**Koneru Lakshmaiah Education Foundation**

**(Deemed to be University)**

**FRESHMAN ENGINEERING DEPARTMENT**

**A Project Based Lab Report**

**On**

**EMPLOYEE MANAGEMENT SYSTEM**

**SUBMITTED BY:**

**I.D NUMBER NAME**

180030314 N.SAI KARTHIK

180030348 M.NARENDRA

180030359 M.HARINI

180030380 K.HARSHINI

**UNDER THE GUIDANCE OF**

**AZMIRA KRISHNA**

**Assistant Professor**



**KL UNIVERSITY**

Green fields, Vaddeswaram – 522502

Guntur Dt,AP,India.

**DEPARTMENT OF BASIC ENGINEERING SCIENCES-1**



**CERTIFICATE**

This is to certify that the project based laboratory report entitled “EMPLOYEE MANAGEMENT SYSTEM” submitted by Mr./Ms**.N.SAI KARTHIK,M.NARENDRA,M.HARINI,K.HARSHINI** bearing Regd. No. **180030314,180030348,180030359,180030380** to the **Department of Basic Engineering Sciences-1, KL University** in partial fulfillment of the requirements for the completion of a project based Laboratory in “TECHNICALSKILLS-2(CODING)”course in I BTech 2 Semester, is a bonafide record of the work carried out by him/her under my supervision during the academic year 2018 – 2019.

PROJECT SUPERVISOR HEAD OF THE DEPARTMENT

**AZMIRA KRISHNA T.VAMSHIDAR**

**ACKNOWLEDGEMENTS**

It is great pleasure for me to express my gratitude to our honorable President **Sri. Koneru Satyanarayana**, for giving the opportunity and platform with facilities in accomplishing the project based laboratory report.

I express the sincere gratitude to our principal **Prof Dr. N.Venkataram** for his administration towards our academic growth.

I express sincere gratitude to HOD-BES-1 **T.Vamshidar** for his leadership and constant motivation provided in successful completion of our academic semester. I record it as my privilege to deeply thank for providing us the efficient faculty and facilities to make our ideas into reality.

I express my sincere thanks to our project supervisor **AZMIRA KRISHNA** for his/her novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it is pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

**Name:** **Regd . No:**

N.SAI KARTHIK 180030314

M.NARENDRA 180030348

M.HARINI 180030359

K.HARSHINI 180030380

**ABSTRACT**

Employees are the backbone of any company therefore their management plays a major role in deciding the success of an organization. Employees Management Software makes it easy for the employer to keep track of all records. This software allows the administrator to edit employees, add new employees, transfer/promote/terminate employees. Each employee in the database is associated with a position can be added and edited when need arises. Employees can be transferred between positions easily without having to retype back their information in the database. You can check to see if there are duplicate positions/employees in the database. Most of all, the employer can assign tasks to employees and assess their progress in order to keep track of employee performance

It is simple to understand and can be used by anyone who is not even familiar with simple employees system. It is user friendly and just asks the user to follow step by step operations by giving easy to follow options. It is fast and can perform many operations for a company.

The problem definition for designing the system is to maintain data of employee, to make easy controlling employees, to divide jobs and access control of employees, to use technology for accurate and timely processing by fully privacy and full authority access. The objective of the project is to set up employee information system about status of employee and attendance of employee and monthly salary process and delivery. To eliminate or reduce as much as possible the hardships of existing system and avoid errors while entering data. In existing method employee management are employee record are maintain in records. It’s a manual process. Complicated to search the employee salary

**INDEX**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PAGE NO** |
| 1 | Introduction | 6 |
| 2 | Aim of the Project | 7 |
| 2.1 | Advantages & Disadvantages | 7 |
| 2.2 | Future Implementation | 7 |
| 3 | Software & Hardware Details | 8 |
| 4 | Data Flow Diagram | 9-11 |

|  |  |  |
| --- | --- | --- |
| 5 | Algorithm | 12 |
| 6 | Implementation | 13-39 |
| 7 | Integration and System Testing | 40-44 |
| 8 | Conclusion | 45 |

**INTRODUCTION**

The objective of this assignment is to develop for an Employee Management System, where it is expected to enter, search, edit and view personnel information of the Employee in an Organization based on different access priority and calculate their salary package.

We are required design the application by assuming that, this system will be installed only in one terminal and used by all types of users, specially the Employee and the administrator of this application. The main purpose of this application is to store and edit the complete personal record of each Employee along with evaluation of salary (annually) in an organization. And all information that is to be added and edited must be handled via text (.txt) or data (.dat) file, so that the information can be uploaded back into the system once activated or saved when the system is exited.

We are also required to incorporate the Validation throughout the application for each entry, from the user in order to avoid logical errors.

We are supposed to describe, justify and implement an object oriented system by the application,

Employee Management system is an application that enables users to create and store Employee Records. The application also provides facilities of a payroll system which enables user to generate Pay slips too. This application is helpful to department of the organization which maintains data of employees related to an organization .

**AIM**

To get the details of employee and display it on the screen(EMPLOYEE MANAGEMENT SYSTEM)

## Advantages:-

1)Records employee information

2)Records the current number of employees according to division / department with the automatic updating of figures when people are employed or leave.

3)Keeps records of inactive employees (those not in the company any longer)

4)Employees can update their own details which are sent for managerial authorization before being added to the system

5)Search facility for easy finding of staff

**Disadvantages:-**

1)Lengthy and complex: ...

2)Demeaning the communication **system** between employer and employee

3) Employees may suffer from low self-esteem: ...

**Future enhancements:-**

1)It provides better and efficient service to members.

2)Reduce the over load of employee.

3)Faster retrival of information about the employee

4)All details of a employee will be available with a click.

**SYSTEM REQUIREMENTS**

* **SOFTWARE REQUIREMENTS:**

The major software requirements of the project are as follows:

Language : UBUNTU,TURBO-C

Operating system**:**Windows Xp or later.

* **HARDWARE REQUIREMENTS:**

The hardware requirements that map towards the software are as follows:

RAM :8GB

Processor :Intel® core ™ - i5 8250U

**DATA FLOW DIAGRAM**

**PRINT MENU**

**INSERTION**

**IF OPT==2**

**CREATION**

**IF OPT==1**

**Takes Input From the User**

**YES**

**NO**

**YES**

**NO**

**CREATES A LIST OF EMPLOYEES**

**INSERTS AN EMPLOYEE IN THE LIST[BIGGINING,MIDDLE,END]**

**IF OPT==3**

**YES**

**NO**

**DELETION**

**IF OPT==4**

**YES**

**DELETES AN EMPLOYEE IN THE LIST[BIGGINING,MIDDLE,END]**

**NO**

**YES**

**IF OPT==5**

**SEARCHES AN EMPLOYEE IN THE LIST**

**NO**

**DISPLAYS EMPLOYEE DETAILS**

**YES**

**IF OPT==6**

**NO**

**YES**

**IF OPT==7**

**SORTS THE GIVEN LIST**

**NO**

**YES**

**IF OPT==8**

**UPDATES AN EMPLOYEE**

**NO**

**IF OPT==9**

**MERGES TWO SEPERATES LISTS**

**ALGORITHM**

Step1:Start

Step2:Display’s the menu

Step3:Takes the input from the user

Step4:

4.1: Creating a list

4.2: Inserting new employee data into the list

4.2.1:Inserts at the beginning of the list

4.2.2:Inserts at the middle of the list

4.2.3:Inserts at the end of the list

4.3: Delets an employee data in the list

4.3.1:Delets at the beginning of the list

4.3.2:Delets at the middle of the list

4.3.3:Delets at the end of the list

4.4: Searches an employee based on id in the list

4.5: Displays all the details of the employee

4.6: Sorts the list based on employee salary

4.7: Updates the details of a particular employee

4.8: Merges two separate lists

4.9: Exit

Step5:Stop

**IMPLEMENTATION**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

void creation();

void creation1();

void insertionatbeginning();

void insertionatmiddle();

void insertionatend();

void deletionatbeginning();

void deletionatmiddle();

void deletionatend();

void search();

void display();

void display1();

void sort();

void update();

void merge();

struct node

{

long long int empid;

char empname[50];

char empdesignation[50];

float empsal;

struct node \*next;

