EMPLOYEE RETIREMENT BENEFIT MANAGEMENT SYSTEM

Name :	Stuti Rawat
Sap Id :	590023168
Batch no :	37
Subject :	Programming in C

ABSTRACT

The *Employee Retirement Benefit Management System* is a C-based application designed to automate the calculation and management of retirement benefits for employees. The project integrates core concepts of structured programming, file handling, basics of string, pointers, functions, and modular design to provide a reliable and user-friendly solution for HR and payroll operations.

The system allows users to add employee records, display stored data, and generate retirement reports for individual employees or the entire workforce. Each report includes detailed calculations of Provident Fund (PF), Gratuity, and the total retirement benefits, based on salary and years of service. By storing employee information in external files, the system ensures data persistence and easy retrieval, making it suitable for long-term record management.

PROBLEM DEFINITION

Managing employee retirement benefits manually is slow, error-prone, and inefficient. HR departments face difficulties in maintaining accurate records, calculating Provident Fund (PF) and gratuity, and generating reports for multiple employees. Traditional methods like paper files or spreadsheets often lead to miscalculations and data loss.

To solve this, the *Employee Retirement Benefit Management System* provides an automated, reliable solution using C programming. It securely stores employee data, performs accurate benefit calculations, and generates reports quickly, ensuring efficiency and accuracy.

SYSTEM DESIGN

ALGORITHM FOR project.h

1. Start
2. Declare struct Employee
3. Declare Employee Functions
   - addEmployee()
   - displayEmployee()
   - saveEmployee()
   - loadEmployee()
   - displayAllEmployees()

4. Declare PF Function
   - calculatePF()

5. Declare Gratuity Function

   - calculateGratuity()

6. Declare Report Functions

   - generateRetirementReport()
   - generateAllRetirementReports()

7. Stop

ALGORITHM FOR employee.c

1. Start
2. addEmployee()
    - Input employee ID, name, salary, and years of service
    - Remove newline from name string
3. saveEmployee()
    - Open file in append mode
    - Write employee record to file
    - Close file
4. loadEmployee()
    - Open file in read mode
    - Search employee by ID
    - If found, return employee record
    - Else return “not found” signal
5. displayAllEmployees()
    - Open file in read mode
    - Read all employee records
    - Call displayEmployee() for each
    - Close file
6. displayEmployee()
    - Print employee ID, name, salary, and years of service
7. Stop

ALGORITHM FOR gratutity.c

1. Start
2. Input employee details (salary, years of service)
3. Apply formula:

   g = (15 * last drawn salary * years of service) / 26
4. Store result in variable g
5. Return gratuity value
6. Stop

ALGORITHM FOR pf.c

1. Start
2. Input employee details (salary, years of service)
3. Apply formula:

   pf = (12% of monthly salary) × (years of service × 12 months)
4. Store result in variable pf
5. Return PF value
6. Stop

ALGORITHM FOR report.c

1. Start
2. generateRetirementReport()
   - Input employee details (ID, name, salary, years of service)
   - Calculate gratuity using calculateGratuity()
   - Calculate PF using calculatePF()
   - Compute total benefits = PF + Gratuity
   - Print formatted retirement report

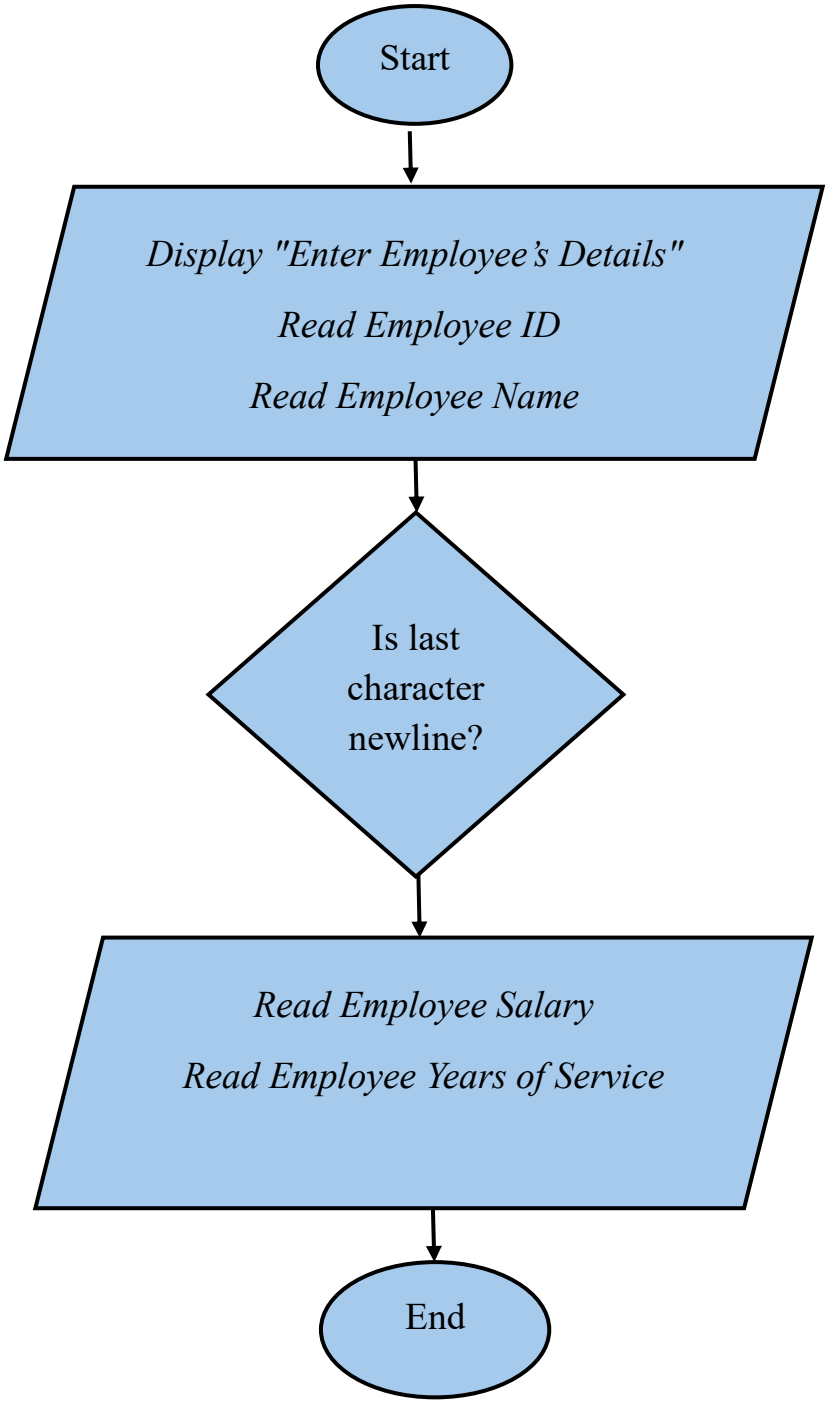
3. generateAllRetirementReports()
    - Open employee data file in read mode
    - Read each employee record sequentially
    - For each record, call generateRetirementReport()
    - Close file
4. Stop

ALGORITHM FOR main.c

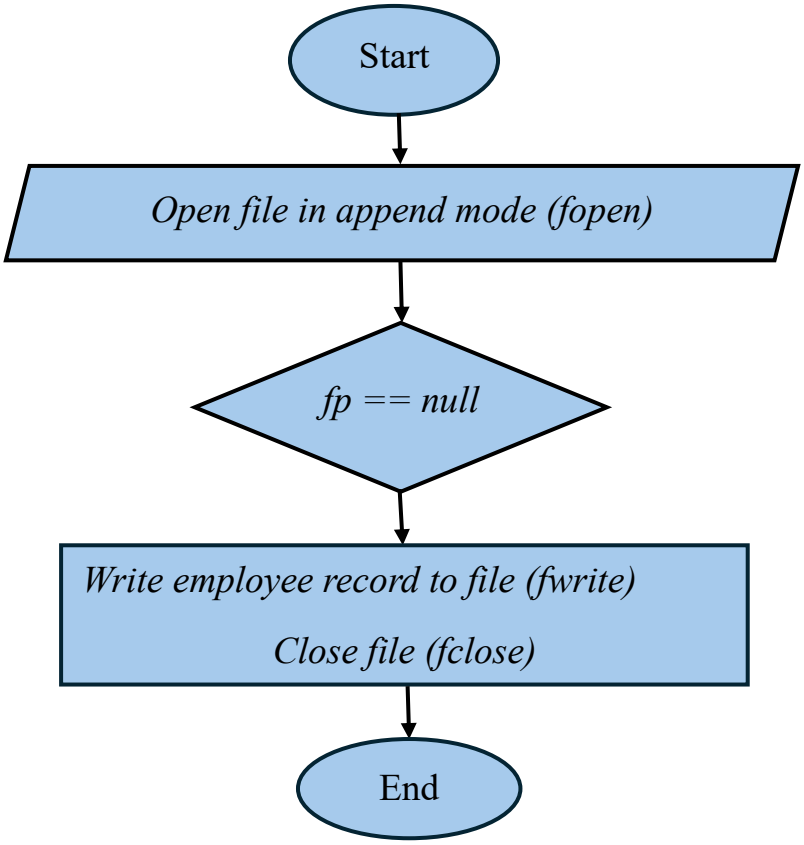
1. Start
2. Initialize variables (choice, emp, filename = docs/employees.dat)
3. Display menu with options:
    1. Add Employee
    2. Display All Employees
    3. Generate Retirement Report for One Employee
    4. Generate Retirement Reports for All Employees
    5. Exit
4. Input choice from user
5. Switch case execution:
    - Case 1 → Call addEmployee() and saveEmployee()
    - Case 2 → Call displayAllEmployees()
    - Case 3 → Input employee ID → Call loadEmployee() → Call generateRetirementReport()
    - Case 4 → Call generateAllRetirementReports()
    - Case 5 → Print exit message → Terminate program
    - Default → Print “Invalid choice”
6. Repeat menu until user exits
7. Stop

FLOWCHART

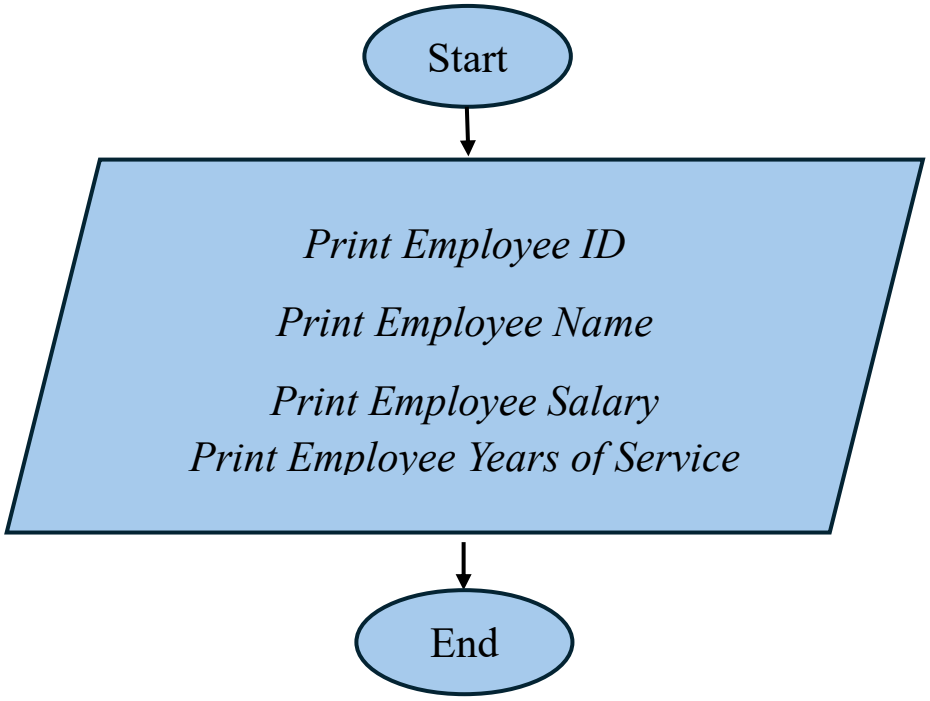
- void addEmployee()



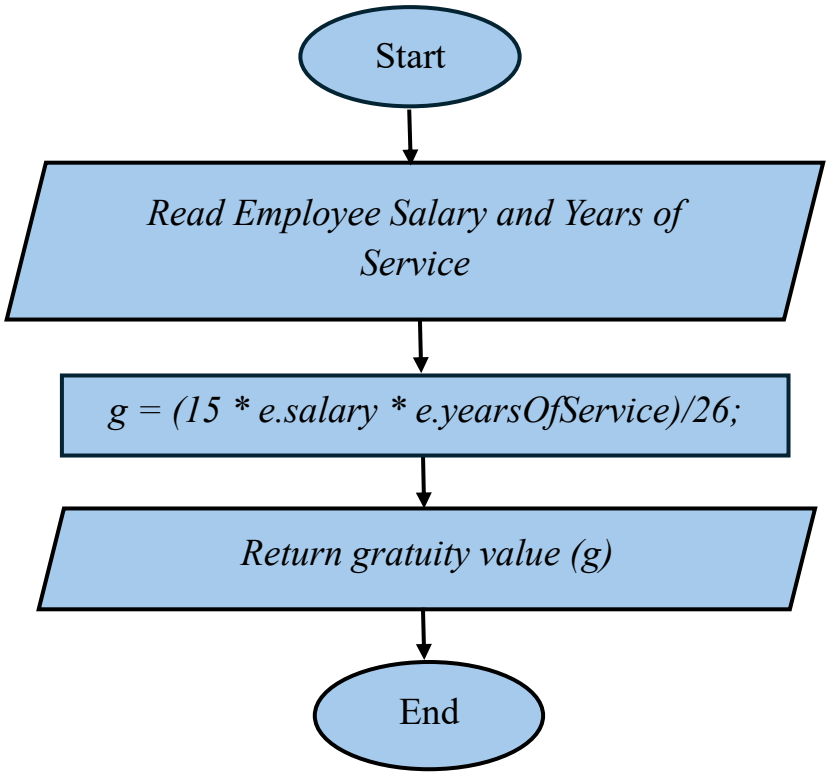
- void saveEmployee()



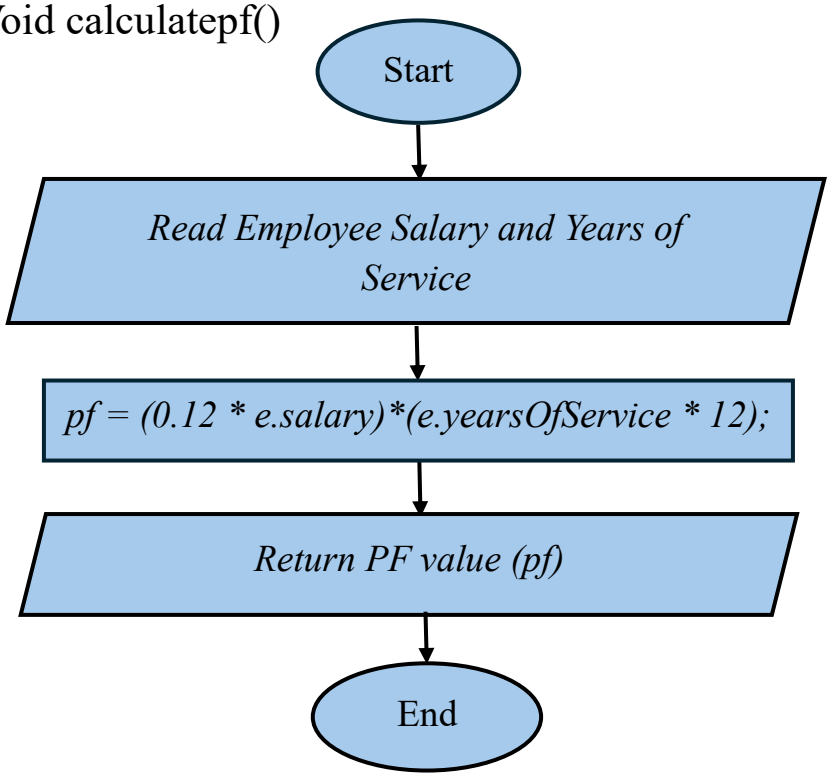
- void displayEmployee()



