**Assignment no 12  
Title: Index Sequential files**

**CODE:**

#include <iostream>

#include <fstream>

#include <cstring>

using namespace std;

class Emp {

int empid;

char name[20];

char design[20];

float salary;

public:

Emp() {

empid = 0;

strcpy(name, "");

strcpy(design, "");

salary = 0.0;

}

int acceptempid() {

return empid;

}

void getdata() {

cout << "\nEnter Emp details:";

cout << "\nEnter emp id: ";

cin >> empid;

cout << "Enter emp name: ";

cin >> name;

cout << "Enter Designation: ";

cin >> design;

cout << "Enter salary: ";

cin >> salary;

}

void display() {

cout << "\nEmp ID: " << empid;

cout << "\nName: " << name;

cout << "\nDesignation: " << design;

cout << "\nSalary: " << salary << "\n";

}

};

class fileop {

public:

void insert();

void show();

void search(int);

int deleterecord(int);

int append(int);

};

void fileop::insert() {

Emp obj;

obj.getdata();

ofstream fout("data.txt", ios::binary | ios::app);

if (!fout) {

cout << "Error opening file for writing.\n";

return;

}

fout.write((char\*)&obj, sizeof(obj));

fout.close();

}

void fileop::show() {

Emp obj;

ifstream fin("data.txt", ios::binary);

if (!fin) {

cout << "Error opening file for reading.\n";

return;

}

bool found = false;

while (fin.read((char\*)&obj, sizeof(obj))) {

obj.display();

found = true;

}

if (!found)

cout << "No records found.\n";

fin.close();

}

void fileop::search(int rno) {

Emp obj;

ifstream fin("data.txt", ios::binary);

if (!fin) {

cout << "Error opening file for reading.\n";

return;

}

bool found = false;

while (fin.read((char\*)&obj, sizeof(obj))) {

if (obj.acceptempid() == rno) {

cout << "\nEmployee Found:";

obj.display();

found = true;

break;

}

}

if (!found)

cout << "Employee not found.\n";

fin.close();

}

int fileop::deleterecord(int rno) {

Emp obj;

ifstream fin("data.txt", ios::binary);

ofstream fout("temp.txt", ios::binary);

if (!fin || !fout) {

cout << "Error opening file.\n";

return 0;

}

bool found = false;

while (fin.read((char\*)&obj, sizeof(obj))) {

if (obj.acceptempid() == rno) {

found = true; // skip writing this one

} else {

fout.write((char\*)&obj, sizeof(obj));

}

}

fin.close();

fout.close();

remove("data.txt");

rename("temp.txt", "data.txt");

if (found) {

cout << "Record deleted successfully.\n";

return 1;

} else {

cout << "Record not found.\n";

return 0;

}

}

int fileop::append(int rno) {

Emp obj;

fstream fs("data.txt", ios::in | ios::out | ios::binary);

if (!fs) {

cout << "Error opening file for updating.\n";

return 0;

}

bool found = false;

while (fs.read((char\*)&obj, sizeof(obj))) {

if (obj.acceptempid() == rno) {

cout << "Current Record:\n";

obj.display();

cout << "\nEnter new details for this employee:\n";

obj.getdata();

// Move the write pointer back to the beginning of this record

fs.seekp((int)fs.tellg() - sizeof(obj), ios::beg);

fs.write((char\*)&obj, sizeof(obj));

found = true;

break;

}

}

fs.close();

if (found) {

cout << "Record updated successfully.\n";

return 1;

} else {

cout << "Record not found.\n";

return 0;

}

}

int main() {

fileop fobj;

char ch = 'y';

int choice, n;

do {

cout << "\n\n======== Main Menu ========";

cout << "\n1. Insert Employee";

cout << "\n2. Display All Employees";

cout << "\n3. Search Employee";

cout << "\n4. Delete Employee";

cout << "\n5. Update Employee";

cout << "\n6. Exit";

cout << "\n============================";

cout << "\nEnter your choice: ";

cin >> choice;

switch (choice) {

case 1:

fobj.insert();

break;

case 2:

fobj.show();

break;

case 3:

cout << "Enter emp id to search: ";

cin >> n;

fobj.search(n);

break;

case 4:

cout << "Enter emp id to delete: ";

cin >> n;

fobj.deleterecord(n);

break;

case 5:

cout << "Enter emp id to update: ";

cin >> n;

fobj.append(n);

break;

case 6:

cout << "Exiting program.\n";

return 0;

default:

cout << "Invalid choice. Try again.\n";

}

cout << "\nDo you want to continue? (y/n): ";

cin >> ch;

} while (ch == 'y' || ch == 'Y');

return 0;

}  
  
output:

