K.GOWTHAM-192372078

16. Develop a C program for implementing random access file for processing the employee details.

Aim

To develop a C program to manage employee records using a random access file for adding, viewing, and modifying employee details efficiently.

Algorithm

1. **Start**
2. Define a structure for employee details with fields like ID, name, and salary.
3. Open a binary file in read/write mode.
4. Provide a menu-driven interface:
    - Add a new employee
    - Display employee details
    - Modify employee details
    - Exit
5. For each menu option:
    - **Add**: Append employee details to the file.
    - **View**: Read the file and display all records.
    - **Modify**: Locate the record by ID, update it, and rewrite it in place.
6. Close the file and end the program.

Procedure

1. Start the program and include the necessary header files.
2. Define a structure for employee details.
3. Open the binary file using `fopen()` in read/write mode.
4. Implement menu-driven functionality:
    - Use `fwrite()` for adding records.
    - Use `fread()` to display or locate records.
    - Use `fseek()` to navigate to specific records for modification.
5. Ensure proper file handling and error checking.
6. Run the program and test the menu options.

Code:

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>
```

```
struct Employee {
    int id;
    char name[50];
    float salary;
};

void addEmployee(FILE *file) {
    struct Employee emp;
    printf("Enter ID: ");
    scanf("%d", &emp.id);
    printf("Enter Name: ");
    scanf("%s", emp.name);
    printf("Enter Salary: ");
    scanf("%f", &emp.salary);
    fseek(file, 0, SEEK_END);
    fwrite(&emp, sizeof(emp), 1, file);
}

void displayEmployees(FILE *file) {
    struct Employee emp;
    rewind(file);
    while (fread(&emp, sizeof(emp), 1, file)) {
```

```
        printf("ID: %d, Name: %s, Salary: %.2f\n", emp.id, emp.name, emp.salary);
    }
}

void modifyEmployee(FILE *file) {
    struct Employee emp;
    int id, found = 0;
    printf("Enter ID to modify: ");
    scanf("%d", &id);
    rewind(file);
    while (fread(&emp, sizeof(emp), 1, file)) {
        if (emp.id == id) {
            found = 1;
            printf("Enter New Name: ");
            scanf("%s", emp.name);
            printf("Enter New Salary: ");
            scanf("%f", &emp.salary);
            fseek(file, -sizeof(emp), SEEK_CUR);
            fwrite(&emp, sizeof(emp), 1, file);
            break;
        }
    }
    if (!found) {
```

```
        printf("Employee with ID %d not found.\n", id);
    }
}

int main() {
    FILE *file = fopen("employees.dat", "rb+");
    if (!file) {
        file = fopen("employees.dat", "wb+");
        if (!file) {
            printf("Error opening file.\n");
            return 1;
        }
    }

    int choice;
    while (1) {
        printf("\n1. Add Employee\n2. Display Employees\n3. Modify Employee\n4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: addEmployee(file); break;
            case 2: displayEmployees(file); break;
```

```
        case 3: modifyEmployee(file); break;

        case 4: fclose(file); return 0;

        default: printf("Invalid choice.\n");

      }

    }

}
```

Result

The program successfully implements a random access file for employee details. It allows adding new employee records, displaying all records, and modifying existing records based on their unique ID.

Output: