Move files from one folder to the respective folders.

In bash

```
$ cd desktop/codingclub/linux-content

$ ls

$ mkdir batch

$ touch abc.txt def.txt ghi.txt

$ mv batch/ destination/ (performing this for three times as we have three files in batch directory)

$ cd destination/

$ ls
```

```
#! /bin/bash

for file in `ls *`;

do

folderName=`echo $file | awk -F. '{print $1}'`;

mkdir $folderName;

mv  $file $foldeName;

# cp  $file $foldeName;


echo moved $file  to $folderName/;

done
```

Check if a folder exists or not. If it's not present, create it

```
#!/bin/bash/

echo -n "enter the directory:"

read dir

if [[ $batch1 == $dir ]]

then

        echo " $dir directory is preset"

else

        mkdir batch1/

        echo "$dir directory is created"

fi
```

Execute command "hello" and "ls" and check its execution status and print whether command executed successful or not.

```
$ cd desktop/codingclub/linux-content

ls

$ echo “hello”

Hello

$ echo “ls”

ls
```

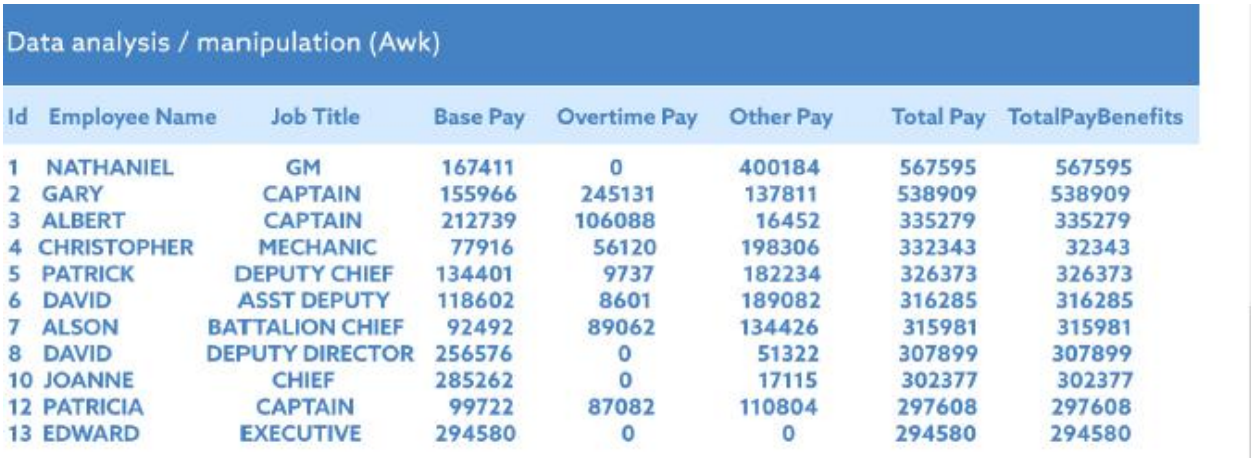
Find the difference between original file and the updated file.
Apply changes to the original file.

```
$ cd desktop/codingclub/linux-content

ls
```

```
mkdir orginal/

cd orginal/

touch orginal-file.sh

cp orginal-file.sh updated/

cd updated/

ls

orginal-file.sh

cd  ..

cd orginal/

ls

orginal-file.sh

notepad orginal-file.sh

echo “this is from from orginal file”

this is from orginal file

cd ..

cd updated/

ls

orginal-file.sh

cat
```

nothing shown in this file

therefore it doesn’t reflect in the updated folder



Data analysis / manipulation (Awk)

Id	Employee Name	Job Title	Base Pay	Overtime Pay	Other Pay	Total Pay	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	32343
5	PATRICK	DEPUTY CHIEF	134401	9737	182234	326373	326373
6	DAVID	ASST DEPUTY	118602	8601	189082	316285	316285
7	ALSON	BATTALION CHIEF	92492	89062	134426	315981	315981
8	DAVID	DEPUTY DIRECTOR	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

i) 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

```
A:) cat data.csv | awk ‘$4>10000’
```

b) Print only Employee Name and TotalPay

```
A:) cat data.csv | awk ‘$4>10000’ | awk ‘{print $2 ” ” $7}’
```

ii) 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'

```
A:) cat data.csv | grep CAPTAIN
```

b) Extract TotalPay and calculate sum. Print the result on terminal.

```
A:) cat data.csv | grep CAPTAIN | awk ‘{sum+=$7} END {print Sum}’
```

iii) 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
betwee*n* 7000-10000

```
A:) cat data.csv | awk ‘$5>7000 && $5<10000’
```

b) Print the result on terminal.

```
A:) cat data.csv | awk ‘$5>7000 && $5<10000’ | awk ‘{print $3 “ ” $5}’
```

iv) 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.
Code is:

```
A:) cat data.csv | awk ’{sum+=$4 } END {print “ ” sum/NR}’
```