NAME: G. GANESH

REG NO: 192373008

EXERICSE-16

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

Aim:

To develop a C program to implement random access file processing for managing employee details.

Algorithm:

1. DefineEmployeeStructure:
   Create a structure to store employee details (ID, name, designation, and salary).
2. MenuOptions:
   Provide options to:
   - Add employee records.
   - View all employee records.
   - Update an employee's details using their ID.
   - Search for an employee by ID.
   - Exit the program.
3. Random Access Operations:
   - AddEmployee:
     Append employee records to the file.
   - ViewEmployees:
     Read and display all records sequentially.
   - UpdateEmployee:
     Locate the record by ID, update the details, and rewrite the record at the specific file position.
   - SearchEmployee:
     Locate and display an employee's details using their ID.
4. ExitProgram:
   End when the user selects the exit option.

Procedure:

1. Open the file in appropriate modes (rb+ for update and ab for appending).
2. Use fseek() to navigate to specific positions for reading, writing, or updating records.
3. Use a loop to display all records or locate a specific record by ID.

4. Implement functions for each menu option to simplify code management.

Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct {
    int id;
    char name[50];
    char designation[50];
    float salary;
} Employee;
void add_employee(FILE *file) {
    Employee emp;
    printf("Enter Employee ID: ");
    scanf("%d", &emp.id);
    printf("Enter Employee Name: ");
    scanf("%s", emp.name);
    printf("Enter Employee Designation: ");
    scanf("%s", emp.designation);
    printf("Enter Employee Salary: ");
    scanf("%f", &emp.salary);
    fseek(file, 0, SEEK_END);
    fwrite(&emp, sizeof(Employee), 1, file);
    printf("Employee added successfully.\n");
}

void view_employees(FILE *file) {
    Employee emp;
    rewind(file);
```

```
    printf("\nEmployee Records:\n");
    while (fread(&emp, sizeof(Employee), 1, file)) {
        printf("ID: %d, Name: %s, Designation: %s, Salary: %.2f\n", emp.id, emp.name, emp.designation, emp.salary);
    }
}
void update_employee(FILE *file) {
    int id, found = 0;
    Employee emp;
    printf("Enter Employee ID to update: ");
    scanf("%d", &id);
    rewind(file);
    while (fread(&emp, sizeof(Employee), 1, file)) {
        if (emp.id == id) {
            found = 1;
            printf("Enter New Name: ");
            scanf("%s", emp.name);
            printf("Enter New Designation: ");
            scanf("%s", emp.designation);
            printf("Enter New Salary: ");
            scanf("%f", &emp.salary);
            fseek(file, -sizeof(Employee), SEEK_CUR);
            fwrite(&emp, sizeof(Employee), 1, file);
            printf("Employee updated successfully.\n");
            break;
        }
    }
    if (!found) {
        printf("Employee with ID %d not found.\n", id);
    }
}
```

```
void search_employee(FILE *file) {
    int id, found = 0;
    Employee emp;
    printf("Enter Employee ID to search: ");
    scanf("%d", &id);
    rewind(file);
    while (fread(&emp, sizeof(Employee), 1, file)) {
        if (emp.id == id) {
            found = 1;
            printf("ID: %d, Name: %s, Designation: %s, Salary: %.2f\n", emp.id, emp.name, emp.designation, emp.salary);
            break;
        }
    }
    if (!found) {
        printf("Employee with ID %d not found.\n", id);
    }
}
int main() {
    FILE *file = fopen("employees.dat", "rb+");
    if (file == NULL) {
        file = fopen("employees.dat", "wb+");
        if (file == NULL) {
            printf("Error opening file.\n");
            return 1;
        }
    }
    int choice;
    do {
        printf("\nEmployee Management System\n");
        printf("1. Add Employee\n");
```

```
        printf("2. View Employees\n");
        printf("3. Update Employee\n");
        printf("4. Search Employee\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                add_employee(file);
                break;
            case 2:
                view_employees(file);
                break;
            case 3:
                update_employee(file);
                break;
            case 4:
                search_employee(file);
                break;
            case 5:
                printf("Exiting program.\n");
                break;
            default:
                printf("Invalid choice! Try again.\n");
        }
    } while (choice != 5);

    fclose(file);
    return 0;
}
```

Result:

The program implements random access file processing to add, view, update, and search employee records successfully.

Output: