**JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY**

**NOIDA-SEC.62**

Logo

Description automatically generated

**(BATCH: B10)**

**PHONE DIRECTORY MANAGEMENT APPLICATION**

**ADITYA JAMWAL**

**20103283**

**SRIJAN GUPTA**

**20103286**

**UTKARSH PATHAK**

**20103289**

**Submitted to: Dr. Vikas**

**INTRODUCTION**

Our project demonstrates the working of contact number applications and also teach about various data structures. Typically, phone directory management application consists of searching, sorting, inserting and deleting operations.

**OBJECTIVES**

The main objective of our project was to develop an application firm that facilitates optimised data maintenance and its retrieval and perform searching, sorting, inserting and deleting operations in an igneous way and developing a user-friendly application that requires minimal user training.

The information contained in the telephone directory records would be name, phone no. and email of the person whose record is entered in the telephone directory application project. So, the simple file handling cumulative telephone directory operations that users can perform include:

* Add records
* Show records
* Modify/update records
* Search records
* Sort Records
* Delete records
* Remove duplicate records

**Data Structures and Techniques used:**

Linked Lists

* Doubly Linked Lists
* Arrays

**Insertion Operation**:

* Time Complexity in between insertion = O(n) time.
* Time Complexity end insertion = O(1) time.
* Space complexity = O(1)

**Search Operation**:

* Linear Search used

Time complexity for unsorted linked list = O(n) time. Time complexity for sorted linked list = O(1) time.

**Deletion Operation:**

* All deletion operations run with time complexity = O(1).
* Space complexity = O(1)

**CODE**

#include<bits/stdc++.h>

#include<windows.h>

using namespace std;

class dnode

{

public:

char number[50];

char gmail[40];

char name[30];

dnode \*prev,\*next;

dnode(char n[],char r[],char g[])

{

strcpy(name,n);

strcpy(number,r);

strcpy(gmail,g);

next=NULL;

prev=NULL;

}

friend class dlist;

};

class dlist

{

dnode \*head,\*temp,\*ptr;

dnode \*ptr1, \*ptr2, \*dup;

public:

dnode \*prevn;

void insert();

void sort();

void deletecontact(char n[20]);

void deletesamenumber();

void deletesamename();

void deletesamegmail();

void searchbyname(char p[20]);

void searchbynumber(char no[30]);

void searchbygmail(char g[20]);

void accept();

void display();

void update(char ch[10]);

dlist()

{

head=NULL;

temp=NULL;

ptr=NULL;

ptr1=NULL;

ptr2=NULL;

dup=NULL;

}

};

void dlist::accept()

{

char number[50];

char gmail[40];

char name[30];

char ans;

do

{

cout<<"ENTER NAME :";

cin>>name;

// cin.getline (name,30);

cout<<"ENTER NUMBER :";

cin>>number;

while(strlen(number)!=10)

{

cout<<"ENTER VALID NUMBER :";

cin>>number;

}

cout<<"ENTER G-MAIL :";

cin>>gmail;

temp=new dnode(name,number,gmail);

if(head==NULL)

{

head=temp;

}

else

{

ptr=head;

while(ptr->next!=NULL)

{

ptr=ptr->next;

}

ptr->next=temp;

temp->prev=ptr;

}

cout<<"DO YOU WANT TO CONTINUE?????????";

cin>>ans;

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

}

void dlist::display()

{

ptr=head;

while(ptr!=NULL)

{

cout<<"\n\nNAME ::\t"<<ptr->name;

cout<<"\nNUMBER::\t+91-"<<ptr->number;

cout<<"\nG-MAIL::\t"<<ptr->gmail;

ptr=ptr->next;

}

}

void dlist::insert()

{

accept();

}

void dlist::sort()

{

dnode \*i,\*j;

int temp;

char n[10];

for(i=head;i->next!=NULL;i=i->next)

{

for(j=i->next;j!=NULL;j=j->next)

{

temp=strcmp(i->name,j->name);

if(temp>0)

{

strcpy(n,i->name);

strcpy(i->name,j->name);

strcpy(j->name,n);

}

}

}

}

void dlist::deletecontact(char s[20])

{

int c=0;

ptr=head;

while(ptr!=NULL)

{

if(strcmp(s,ptr->name)==0)

{

c=1;

break;

}

else

{

c=2;

}

ptr=ptr->next;

}

if(c==1 && ptr!=head && ptr->next!=NULL)

{

ptr->prev->next=ptr->next;

ptr->next->prev=ptr->prev;

delete(ptr);

cout<<"YOUR CONTACT IS SUCCESSFULLY DELETED\n\n";

}

if(ptr==head)

{

head=head->next;

head->prev=NULL;

delete(ptr);

cout<<"YOUR CONTACT IS SUCCESSFULLY DELETED\n\n";

}

if(ptr->next==NULL)

{

ptr->prev->next=NULL;

ptr->prev=NULL;

delete(ptr);

cout<<"YOUR CONTACT IS SUCCESSFULLY DELETED\n\n";

}

if(c==2)

{

cout<<"YOUR ENTERED NAME IS NOT IN THE LIST...";

}

}

void dlist::deletesamename()

{

ptr1=head;

while (ptr1 != NULL && ptr1->next != NULL)

{

ptr2 = ptr1;

while (ptr2->next != NULL)

{

if (strcmp(ptr1->name,ptr2->next->name)==0)

{

dup = ptr2->next;

ptr2->next = ptr2->next->next;

delete(dup);

}

else

{

ptr2 = ptr2->next;

}

}

ptr1 = ptr1->next;

}

}

void dlist::deletesamegmail()

{

ptr1=head;

while (ptr1 != NULL && ptr1->next != NULL)

{

ptr2 = ptr1;

while (ptr2->next != NULL)

{

if (strcmp(ptr1->gmail,ptr2->next->gmail)==0)

{

dup = ptr2->next;

ptr2->next = ptr2->next->next;

delete(dup);

}

else

{

ptr2 = ptr2->next;

}

}

ptr1 = ptr1->next;

}

}

void dlist::deletesamenumber()

{

ptr1=head;

while (ptr1 != NULL && ptr1->next != NULL)

{

ptr2 = ptr1;

while (ptr2->next != NULL)

{

if (strcmp(ptr1->number,ptr2->number)==0)

{

dup = ptr2->next;

ptr2->next = ptr2->next->next;

delete(dup);

}

else

{

ptr2 = ptr2->next;

}

}

ptr1 = ptr1->next;

}

}

void dlist::searchbyname(char na[10])

{

ptr=head;

while(ptr!=NULL)

{

if(strcmp(na,ptr->name)==0)

{

cout<<"NAME FOUND"<<endl;

cout<<"CONTACT DETAILS ARE BELOW:\n"<<endl;

cout<<"\n\nNAME ::\t"<<ptr->name;

cout<<"\nNUMBER::\t+91-"<<ptr->number;

cout<<"\nG-MAIL::\t"<<ptr->gmail;

}

ptr=ptr->next;

}

}

void dlist::searchbynumber(char num[20])

{

ptr=head;

while(ptr!=NULL)

{

if(strcmp(num,ptr->number)==0)

{

cout<<"NUMBER FOUND\n"<<endl;

cout<<"CONTACT DETAILS ARE BELOW:\n"<<endl;

cout<<"\n\nNAME ::\t"<<ptr->name;

cout<<"\nNUMBER::\t+91-"<<ptr->number;

cout<<"\nG-MAIL::\t"<<ptr->gmail;

}

ptr=ptr->next;

}

}

void dlist::searchbygmail(char gm[20])

{

ptr=head;

while(ptr!=NULL)

{

if(strcmp(gm,ptr->gmail)==0)

{

cout<<"G-MAIL FOUND\n"<<endl;

cout<<"CONTACT DETAILS ARE BELOW:\n"<<endl;

cout<<"\n\nNAME ::\t"<<ptr->name;

cout<<"\nNUMBER::\t+91-"<<ptr->number;

cout<<"\nG-MAIL::\t"<<ptr->gmail;

}

ptr=ptr->next;

}

}

void dlist::update(char n[20])

{

char ans;

int c;

ptr=head;

while(ptr!=NULL)

{

if(strcmp(n,ptr->name)==0)

{

do

{

cout<<"\n WHAT DO YOU WANT TO UPDATE? \n 1.NAME \n 2.PHONE NUMBER \n 3.G-MAIL \n";

cin>>c;

switch(c)

{

case 1:

cout<<"ENTER NEW-NAME=";

cin>>ptr->name;

break;

case 2:

cout<<"ENTER NEW PHONE-NUMBER?";

cin>>ptr->number;

while(strlen(ptr->number)!=10)

{

cout<<"ENTER VALID NUMBER :";

cin>>ptr->number;

}

break;

case 3:

cout<<"ENTER NEW G-MAIL";

cin>>ptr->gmail;

break;

}

cout<<"DO YOU WANT TO CONTINUE UPDATING?";

cin>>ans;

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

}

ptr=ptr->next;

}

}

int main()

{

system("COLOR 8F");

cout<<"\n\n\n\n\n\n\n\n\n\n\n\tPHONE BOOK.....";

Sleep(500);

system("cls");

for(int i=1;i<=100;++i)

{

Sleep(2);

system("cls");

cout<<"\n\n\n\n\n\n\n\n\n\n\n\t\t\t\tWELCOME TO PHONE BOOK...";

cout<<"\n\n\n\n\n\n\n\n\n\n\n\n\t\t\t\t\t\t\t\tLOADING "<<i<<"%";

if(i==100)

Sleep(500);

}

system("cls");

char n[20];

char nam[20];

char name[10];

char number[10];

char gmail[20];

dlist d1;

char ans;

int ch,a;

cout<<"\*\*\*\*\* PHONE BOOK \*\*\*\*\*\*\*";

cout<<"\n\nWHAT IS YOUR NAME?\n";

cin.getline(name,20);

cout<<"\n\n!!!!!!!!!!!!!!!!!!!!!!! WELCOME "<<name<<" !!!!!!!!!!!!!!!!!!!!!";

cout<<"\n\n\nLET'S CREATE OUR PHONEBOOK "<<name<<" \n\n";

d1.accept();

d1.sort();

do

{

cout<<"\n\n\n\n 1) DISPLAY YOUR PHONE BOOK \n 2) INSERT NEW CONTACT \n 3) UPDATE DETAILS ON EXISTING CONTACT \n 4) DELETE CONTACT \n 5) DELETE DUPLICATE NAME IN PHONEBOOK \n 6) DELETE DUPLICATE NUMBERS IN PHONEBOOK \n 7) DELETE DUPLICATE EMAIL IN PHONEBOOK \n 8) SEARCH \n 9) SORT \n";

cin>>ch;

switch(ch)

{

case 2:

d1.insert();

break;

case 1:

d1.display();

break;

case 9:

d1.sort();

d1.display();

break;

case 3:

cout<<"\n\nENTER THE NAME OF PERSON WHOSE DETAILS YOU WANT TO UPDATE...\n";

cin>>n;

d1.update(n);

d1.sort();

break;

case 4:

cout<<"\nENTER THE NAME YOU WANT TO DELETE FROM PHONEBOOK\n";

cin>>name;

d1.deletecontact(name);

break;

case 5:

d1.deletesamename();

d1.display();

break;

case 6:

d1.deletesamenumber();

d1.display();

break;

case 7:

d1.deletesamegmail();

d1.display();

break;

case 8:

do

{

cout<<"\n 1.SEARCH BY NAME \n 2.SEARCH BY NUMBER \n 3.SEARCH BY GMAIL";

cin>>a;

switch(a)

{

case 1:

cout<<"ENTER THE NAME TO BE SEARCHED\n";

cin>>name;

d1.searchbyname(name);

break;

case 2:

cout<<"ENTER THE NAME TO BE SEARCHED\n";

cin>>number;

d1.searchbynumber(number);

break;

case 3:

cout<<"ENTER THE NAME TO BE SEARCHED\n";

cin>>gmail;

d1.searchbygmail(gmail);

break;

default:cout<<"\nNO PROPER INPUT GIVEN.....\n";

}

cout<<"DO YOU WANT TO CONTINUE SEARCHING?????????";

cin>>ans;

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

break;

default:cout<<"\nNO PROPER INPUT GIVEN..\n";

}

cout<<"\n\nDO YOU WANT TO CONTINUE OPERATIONS?????????";

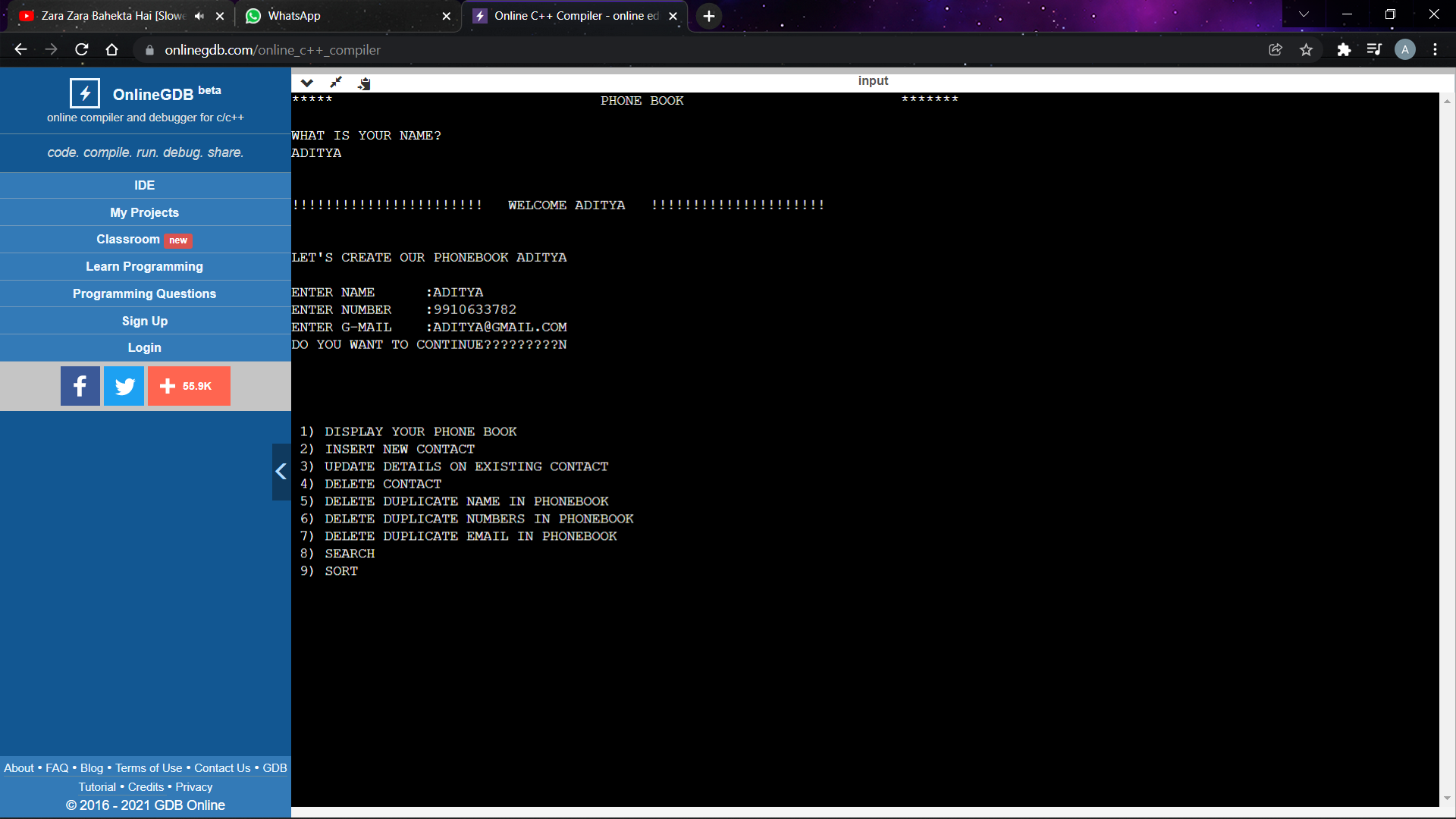
cin>>ans;

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

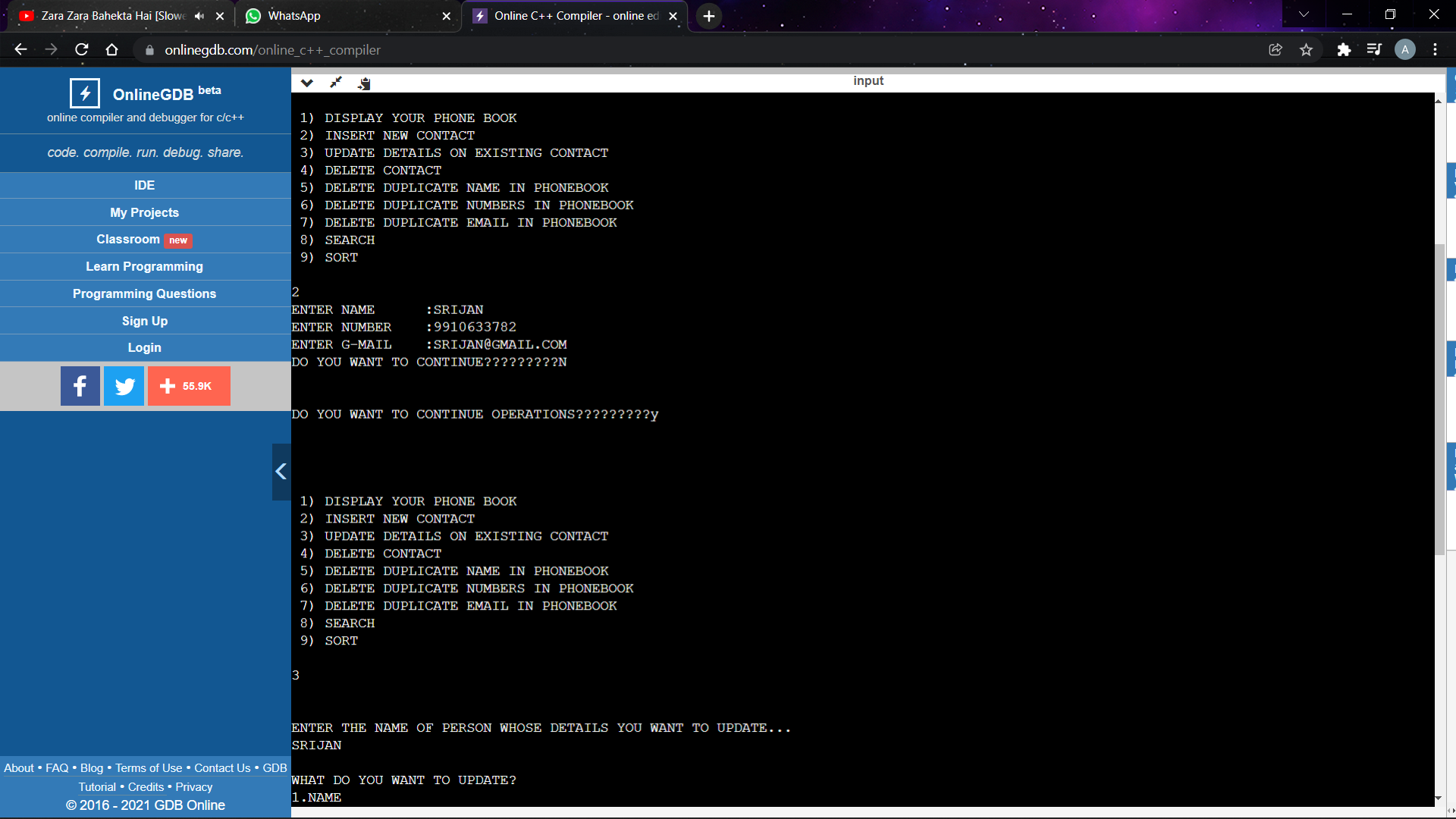
}

**OUTPUTS**

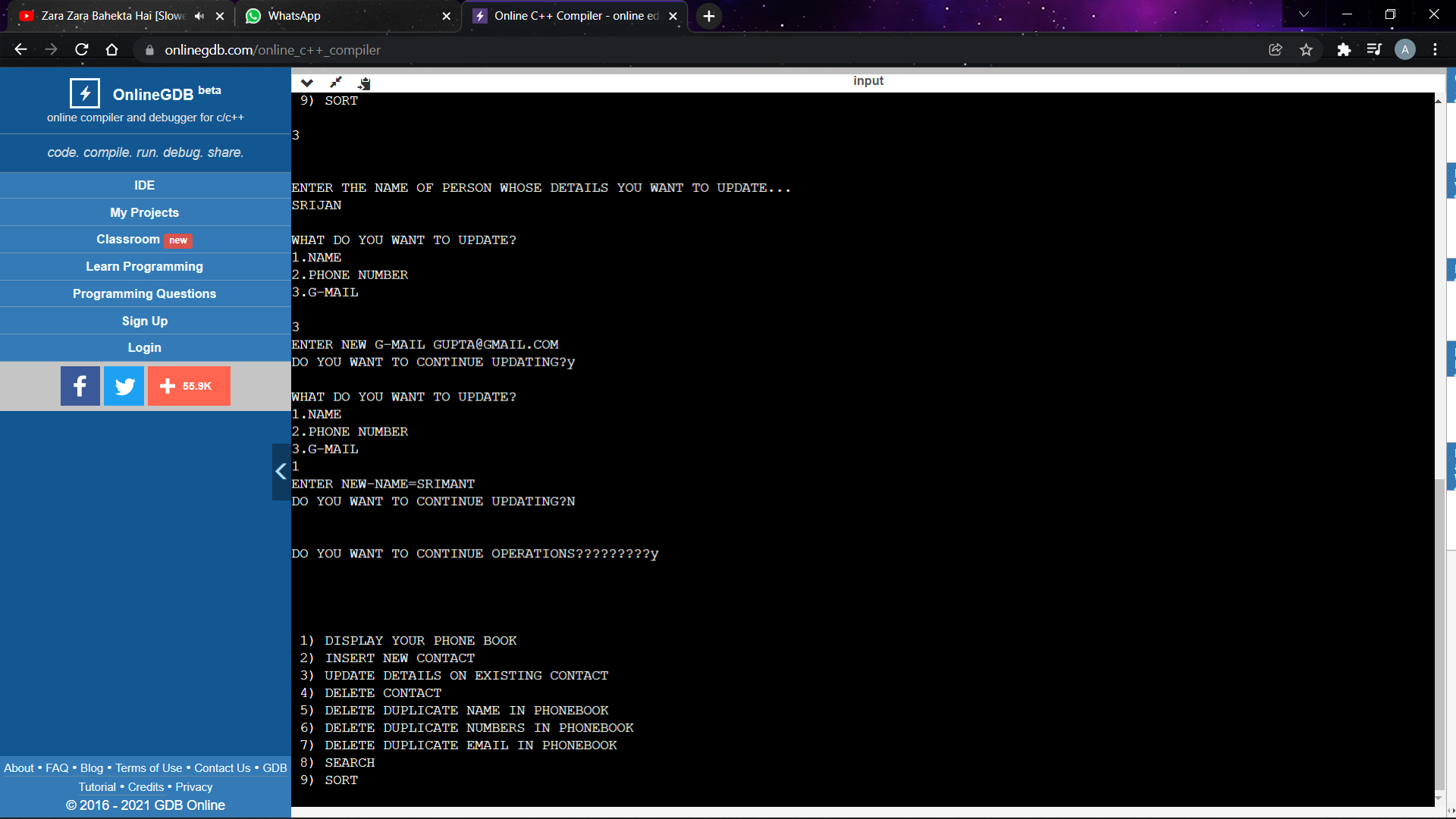
**HOME DIRECTORY PAGE (OPTIONS)**



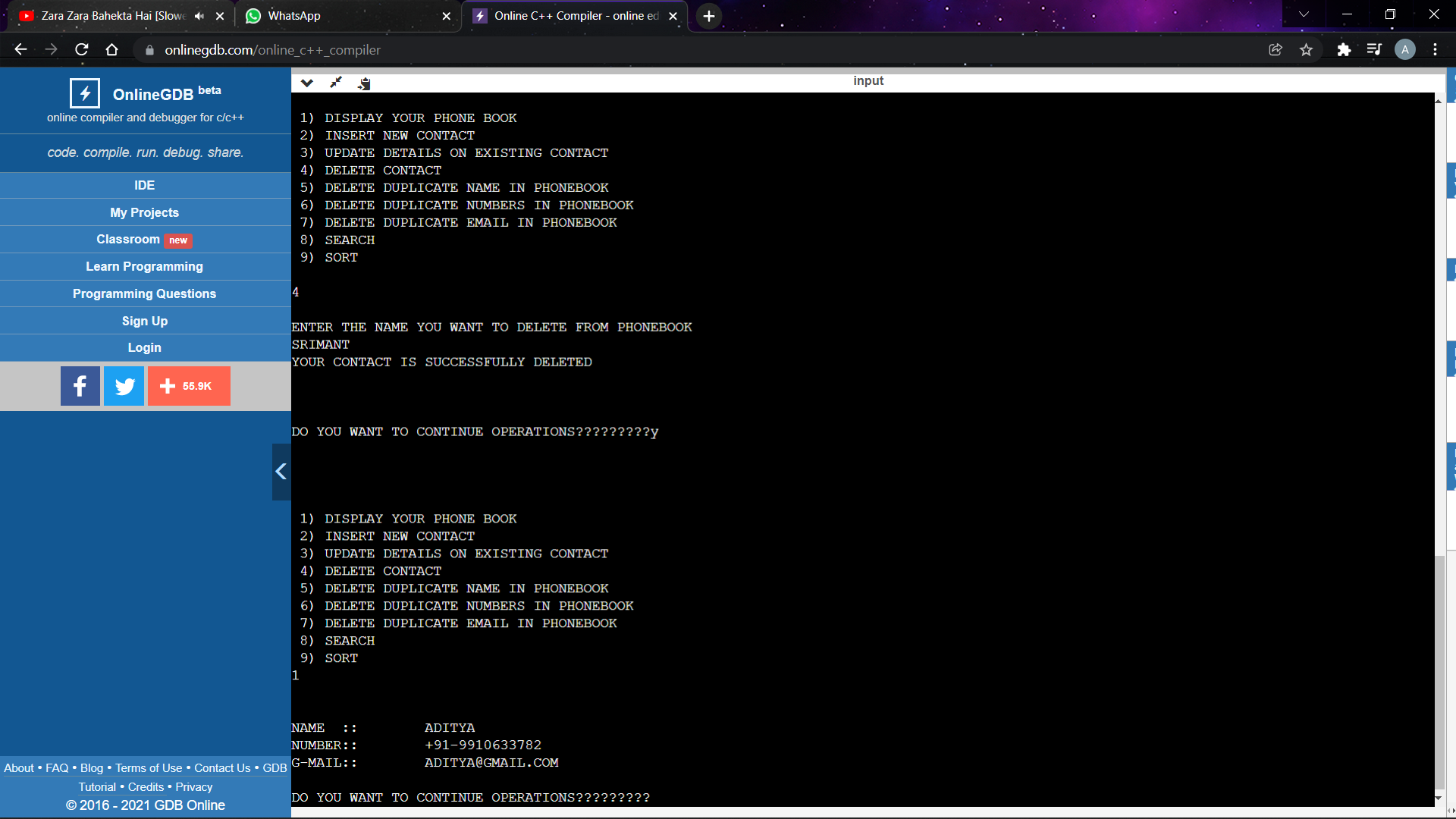
**INSERTING NEW CONTACT**



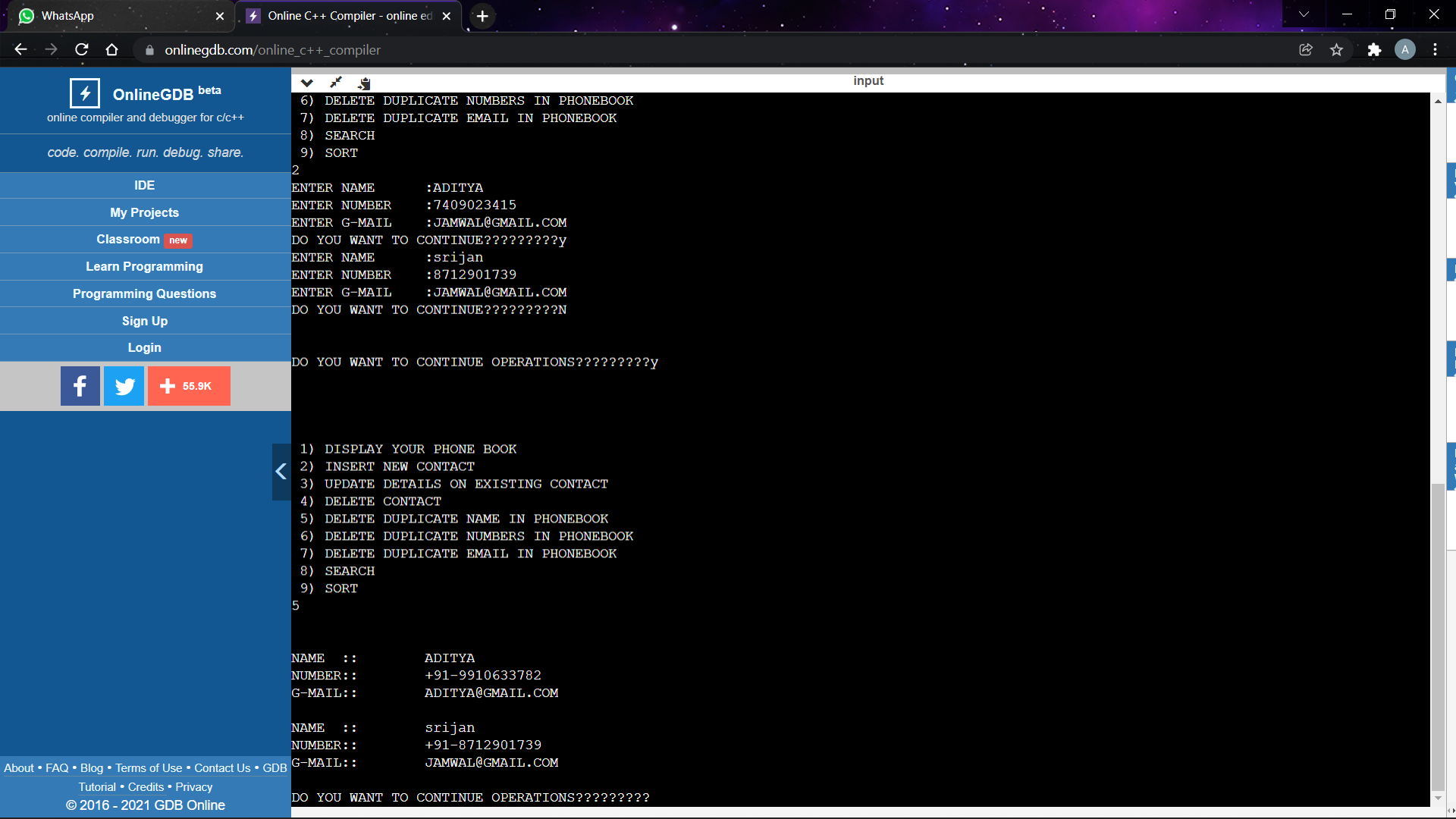
**UPDATING DETAILS (EXISTING CONTACT)**



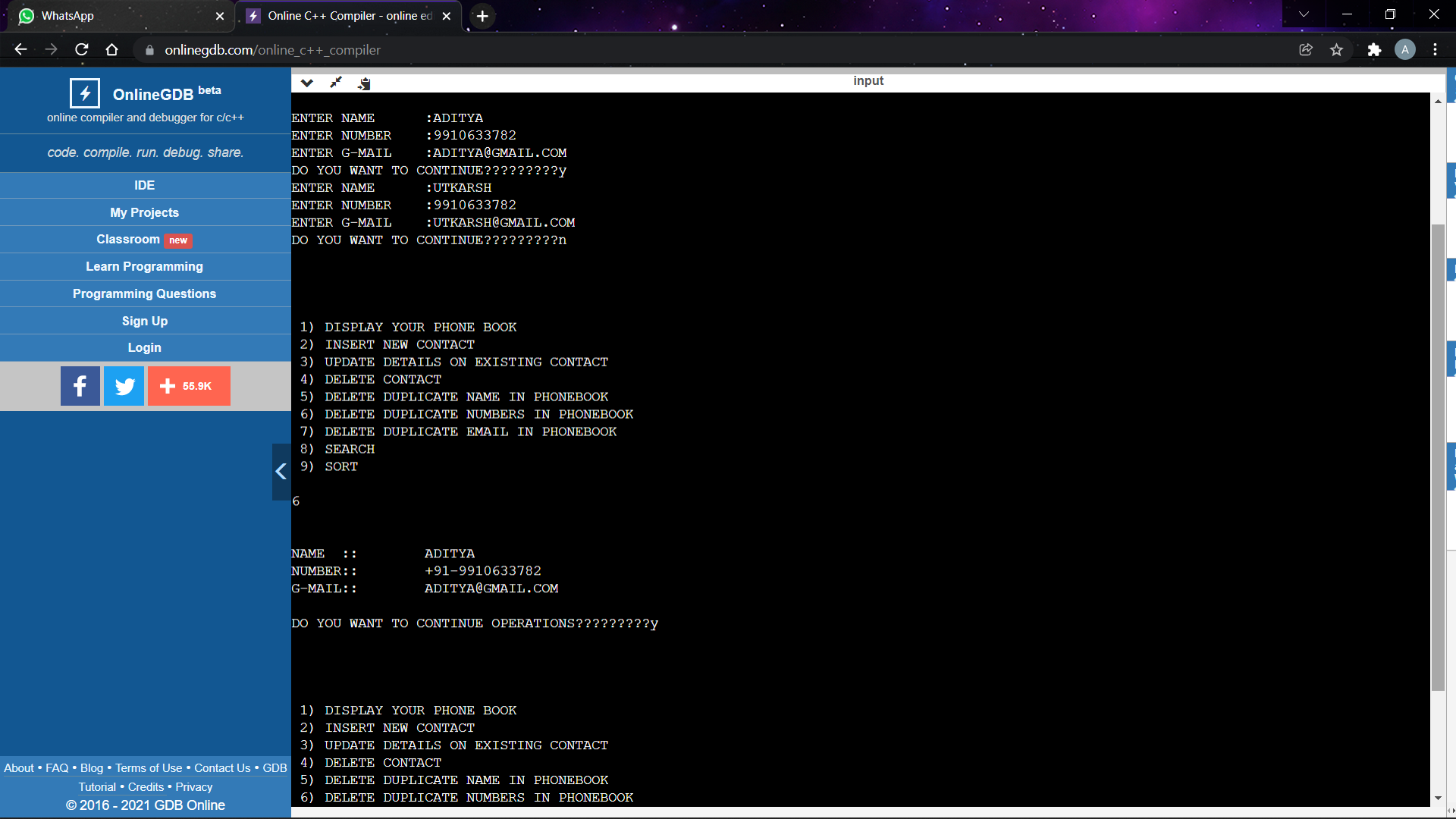
**DELETING A CONTACT BY NAME**



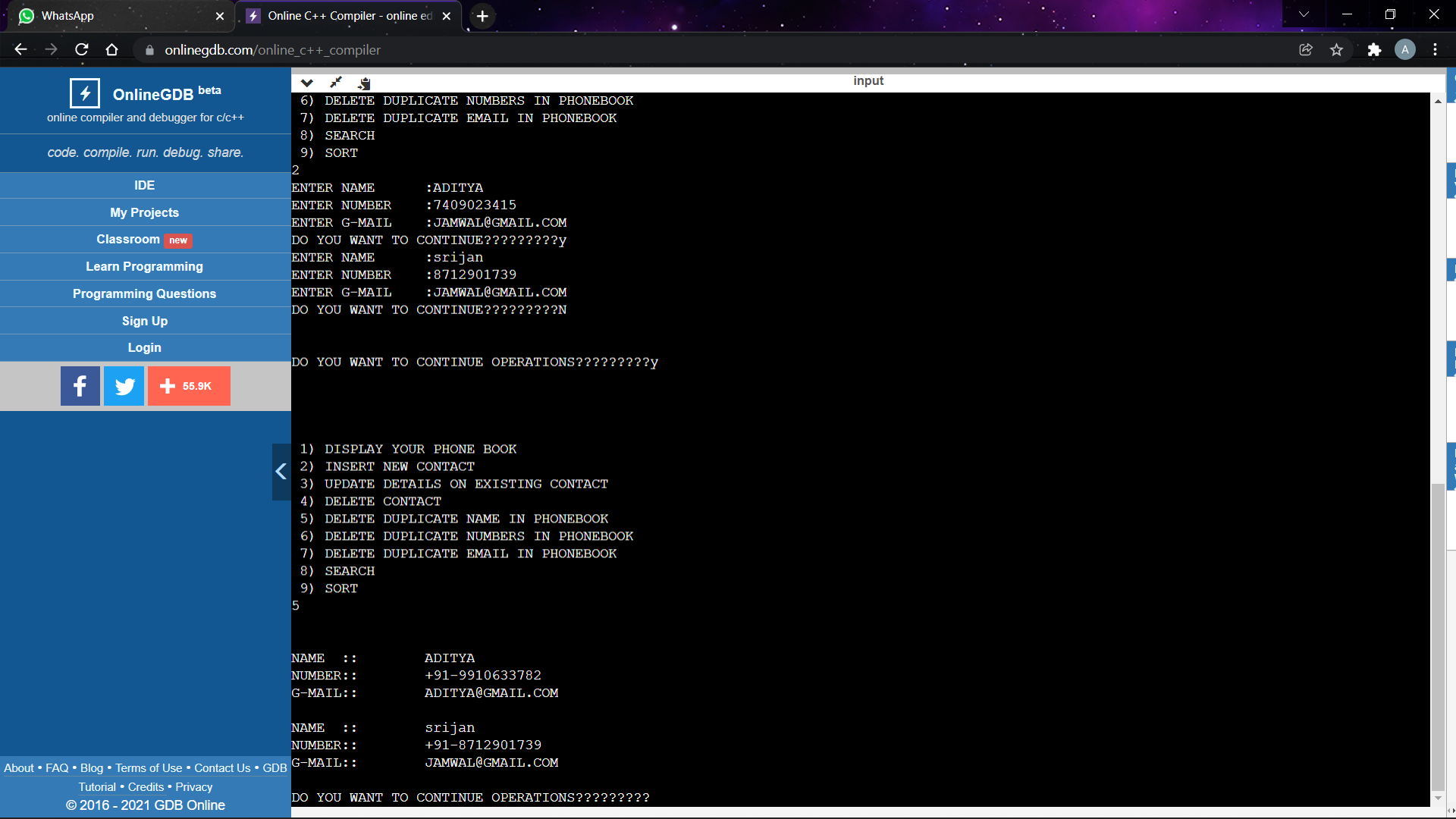
