DLITHE PROJECT REPORT

PROJECT ID : CP027

PROJECT TITLE : Employee Record System

TEAM MEMBERS : Ankita (4MT21CS025)

Aruna S (4MT21CS028)

Chaithrashree (4MTCS21037)

Divya S Suvarna (4MT19CS050)

Dwithi Shetty(4MT21CS052)

ABSTRACT

The Employee Record System is a software solution designed to efficiently manage and maintain employee data within an organization. This system serves as a central repository for all employee-related information, streamlining HR processes and improving overall workforce management. The Employee Record System provides a secure, user-friendly interface for HR professionals and authorized personnel to input, retrieve, and manage employee records.

It streamlines HR processes, enhances data security, and provides valuable insights for strategic decision-making. The Employee Record System contributes to improved employee satisfaction, regulatory compliance, and overall organizational efficiency.

INTRODUCTION

In this series of C Projects Source Code, we'll look at how to build an Employee Management System in C. We may manage the information of workers working in a firm or organization using this Employee Management System. The file handling technique is used here to save the data in a particular file, and you get the notion of this project as soon as you hear the name.

This project uses the Insert, Edit, and Delete file actions, but the sole constraint is that you can only display the data, not search for any data item in particular.

The following modules are included in this project.

- Add Employee Details
- Edit Employee details
- Modify Employee
- Delete Employee

Background :

Employee Record Systems reflects the dynamic nature of workforce management and the continuous adaptation of technology to meet the evolving needs of organizations.The development and implementation of Employee Record Systems have evolved significantly over the years in response to the changing needs of organizations and advancements in technology. Employee Record Systems is a critical component of HR management, facilitating efficient, compliant, and strategic handling of employee data and processes.

Objectives :

- The main objectives of the Employee Record Systems are as follows:
- To maintain centralized data management
- To maintain efficient HR operations
- To ensure data security and compliance
- To provide enhanced decision-making
- To access employee self-service and engagement
- To facilitate scalability and integration.

TECHNOLOGYIES USED

The Employee Record Systems is implemented using the following technologies:

- C programming language
- File handling for data storage

SYSTEM ARCHITECTURE

Database:

Whether you’re an HR professional or a small business owner wearing many hats, you probably know that keeping a record of all your employees’ information can be a time-consuming task. Our Employee Database template offers a single space to store all types of employee data, from emails to hire dates.

Using Employee Database template, you can:

- Store all employee information in one place,
- Create custom labels like food allergies, type of employment,
- Get an overview of all members by team,
- Call or text an employee directly from the database,
- Manage hire dates, layoff, etc. in a calendar,
- Filter employees by department, location, or custom attribute.

Benefits of Using Our Employee Database Template

- Single point of storage for all employee data,
- Faster access to relevant information,
- Managing employees by team or department,
- Simplified HR processes.

PROJECT MODULES

The project consists of the following modules:

- Module 1: Employee record management

  - Add employee details

  - List employee details

  - Modify employee details

  - Delete employee

DESIGN AND IMPLEMENTATION

A properly designed database provides you with access to up-to-date, accurate information. Because a correct design is essential to achieving your goals in working with a database, investing the time required to learn the principles of good design makes sense. In the end, you are much more likely to end up with a database that meets your needs and can easily accommodate change.

Using Employee Database template, you can:

- Store all employee information in one place,
- Create custom labels like food allergies, type of employment,
- Get an overview of all members by team,
- Call or text an employee directly from the database,
- Manage hire dates, layoff, etc. in a calendar,
- Filter employees by department, location, or custom attribute.

Benefits of Using Our Employee Database Template

- Single point of storage for all employee data,
- Faster access to relevant information,
- Managing employees by team or department,
- Simplified HR processes.

FEATURES AND IMPLEMENTATION

- Feature 1: Employee record

  - Add new employee details to the database.

  -Display a list of all employees with details.

  -Modify existing employee records.

  -Delete employee records.

TESTING

Unit Testing

- Unit testing is the first level of testing usually performed by the developers.
- In unit testing, a module or component is tested in isolation.
- As the testing is limited to a particular module or component, exhaustive testing is possible.
- Advantage – Error can be detected at an early stage saving time and money to fix it.
- Limitation – Integration issues are not detected in this stage, modules may work perfectly on isolation but can have issues in interfacing between the modules.

Integration Testing

- Integration testing is the second level of testing in which we test a group of related modules.
- It aims at finding interfacing issues b/w the modules i.e. if the individual units can be integrated into a sub-system correctly.

User Acceptance Testing

- Acceptance testing is the final and one of the most important levels of testing on successful completion of which the application is released to production.
- It aims at ensuring that the product meets the specified business requirements within the defined standard of quality.
- There are two kinds of acceptance testing- alpha testing and beta testing.
  1. When acceptance testing is carried out by testers or some other internal employees of the organization at the developer's site it is known as **alpha testing**.
  2. User acceptance testing done by end-users at the end-user's site is called **beta testing**.

CHALLENGES FACED

- Implementing file handling for data storage and retrieval.
- Ensuring data consistency and accuracy.
- Handling errors and exceptions gracefully.
- Future Enhancements
- Implementing a graphical user interface (GUI) for a more user-friendly experience.
- Adding features for generating reports and statistics.
- Implementing security measures to protect data.

FUTURE ENHANCEMENT

Enhancing an employee record system using file handling in programming can be done in several ways. Here are some future enhancements you can consider:

Data Encryption: Implement data encryption to protect sensitive employee information stored in files, ensuring data security and compliance with privacy regulations.

Search and Sorting: Add advanced search and sorting functionality to quickly locate and organize employee records based on various criteria like name, department, or hire date.

Data Validation: Implement robust data validation to ensure that only valid and consistent data is entered into the system, preventing errors and inconsistencies in records.

User Authentication: Enhance security by implementing user authentication mechanisms to control access to the employee record system, ensuring only authorized personnel can view or modify data.

Audit Trails: Keep a log of all changes made to employee records, including who made the changes and when, for tracking and auditing purposes.

Reporting and Analytics: Develop tools for generating reports and analytics on employee data, allowing for better decision-making and trend analysis.

Backup and Recovery: Implement automated backup and recovery processes to prevent data loss in case of system failures or errors.

Integration: Integrate the employee record system with other HR or payroll systems to streamline data sharing and reduce redundancy.

User-Friendly Interface: Continuously improve the user interface for ease of use and accessibility, incorporating user feedback and usability testing.

CONCLUSION

Conclusion. Software for employee management systems helps your organization improve workforce productivity and boost overall well-being by tracking and monitoring the daily working activities of every employee. The purpose of an employee management system is to help improve workforce productivity, identify ways to engage and retain talent, and alleviate administrative burdens for HR professionals. Achieving greater efficiency through the use of technology can also help control costs and minimize compliance risks.

APPENDIX

Appendix A : Sample Data Files

Sample data files for employee used for testing the program.

employee.data: Contains sample medicine records.

Appendix B : User Manual

A user manual providing instructions on how to use the employee Management System, including detailed explanations of each menu option and functionality.

Appendix C : Test Cases

A list of test cases and their expected results used for unit testing and integration testing of the program.

Appendix D : Flowcharts

Flowcharts illustrating the flow of control and data in the program for each module (employee record).

Appendix E : Error Handling

A document detailing the error handling mechanisms implemented in the program, including how errors are detected and how they are handled.

Appendix F : Future Enhancements

A document outlining potential future enhancements and features that can be added to the program to improve its functionality and usability.

Appendix G : Sample Reports

Sample reports generated by the program, showcasing the potential reporting capabilities of the system.

Appendix H : Data Dictionary

A data dictionary providing descriptions of the data structures used in the program, including the fields and their data types.

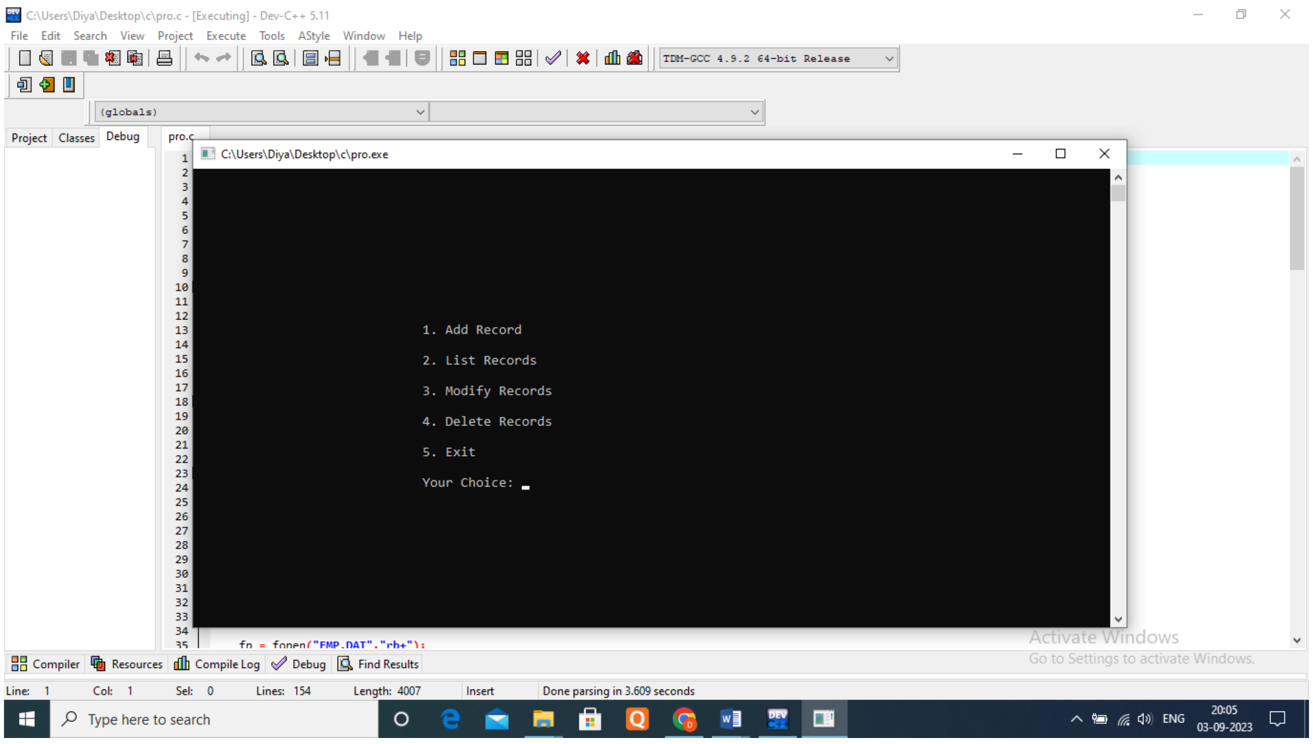
Appendix I : Glossary

A glossary of terms and abbreviations used in the program, helping users understand the terminology used in the system.

Appendix J : References

A list of references and sources of information used in the development of the progra

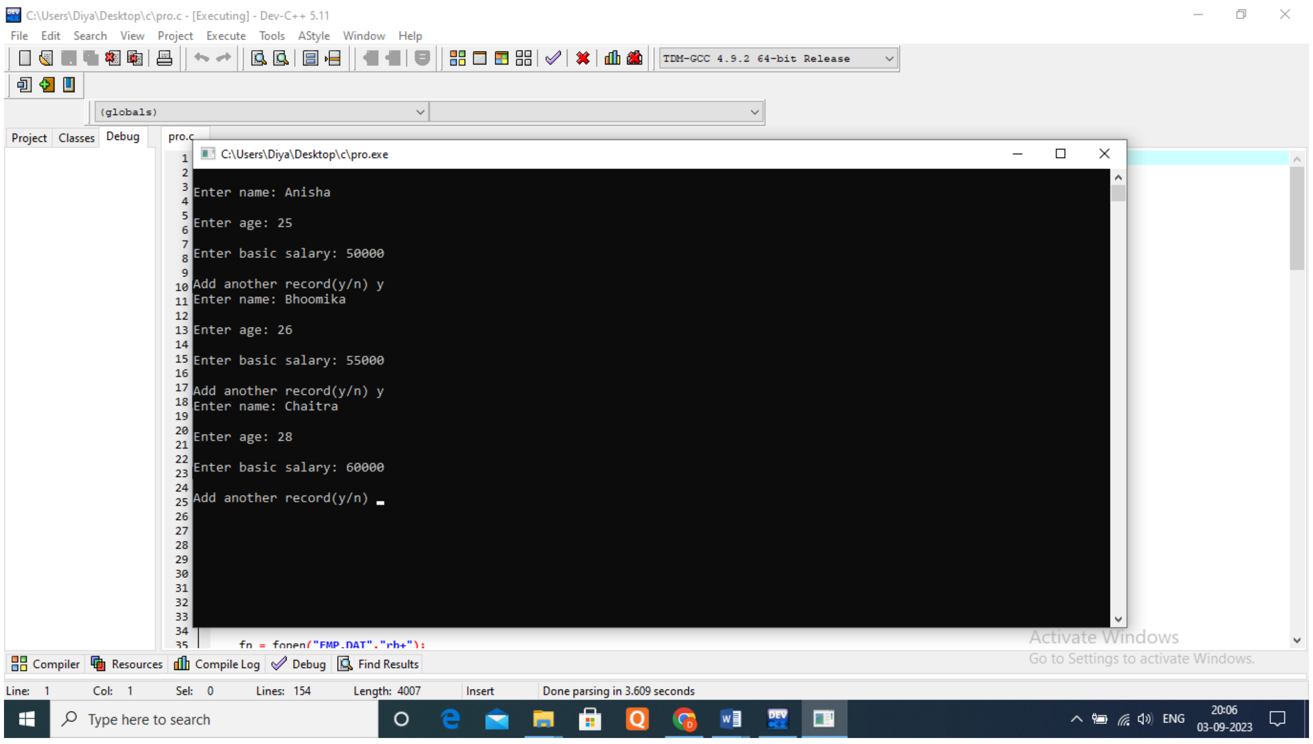
SCREENSHOTS



C:\Users\Diya\Desktop\c\pro.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
C:\Users\Diya\Desktop\c\pro.exe
Activate Windows
Go to Settings to activate Windows.
Type here to search
20:05
03-09-2023

```
1. Add Record

2. List Records

3. Modify Records

4. Delete Records

5. Exit

Your Choice: _
```

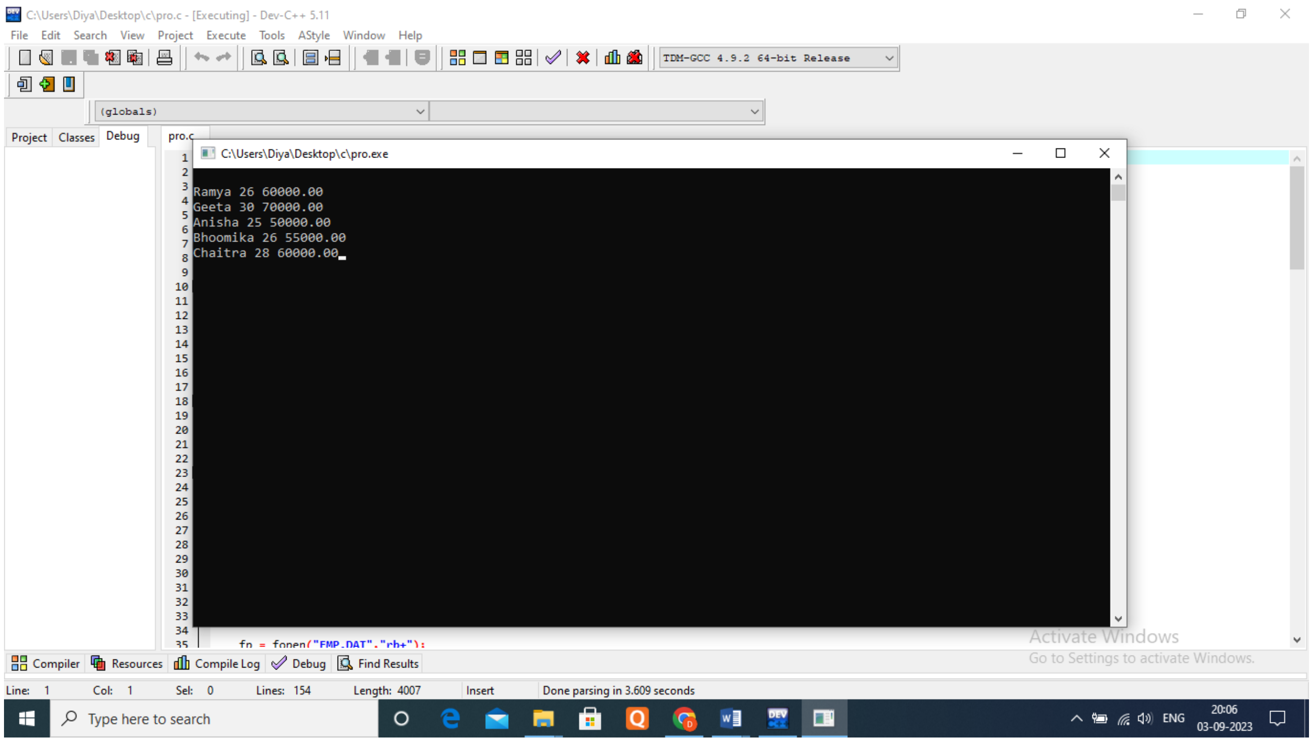
Adding record:



C:\Users\Diya\Desktop\c\pro.c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
C:\Users\Diya\Desktop\c\pro.exe
Activate Windows
Go to Settings to activate Windows.
Type here to search
20:06
03-09-2023

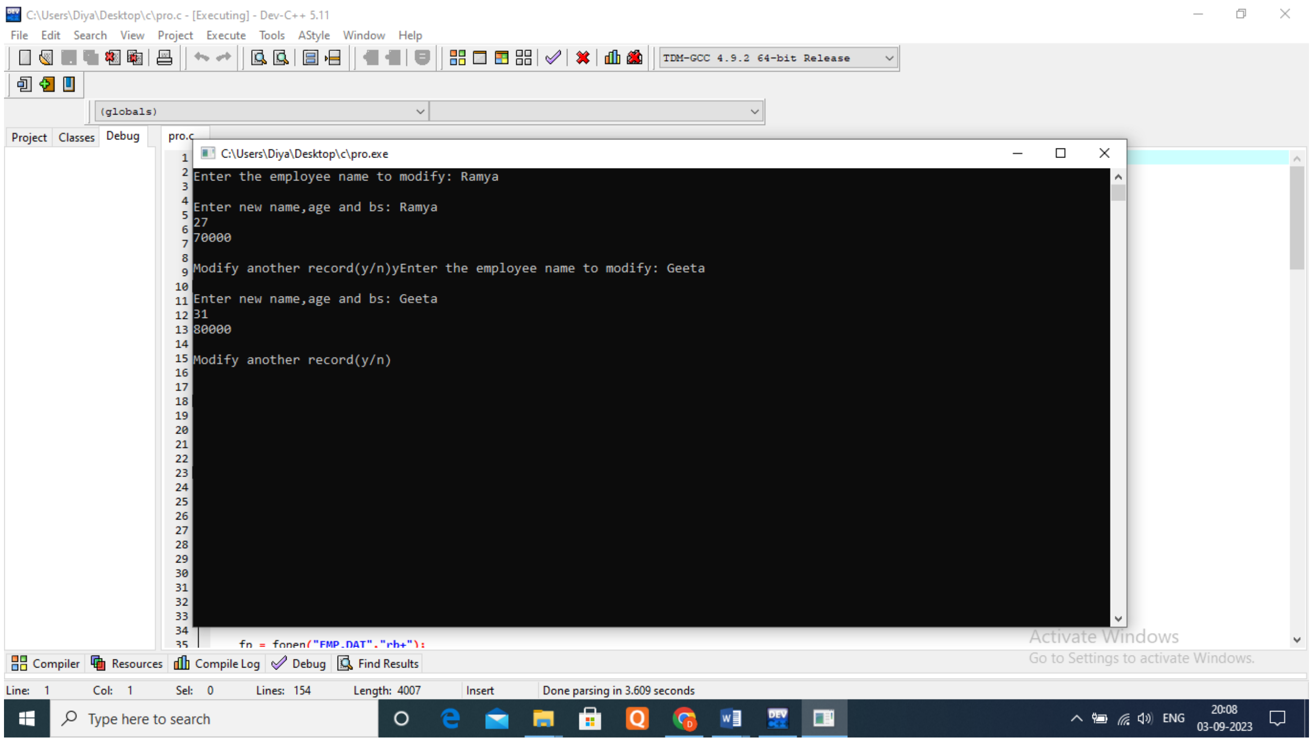
```
Enter name: Anisha

Enter age: 25

Enter basic salary: 50000

Add another record(y/n) y
Enter name: Bhoomika

Enter age: 26

Enter basic salary: 55000

Add another record(y/n) y
Enter name: Chaitra

Enter age: 28

Enter basic salary: 60000

Add another record(y/n) _
```

Displaying Record:



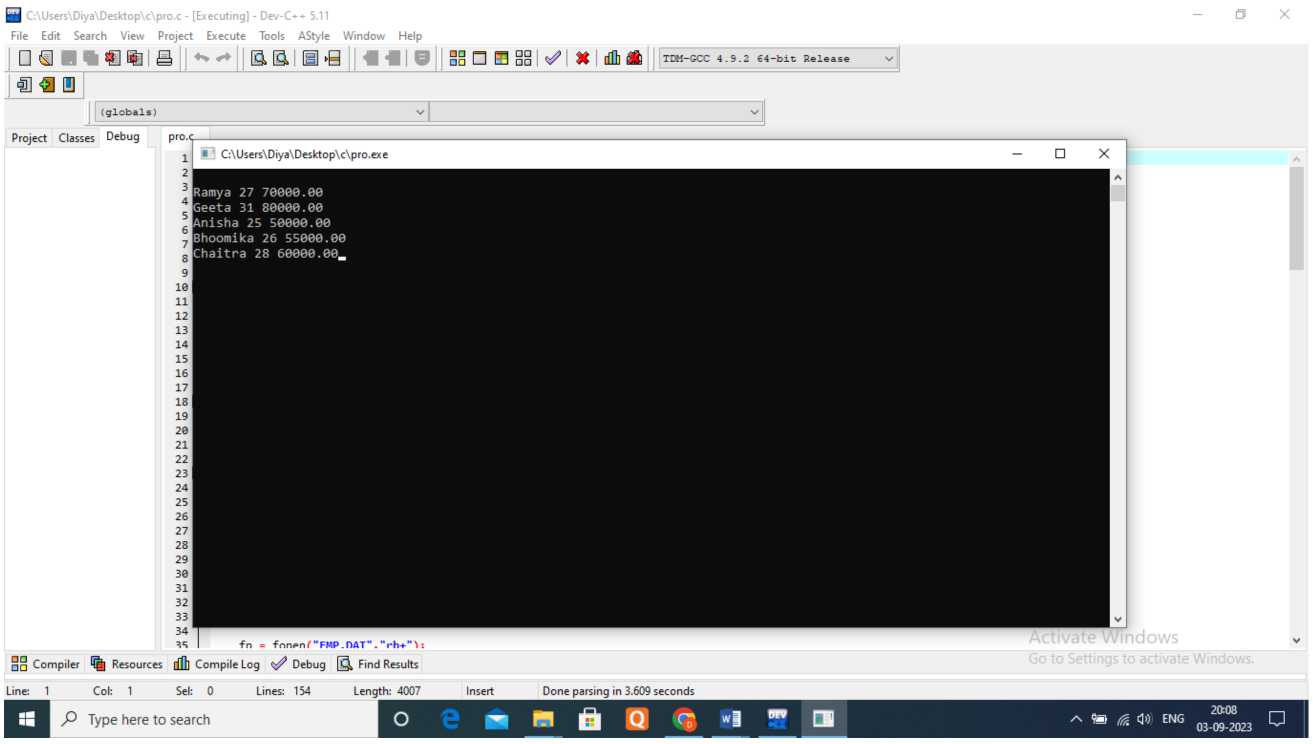
```
Ramya 26 60000.00
Geeta 30 70000.00
Anisha 25 50000.00
Bhoomika 26 55000.00
Chaitra 28 60000.00
```

Modifying Record:



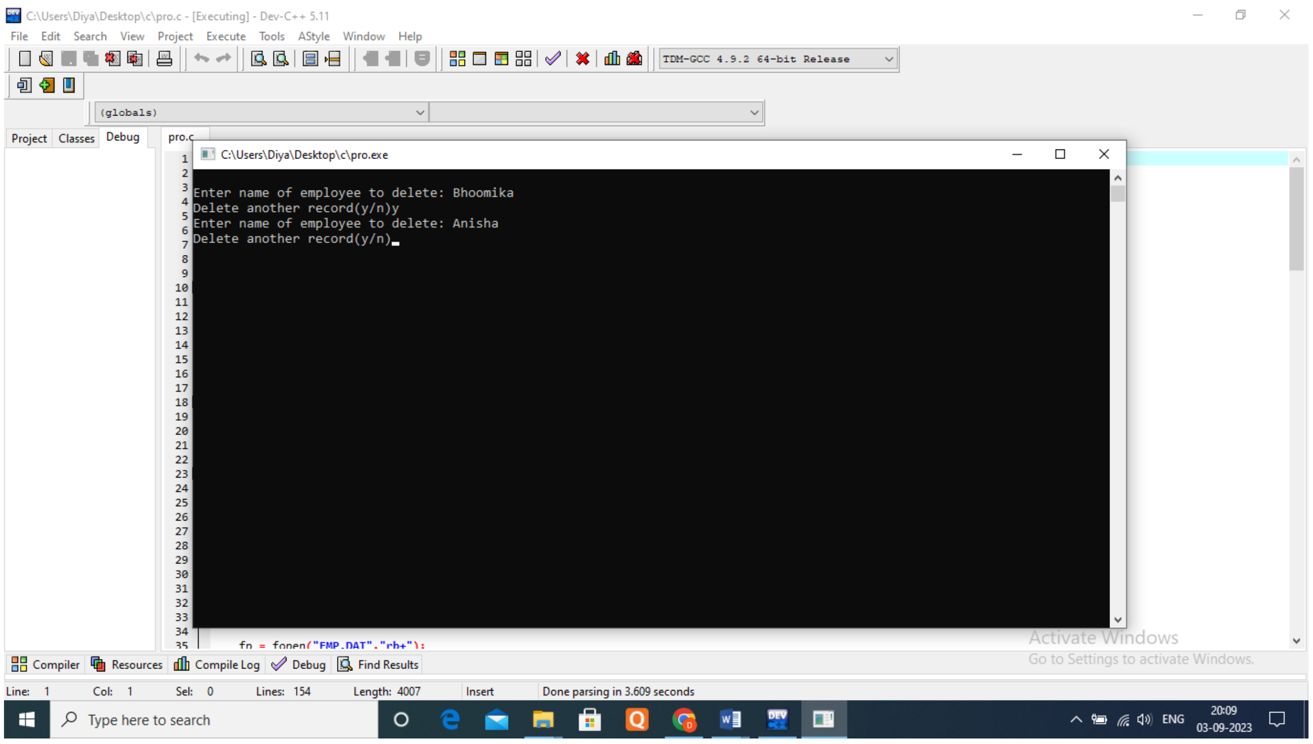
```
Enter the employee name to modify: Ramya

Enter new name,age and bs: Ramya
27
70000

Modify another record(y/n)yEnter the employee name to modify: Geeta

Enter new name,age and bs: Geeta
31
80000

Modify another record(y/n)
```

Displaying after modifying:



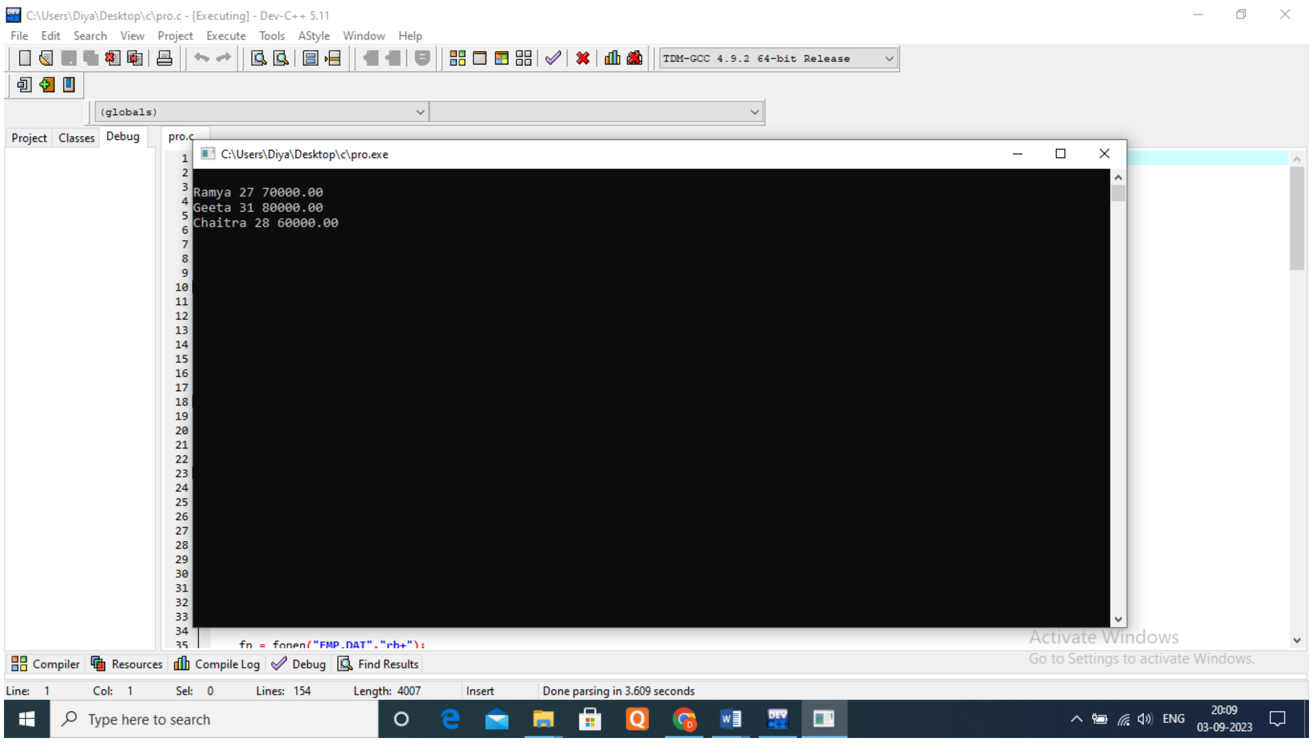
```
Ramya 27 70000.00
Geeta 31 80000.00
Anisha 25 50000.00
Bhoomika 26 55000.00
Chaitra 28 60000.00
```

Deleting Employee record:



```
Enter name of employee to delete: Bhoomika
Delete another record(y/n)y
Enter name of employee to delete: Anisha
Delete another record(y/n)
```

Displaying Record after deletion:



```
Ramya 27 70000.00
Geeta 31 80000.00
Chaitra 28 60000.00
```

C:\Users\Diya\Desktop\c\pro.c - [Executing] - Dev-C++ 5.11
C:\Users\Diya\Desktop\c\pro.exe
TDM-GCC 4.9.2 64-bit Release
(globals)
fp = fopen("EMP.DAT","rb+");
Line: 1 Col: 1 Sel: 0 Lines: 154 Length: 4007 Insert Done parsing in 3.609 seconds

CODE SNIPPET

```
#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <windows.h>

#include <string.h>


COORD coord = {0,0};


void gotoxy(int x,int y)

{

   coord.X = x;

   coord.Y = y;

   SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),coord);

}




int main()

{

   FILE *fp, *ft;

   char another, choice;


   struct emp

   {

      char name[40];

      int age;

      float bs;

   };


   struct emp e;
```

```
char empname[40];

long int recsize;

fp = fopen("EMP.DAT","rb+");
if(fp == NULL)
{
    fp = fopen("EMP.DAT","wb+");
    if(fp == NULL)
    {
        printf("Connot open file");
        exit(1);
    }
}

recsize = sizeof(e);

while(1)
{
    system("cls");
    gotoxy(30,10);
    printf("1. Add Record");
    gotoxy(30,12);
    printf("2. List Records");
    gotoxy(30,14);
    printf("3. Modify Records");
    gotoxy(30,16);
    printf("4. Delete Records");
    gotoxy(30,18);
    printf("5. Exit");
    gotoxy(30,20);
```

```
printf("Your Choice: ");
fflush(stdin);
choice  = getche();
switch(choice)
{
case '1':
    system("cls");
    fseek(fp,0,SEEK_END);

    another = 'y';
    while(another == 'y')
    {
        printf("\nEnter name: ");
        scanf("%s",e.name);
        printf("\nEnter age: ");
        scanf("%d", &e.age);
        printf("\nEnter basic salary: ");
        scanf("%f", &e.bs);

        fwrite(&e,recsize,1,fp);

        printf("\nAdd another record(y/n) ");
        fflush(stdin);
        another = getche();
    }
    break;
case '2':
    system("cls");
    rewind(fp);
    while(fread(&e,recsize,1,fp)==1)
    {
        printf("\n%s %d %.2f",e.name,e.age,e.bs);
```

```
          }
          getch();
          break;

      case '3':
          system("cls");
          another = 'y';
          while(another == 'y')
          {
              printf("Enter the employee name to modify: ");
              scanf("%s", empname);
              rewind(fp);
              while(fread(&e,recsize,1,fp)==1)
              {
                  if(strcmp(e.name,empname) == 0)
                  {
                      printf("\nEnter new name,age and bs: ");
                      scanf("%s%d%f",e.name,&e.age,&e.bs);
                      fseek(fp,-recsize,SEEK_CUR);
                      fwrite(&e,recsize,1,fp);
                      break;
                  }
              }
              printf("\nModify another record(y/n)");
              fflush(stdin);
              another = getche();
          }
          break;
      case '4':
          system("cls");
          another = 'y';
          while(another == 'y')
```

```
            {
                printf("\nEnter name of employee to delete: ");
                scanf("%s",empname);
                ft = fopen("Temp.dat","wb");
                rewind(fp);
                while(fread(&e,recsize,1,fp) == 1)
                {
                    if(strcmp(e.name,empname) != 0)
                    {
                        fwrite(&e,recsize,1,ft);
                    }
                }
                fclose(fp);
                fclose(ft);
                remove("EMP.DAT");
                rename("Temp.dat","EMP.DAT");
                fp = fopen("EMP.DAT", "rb+");
                printf("Delete another record(y/n)");
                fflush(stdin);
                another = getche();
            }
            break;
        case '5':
            fclose(fp);
            exit(0);
        }
    }
    return 0;
}
```