Sri Lanka Institute of Information Technology

Faculty of Computing

Computer System Engineering Department

IE2050 – Operating System
Assignment 1 (CSNE)

Name	: D.M. Kolitha Kasun Dasanayaka
Index Number	: IT18184068
Group	: Group 06.1 (CSNE)
Submission Date	: 20th Sep 2019

Task 1

1. First need to compile OS_Task_1A.c using,

   gcc -o write OS_Task_1A.c -l pthread

This program will read data from OS.txt file and save it in the local variable and then write in the shared memory.

2. Secondly need to compile OS_Task_1B.c using

   gcc -o read OS_Task_1B.c

This program is written to read data from the Shared Memory and to delete shared memory.

In the OS_Task_1A.c

- It's has 4 threads; 3 threads are used to scan data from the "OS.txt" file and one is used to print details, 1 method to create shared memory and the main method.

I have declared variables:

- **name structure:** used to mapped to the shared memory
- Global variables to store data within the program.
  - **Name** char 2D array
  - **City** char 2D array
  - **Age** integer array
  - **whole** char one-dimension array.
- Shared Memory Variables:
  - **Shm_id:** this variable is used to hold the returned segment identifier.
  - **Key:** key_t type variable, which is hardcoded to use "2331".
  - **shm_addr**
  - **string_num** integer pointer
  - **strings:** object of the name struct
- I have used 3 threads to read "OS.txt" file using fscanf function
  1. readnames
  2. readcities
  3. readages

- after that I created a ran a function "create_sh()" to create a shared memory and put the values I obtained with functions which ran on threads.
- Then I created another thread to print details (values that obtained and inserted to the shared memory)

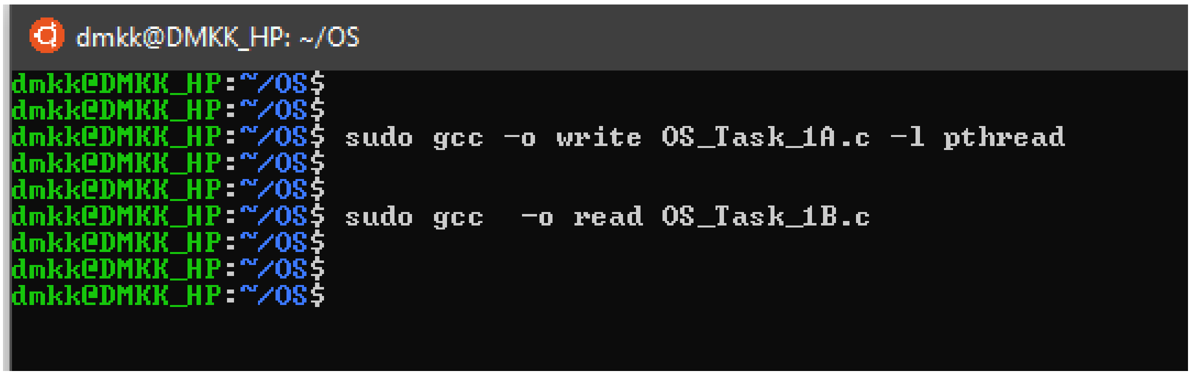
- **readNames() :** function reads names in the file
- **readCities() :** function reads cities in the file
- **readAges() :** function reads ages in the file
- **create_sm() :** function creates a shared memory and inserts the values to it
- **printDetails() :** function prints the values in the shared memory.
- I used **pthread_mutex_lock()** predefined function to secure the resource from other threads and **pthread_mutex_unlock()** function to unlock resources.

<u>In the OS_Task_1B.c</u>

I created the shared memory as in the OS_Task_1B.c and in the main program using printf, I displayed the details in the shared memory.

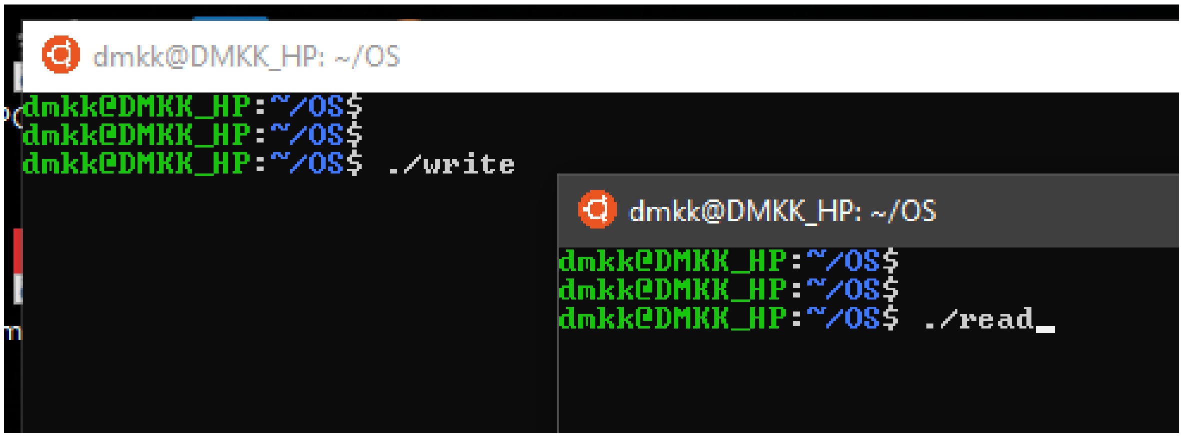
Then asked from the user that he want to delete the shred memory or not.

<u>Screenshots of Task 1</u>



```
dmkk@DMKK_HP: ~/OS
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ sudo gcc -o write OS_Task_1A.c -l pthread
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ sudo gcc  -o read OS_Task_1B.c
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
```

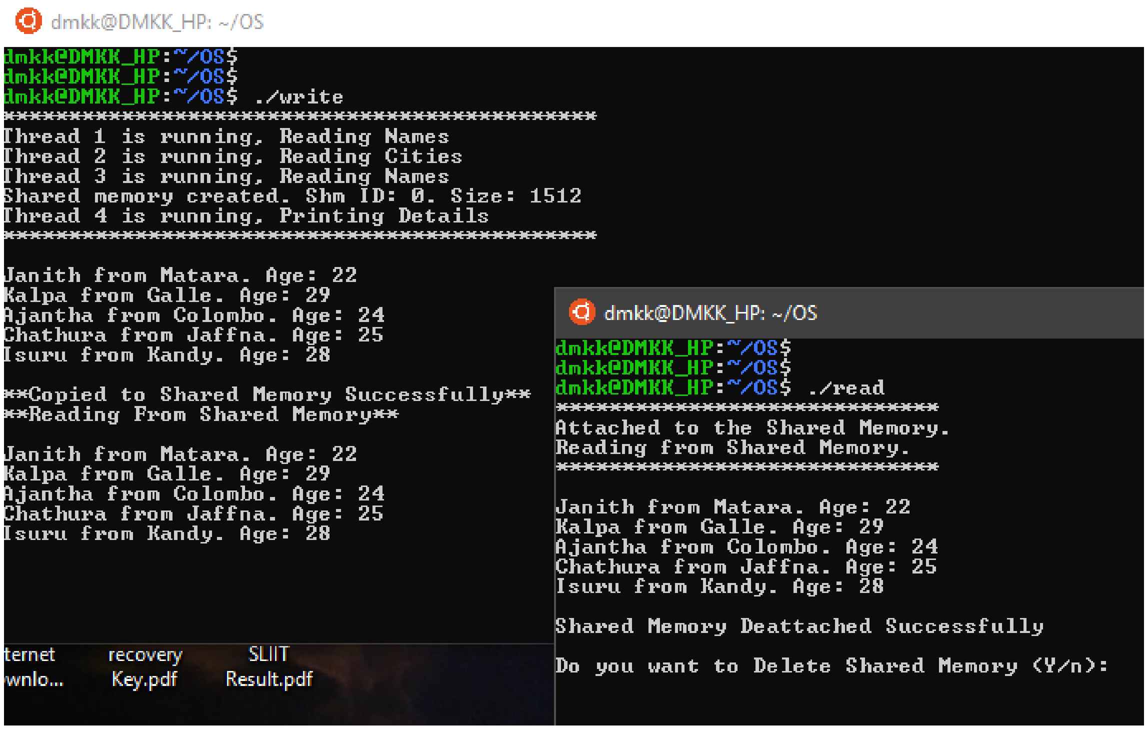
Fig:1 Compiling two c files of Task 1



```
dmkk@DMKK_HP: ~/OS
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ ./write
```

```
dmkk@DMKK_HP: ~/OS
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ ./read_
```

Fig:2 Executing 2 Files (write must execute 1st)



dmkk@DMKK_HP: ~/OS

```
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ ./write
*********************************************
Thread 1 is running, Reading Names
Thread 2 is running, Reading Cities
Thread 3 is running, Reading Names
Shared memory created. Shm ID: 0. Size: 1512
Thread 4 is running, Printing Details
*********************************************

Janith from Matara. Age: 22
Kalpa from Galle. Age: 29
Ajantha from Colombo. Age: 24
Chathura from Jaffna. Age: 25
Isuru from Kandy. Age: 28

**Copied to Shared Memory Successfully**
**Reading From Shared Memory**

Janith from Matara. Age: 22
Kalpa from Galle. Age: 29
Ajantha from Colombo. Age: 24
Chathura from Jaffna. Age: 25
Isuru from Kandy. Age: 28
```

dmkk@DMKK_HP: ~/OS

```
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ ./read
*****************************
Attached to the Shared Memory.
Reading from Shared Memory.
*****************************

Janith from Matara. Age: 22
Kalpa from Galle. Age: 29
Ajantha from Colombo. Age: 24
Chathura from Jaffna. Age: 25
Isuru from Kandy. Age: 28

Shared Memory Deattached Successfully

Do you want to Delete Shared Memory (Y/n):
```

Fig:3 After executing

Task 2

- First need to compile the OS_Task_2.c file using,
  - **gcc -o task2 OS_Task_2.c -l pthread**
- **Need to run the same task2 file in 2 or more terminals to use concurrent accessing.**
- When accessing usin HR option Accounts cannot access to the database. Accountants need to wait or exit when HR is editing. (I have attached some screenshots.
- I have created a Emp struct which contain variables to store data for the program.
- Special integer x variable has created to monitor shared memory to keep track of which mode is it in now.

  X = 0: Shared Memory is empty

  X = 1: Shared Memory is used by HR

  X = 2: Shared Memory is not shared by HR but not empty
- **create_sm() :**
  - **shmget** function will allocate a memory segment
  - **shmat** function will attach the shared memory segment
  - Mapping to the shared memory will be done by the following lines

    Emp_num = (int*) shm_addr;

    ****Emp_num = 0;***

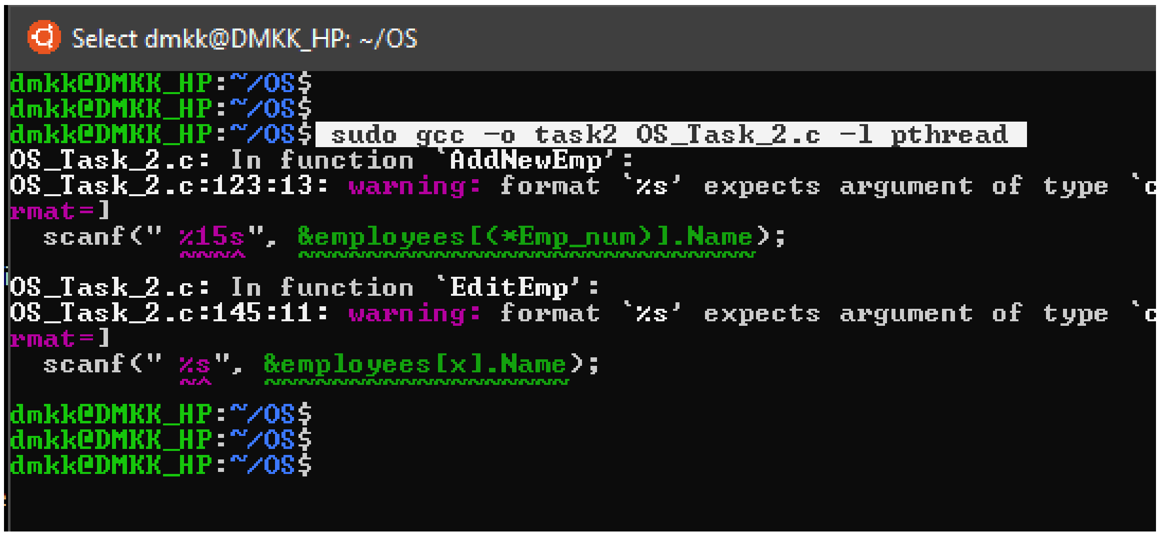
    employees = (struct Emp*) ((void*)shm_addr+sizeof(int));
- **readDB() :**
  - It's a function for a thread
  - This function is to read the database (data2.txt) file and store it in the shared memory.

- **DisplayDB() :**
  - This function will display the database that stored in the shared variable
- **AddNewEmp() :**
  - This method add a new employee
- **EditEmp() :**
  - This method to edit employee details
- **removeEmp() :**
  - This method to remove employee
- **editEMpSal():**
  - This method to edit employee salary variable
- **editEmpAttend()**
  - This method to edit employee attendance variable
- **viewTotalSal()**
  - This method to edit employee total variable
- **viewAttendance()**
  - This method to view employee attendance variable
- **viewEmployee()**
  - This method to view given employee details
- **viewSalDay()**

    - This method to view employee salpaerday variable
- **calculateTotal()**
    - This method to calculate employee total variable
- **editTotal()**
    - This method to edit employee total variable

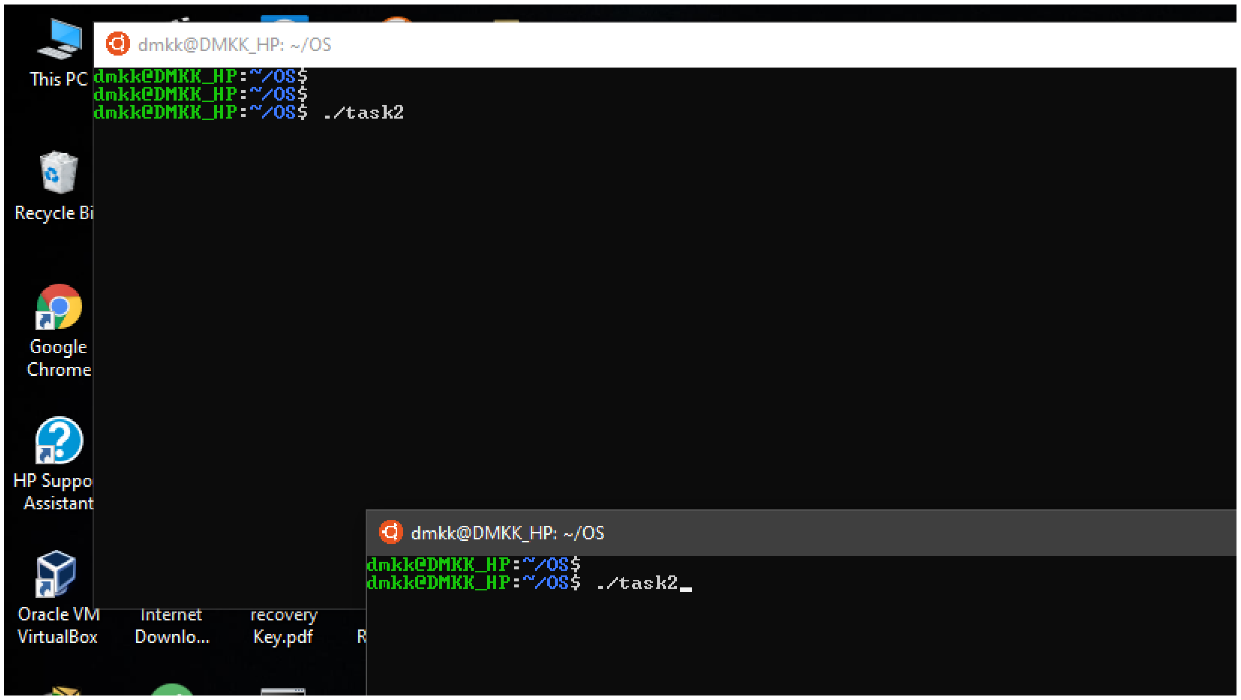
- **Exit()**
    - This method to exit from the program and save the data in the data2.txt (database) file
- **hrMenu()**
    - This method to display HR Menu and Control HR functions
- **Menu()**
    - This method to display Menu and Control Menu functions
- **accMenu()**
    - This method to display Accountant Menu and Control Accountant functions
- **contHR()**
    - To create a continue method for HR
- **contAcc()**
    - To create a continue method for Accountant
- In the **main** method:
    - **First creating the shared memory**
    - **Menu Function is called**

Screenshots of Task 2



```
Select dmkk@DMKK_HP: ~/OS
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ sudo gcc -o task2 OS_Task_2.c -l pthread
OS_Task_2.c: In function `AddNewEmp':
OS_Task_2.c:123:13: warning: format `%s' expects argument of type `c
rmat=]
   scanf(" %15s", &employees[(*Emp_num)].Name);
           ~~~~^  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

OS_Task_2.c: In function `EditEmp':
OS_Task_2.c:145:11: warning: format `%s' expects argument of type `c
rmat=]
   scanf(" %s", &employees[x].Name);
           ~^   ~~~~~~~~~~~~~~~~~~

dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
```

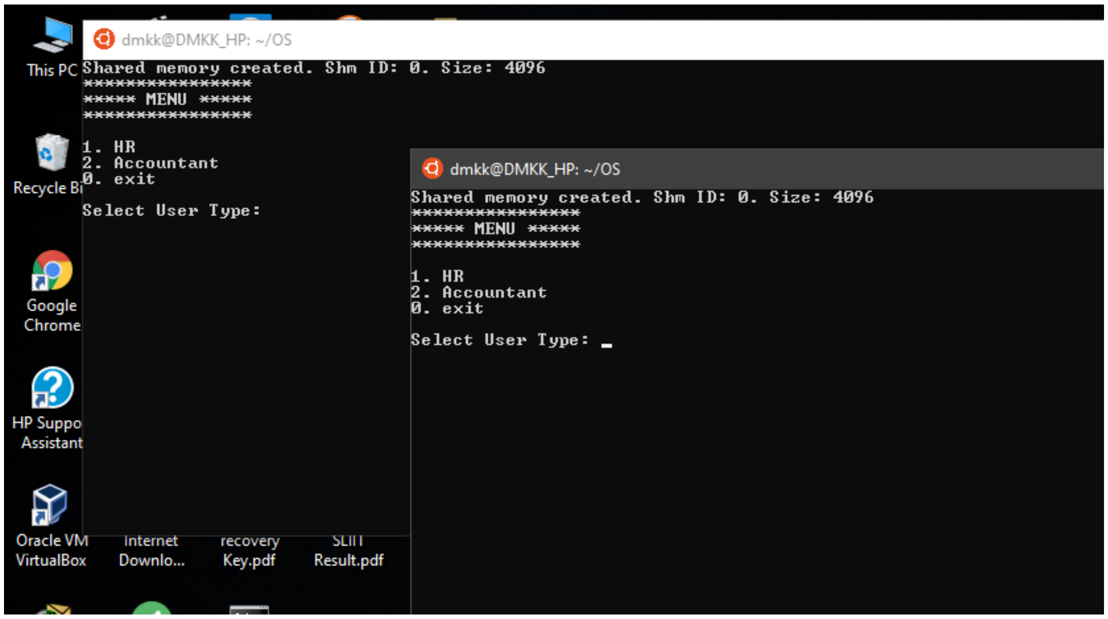
Fig:4 compiling Task 2



```
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ ./task2
```

```
dmkk@DMKK_HP:~/OS$
dmkk@DMKK_HP:~/OS$ ./task2
```