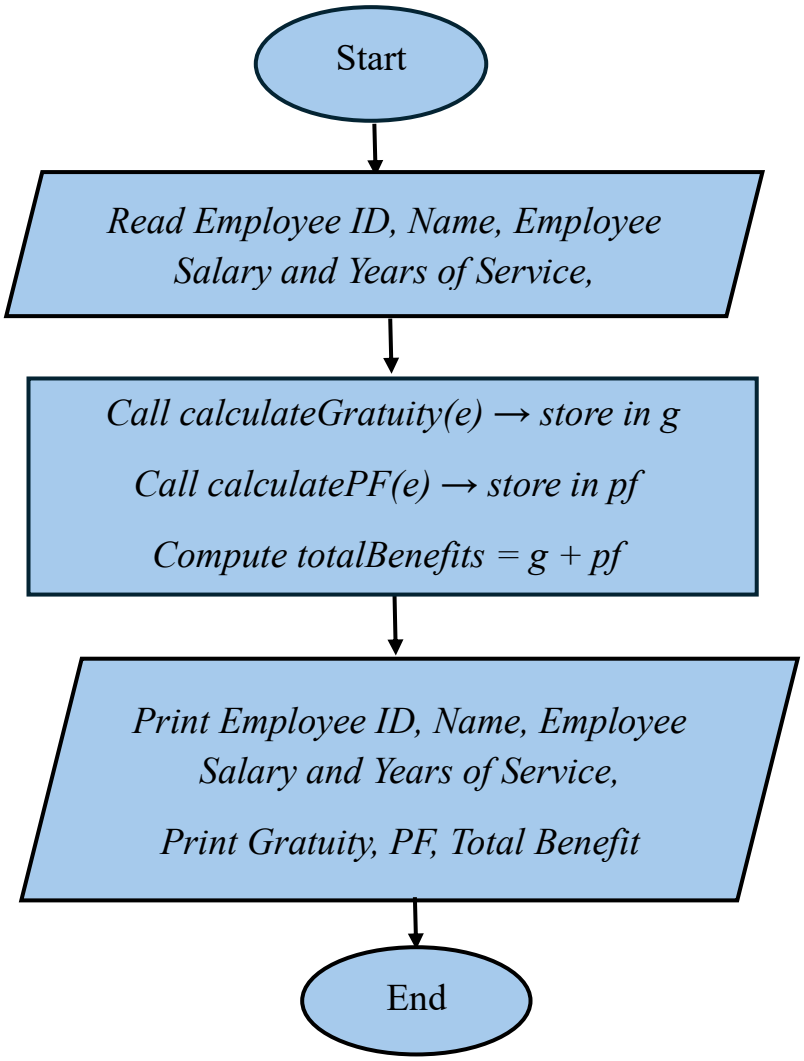
- void calculateGratuity()



- Void calculatepf()



- Void generateRetirementReport()



IMPLEMENTATION DETAILS

1. **Overall Structure :-**
   - **main.c -** It contains the core of the program which handles the menu, input and overall flow**.**
   - **project.h -** a header file where all the functions are declared.
   - **employee.**c - Implements the employee-related functions such as adding employee details, saving records to file, loading employee data by ID, displaying employee information, and generating retirement reports (PF and gratuity calculations).
   - **gratuity.**c – To calculate gratuity.
   - **pf.c –** To calculate pf (Providient Fund).
   - **Report.c –** To generate retirement reports (Employee Details, PF, Gratuity, Total Benefit).