}s;

struct node1

{

long long int empid1;

char empname1[50];

char empdesignation1[50];

float empsal1;

struct node1 \*next;

}s1;

struct node \*start=NULL,\*ptr;

struct node1 \*start1=NULL,\*ptr1;

void main()

{

int opt;

while(1)

{

printf("Enter an option\n");

printf("1.Creation 2.Insertion 3.Deletion 4.Search 5.Display 6.Sort 7.Update 8.Merge 9.Exit\n");

scanf("%d",&opt);

switch(opt)

{

case 1:if(opt==1)

creation();

break;

case 2:if(opt==2)

{

int choice;

printf("Enter a choice\n");

printf("1.Insert at Beginning 2.Insert at Middle 3.Insert at End\n");

scanf("%d",&choice);

switch(choice)

{

case 1:if(choice==1)

insertionatbeginning();

break;

case 2:if(choice==2)

insertionatmiddle();

break;

case 3:if(choice==3)

insertionatend();

break;

default:

printf("Enter a valid choice\n");

break;

}

}

break;

case 3:if(opt==3)

{

int choice;

printf("Enter a choice\n");

printf("1.Delete at Beginning 2.Delete at Middle 3.Delete at End\n");

scanf("%d",&choice);

switch(choice)

{

case 1:if(choice==1)

deletionatbeginning();

break;

case 2:if(choice==2)

deletionatmiddle();

break;

case 3:if(choice==3)

deletionatend();

break;

default:

printf("Enter a valid choice\n");

break;

}

}

break;

case 4:if(opt==4)

search();

break;

case 5:if(opt==5)

display();

break;

case 6:if(opt==6)

sort();

break;

case 7:if(opt==7)

update();

break;

case 8:if(opt==8)

merge();

break;

case 9:if(opt==9)

exit(0);

break;

default:

printf("Enter a valid option\n");

break;

}

}

}

void creation()

{

struct node \*newnode;

long long int eid;

char ename[50];

char edsg[50];

float esal;

newnode=(struct node \*)malloc(sizeof(struct node));

printf("Enter The Details Of The Employee\n");

printf("Enter Employee Id\n");

scanf("%lld",&eid);

printf("Enter Employee Name\n");

scanf("%s",ename);

printf("Enter Employee designation\n");

scanf("%s",edsg);

printf("Enter Employee Salary\n");

scanf("%f",&esal);

newnode->empid=eid;

strcpy(newnode->empname,ename);

strcpy(newnode->empdesignation,edsg);

newnode->empsal=esal;

newnode->next=NULL;

if(start==NULL)

start=newnode;

else

{

ptr=start;

while(ptr->next!=NULL)

ptr=ptr->next;

ptr->next=newnode;

}

printf("Newnode is Created\n");

}

void creation1()

{

struct node1 \*newnode;

long long int eid;

char ename[50];

char edsg[50];

float esal;

newnode=(struct node1 \*)malloc(sizeof(struct node1));

printf("Enter The Details Of The Employee\n");

printf("Enter Employee Id\n");

scanf("%lld",&eid);

printf("Enter Employee Name\n");

scanf("%s",ename);

printf("Enter Employee designation\n");

scanf("%s",edsg);

printf("Enter Employee Salary\n");

scanf("%f",&esal);

newnode->empid1=eid;

strcpy(newnode->empname1,ename);

strcpy(newnode->empdesignation1,edsg);

newnode->empsal1=esal;

newnode->next=NULL;

if(start1==NULL)

start1=newnode;

else

{

ptr1=start1;

while(ptr1->next!=NULL)

ptr1=ptr1->next;

ptr1->next=newnode;