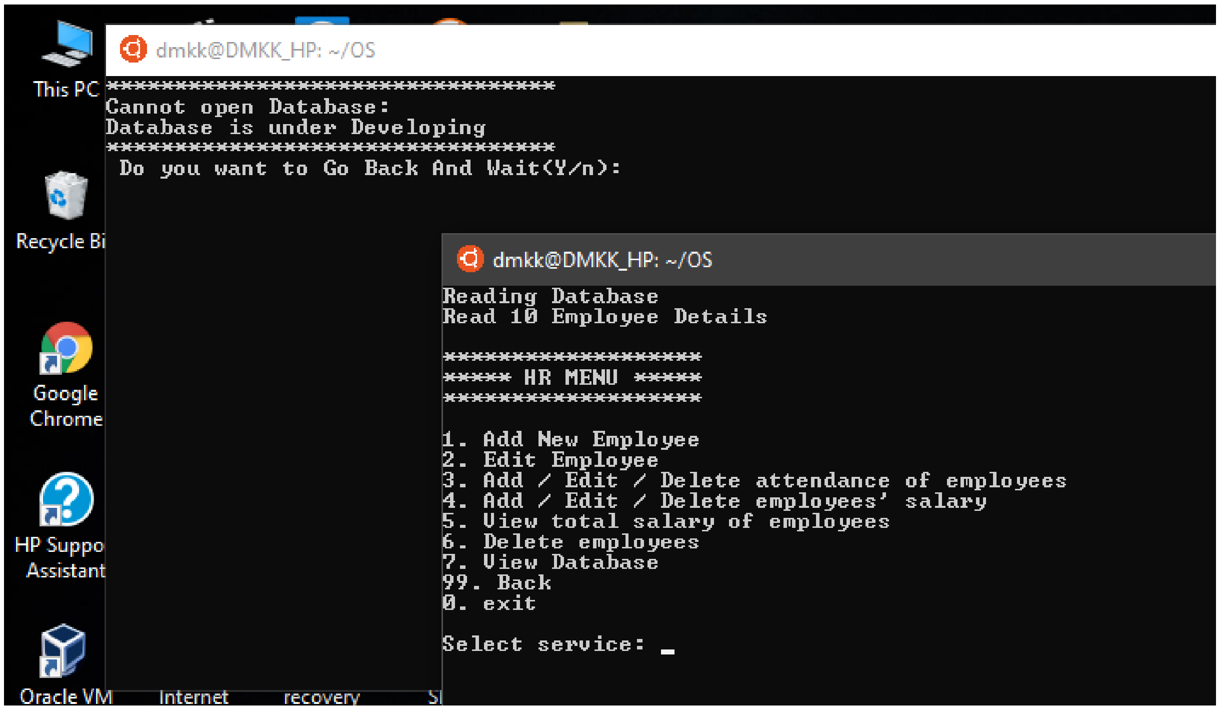
Fig: 5 executing task 2



```
Shared memory created. Shm ID: 0. Size: 4096
****************
***** MENU *****
****************

1. HR
2. Accountant
0. exit

Select User Type:
```

```
Shared memory created. Shm ID: 0. Size: 4096
****************
***** MENU *****
****************

1. HR
2. Accountant
0. exit

Select User Type: _
```

