Assignment No.06

Name: - Omprakash Khaswhi

****

USE CASE 1: - Check Employee is Present or Absent - Use ((RANDOM)) for Attendance Check

Code: -

echo "Welcome"

ispresent=1

randomcheck=$((RANDOM%2))

if [[$ispresent -eq $randomcheck]]

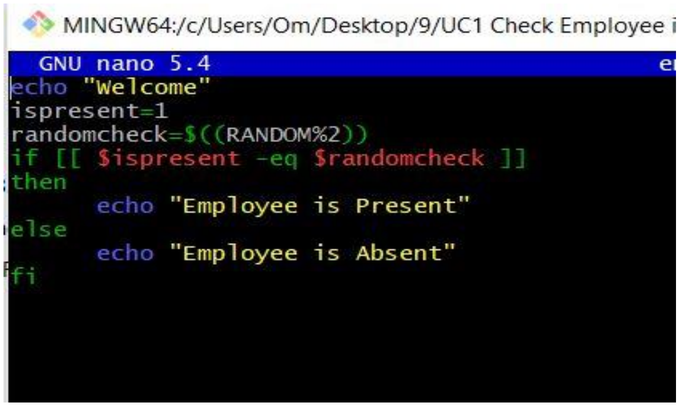
then

echo "Employee is Present"

else

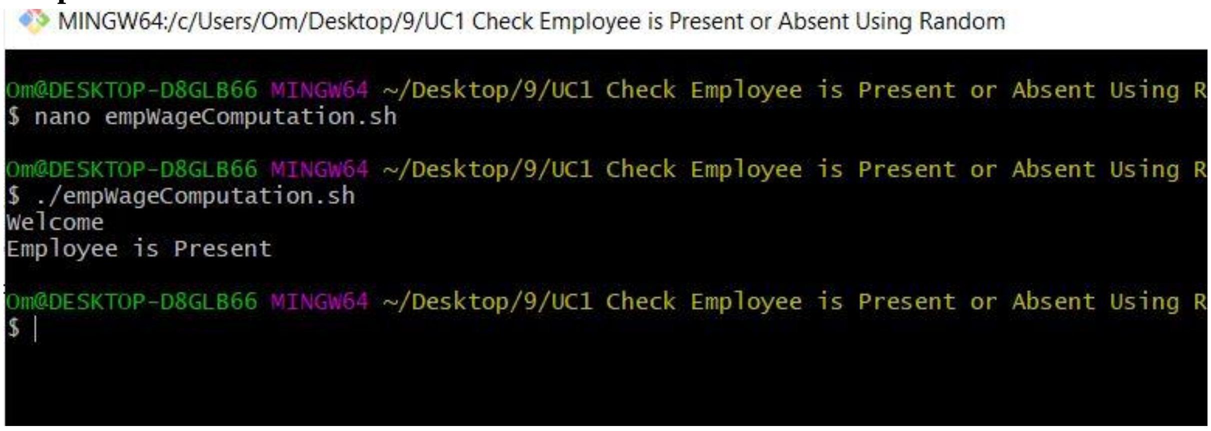
echo "Employee is Absent"

fi



```
MINGW64:/c/Users/Om/Desktop/9/UC1 Check Employee i
  GNU nano 5.4                                    e
echo "Welcome"
ispresent=1
randomcheck=$((RANDOM%2))
if [[ $ispresent -eq $randomcheck ]]
then
        echo "Employee is Present"
else
        echo "Employee is Absent"
fi
```

Output: -

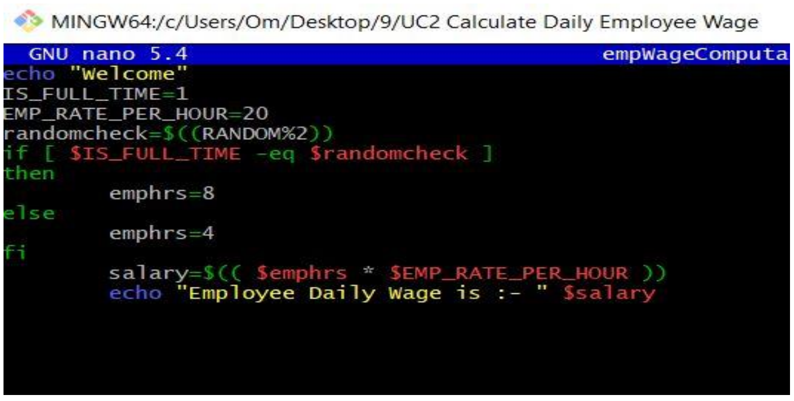


```
MINGW64:/c/Users/Om/Desktop/9/UC1 Check Employee is Present or Absent Using Random

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC1 Check Employee is Present or Absent Using R
$ nano empWageComputation.sh

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC1 Check Employee is Present or Absent Using R
$ ./empWageComputation.sh
Welcome
Employee is Present

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC1 Check Employee is Present or Absent Using R
$ |
```

USE CASE 2: - Calculate Daily Employee Wage - Assume Wage per Hour is 20 - Assume Full Day Hour is 8

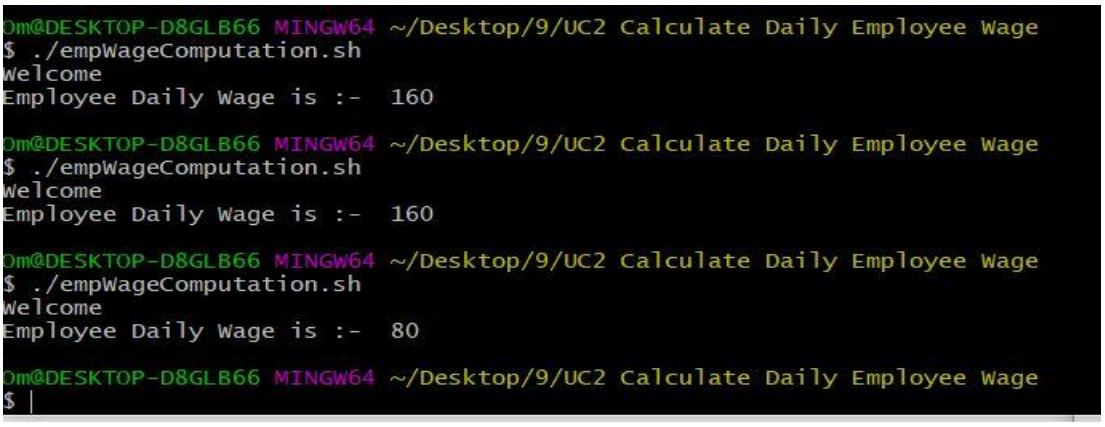
Code: -

```
echo "Welcome"
IS_FULL_TIME=1
EMP_RATE_PER_HOUR=20
randomcheck=$((RANDOM%2))
if [ $IS_FULL_TIME -eq $randomcheck ]
then
emphrs=8
else
emphrs=4
fi
salary=$(( $emphrs * $EMP_RATE_PER_HOUR ))
echo "Employee Daily Wage is :- " $salary
```



```
MINGW64:/c/Users/Om/Desktop/9/UC2 Calculate Daily Employee Wage
  GNU nano 5.4                                    empWageComputa
echo "Welcome"
IS_FULL_TIME=1
EMP_RATE_PER_HOUR=20
randomcheck=$((RANDOM%2))
if [ $IS_FULL_TIME -eq $randomcheck ]
then
        emphrs=8
else
        emphrs=4
fi
        salary=$(( $emphrs * $EMP_RATE_PER_HOUR ))
        echo "Employee Daily Wage is :- " $salary
```

Output: -

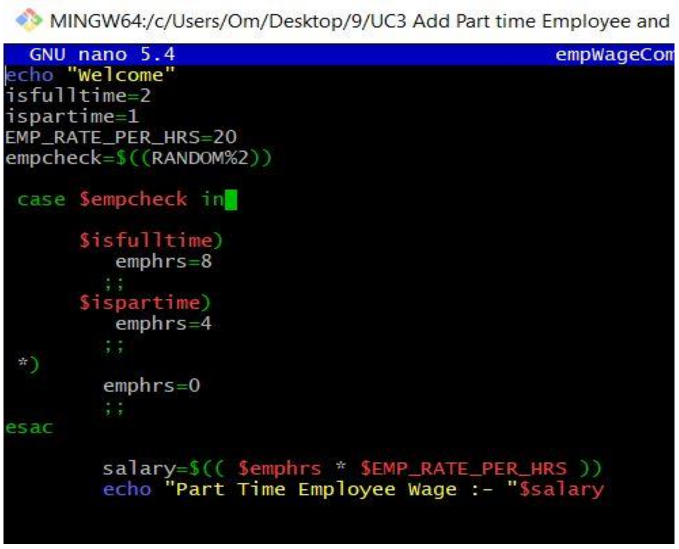


```
Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC2 Calculate Daily Employee Wage
$ ./empWageComputation.sh
Welcome
Employee Daily Wage is :-  160

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC2 Calculate Daily Employee Wage
$ ./empWageComputation.sh
Welcome
Employee Daily Wage is :-  160

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC2 Calculate Daily Employee Wage
$ ./empWageComputation.sh
Welcome
Employee Daily Wage is :-  80

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC2 Calculate Daily Employee Wage
$ |
```

USE CASE 3: - Add Part time Employee & Wage - Assume Part time Hour is 8

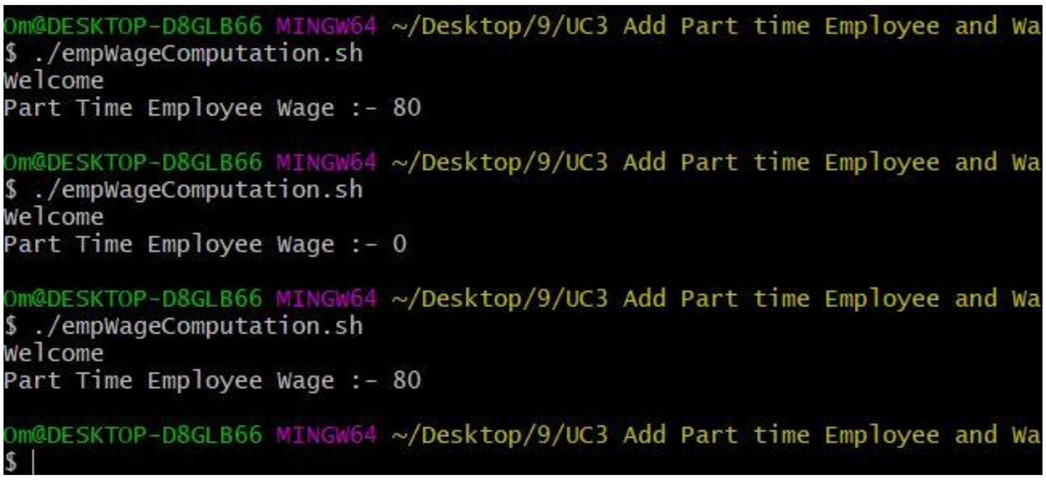
Code: -

```
echo "Welcome"
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
empcheck=$((RANDOM%2))
case $empcheck in
$isfulltime)
emphrs=8
;;
$ispartime)
emphrs=4
;;
*)
emphrs=0
;;
esac
salary=$(( $emphrs * $EMP_RATE_PER_HRS ))
echo "Part Time Employee Wage :- "$salary
```



```
MINGW64:/c/Users/Om/Desktop/9/UC3 Add Part time Employee and
  GNU nano 5.4                                       empWageCom
echo "Welcome"
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
empcheck=$((RANDOM%2))

 case $empcheck in

      $isfulltime)
         emphrs=8
        ;;
      $ispartime)
         emphrs=4
        ;;
 *)
        emphrs=0
        ;;
esac

        salary=$(( $emphrs * $EMP_RATE_PER_HRS ))
        echo "Part Time Employee Wage :- "$salary
```

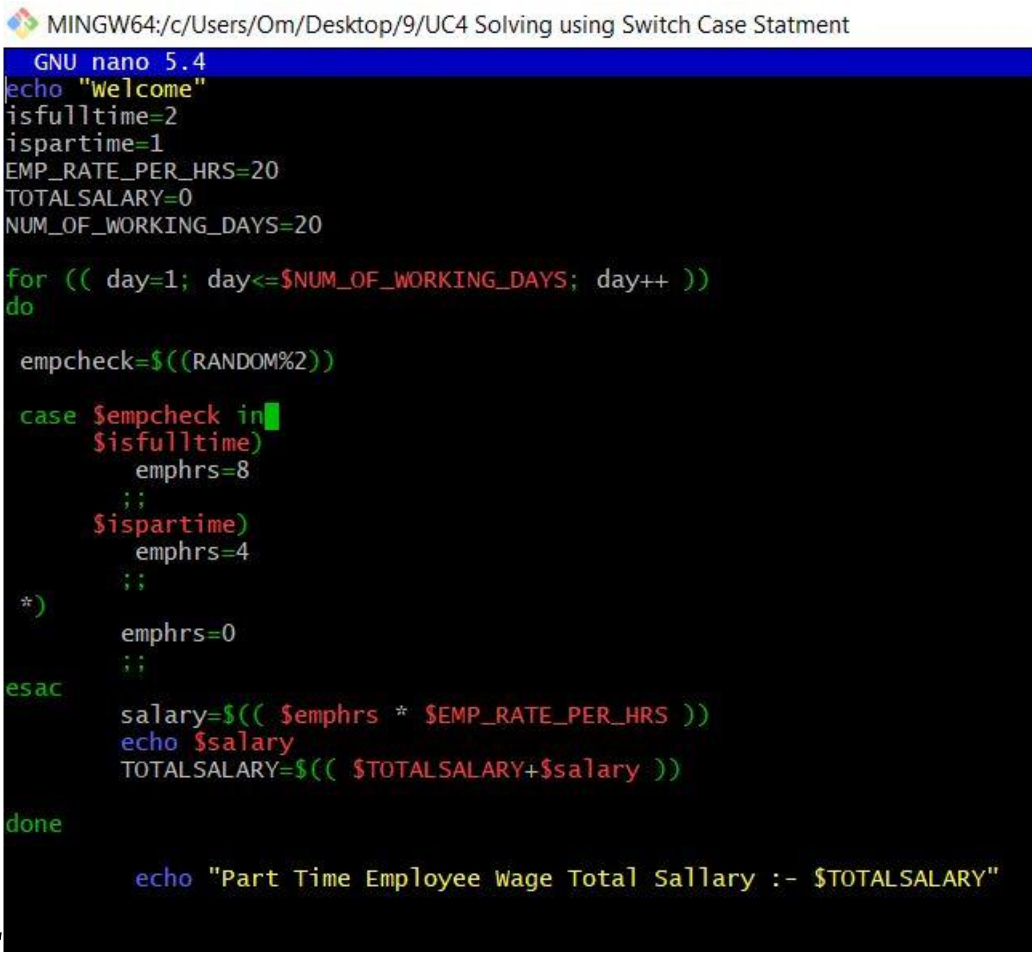
Output: -


```
Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC3 Add Part time Employee and Wa
$ ./empWageComputation.sh
Welcome
Part Time Employee Wage :- 80

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC3 Add Part time Employee and Wa
$ ./empWageComputation.sh
Welcome
Part Time Employee Wage :- 0

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC3 Add Part time Employee and Wa
$ ./empWageComputation.sh
Welcome
Part Time Employee Wage :- 80

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC3 Add Part time Employee and Wa
$ |
```

USE CASE 4: - Solving using Switch Case Statement

Code: -

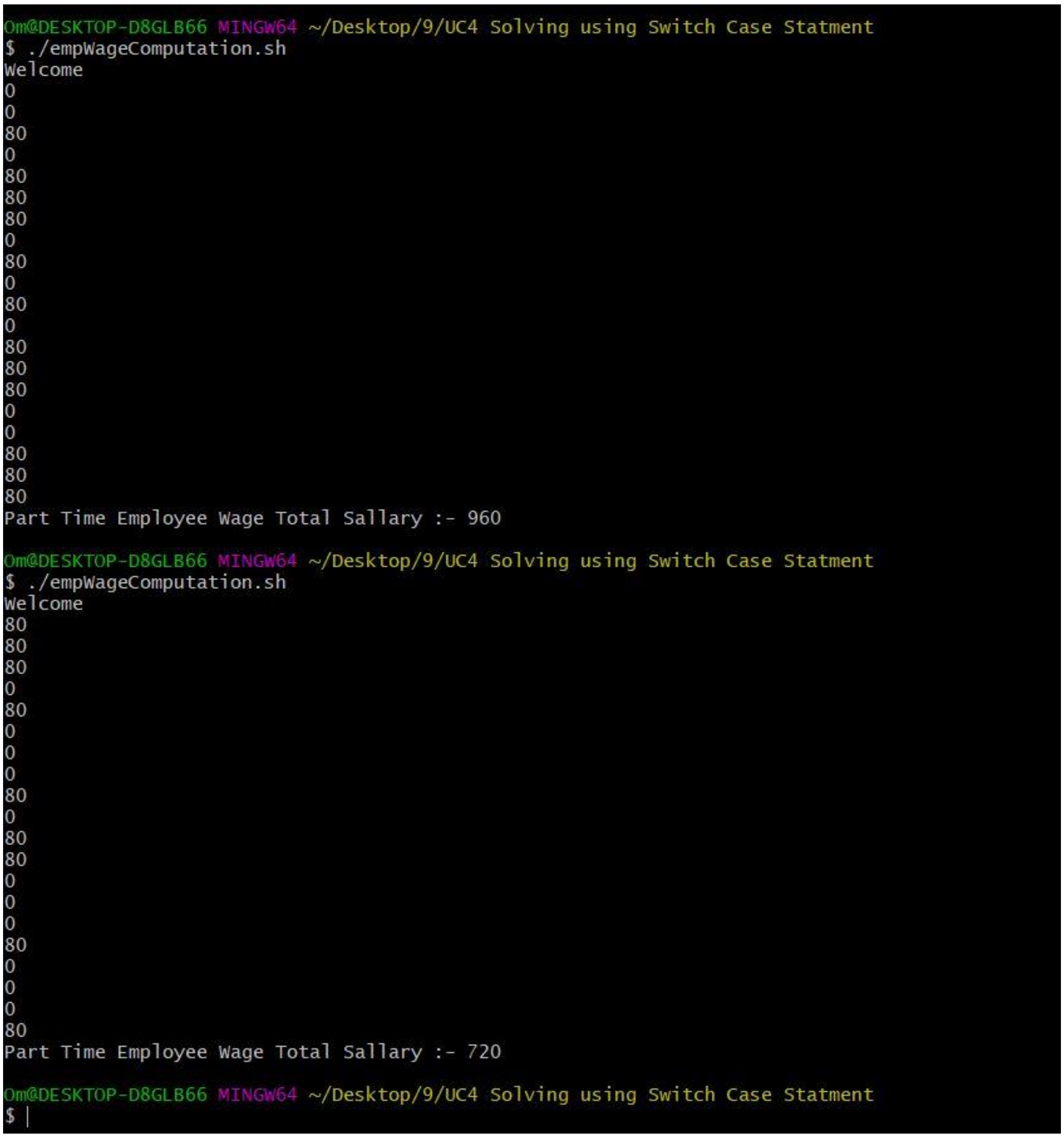
```
echo "Welcome"
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
TOTALSALARY=0
NUM_OF_WORKING_DAYS=20
for (( day=1; day<=$NUM_OF_WORKING_DAYS; day++ ))
do
empcheck=$((RANDOM%2))
case $empcheck in
$isfulltime)
emphrs=8
;;
$ispartime)
emphrs=4
;;
*)
emphrs=0
;;
esac
salary=$(( $emphrs * $EMP_RATE_PER_HRS ))
echo $salary
TOTALSALARY=$(( $TOTALSALARY+$salary ))
Done
echo "Part Time Employee Wage Total Sallary :- $TOTALSALARY”
```



MINGW64:/c/Users/Om/Desktop/9/UC4 Solving using Switch Case Statment

```
  GNU nano 5.4
echo "Welcome"
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
TOTALSALARY=0
NUM_OF_WORKING_DAYS=20

for (( day=1; day<=$NUM_OF_WORKING_DAYS; day++ ))
do

 empcheck=$((RANDOM%2))

 case $empcheck in
      $isfulltime)
         emphrs=8
        ;;
      $ispartime)
         emphrs=4
        ;;
 *)
        emphrs=0
        ;;
esac
        salary=$(( $emphrs * $EMP_RATE_PER_HRS ))
        echo $salary
        TOTALSALARY=$(( $TOTALSALARY+$salary ))

done

         echo "Part Time Employee Wage Total Sallary :- $TOTALSALARY"
```

Output :-

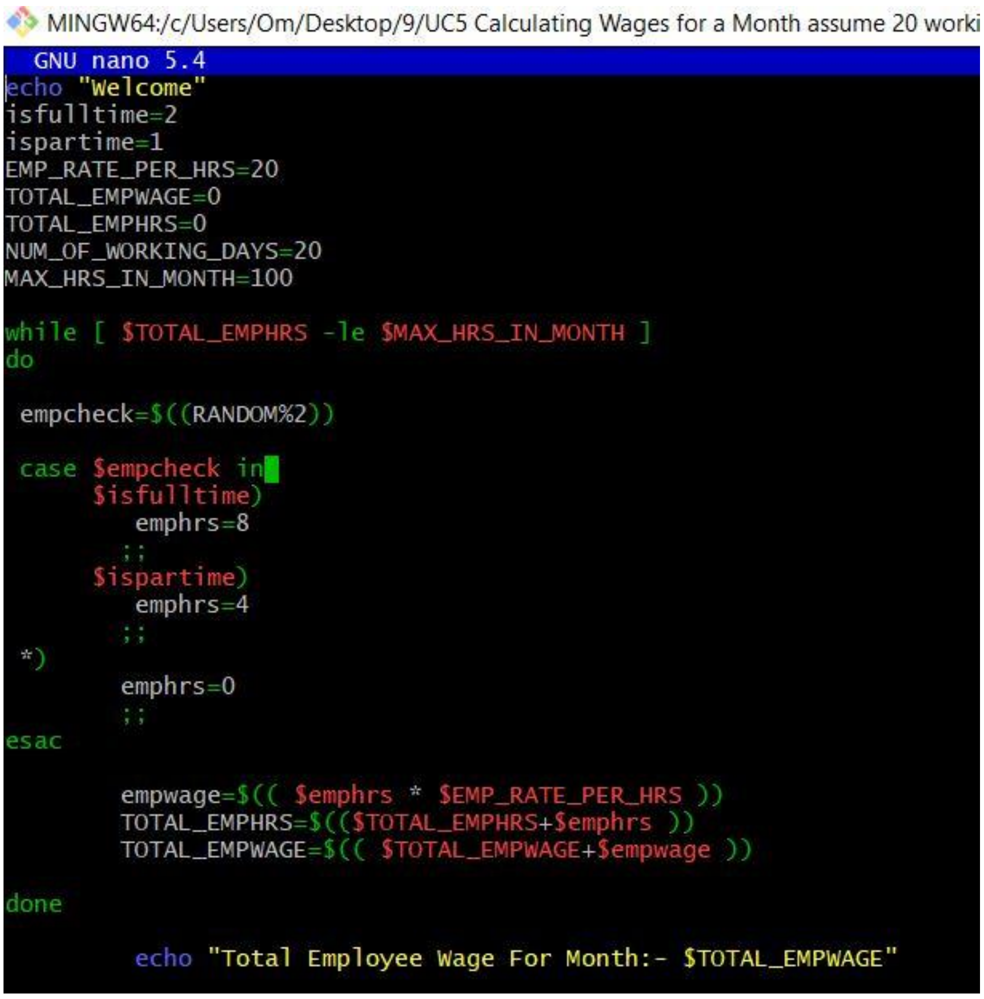


```
Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC4 Solving using Switch Case Statment
$ ./empWageComputation.sh
Welcome
0
0
80
0
80
80
80
0
80
0
80
0
80
80
80
0
0
80
80
80
Part Time Employee Wage Total Sallary :- 960

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC4 Solving using Switch Case Statment
$ ./empWageComputation.sh
Welcome
80
80
80
0
80
0
0
0
80
0
80
80
0
0
0
80
0
0
0
80
Part Time Employee Wage Total Sallary :- 720

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC4 Solving using Switch Case Statment
$ |
```

USE CASE 5: - Calculating Wages for a Month - Assume 20 Working Day per Month

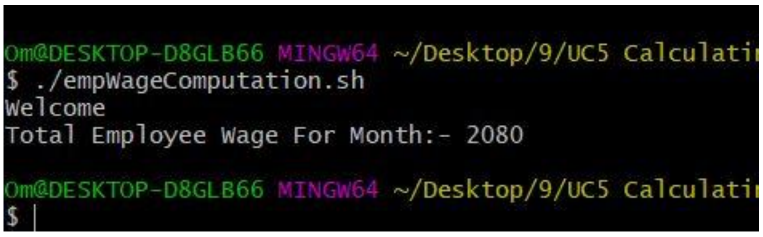
Code: -

```
echo "Welcome"
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
TOTAL_EMPWAGE=0
TOTAL_EMPHRS=0
NUM_OF_WORKING_DAYS=20
MAX_HRS_IN_MONTH=100
while [ $TOTAL_EMPHRS -le $MAX_HRS_IN_MONTH ]
do
empcheck=$((RANDOM%2))
case $empcheck in
$isfulltime)
emphrs=8
;;
$ispartime)
emphrs=4
;;
*)
emphrs=0
;;
esac
empwage=$(( $emphrs * $EMP_RATE_PER_HRS ))
TOTAL_EMPHRS=$(($TOTAL_EMPHRS+$emphrs ))
TOTAL_EMPWAGE=$(( $TOTAL_EMPWAGE+$empwage ))
done
echo "Total Employee Wage For Month:- $TOTAL_EMPWAGE"
```



```
MINGW64:/c/Users/Om/Desktop/9/UC5 Calculating Wages for a Month assume 20 worki
  GNU nano 5.4
echo "Welcome"
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
TOTAL_EMPWAGE=0
TOTAL_EMPHRS=0
NUM_OF_WORKING_DAYS=20
MAX_HRS_IN_MONTH=100

while [ $TOTAL_EMPHRS -le $MAX_HRS_IN_MONTH ]
do

 empcheck=$((RANDOM%2))

 case $empcheck in
      $isfulltime)
         emphrs=8
        ;;
      $ispartime)
         emphrs=4
        ;;
 *)
        emphrs=0
        ;;
esac

        empwage=$(( $emphrs * $EMP_RATE_PER_HRS ))
        TOTAL_EMPHRS=$(($TOTAL_EMPHRS+$emphrs ))
        TOTAL_EMPWAGE=$(( $TOTAL_EMPWAGE+$empwage ))

done

         echo "Total Employee Wage For Month:- $TOTAL_EMPWAGE"
```

Output: -



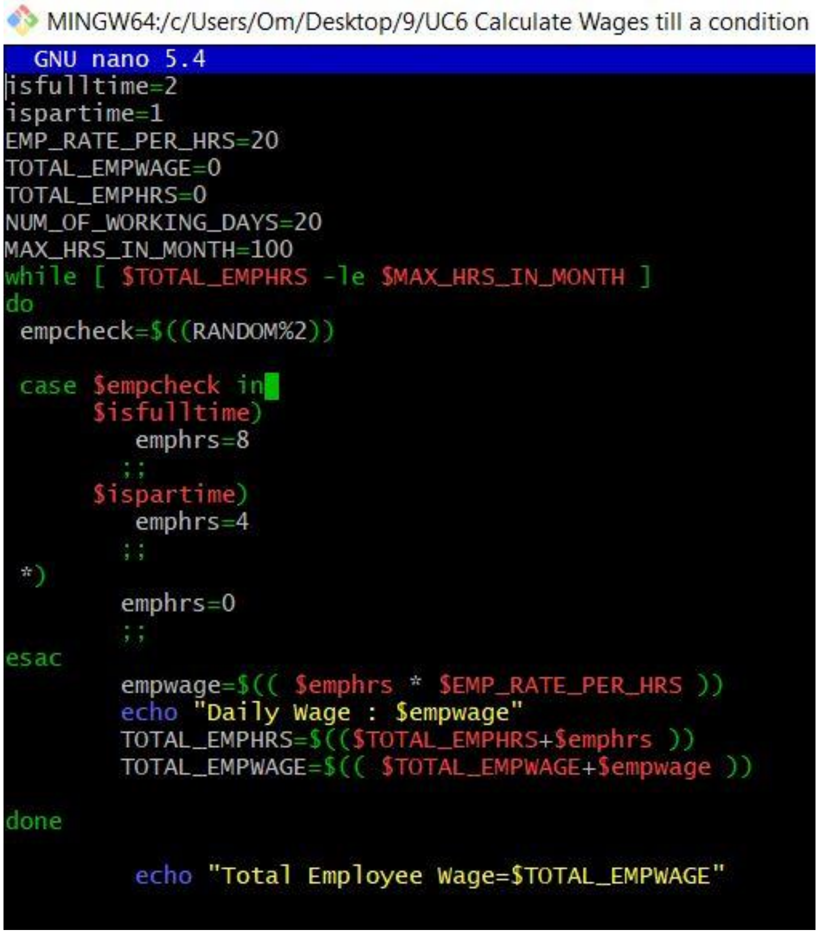
```
Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC5 Calculati
$ ./empWageComputation.sh
Welcome
Total Employee Wage For Month:- 2080

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC5 Calculati
$ |
```

USECASE 6: - Calculate Wages till a condition of total working hours or days is reached for a month - Assume 100 hours and 20 days

Code: -

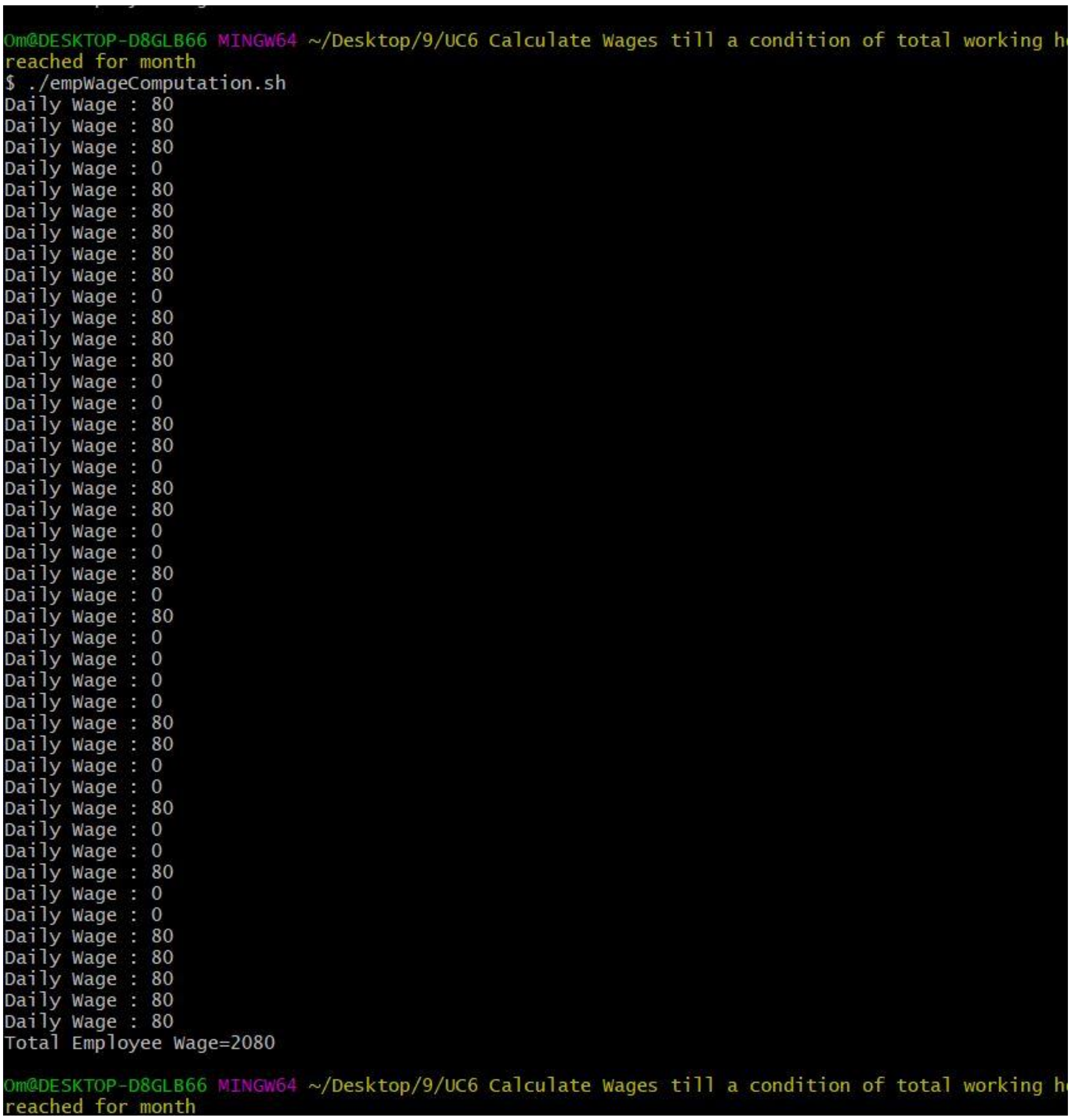
```
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
TOTAL_EMPWAGE=0
TOTAL_EMPHRS=0
NUM_OF_WORKING_DAYS=20
MAX_HRS_IN_MONTH=100
while [ $TOTAL_EMPHRS -le $MAX_HRS_IN_MONTH ]
do
empcheck=$((RANDOM%2))
case $empcheck in
$isfulltime)
emphrs=8
;;
$ispartime)
emphrs=4
;;
*)
emphrs=0
;;
esac
empwage=$(( $emphrs * $EMP_RATE_PER_HRS ))
echo "Daily Wage : $empwage"
TOTAL_EMPHRS=$(($TOTAL_EMPHRS+$emphrs ))
TOTAL_EMPWAGE=$(( $TOTAL_EMPWAGE+$empwage ))
done
```

