all the files which end with '.txt' in current directory

3. `fname=`echo "$file" | awk -F. '{print $1}' `

This will first read the file name, then with awk command ONLY pass the file name without extension to fname variable.

4. `mkdir -p "$fname"`

Will create a directory with the value of fname variable. –p means that it won't give any error if the folder / directory exists already.

5. `mv "$file" "$fname/$file"`

Move the txt file into folder with name of that file.

Here file variable has full name with the extension and the fname (fodlername) has name without the extension.

2.

```
Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ ls
abc/        data.csv  ghi/  linux_chit_sheet.pdf     README.md
access.log  def/      jkl/  linux_problem_sheet.pdf

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ cat data.csv
Id EmployeeName JobTitle        BasePay OvertimePay OtherPay TotalPay TotalPayBenefits
1  NATHANIEL    GM              167411  0           400184   567595   567595
2  GARY         CAPTAIN         155966  245131      137811   538909   538909
3  ALBERT       CAPTAIN         212739  106088      16452    335279   335279
4  CHRISTOPHER  MECHANIC        77916   56120       198306   332343   332343
5  PATRICK      DEPUTYCHIEF     134401  9737        182234   326373   326373
6  DAVID        ASSTDEPUTY      118602  8601        189082   316285   316285
7  ALSON        BATTALIONCHIEF  92492   89062       134426   315981   315981
8  DAVID        DEPUTYDIRECTOR  256576  0           51322    307899   307899
10 JOANNE       CHIEF           285262  0           17115    302377   302377
12 PATRICIA     CAPTAIN         99722   87082       110804   297608   297608
13 EDWARD       EXECUTIVE       294580  0           0        294580   294580

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$
```

Tasks:

1.> print EmployeeName and TotalPay who has Basepay greater than 10000
a) Read data file 'data.csv' from command line and extract rows which have BasePay > 10000
b) print only EmployeeName and TotalPay

```
Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ awk -F ' +' '$4 > 10000 {print $2,"  ", $7}' data.csv
EmployeeName    TotalPay
NATHANIEL    567595
GARY    538909
ALBERT    335279
CHRISTOPHER    332343
PATRICK    326373
DAVID    316285
ALSON    315981
DAVID    307899
JOANNE    302377
PATRICIA    297608
EDWARD    294580

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$
```

Explanation:

a. `awk -F ' +' '$4 > 10000`

$4 > 10000 filters rows where 4th column is greater than 10000

b. `{print $2,"  ", $7}' data.csv`

print only EmployeeName and TotalPay

2.> What is the aggregate TotalPay Of employees whose jobtitle is 'CAPTAIN'
a) Read data file 'data.csv' from command line and extract rows which have 'CAPTAIN' in the column 'jobtitle'
b) Extract TotalPay and calculate sum. Print the result on terminal.

```
Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ cat data.csv
Id EmployeeName JobTitle        BasePay OvertimePay OtherPay TotalPay TotalPayBenefits
1  NATHANIEL    GM              167411  0           400184   567595   567595
2  GARY         CAPTAIN         155966  245131      137811   538909   538909
3  ALBERT       CAPTAIN         212739  106088      16452    335279   335279
4  CHRISTOPHER  MECHANIC        77916   56120       198306   332343   332343
5  PATRICK      DEPUTYCHIEF     134401  9737        182234   326373   326373
6  DAVID        ASSTDEPUTY      118602  8601        189082   316285   316285
7  ALSON        BATTALIONCHIEF  92492   89062       134426   315981   315981
8  DAVID        DEPUTYDIRECTOR  256576  0           51322    307899   307899
10 JOANNE       CHIEF           285262  0           17115    302377   302377
12 PATRICIA     CAPTAIN         99722   87082       110804   297608   297608
13 EDWARD       EXECUTIVE       294580  0           0        294580   294580

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ awk -F ' +' '$3 == "CAPTAIN" {sum += $6} END {print sum}' data.csv
265067
```

Explanation:

a. `$3 == "CAPTAIN"`

This searches the 3rd column for value “CAPTAIN”

b. `{sum += $6} END {print sum}`

Adds the value of 6th column. And AT THE END prints the total.

3.> Print JobTitle and Overtimepay who has Overtimepay is between 7000 and 10000
a) Read data file 'data.csv' from command line and extract jobtitle and overtimepay for column value range between 7000-10000
b) print the result on terminal.

```
Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ cat data.csv
Id EmployeeName JobTitle        BasePay OvertimePay OtherPay TotalPay TotalPayBenefits
1  NATHANIEL    GM              167411  0           400184   567595   567595
2  GARY         CAPTAIN         155966  245131      137811   538909   538909
3  ALBERT       CAPTAIN         212739  106088      16452    335279   335279
4  CHRISTOPHER  MECHANIC        77916   56120       198306   332343   332343
5  PATRICK      DEPUTYCHIEF     134401  9737        182234   326373   326373
6  DAVID        ASSTDEPUTY      118602  8601        189082   316285   316285
7  ALSON        BATTALIONCHIEF  92492   89062       134426   315981   315981
8  DAVID        DEPUTYDIRECTOR  256576  0           51322    307899   307899
10 JOANNE       CHIEF           285262  0           17115    302377   302377
12 PATRICIA     CAPTAIN         99722   87082       110804   297608   297608
13 EDWARD       EXECUTIVE       294580  0           0        294580   294580

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$  awk -F ' +' '$5 >= 7000 && $5 <= 10000 {print $3, $5}' data.csv
DEPUTYCHIEF 9737
ASSTDEPUTY 8601
```

Explanation:

a. `$5 >= 7000 && $5 <= 10000`

Filters the rows where the 5th column value is greater or equal to 7000 AND less or equal to 10000.

b. `{print $3, $5}`

Prints 3rd, 5th column value which is JobTitle, Overtimepay.

4.> Print average BasePay

a)Read data file 'data.csv' from command line and extract BasePay values and calculate its average

b)Print the result on terminal.

```
Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ cat data.csv
Id EmployeeName JobTitle        BasePay OvertimePay OtherPay TotalPay TotalPayBenefits
1  NATHANIEL    GM              167411  0           400184   567595   567595
2  GARY         CAPTAIN         155966  245131      137811   538909   538909
3  ALBERT       CAPTAIN         212739  106088      16452    335279   335279
4  CHRISTOPHER  MECHANIC        77916   56120       198306   332343   332343
5  PATRICK      DEPUTYCHIEF     134401  9737        182234   326373   326373
6  DAVID        ASSTDEPUTY      118602  8601        189082   316285   316285
7  ALSON        BATTALIONCHIEF  92492   89062       134426   315981   315981
8  DAVID        DEPUTYDIRECTOR  256576  0           51322    307899   307899
10 JOANNE       CHIEF           285262  0           17115    302377   302377
12 PATRICIA     CAPTAIN         99722   87082       110804   297608   297608
13 EDWARD       EXECUTIVE       294580  0           0        294580   294580

Shri@PRODUCTIVITY-4 MINGW64 ~/Testing_Bridge/linux-content (master)
$ awk -F ' +' '{sum += $4; count++} END {print sum/count}' data.csv
157972
```

Explanation:

a. `{sum += $4; count++}`

Adds values of 4th column to sum variable, and increases count variable by 1 each time.

b. `END {print sum/count}`

Prints at the end / after passing of all rows, the sum divided by count = average (as we wanted).