2. **Menu :-**

```c
while (1)
{

    printf("\n===== Employee Retirement Benefit Management System =====\n\n");
    printf("1. Add Employee\n");
    printf("2. Display All Employees\n");
    printf("3. Generate Retirement Report for One Employee\n");
    printf("4. Generate Retirement Reports for All Employees\n");
    printf("5. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);
```

3. To add Employee Details :-

```c
void addEmployee(struct Employee *e)
{
    printf(" \nEnter Employee's Details: \n");
    printf("Employee ID: ");
    scanf("%d", &e->id);
    getchar();  // clear leftover newline

    printf("Employee Name: ");
    fgets(e->name, sizeof(e->name), stdin);

    // Remove trailing newline from name
    size_t len = strlen(e->name);
    if (len > 0 && e->name[len - 1] == '\n')
    {
        e->name[len - 1] = '\0';
    }

    printf("Employee Salary: ");
    scanf("%f", &e->salary);
    printf("Employee Years of Service: ");
    scanf("%d", &e->yearsOfService);
}
```

4. Gratuity Calculation :-

gratuity.c > ...

```c
// -------------------- To Calculate Gratuity ---------------------

#include "project.h"
#include <stdio.h>

float calculateGratuity(struct Employee e)
{
    /*
      Standard gratuity formula (India):
      Gratuity = (15 * last drawn salary * years of service) / 26
      - Salary here is taken as monthly basic salary
      - 26 represents the number of working days in a month
    */

    float g=1;
    g = (15 * e.salary * e.yearsOfService)/26;      // Compute Gratuity

    // Return Gratuity
    return g;

}
```

5. PF Calculation :-

```c
// pf.c > ...
// -------------------- To Calculate (PF) Providient Fund ---------------------

#include "project.h"
#include <stdio.h>

float calculatePF(struct Employee e)
{
    /*
      PF is generally calculated as 12% of basic salary per month.
      For simplicity in this project:
      PF = (12% of monthly salary) × (years of service × 12 months)
    */

    // Compute PF (Providient Fund)
    float pf = (0.12 * e.salary)*(e.yearsOfService * 12);

    // Return PF
    return pf;

}
```

6. Report Generation :-

```c
// report.c > generateRetirementReport(Employee)
// -------------------- To Display Retirement Report ---------------------

#include "project.h"
#include <stdio.h>

void generateRetirementReport(struct Employee e)
{
    float g = calculateGratuity(e);
    float pf = calculatePF(e);

    float totalBenefits = g + pf;

    printf("\n=========================================================\n");
    printf("        Retirement Report for Employee ID: %d\n", e.id);
    printf("\n Name: %s\n", e.name);
    printf(" Salary: %.2f\n", e.salary);
    printf(" Years of Service: %d\n", e.yearsOfService);
    printf("\n-----------------------------------\n\n");
    printf(" Provident Fund (PF): %.2f\n", pf);
    printf(" Gratuity: %.2f\n", g);
    printf(" Total Retirement Benefits: %.2f\n", totalBenefits);
    printf("\n=========================================================\n\n");

}
```

TESTING AND RESULTS

Case 1: Add Employee.

```
PS C:\Users\Stuti Rawat\OneDrive\Documents\Major-C-Project> .\employee_system.exe

===== Employee Retirement Benefit Management System =====

1. Add Employee
2. Display All Employees
3. Generate Retirement Report for One Employee
4. Generate Retirement Reports for All Employees
5. Exit
Enter your choice: 1

Enter Employee's Details:
Employee ID: 62452
Employee Name: Sumit Mehra
Employee Salary: 75000
Employee Years of Service: 20
```

Case 2: Display All Employees.

```
===== Employee Retirement Benefit Management System =====

1. Add Employee
2. Display All Employees
3. Generate Retirement Report for One Employee
4. Generate Retirement Reports for All Employees
5. Exit
Enter your choice: 2

===== Employee List =====

-------------------------------------
 Employee ID       : 23168
 Name              : Stuti Rawat
 Salary            : 80000.00
 Years of Service  : 20
-------------------------------------

-------------------------------------
 Employee ID       : 22651
 Name              : Lakshita Negi
 Salary            : 120000.00
 Years of Service  : 30
-------------------------------------

-------------------------------------
 Employee ID       : 43456
 Name              : Abhishek Gupta
 Salary            : 55000.00
 Years of Service  : 15
-------------------------------------
```

Case 3: Generate Retirement Report for One Employee.

```
===== Employee Retirement Benefit Management System =====

1. Add Employee
2. Display All Employees
3. Generate Retirement Report for One Employee
4. Generate Retirement Reports for All Employees
5. Exit
Enter your choice: 3
Enter Employee ID to generate report: 23168

=======================================================
        Retirement Report for Employee ID: 23168

 Name: Stuti Rawat
 Salary: 80000.00
 Years of Service: 20

-----------------------------------

 Provident Fund (PF): 2304000.00
 Gratuity: 923076.94
 Total Retirement Benefits: 3227077.00

=======================================================
```

Case 4: Generate Retirement Reports for All Employees.

```
===== Employee Retirement Benefit Management System =====

1. Add Employee
2. Display All Employees
3. Generate Retirement Report for One Employee
4. Generate Retirement Reports for All Employees
5. Exit
Enter your choice: 4

=======================================================
        Retirement Report for Employee ID: 23168

 Name: Stuti Rawat
 Salary: 80000.00
 Years of Service: 20

-----------------------------------

 Provident Fund (PF): 2304000.00
 Gratuity: 923076.94
 Total Retirement Benefits: 3227077.00

=======================================================
```

```
=========================================================
        Retirement Report for Employee ID: 22651

 Name: Lakshita Negi
 Salary: 120000.00
 Years of Service: 30

-----------------------------------

 Provident Fund (PF): 5184000.00
 Gratuity: 2076923.12
 Total Retirement Benefits: 7260923.00

=========================================================


=========================================================
        Retirement Report for Employee ID: 43456

 Name: Abhishek Gupta
 Salary: 55000.00
 Years of Service: 15

-----------------------------------

 Provident Fund (PF): 1188000.00
 Gratuity: 475961.53
 Total Retirement Benefits: 1663961.50

=========================================================


=========================================================
        Retirement Report for Employee ID: 55724

 Name: Sneha Chauhan
 Salary: 100000.00
 Years of Service: 25

-----------------------------------

 Provident Fund (PF): 3600000.00
 Gratuity: 1442307.75
 Total Retirement Benefits: 5042308.00

=========================================================
```

Case 5: Exit.

```
===== Employee Retirement Benefit Management System =====

1. Add Employee
2. Display All Employees
3. Generate Retirement Report for One Employee
4. Generate Retirement Reports for All Employees
5. Exit
Enter your choice: 5

===== Thank You For Using Employee Retirement Benefit Management System =====
```

Default Case:

```
PS C:\Users\Stuti Rawat\OneDrive\Documents\Major-C-Project> .\employee_system.exe

===== Employee Retirement Benefit Management System =====

1. Add Employee
2. Display All Employees
3. Generate Retirement Report for One Employee
4. Generate Retirement Reports for All Employees
5. Exit
Enter your choice: 6
Invalid choice. Please try again.
```

CONCLUSION

This project successfully demonstrates how **C programming** can be applied to manage employee records and calculate retirement benefits like **PF** and **Gratuity**. Using structures, file handling, and modular design, the system ensures data storage, accuracy, and easy report generation. It strengthens core programming skills while showing practical use in HR and payroll management.

"This project showcases the power of C programming in solving real-world HR problems with precision and efficiency."

REFERENCES

1. Let Us C - Yashavant Kanetkar
2. Class PPTs
3. Learn-C.org
4. Github