B.Tech Program Third Year

Course: Design and Analysis of Algorithms

Course Code: CS3102

## 

## **Phonebook Directory**

by

**Divyaant Kumar Jain**

**Samarth Agarwal**

**Pankhuri**

Under the guidance

of

**Mr. Ajit Noonua**

**Department of Computer Science and Engineering**

**School of Computer Science**

**Faculty of Engineering**

**Manipal University Jaipur, India**

Nov 2022

**Acknowledgment**

**Index**

**Introduction**

This project is developed using the general need required by the user while using the phone directory book. In order to keep the phone book directory updated, it will have various operations such as add phone records, display records, search any particular record, delete record, modify existing record etc.

**Software and Hardware Requirements**

# Software Requirements: -

# Operating System - Windows 7/10

# Language – C++

# Compiler – VS Code

# Hardware Requirements: -

# Processor – Intel Pentium 3.0 Ghz or higher

# RAM – 512 Mb or more

# Hard Disk – 10GB or more

**Code**

#include <iostream>

#include <stdlib.h>

#include <string>

using namespace std;

struct node

{

string name, number;

node \*next;

};

node \*head = NULL, \*newnode, \*temp;

int len = 0;

void add\_contact()

{

newnode = new node;

cout << " Enter Name ";

cin >> newnode->name;

cout << " Enter number ";

cin >> newnode->number;

cout << "\n";

newnode->next = NULL;

if (head == NULL)

{

head = newnode;

temp = newnode;

}

else

{

temp->next = newnode;

temp = newnode;

}

}

void display()

{

if (head == NULL)

{

cout << " Contact list is Empty " << endl;

}

else

{

temp = head;

while (temp != NULL)

{

cout << "\n\tFull Name: " << temp->name << endl;

cout << "\tPhone Number: " << temp->number << endl;

cout << "\n";

temp = temp->next;

len++;

}

cout << "Total contacts in the list = " << len << endl;

cout << "\n";

}

}

void search\_contact()

{

node \*search\_node = head;

string srch;

int count = 0;

cout << "Enter your desired contact you want to search: ";

cin >> srch;

bool found = true;

if (head == NULL)

{

cout << "\nList is Empty " << endl;

}

else

{

while (search\_node != NULL)

{

if (srch == search\_node->name || srch == search\_node->number)

{

cout << "\n\tFull Name: " << search\_node->name << endl;

cout << "\tPhone Number: " << search\_node->number << endl;

found = true;

break;

}

search\_node = search\_node->next;

count++;

}

}

if (found == true)

{

cout << "\tIndex of Contact = " << count << endl;

cout << "\n";

}

else

{

cout << "Desired contact not found" << endl;

}

}

void at\_given()

{

int pos;

node \*next\_node;

temp = head;

cout << "Enter your desired position from where you want to delete contact" << endl;

cin >> pos;

if (head == NULL)

{

cout << "List is empty" << endl;

}

else if (pos > len)

{

cout << "Invalid Position" << endl;

}

else if (pos == 0)

{

temp = head;

head = head->next;

delete temp;

cout << "Contact has been deleted" << endl;

}

else

{

for (int i = 1; i < pos; i++)

{

temp = temp->next;

}

next\_node = temp->next;

temp->next = next\_node->next;

delete next\_node;

cout << "Contact has been deleted" << endl;

}

}

void clear\_all()

{

if (head == NULL)

{

cout << "List is Empty" << endl;

}

else

{

temp = head;

while (head != NULL)

{

head = head->next;

delete temp;

}

cout << "All Contact list has been deleted" << endl;

}

}

void EditContacts()

{

node \*temp = head;

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

cout << " Press 1 if you want to Edit By Name" << endl;

cout << " Press 2 if you want to Edit By Number" << endl;

int Ecommand;

cout << " Enter the Command: ";

cin >> Ecommand;

string x, y;

if (Ecommand == 1)

{

bool Echeck = false;

cout << " Enter the Name to Edit: ";

cin >> x;

while (temp != NULL)

{

if (temp->name == x)

{

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

cout << "Name: " << temp->name << endl;

cout << "Phone Number: " << temp->number << endl;

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

Echeck = true;

break;

}

temp = temp->next;

}

if (Echeck == true)

{

int command;

cout << " Press 1 to Edit the Contact: ";

cin >> command;

if (command == 1)

{

cout << " Enter New Name: ";

cin >> x;

cout << " Enter New Number: ";

cin >> y;

temp->name = x;

temp->number = y;

cout << " Contact Edited Success Fully" << endl;

}

else

{

cout << " You Enter Wrong Command ... Try Again" << endl;

}

}

else if (Echeck == false)

{

cout << " Contact Not Found" << endl;

}

}

else if (Ecommand == 2)

{

bool Echeck = false;

cout << " Enter the Number to Edit: ";

cin >> y;

while (temp != NULL)

{

if (temp->number == y)

{

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

cout << "Name: " << temp->name << endl;

cout << "Phone Number: " << temp->number << endl;

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

Echeck = true;

break;

}

temp = temp->next;

}

if (Echeck == true)

{

int command;

cout << " Press 1 to Edit the Contact: ";

cin >> command;

if (command == 1)

{

cout << " Enter New Name: ";

cin >> x;

cout << " Enter New Number: ";

cin >> y;

temp->name = x;

temp->number = y;

cout << " Contact Edited Success Fully" << endl;

}

else

{

cout << " You Enter Wrong Command" << endl;

}

}

else if (Echeck == false)

{

cout << " There is No Contact of this Number." << endl;

}

}

else

{

cout << " You Enter Wrong Command ... Try Again" << endl;

}

}

void menu()

{

cout << "Enter 1 to add contact " << endl;

cout << "Enter 2 to display all contact" << endl;

cout << "Enter 3 to search contact" << endl;

cout << "Enter 4 to delete contact" << endl;

cout << "Enter 5 to clear ALL records" << endl;

cout << "Enter 6 to Edit Contact" << endl;

}

int main()

{

int op;

while (true)

{

menu();

cin >> op;

switch (op)

{

case 1:

cout << "\n";

add\_contact();

break;

case 2:

cout << "\n";

len = 0;

display();

break;

case 3:

cout << "\n";

search\_contact();

break;

case 4:

cout << "\n";

at\_given();

break;

case 5:

clear\_all();

break;

case 6:

EditContacts();

break;

default:

cout << "Invalid Option" << endl;

}

}

}

**Output**

**Conclusion**

**Future Prospects**

**Bibliography**