**PROJECT CODE**

#include <iostream>

#include <string>

using namespace std;

// Define the node struct for the single linked list

struct Node {

string name;

string phone;

Node\* next;

};

// Define the linked list class

class LinkedList {

private:

Node\* head;

public:

// Constructor to initialize an empty linked list

LinkedList() {

head = NULL;

}

// Function to insert a new entry at the end of the list

void addRecord(string name, string phone) {

Node\* newNode = new Node;

newNode->name = name;

newNode->phone = phone;

newNode->next = NULL;

if (head == NULL) {

head = newNode;

}

else {

Node\* current = head;

while (current->next != NULL) {

current = current->next;

}

current->next = newNode;

}

}

// Function to update an entry by name

void updateRecord(string name, string phone) {

Node\* current = head;

while (current != NULL && current->name != name) {

current = current->next;

}

if (current != NULL) {

// Update the phone number of the current entry

current->phone = phone;

}

}

// Function to search for an entry by name

string searchRecord(string name) {

Node\* current = head;

while (current != NULL && current->name != name) {

current = current->next;

}

if (current != NULL) {

return "\t Phone Number : " + current->phone;

}

else {

return "\t Record Not Found ";

}

}

// Function to delete an entry by name

void delRecord(string name) {

if (head == NULL) {

return;

}

if (head->name == name) {

Node\* temp = head;

head = head->next;

delete temp;

return;

}

Node\* current = head;

Node\* predecessor = NULL;

while (current != NULL && current->name != name) {

predecessor = current;

current = current->next;

}

if (current == NULL) {

return;

}

predecessor->next = current->next;

delete current;

}

void displayAllRecords() {

Node\* current = head;

while (current != NULL) {

cout << "Name : " << current->name << " | ";

cout << "Phone Number : " << current->phone << endl;

current = current->next;

}

}

};

int main() {

// Create an empty linked list

LinkedList directory;

int choice = 0;

string name, n;

while (choice != 6) {

cout << "\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\t \t Telephone Directory \n";

cout << "\t Press 1 to Add a New Record in Directory " << endl;

cout << "\t Press 2 to Display all Records " << endl;

cout << "\t Press 3 to Update a Record " << endl;

cout << "\t Press 4 to Delete a Record " << endl;

cout << "\t Press 5 to Search a Record " << endl;

cout << "\t Press 6 to Exit the Directory " << endl;

cout << "\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cin >> choice;

switch (choice)

{

case 1:

system("cls");

cout << "\t----Adding a New Record----\n " << endl;

cout << " Enter Name : ", cin >> name;

cout << " Enter Phone Number : ", cin >> n;

directory.addRecord(name, n);

cout << "\n\t----Record Added Successfully----" << endl;

break;

case 2:

system("cls");

cout << "\t Displaying all Records :- \n\n";

directory.displayAllRecords();

break;

case 3:

system("cls");

cout << "Enter Name of Record that is to be updated ";

cin >> name;

cout << "Enter New Phone Number ";

cin >> n;

directory.updateRecord(name, n);

break;

case 4:

system("cls");

cout << "Enter Name of Record that is to be Deleted ";

cin >> name;

directory.delRecord(name);

cout << "\t----Record Deleted Successfully----\n";

break;

case 5:

system("cls");

cout << "\t----Searching Records----\n";

cout << "\t Enter Name of Record that is to be Searched : ";

cin >> name;

cout << directory.searchRecord(name);

cout << " " << endl;

break;

case 6:

cout << "\t----Exit----\n";

break;

default:

break;

}

}

return 0;

}