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

details.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_NAME\_LENGTH 100

#define FILE\_NAME "employees.dat"

typedef struct {

int id;

char name[MAX\_NAME\_LENGTH];

float salary;

} Employee;

void addEmployee();

void displayEmployee(int id);

void updateEmployee(int id);

void deleteEmployee(int id);

void handleError(const char \*message);

int main() {

int choice, id;

while (1) {

printf("\nEmployee Management System\n");

printf("1. Add Employee\n");

printf("2. Display Employee\n");

printf("3. Update Employee\n");

printf("4. Delete Employee\n");

printf("5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

addEmployee();

break;

case 2:

printf("Enter Employee ID to display: ");

scanf("%d", &id);

displayEmployee(id);

break;

case 3:

printf("Enter Employee ID to update: ");

scanf("%d", &id);

updateEmployee(id);

break;

case 4:

printf("Enter Employee ID to delete: ");

scanf("%d", &id);

deleteEmployee(id);

break;

case 5:

exit(0);

default:

printf("Invalid choice. Please try again.\n");

}

}

return 0;

}

void addEmployee() {

FILE \*file = fopen(FILE\_NAME, "ab");

if (!file) {

handleError("Unable to open file for writing.");

}

Employee emp;

printf("Enter Employee ID: ");

scanf("%d", &emp.id);

printf("Enter Employee Name: ");

getchar(); // to consume newline character

fgets(emp.name, MAX\_NAME\_LENGTH, stdin);

emp.name[strcspn(emp.name, "\n")] = 0; // remove newline character

printf("Enter Employee Salary: ");

scanf("%f", &emp.salary);

fwrite(&emp, sizeof(Employee), 1, file);

fclose(file);

printf("Employee added successfully.\n");

}

void displayEmployee(int id) {

FILE \*file = fopen(FILE\_NAME, "rb");

if (!file) {

handleError("Unable to open file for reading.");

}

Employee emp;

while (fread(&emp, sizeof(Employee), 1, file)) {

if (emp.id == id) {

printf("Employee ID: %d\n", emp.id);

printf("Employee Name: %s\n", emp.name);

printf("Employee Salary: %.2f\n", emp.salary);

fclose(file);

return;

}

}

fclose(file);

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

}

void updateEmployee(int id) {

FILE \*file = fopen(FILE\_NAME, "r+b");

if (!file) {

handleError("Unable to open file for updating.");

}

Employee emp;

int found = 0;

while (fread(&emp, sizeof(Employee), 1, file)) {

if (emp.id == id) {

found = 1;

printf("Updating Employee ID: %d\n", emp.id);

printf("Enter new Employee Name: ");

getchar(); // to consume newline character

fgets(emp.name, MAX\_NAME\_LENGTH, stdin);

emp.name[strcspn(emp.name, "\n")] = 0; // remove newline character

printf("Enter new Employee Salary: ");

scanf("%f", &emp.salary);

fseek(file, -sizeof(Employee), SEEK\_CUR);

fwrite(&emp, sizeof(Employee), 1, file);

printf("Employee updated successfully.\n");

break;

}

}

fclose(file);

if (!found) {

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

}

}

void deleteEmployee(int id) {

FILE \*file = fopen(FILE\_NAME, "rb");

if (!file) {

handleError("Unable to open file for reading.");

}

FILE \*tempFile = fopen("temp.dat", "wb");

if (!tempFile) {

fclose(file);

handleError("Unable to open temporary file for writing.");

}

Employee emp;

int found = 0;

while (fread(&emp, sizeof(Employee), 1, file)) {

if (emp.id == id) {

found = 1;

printf("Deleting Employee ID: %d\n", emp.id);

continue; // skip writing this employee

}

fwrite(&emp, sizeof(Employee), 1, tempFile);

}

fclose(file);

fclose(tempFile);

remove(FILE\_NAME);

rename("temp.dat", FILE\_NAME);

if (found) {

printf("Employee deleted successfully.\n");

} else {

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

}

}

void handleError(const char \*message) {

perror(message);

exit(EXIT\_FAILURE);

}

