t

NAME: HAMNA WAQAS

REG NO: 014

FINAL LAB TASK

#include <iostream>

#include <string>

using namespace std;

struct Employee {

string name;

int age;

float salary;

int id;

};

const int MAX\_EMPLOYEES = 10;

void addEmployee(Employee employees[], int& count) {

if (count >= MAX\_EMPLOYEES) {

cout << "Employee list is full!" << endl;

return;

}

cout << "Enter Employee Name: ";

cin >> employees[count].name;

cout << "Enter Employee Age: ";

cin >> employees[count].age;

cout << "Enter Employee Salary: ";

cin >> employees[count].salary;

cout << "Enter Employee ID: ";

cin >> employees[count].id;

count++;

}

void displayEmployees(const Employee employees[], int count) {

if (count == 0) {

cout << "No employee data available." << endl;

return;

}

for (int i = 0; i < count; i++) {

cout << "Employee ID: " << employees[i].id << endl;

cout << "Name: " << employees[i].name << endl;

cout << "Age: " << employees[i].age << endl;

cout << "Salary: " << employees[i].salary << endl;

cout << "------------------------------" << endl;

}

}

int searchEmployeeById(const Employee employees[], int count, int id) {S

for (int i = 0; i < count; i++) {

if (employees[i].id == id) {

return i;

}

}

return -1;

}

void updateEmployeeId(Employee employees[], int count, int oldId, int newId) {

int index = searchEmployeeById(employees, count, oldId);

if (index != -1) {

employees[index].id = newId;

cout << "Employee ID updated successfully." << endl;

} else {

cout << "Employee not found." << endl;

}

}

void deleteEmployeeById(Employee employees[], int& count, int id) {

int index = searchEmployeeById(employees, count, id);

if (index != -1) {

for (int i = index; i < count - 1; i++) {

employees[i] = employees[i + 1];

}

count--;

cout << "Employee deleted successfully." << endl;

} else {

cout << "Employee not found." << endl;

}

}

int main() {

Employee employees[MAX\_EMPLOYEES];

int count = 0;

int choice;

do {

cout << "\nEmployee Management System\n";

cout << "1. Add Employee\n";

cout << "2. Display All Employees\n";

cout << "3. Search Employee by ID\n";

cout << "4. Update Employee ID\n";

cout << "5. Delete Employee by ID\n";

cout << "6. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch(choice) {

case 1:

addEmployee(employees, count);

break;

case 2:

displayEmployees(employees, count);

break;

case 3:

{

int searchId;

cout << "Enter Employee ID to search: ";

cin >> searchId;

int index = searchEmployeeById(employees, count, searchId);

if (index != -1) {

cout << "Employee found: " << employees[index].name << endl;

} else {

cout << "Employee not found." << endl;

}

}

break;

case 4:

{

int oldId, newId;

cout << "Enter current Employee ID: ";

cin >> oldId;

cout << "Enter new Employee ID: ";

cin >> newId;

updateEmployeeId(employees, count, oldId, newId);

}

break;

case 5:

{

int deleteId;

cout << "Enter Employee ID to delete: ";

cin >> deleteId;

deleteEmployeeById(employees, count, deleteId);

}

break;

case 6:

cout << "Exiting program." << endl;

break;

default:

cout << "Invalid choice, please try again." << endl;

}

} while (choice != 6);

return 0;

}

output

Employee Management System

1. Add Employee

2. Display All Employees

3. Search Employee by ID

4. Update Employee ID

5. Delete Employee by ID

6. Exit

Enter your choice: 1

Enter Employee Name: hamna

Enter Employee Age: 18

Enter Employee Salary: 20000

Enter Employee ID: 2

Employee Management System

1. Add Employee

2. Display All Employees

3. Search Employee by ID

4. Update Employee ID

5. Delete Employee by ID

6. Exit

Enter your choice: 0

Invalid choice, please try again.