Employee Payroll System

Report

(Employee Payroll System Using C Programming)

Name : Akash Ani

Roll Number : 04

Branch : CSE

Class : s3 CSE-B

Date : 15-07-2024

Introduction

Brief Overview of the Project

The Employee Payroll System is a C programming project designed to manage employee details, calculate salaries, handle allowances and deductions, and generate pay slips. The system also includes functionalities to save data to a file and read from it, ensuring data persistence.

Problem Statement

Manual payroll management is time-consuming and prone to errors. Automating the payroll process can significantly enhance accuracy and efficiency, reducing the workload on HR personnel.

Objective

To develop a C program that automates the payroll management process, including adding employee details, calculating salaries, generating pay slips, and saving/loading data from a file.

System Requirements

Hardware Requirements

- **A computer with at least 1 GB of RAM**
- **A processor with a minimum speed of 1 GHz**
- **Sufficient storage to save employee data files**

Software Requirements

- **An operating system (Windows, Linux, or macOS)**
- **A C compiler (GCC, Clang, or similar)**
- **A text editor or IDE (Code::Blocks, Visual Studio Code, etc.)**

Design and Development

Description of the Program Logic

The program uses a struct to store employee details, including ID, name, base salary, allowances, deductions, and net salary. The main menu offers options to add an employee, generate pay slips, display all employees, and save and exit. Data is stored in a binary file to ensure persistence.

Key Functions:

1. **addEmp: Adds a new employee to the system.**
2. **calcSal: Calculates the net salary of an employee.**
3. **genPaySlip: Generates and displays a pay slip for a given employee.**
4. **dispEmps: Displays details of all employees.**
5. **saveEmpsToFile: Saves employee data to a file.**
6. **loadEmpsFromFile: Loads employee data from a file.**

Flowchart or Pseudocode

Pseudocode:

```
START
Initialize employee array and count
Load employees from file

WHILE true
   Display menu
   Get user choice
```

```
  IF choice == 1
    Add employee
  ELSE IF choice == 2
    Generate pay slip for given employee ID
  ELSE IF choice == 3
    Display all employees
  ELSE IF choice == 4
    Save employees to file
    Exit program
  ELSE
    Display invalid choice message
  ENDIF

  Clear input buffer
ENDWHILE

STOP
```

Testing and Results

Test Cases

1. **Add Employee:**
    - **Input: Employee ID = 1, Name = "Akash", Base Salary = 1000000000, Allowances = 1000000, Deductions =1**
    - **Expected Output: Employee added successfully, net salary calculated correctly.**
2. **Generate Pay Slip:**
    - **Input: Employee ID = 1**
    - **Expected Output: Pay slip displayed with correct details.**
3. **Display All Employees:**
    - **Input: 3**
    - **Expected Output: List of all employees with correct details.**
4. **Save and Exit:**
    - **Input: 4**
    - **Expected Output: Employee data saved to file, program exits.**

Output Screenshots or Results

Add Employee:

```
Employee Payroll System
1. Add Employee
2. Generate Pay Slip
3. Display All Employees
4. Save and Exit
Enter your choice: 1
Enter employee ID: 1
Enter employee name: Akash
Enter base salary: 1000000000
Enter allowances: 1000000
Enter deductions: 1
```

Generate Pay Slip:

```
Employee Payroll System
1. Add Employee
2. Generate Pay Slip
3. Display All Employees
4. Save and Exit
Enter your choice: 2
Enter employee ID for pay slip: 1

Pay Slip for Employee ID: 1
Name: Akash
Base Salary: 1000000000.00
Allowances: 1000000.00
Allowances: 1000000.00
Deductions: 1.00
Net Salary: 1001000000.00
----------------------------
```

Display All Employees:

```
Employee Payroll System
1. Add Employee
2. Generate Pay Slip
1. Add Employee
2. Generate Pay Slip
3. Display All Employees
4. Save and Exit
Enter your choice: 3
4. Save and Exit
Enter your choice: 3
Employee ID: 1
Name: Akash
Base Salary: 1000000000.00
Base Salary: 1000000000.00
Allowances: 1000000.00
Deductions: 1.00
Net Salary: 1001000000.00
---------------------------
Employee ID: 2
Net Salary: 1001000000.00
---------------------------
Employee ID: 2
Name: Adarsh
Base Salary: 1000000.00
Allowances: 100000.00
Deductions: 9900.00
Net Salary: 1090100.00
---------------------------
```

Discussion of Results

The program performs all intended functionalities correctly. The employee data is accurately added, calculated, displayed, and saved. The file handling ensures that data is preserved between sessions, validating the persistence feature.

Conclusion

Summary of the Project

The Employee Payroll System effectively automates payroll management, reducing errors and improving efficiency. The project demonstrates the practical use of C programming for data management, file handling, and user interaction.

Future Enhancements

1. **User Authentication: Adding login features to enhance security.**
2. **GUI Interface: Developing a graphical user interface for better user experience.**
3. **Database Integration: Using a database system for more robust data storage and management.**
4. **Advanced Reporting: Generating comprehensive reports and analytics for HR management.**

References

- **Study Materials by Prof Smitha Jacob**

Appendices

Code Listing:

Final.c

Microproject > Final.c > ...

```c
#include <stdio.h>
#include <string.h>

#define MAX_EMP 100
#define EMP_FILE "empdata.dat"

typedef struct {
    int id;
    char name[50];
    float base_sal;
    float allow;
    float deduct;
    float net_sal;
} Emp;

void addEmp(Emp emps[], int *num_emp);
void calcSal(Emp *emp);
void genPaySlip(Emp emp);
void dispEmps(Emp emps[], int num_emp);
void saveEmpsToFile(Emp emps[], int num_emp);
int loadEmpsFromFile(Emp emps[]);

void addEmp(Emp emps[], int *num_emp) {
    if (*num_emp >= MAX_EMP) {
        printf("Max number of employees reached.\n");
        return;
    }

    Emp new_emp;
    printf("Enter employee ID: ");
    if (scanf("%d", &new_emp.id) != 1) {
        printf("Invalid input for employee ID.\n");
        while (getchar() != '\n'); // Clear input buffer
        return;
    }

    printf("Enter employee name: ");
```

```c
    if (scanf("%s", new_emp.name) != 1) {
        printf("Invalid input for employee name.\n");
        while (getchar() != '\n'); // Clear input buffer
        return;
    }

    printf("Enter base salary: ");
    if (scanf("%f", &new_emp.base_sal) != 1) {
        printf("Invalid input for base salary.\n");
        while (getchar() != '\n'); // Clear input buffer
        return;
    }

    printf("Enter allowances: ");
    if (scanf("%f", &new_emp.allow) != 1) {
        printf("Invalid input for allowances.\n");
        while (getchar() != '\n'); // Clear input buffer
        return;
    }

    printf("Enter deductions: ");
    if (scanf("%f", &new_emp.deduct) != 1) {
        printf("Invalid input for deductions.\n");
        while (getchar() != '\n'); // Clear input buffer
        return;
    }

    calcSal(&new_emp);
    emps[*num_emp] = new_emp;
    (*num_emp)++;
}

void calcSal(Emp *emp) {
    emp->net_sal = emp->base_sal + emp->allow - emp->deduct;
}
```

```c
void genPaySlip(Emp emp) {
    printf("\nPay Slip for Employee ID: %d\n", emp.id);
    printf("Name: %s\n", emp.name);
    printf("Base Salary: %.2f\n", emp.base_sal);
    printf("Allowances: %.2f\n", emp.allow);
    printf("Deductions: %.2f\n", emp.deduct);
    printf("Net Salary: %.2f\n", emp.net_sal);
    printf("-----------------------------\n");
}

void dispEmps(Emp emps[], int num_emp) {
    for (int i = 0; i < num_emp; i++) {
        printf("\n");
        printf("Employee ID: %d\n", emps[i].id);
        printf("Name: %s\n", emps[i].name);
        printf("Base Salary: %.2f\n", emps[i].base_sal);
        printf("Allowances: %.2f\n", emps[i].allow);
        printf("Deductions: %.2f\n", emps[i].deduct);
        printf("Net Salary: %.2f\n", emps[i].net_sal);
        printf("-----------------------------\n");
    }
}

void saveEmpsToFile(Emp emps[], int num_emp) {
    FILE *file = fopen(EMP_FILE, "wb");
    if (file == NULL) {
        perror("Error opening file for writing");
        return;
    }

    fwrite(&num_emp, sizeof(int), 1, file);
    fwrite(emps, sizeof(Emp), num_emp, file);

    fclose(file);
}

int loadEmpsFromFile(Emp emps[]) {
```

```c
    FILE *file = fopen(EMP_FILE, "rb");
    if (file == NULL) {
        // File doesn't exist, initialize to 0 employees
        return 0;
    }

    int num_emp;
    fread(&num_emp, sizeof(int), 1, file);
    fread(emps, sizeof(Emp), num_emp, file);

    fclose(file);
    return num_emp;
}

int main() {
    Emp emps[MAX_EMP];
    int num_emp = loadEmpsFromFile(emps);
    int choice;

    while (1) {
        printf("\nEmployee Payroll System\n");
        printf("1. Add Employee\n");
        printf("2. Generate Pay Slip\n");
        printf("3. Display All Employees\n");
        printf("4. Save and Exit\n");
        printf("Enter your choice: ");

        if (scanf("%d", &choice) != 1) {
            printf("Invalid input. Please enter a number.\n");
            while (getchar() != '\n'); // Clear input buffer
            continue;
        }

        switch (choice) {
            case 1:
                addEmp(emps, &num_emp);
                break;
            case 2: {
                int emp_id;
                printf("Enter employee ID for pay slip: ");
                if (scanf("%d", &emp_id) != 1) {
                    printf("Invalid input for employee ID.\n");
                    while (getchar() != '\n'); // Clear input buffer
                    continue;
                }
                int found = 0;
                for (int i = 0; i < num_emp; i++) {
                    if (emps[i].id == emp_id) {
                        genPaySlip(emps[i]);
                        found = 1;
                        break;
                    }
                }
                if (!found) {
                    printf("Employee not found.\n");
                }
                break;
            }
            case 3:
                dispEmps(emps, num_emp);
                break;
            case 4:
                saveEmpsToFile(emps, num_emp);
                printf("Employee data saved. Exiting...\n");
                return 0;
            default:
                printf("Invalid choice. Please try again.\n");
                while (getchar() != '\n'); // Clear input buffer
        }
    }
}
```

THANKYOU