}

printf("Newnode is Created\n");

}

void insertionatbeginning()

{

struct node \*newnode;

long long int eid;

char ename[50];

char edsg[50];

float esal;

newnode=(struct node \*)malloc(sizeof(struct node));

printf("Enter The Details Of The Employee\n");

printf("Enter Employee Id\n");

scanf("%lld",&eid);

printf("Enter Employee Name\n");

scanf("%s",ename);

printf("Enter Employee designation\n");

scanf("%s",edsg);

printf("Enter Employee Salary\n");

scanf("%f",&esal);

newnode->empid=eid;

strcpy(newnode->empname,ename);

strcpy(newnode->empdesignation,edsg);

newnode->empsal=esal;

newnode->next=NULL;

if(start==NULL)

{

start=newnode;

}

else

{

newnode->next=start;

start=newnode;

}

}

void insertionatmiddle()

{

struct node \*newnode,\*pptr;

long long int eid;

char ename[50];

char edsg[50];

float esal;

newnode=(struct node \*)malloc(sizeof(struct node));

printf("Enter The Details Of The Employee\n");

printf("Enter Employee Id\n");

scanf("%lld",&eid);

printf("Enter Employee Name\n");

scanf("%s",ename);

printf("Enter Employee designation\n");

scanf("%s",edsg);

printf("Enter Employee Salary\n");

scanf("%f",&esal);

newnode->empid=eid;

strcpy(newnode->empname,ename);

strcpy(newnode->empdesignation,edsg);

newnode->empsal=esal;

newnode->next=NULL;

int pos,i=1;

printf("Enter the position where the node is to be inserted\n");

scanf("%d",&pos);

ptr=start;

if(start==NULL)

start=newnode;

else

{

while(i<pos)

{

pptr=ptr;

ptr=ptr->next;

i+=1;

}

newnode->next=pptr->next;

pptr->next=newnode;

}

}

void insertionatend()

{

struct node \*newnode;

long long int eid;

char ename[50];

char edsg[50];

float esal;

newnode=(struct node \*)malloc(sizeof(struct node));

printf("Enter The Details Of The Employee\n");

printf("Enter Employee Id\n");

scanf("%lld",&eid);

printf("Enter Employee Name\n");

scanf("%s",ename);

printf("Enter Employee designation\n");

scanf("%s",edsg);

printf("Enter Employee Salary\n");

scanf("%f",&esal);

newnode->empid=eid;

strcpy(newnode->empname,ename);

strcpy(newnode->empdesignation,edsg);

newnode->empsal=esal;

newnode->next=NULL;

if(start==NULL)

start=newnode;

else

{

ptr=start;

while(ptr->next!=NULL)

ptr=ptr->next;

ptr->next=newnode;

}

}

void deletionatbeginning()

{

ptr=start;

start=start->next;

free(ptr);

}

void deletionatmiddle()

{

struct node \*pptr;

int pos,i=1;

printf("Enter the postion of the node to be deleted\n");

scanf("%d",&pos);

ptr=start;

while(i<pos)

{

pptr=ptr;

ptr=ptr->next;

i+=1;

}

pptr->next=ptr->next;

free(ptr);

}

void deletionatend()

{

struct node \*pptr;

ptr=start;

while(ptr->next!=NULL)

{

pptr=ptr;

ptr=ptr->next;

}

pptr->next=NULL;

free(ptr);

}

void search()

{

long long int id;

printf("Enter Employee Id to search in the list\n");

scanf("%lld",&id);

ptr=start;

while(ptr->next!=NULL)

{

if(ptr->empid==id)

{

printf("Employee Found\n");

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr->empid,ptr->empname,ptr->empdesignation,ptr->empsal);

ptr=ptr->next;

break;

}

ptr=ptr->next;

}

if(ptr->empid==id)

{

printf("Employee Found\n");

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr->empid,ptr->empname,ptr->empdesignation,ptr->empsal);