echo "Total Employee Wage=$TOTAL_EMPWAGE"



```
MINGW64:/c/Users/Om/Desktop/9/UC6 Calculate Wages till a condition
  GNU nano 5.4
isfulltime=2
ispartime=1
EMP_RATE_PER_HRS=20
TOTAL_EMPWAGE=0
TOTAL_EMPHRS=0
NUM_OF_WORKING_DAYS=20
MAX_HRS_IN_MONTH=100
while [ $TOTAL_EMPHRS -le $MAX_HRS_IN_MONTH ]
do
 empcheck=$((RANDOM%2))

 case $empcheck in
      $isfulltime)
         emphrs=8
        ;;
      $ispartime)
         emphrs=4
        ;;
 *)
        emphrs=0
        ;;
esac
        empwage=$(( $emphrs * $EMP_RATE_PER_HRS ))
        echo "Daily Wage : $empwage"
        TOTAL_EMPHRS=$(($TOTAL_EMPHRS+$emphrs ))
        TOTAL_EMPWAGE=$(( $TOTAL_EMPWAGE+$empwage ))

done

         echo "Total Employee Wage=$TOTAL_EMPWAGE"
```

Output:-



```
Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC6 Calculate Wages till a condition of total working h
reached for month
$ ./empWageComputation.sh
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Daily Wage : 0
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Daily Wage : 0
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Daily Wage : 0
Daily Wage : 0
Daily Wage : 80
Daily Wage : 80
Daily Wage : 0
Daily Wage : 80
Daily Wage : 80
Daily Wage : 0
Daily Wage : 0
Daily Wage : 80
Daily Wage : 0
Daily Wage : 80
Daily Wage : 0
Daily Wage : 0
Daily Wage : 0
Daily Wage : 0
Daily Wage : 80
Daily Wage : 80
Daily Wage : 0
Daily Wage : 0
Daily Wage : 80
Daily Wage : 0
Daily Wage : 0
Daily Wage : 80
Daily Wage : 0
Daily Wage : 0
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Daily Wage : 80
Total Employee Wage=2080

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC6 Calculate Wages till a condition of total working h
reached for month
```

USW CASE 7: - Refactor the Code to write a function to get work hours

Code: -

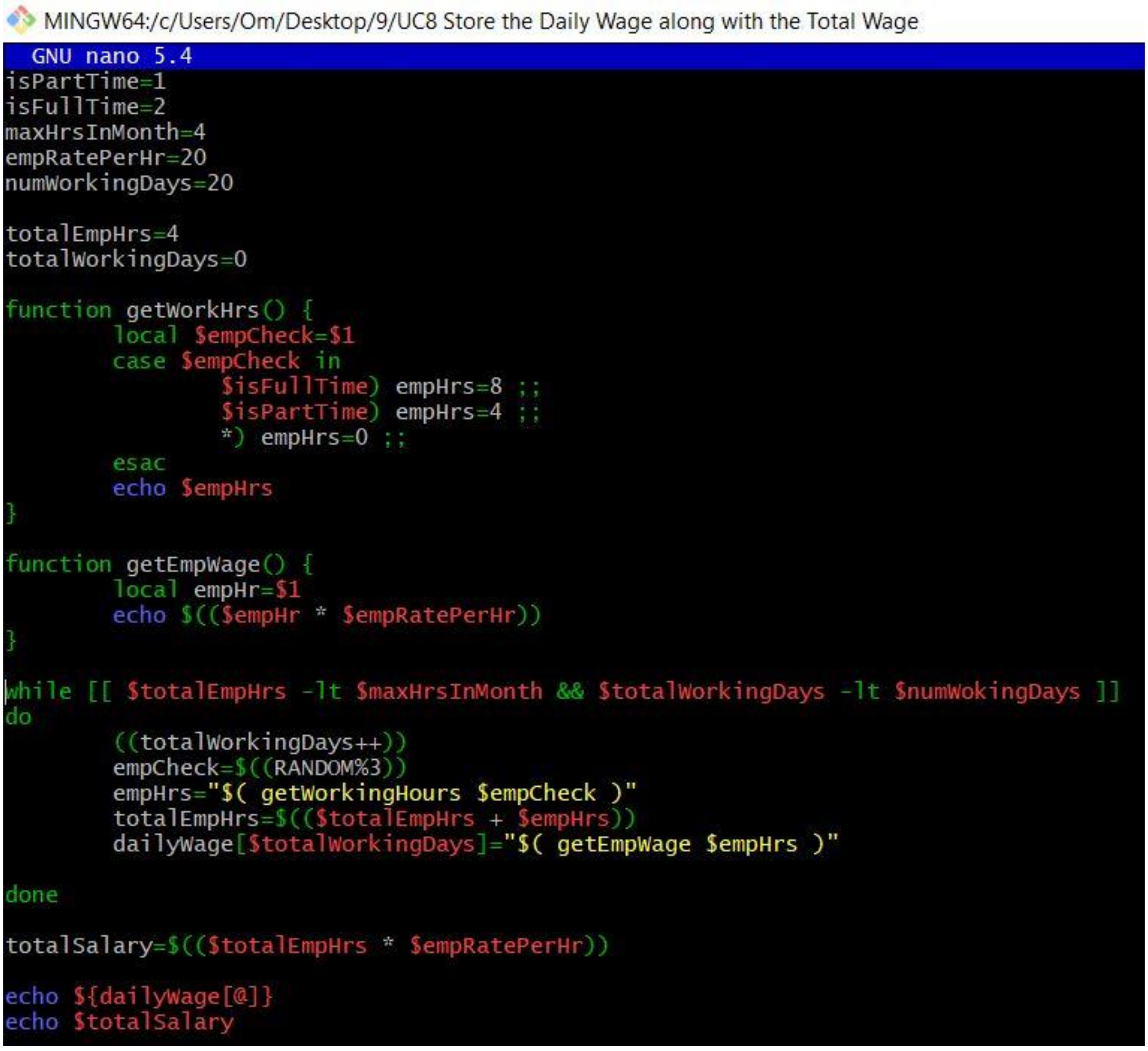
```
#!/bin/bash -x

echo "Welcome"

isPartTime=1

isFullTime=2

maxHrsInMonth=10

empRatePerHr=20

numWorkingDays=20

totalEmpHrs=100

totalWorkingDays=20

function getWorkingHours() {

case $1 in

$isFullTime) workHrs=8 ;;

$isPartTime) workHrs=4 ;;

*) workHrs=0 ;;

esac

echo "Working Hours :" $workHrs

}

while [[ $totalWorkHrs -lt $maxHrsInMonth && $totalWorkingDays -lt $numWokingDays
]]

do

((totalWorkingDays++))

workHrs"$( getWorkingHours $((RANDOM)) )"

totalWorkHrs=$(($totalWorkHrs * $workHrs))

done

totalSalary=$(($totalWorkHrs * $workHrs))

echo "Total Salary:" $totalSalary
```

USE CASE 8: - Store the Daily Wage along with the Total Wage

Code: -

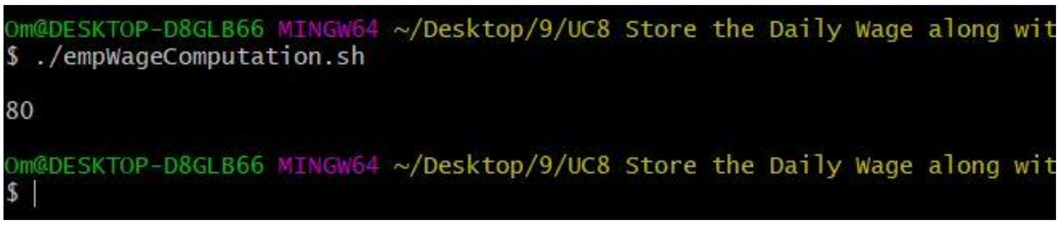
```
isPartTime=1
isFullTime=2
maxHrsInMonth=4
empRatePerHr=20
numWorkingDays=20
totalEmpHrs=4
totalWorkingDays=0
function getWorkHrs() {
local $empCheck=$1
case $empCheck in
$isFullTime) empHrs=8 ;;
$isPartTime) empHrs=4 ;;
*) empHrs=0 ;;
esac
echo $empHrs
}
function getEmpWage() {
local empHr=$1
echo $(($empHr * $empRatePerHr))
}
while [[ $totalEmpHrs -lt $maxHrsInMonth && $totalWorkingDays -lt $numWokingDays ]]
do
((totalWorkingDays++))
empCheck=$((RANDOM%3))
empHrs="$( getWorkingHours $empCheck )"
totalEmpHrs=$(($totalEmpHrs + $empHrs))
dailyWage[$totalWorkingDays]="$( getEmpWage $empHrs )"
```

```
done

totalSalary=$(($totalEmpHrs * $empRatePerHr))

echo ${dailyWage[@]}

echo $totalSalary
```



```
MINGW64:/c/Users/Om/Desktop/9/UC8 Store the Daily Wage along with the Total Wage
  GNU nano 5.4
isPartTime=1
isFullTime=2
maxHrsInMonth=4
empRatePerHr=20
numWorkingDays=20

totalEmpHrs=4
totalWorkingDays=0

function getWorkHrs() {
        local $empCheck=$1
        case $empCheck in
                $isFullTime) empHrs=8 ;;
                $isPartTime) empHrs=4 ;;
                *) empHrs=0 ;;
        esac
        echo $empHrs
}

function getEmpWage() {
        local empHr=$1
        echo $(($empHr * $empRatePerHr))
}

while [[ $totalEmpHrs -lt $maxHrsInMonth && $totalWorkingDays -lt $numWokingDays ]]
do
        ((totalWorkingDays++))
        empCheck=$((RANDOM%3))
        empHrs="$( getWorkingHours $empCheck )"
        totalEmpHrs=$(($totalEmpHrs + $empHrs))
        dailyWage[$totalWorkingDays]="$( getEmpWage $empHrs )"

done

totalSalary=$(($totalEmpHrs * $empRatePerHr))

echo ${dailyWage[@]}
echo $totalSalary
```

Output: -



```
Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC8 Store the Daily Wage along wit
$ ./empWageComputation.sh

80

Om@DESKTOP-D8GLB66 MINGW64 ~/Desktop/9/UC8 Store the Daily Wage along wit
$ |
```

USE CASE 9: - Store the Day and the Daily Wage along with the Total Wage

Code: -

```
isPartTime=1
isFullTime=2
maxHrsInMonth=4
empRatePerHr=20
numWorkingDays=20
totalEmpHrs=0
totalWorkingDays=0
declare -A dailyWage
function getWorkHrs() {
local $empCheck=$1
case $empCheck in
$isFullTime) empHrs=8 ;;
$isPartTime) empHrs=4 ;;
*) empHrs=0 ;;
esac
echo $empHrs
}
function getEmpWage() {
local empHr=$1
echo $(($empHr*$empRatePerHr))
}
while [[ $totalEmpHrs -lt $maxHrsInMonth && $totalWorkingDays -lt $numWokingDays ]]
do
((totalWorkingDays++))
empCheck=$((RANDOM%3))
empHrs="$( getWorkingHours $empCheck )"
totalEmpHrs=$(($totalEmpHrs+$empHrs))
```

```
dailyWage["Day"$totalWorkingDays]="$( getEmpWage $empHrs )"
done
totalSalary=$(($totalEmpHrs*$empRatePerHr))
echo "${dailyWage[@]}"
echo "${!dailyWage[@]}"
```