

**Project Data Structures**

**Submitted To: Sir Adnan Rafique**

**Submitted By: Hamza Murtaza**

**Faisal Aslam**

**Roll numbers: (04R,15R)**

**Session : 2020-2024**

**Subject Data Structures & Algorithms**

**University of Poonch Department of CS&IT**

**Q No. 1. Write a code (C++) to implement a phone directory application using a linked list.**

**Description:**

This project can demonstrate the working of contact book applications and also teach about data structures like arrays, linked lists, stacks, and queues. Typically, phone book management encompasses **searching, sorting, inserting, and deleting** operations. A distinctive feature of the search queries here is that the user sees suggestions from the contact list after entering each character.

**CODE:**

#include <iostream>

#include <string>

using namespace std;

// Define a structure for a contact in the phone directory

struct Contact {

string name;

string phoneNumber;

Contact\* next;

};

// Define a class for the phone directory

class PhoneDirectory

{ private:

Contact\* head;

public:

PhoneDirectory() {

head = NULL;

}

// Insert a new contact in the directory

void insertContact(string name, string phoneNumber) {

Contact\* newContact = new Contact();

newContact->name = name;

newContact->phoneNumber = phoneNumber;

newContact->next = NULL;

if (head == NULL) {

head = newContact;

return;

}

Contact\* current = head;

while (current->next != NULL) {

current = current->next;

}

current->next = newContact;

}

// Display all contacts in the directory

void displayContacts() {

if (head == NULL) {

cout << "Phone directory is empty." << endl;

return;

}

Contact\* current = head;

cout << "Phone directory: " << endl;

while (current != NULL) {

cout << "Contact Name: " << current->name << "\n Phone Number: " << current->phoneNumber << endl; current = current->next;

}

}

void searchContact(string name) {

Contact\* current = head;

bool found = false;

string suggestion = "";

while (current != NULL) {

if (current->name.find(name) == 0) {

found = true;

suggestion = current->name;

cout << suggestion <<endl;

}

current = current->next;

}

if (!found) {

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

}

else if (suggestion == name) {

cout << "Exact match found." << endl;

}

}

// Delete a contact from the directory

void deleteContact(string name) { if (head == NULL) {

cout << "Phone directory is empty." << endl; return;

}

if (head->name == name) {

Contact\* temp = head;

head = head->next;

delete temp;

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

return;

}

Contact\* current = head; while (current->next != NULL) {

if (current->next->name == name) {

Contact\* temp = current->next;

current->next = current->next->next;

delete temp;

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

return;

}

current = current->next;

}

cout << "Contact not found in phone directory." << endl;

}

};

int main() {

PhoneDirectory phoneDir;

cout<<"\*\*\*\*\*\*\*\*Welcome to the Phone directory\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

while (true) {

cout << "Please select what do you want:" << endl;

cout << "1. Add a contact Number" << endl;

cout << "2. Display all Contacts Numbers" << endl;

cout << "3. Search for a Contact Number" << endl;

cout << "4. Delete a Contact Number" << endl;

cout << "5. Quit" << endl;

int option;

cin >> option;

if (option == 1) {

cout << "Enter name: ";

string name;

cin >> name;

cout << "Enter phone number: ";

string phoneNumber;

cin >> phoneNumber;

phoneDir.insertContact(name, phoneNumber);

} else if (option == 2) {

phoneDir.displayContacts(); }

else if (option == 3) {

cout << "Enter name to search: ";

string name; cin >> name;

phoneDir.searchContact(name);

} else if (option == 4) {

cout << "Enter name to delete: ";

string name;

cin >> name;

phoneDir.deleteContact(name); }

else if (option == 5) {

break; } else

{

cout << "Invalid option. Please try again." << endl;

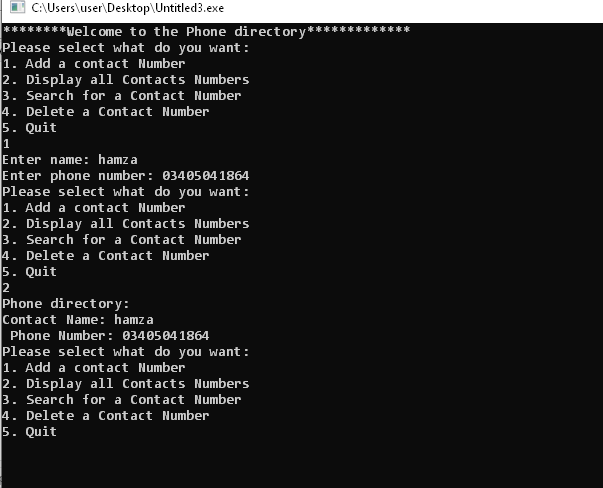
}

}

return 0;

}

**OUTPUT**



**THE END**