}

}

void display()

{

ptr=start;

while(ptr->next!=NULL)

{

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr->empid,ptr->empname,ptr->empdesignation,ptr->empsal);

ptr=ptr->next;

}

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr->empid,ptr->empname,ptr->empdesignation,ptr->empsal);

}

void display1()

{

ptr=start;

while(ptr->next!=NULL)

{

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr->empid,ptr->empname,ptr->empdesignation,ptr->empsal);

ptr=ptr->next;

}

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr->empid,ptr->empname,ptr->empdesignation,ptr->empsal);

ptr1=start1;

while(ptr1->next!=NULL)

{

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr1->empid1,ptr1->empname1,ptr1->empdesignation1,ptr1->empsal1);

ptr1=ptr1->next;

}

printf("Employee Id:%lld Employee Name:%s Employee Designation:%s Employee Salary:%0.2f\n",ptr1->empid1,ptr1->empname1,ptr1->empdesignation1,ptr1->empsal1);

}

void sort()

{

struct node \*pptr;

struct node \*temp;

temp=(struct node \*)malloc(sizeof(struct node));

for(ptr=start;ptr->next!=NULL;ptr=ptr->next)

{

for(pptr=ptr->next;pptr->next!=NULL;pptr=pptr->next)

{

if(pptr->empsal<ptr->empsal)

{

temp->empid=ptr->empid;

strcpy(temp->empname,ptr->empname);

strcpy(temp->empdesignation,ptr->empdesignation);

temp->empsal=ptr->empsal;

ptr->empid=pptr->empid;

strcpy(ptr->empname,pptr->empname);

strcpy(ptr->empdesignation,pptr->empdesignation);

ptr->empsal=pptr->empsal;

pptr->empid=temp->empid;

strcpy(pptr->empname,temp->empname);

strcpy(pptr->empdesignation,temp->empdesignation);

pptr->empsal=temp->empsal;

}

}

if(pptr->next==NULL)

{

if(pptr->empsal<ptr->empsal)

{

temp->empid=ptr->empid;

strcpy(temp->empname,ptr->empname);

strcpy(temp->empdesignation,ptr->empdesignation);

temp->empsal=ptr->empsal;

ptr->empid=pptr->empid;

strcpy(ptr->empname,pptr->empname);

strcpy(ptr->empdesignation,pptr->empdesignation);

ptr->empsal=pptr->empsal;

pptr->empid=temp->empid;

strcpy(pptr->empname,temp->empname);

strcpy(pptr->empdesignation,temp->empdesignation);

pptr->empsal=temp->empsal;

}

}

}

}

void update()

{

struct node \*pptr;

int x;

long long int eid,id;

char ename[50],name[50];

char edsg[50],dsg[50];

float esal,sal;

printf("Choose an option\n");

printf("1.Employee Id 2.Employee Name 3.Employee Designation 4.Employee Salary\n");

scanf("%d",&x);

if(x==1)

{

printf("Enter the Id number of the employee to be updated\n");

scanf("%lld",&id);

printf("Enter New Employee Id\n");

scanf("%lld",&eid);

for(ptr=start;ptr->next!=NULL;ptr=ptr->next)

{

if(ptr->empid==id)

{

pptr=ptr;

break;

}

}

if(ptr->empid==id)

pptr=ptr;

pptr->empid=eid;

}

else if(x==2)

{

printf("Enter Employee Name to be updated\n");

scanf("%s",name);

printf("Enter New Employee Name\n");

scanf("%s",ename);

for(ptr=start;ptr->next!=NULL;ptr=ptr->next)

{

if(strcmp(ptr->empname,name)==0)

{

pptr=ptr;

break;

}

}

if(strcmp(ptr->empname,name)==0)

pptr=ptr;

strcpy(pptr->empname,ename);

}

else if(x==3)

{

printf("Enter Employee Designation to be updated\n");

scanf("%s",dsg);

printf("Enter New Employee designation\n");

scanf("%s",edsg);

for(ptr=start;ptr->next!=NULL;ptr=ptr->next)

{

if(strcmp(ptr->empdesignation,dsg)==0)

{

pptr=ptr;

break;

}

}

if(strcmp(ptr->empdesignation,dsg)==0)

pptr=ptr;

strcpy(pptr->empdesignation,edsg);

}

else

{

printf("Enter Employee Salary to be updated\n");

scanf("%f",sal);

printf("Enter New Employee Salary\n");

scanf("%f",esal);

for(ptr=start;ptr->next!=NULL;ptr=ptr->next)

{

if(ptr->empsal==sal)

{

pptr=ptr;

break;

}

}

if(ptr->empsal==sal)

pptr=ptr;

pptr->empsal=esal;

}

}

void merge()

{

int opt;

while(1)

{

printf("choose an option\n");

printf("1.Creation 2.Exit\n");

scanf("%d",&opt);

if(opt==1)

creation1();

else

break;

}

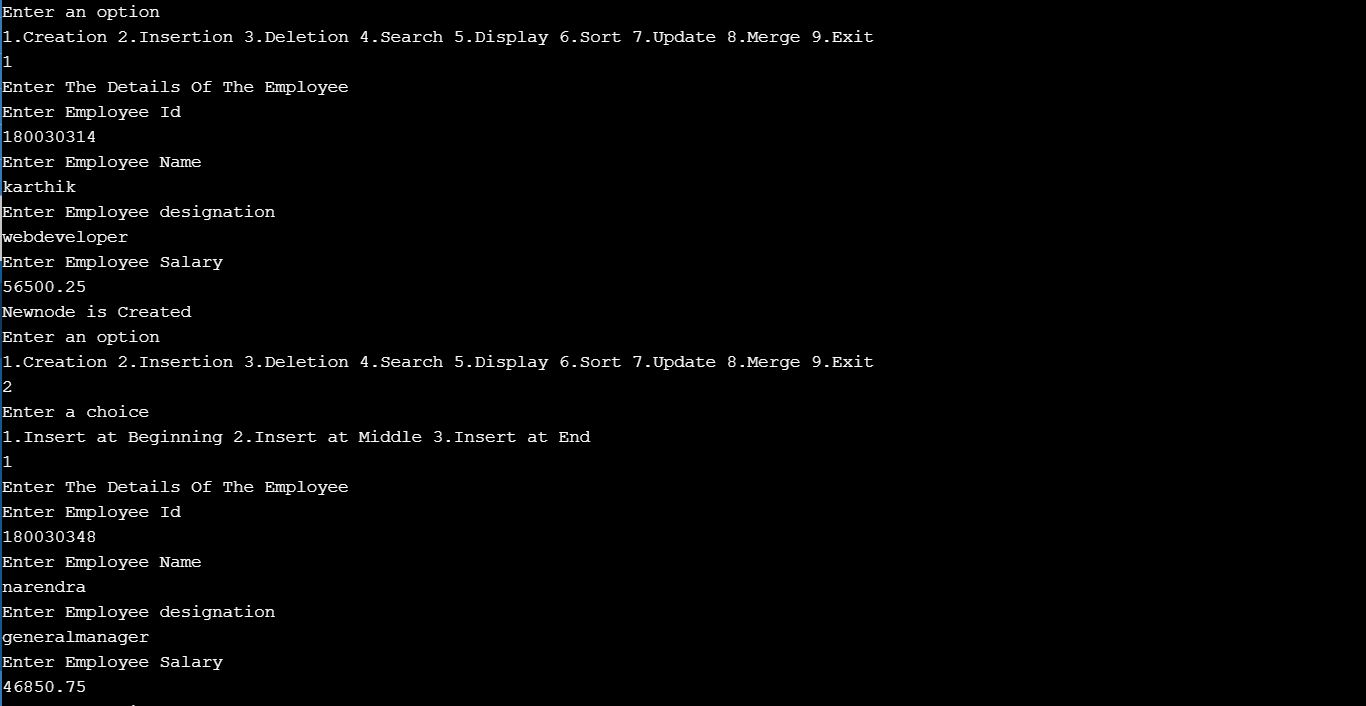
display1();

}

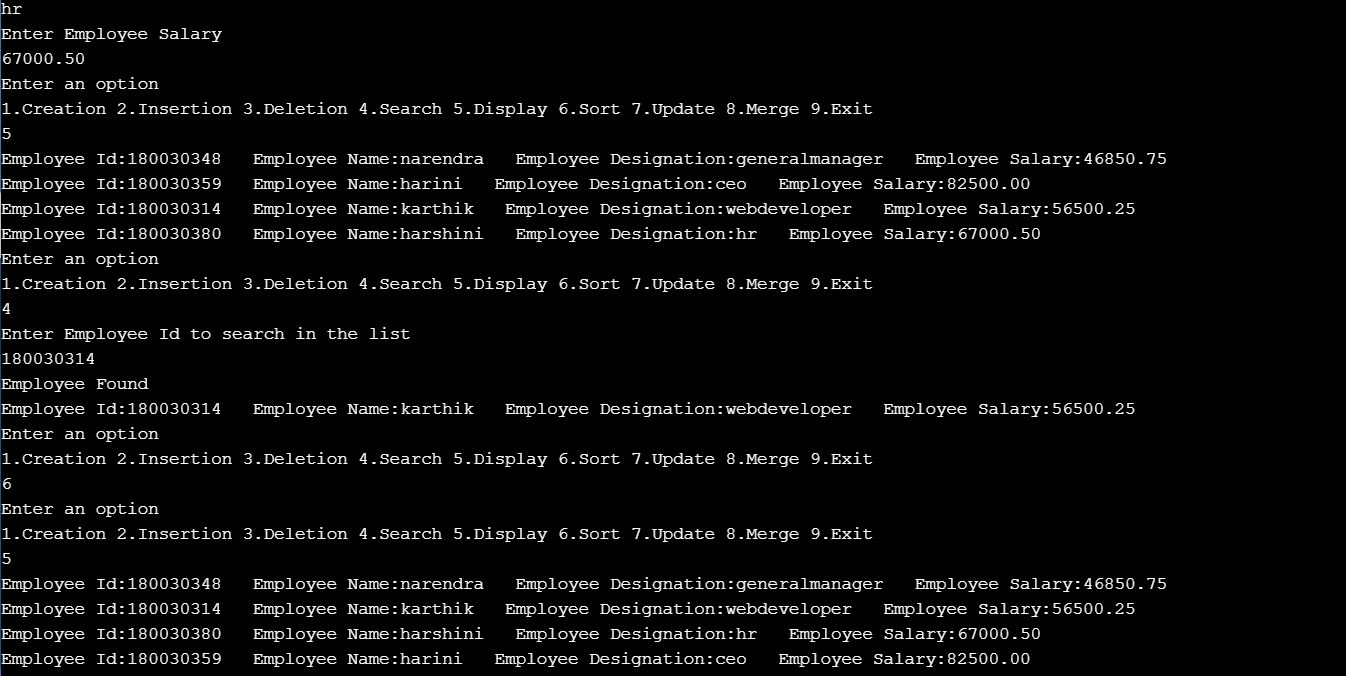
**INTEGRATION AND SYSTEM TESTING**

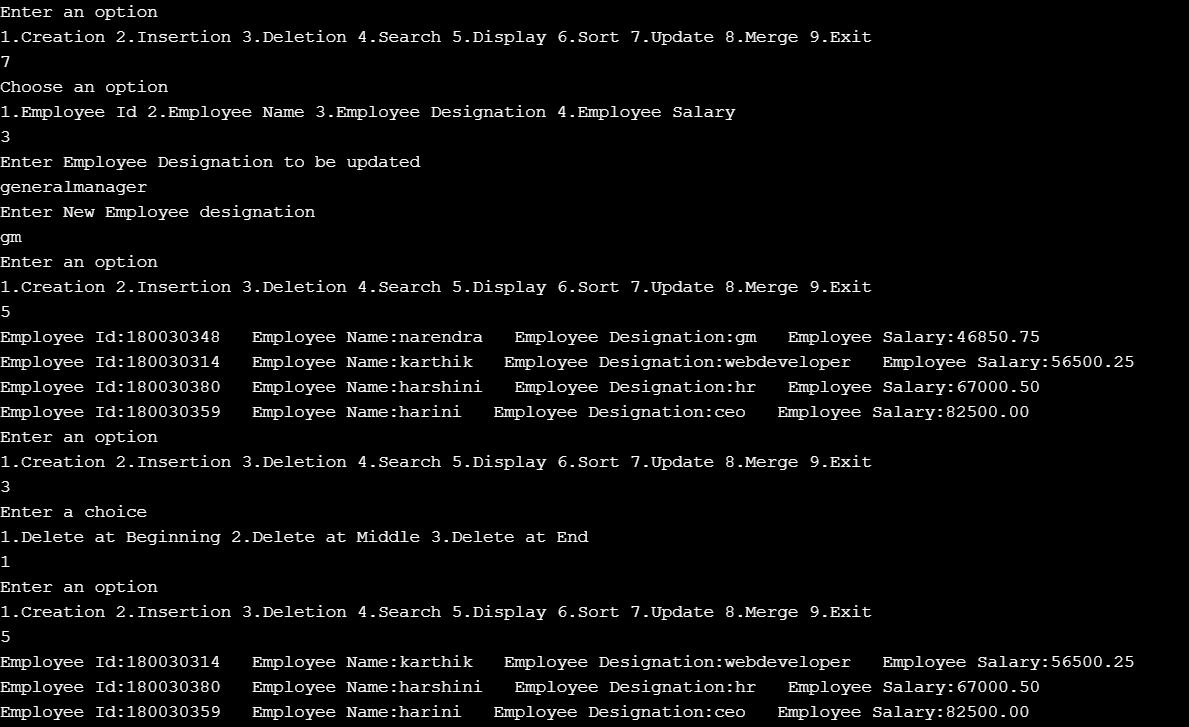
**Outputs:-**

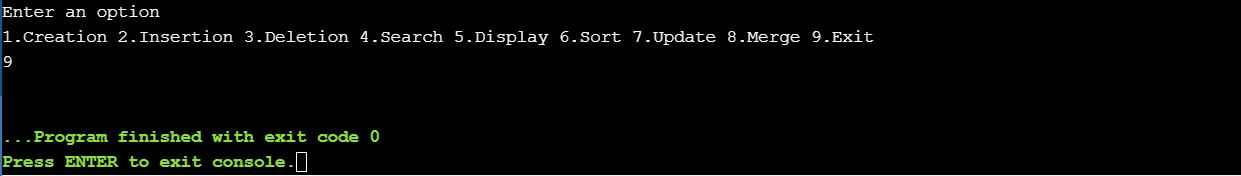
Screen Shots:



****

****

****

****

**CONCLUSION**

According to managers and employers, employees leave the organization more because of internal reasons within the organization and less because of external reasons or their own personal reasons. It can also be inferred that as majority of the reasons of employee turnover are from within the organization, they are controllable. If the organization or management makes effort by changing or modifying HR policies of the company then they may be able to control employee turnover to a large extent. Further, managers perceive that employees are comfortable when the workload is reasonable and increase in it may influence their decision to discontinue the work. Equally important is the work shift timing in the organization. As it is evident from this research that work life balance is also important impacting the employee decision to continue or quit. Further conclusions include: managers consider salary as the major reason behind employee turnover. Managers also opine that ambience at the work place influences heavily on the employee turnover in the organization. Majority of the managers opine that employees leave due to the style of operation of the manager. They need to maintain good rapport and good relationship with the employees to decrease the employee turnover.