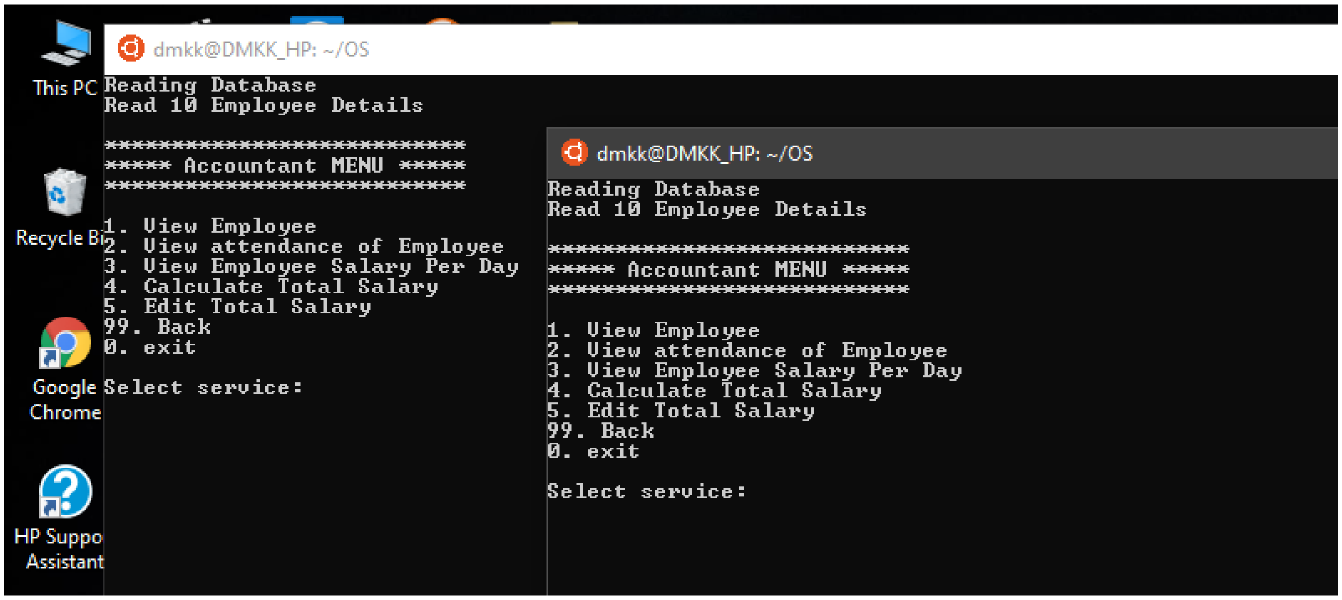
Fig: 6 after executing task 2



```
**********************************
Cannot open Database:
Database is under Developing
**********************************
 Do you want to Go Back And Wait(Y/n):
```

```
Reading Database
Read 10 Employee Details

********************
***** HR MENU *****
********************

1. Add New Employee
2. Edit Employee
3. Add / Edit / Delete attendance of employees
4. Add / Edit / Delete employees' salary
5. View total salary of employees
6. Delete employees
7. View Database
99. Back
0. exit

Select service: _
```

Fig: 7 When HR is accessing Accountant cannot access



```
Reading Database
Read 10 Employee Details

***************************
***** Accountant MENU *****
***************************

1. View Employee
2. View attendance of Employee
3. View Employee Salary Per Day
4. Calculate Total Salary
5. Edit Total Salary
99. Back
0. exit

Select service:
```

```
Reading Database
Read 10 Employee Details

***************************
***** Accountant MENU *****
***************************

1. View Employee
2. View attendance of Employee
3. View Employee Salary Per Day
4. Calculate Total Salary
5. Edit Total Salary
99. Back
0. exit

Select service:
```

Fig: 8 Two Accountants Concurrent Accessing

References

[1] Shared Memory (online) - [https://www.tutorialspoint.com/inter_process_communication/inter_process_communication_shared_memory.htm (visited- 2019-09-02)]

[2] Shared Memory (Online) – [http://www.cs.kent.edu/~ruttan/sysprog/lectures/shmem/shmem.html (visited- 2019-09-02)]