**DISPLAY PHONEBOOK**



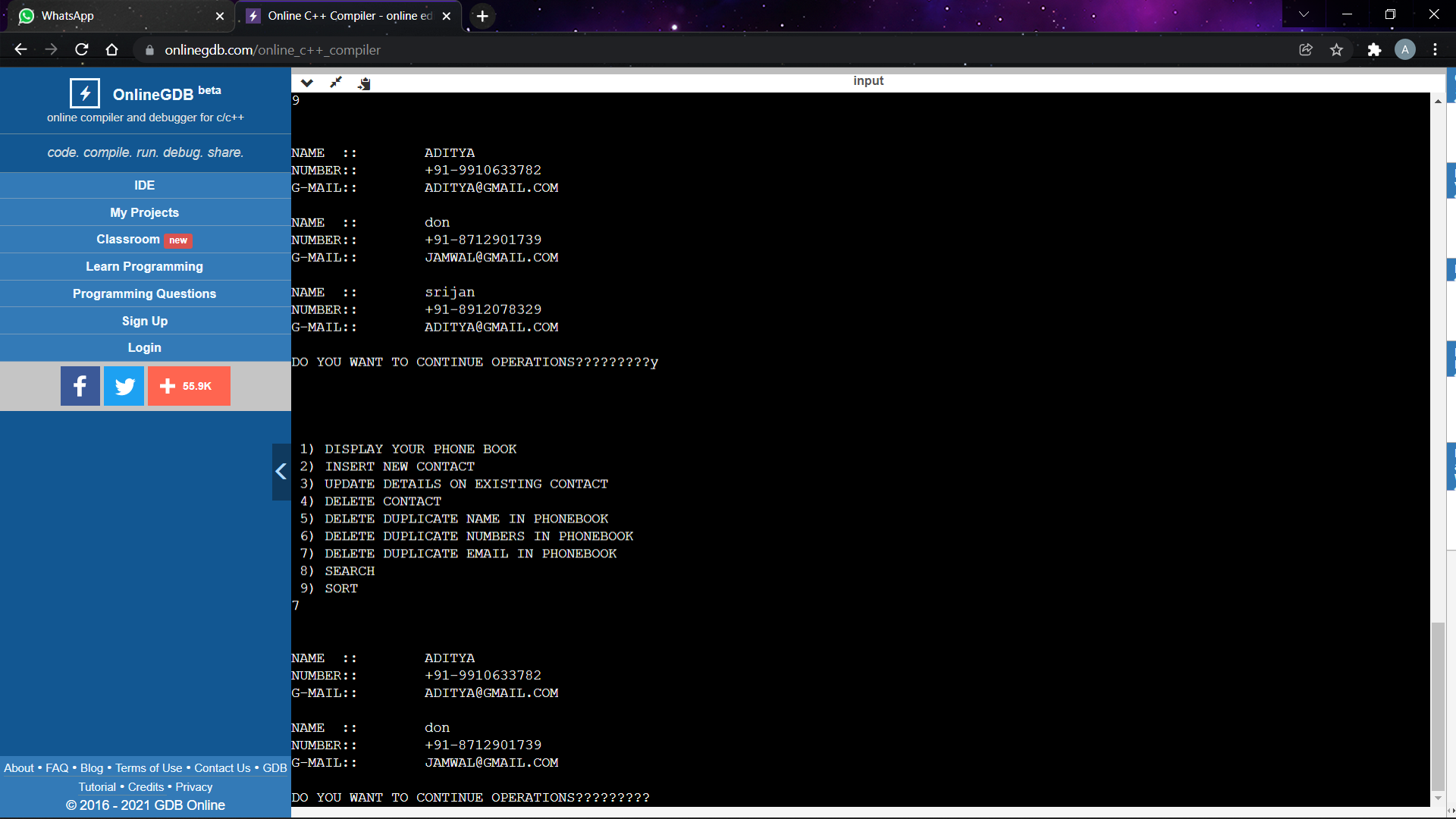
**DELETING DUPLICATE NUMBERS**



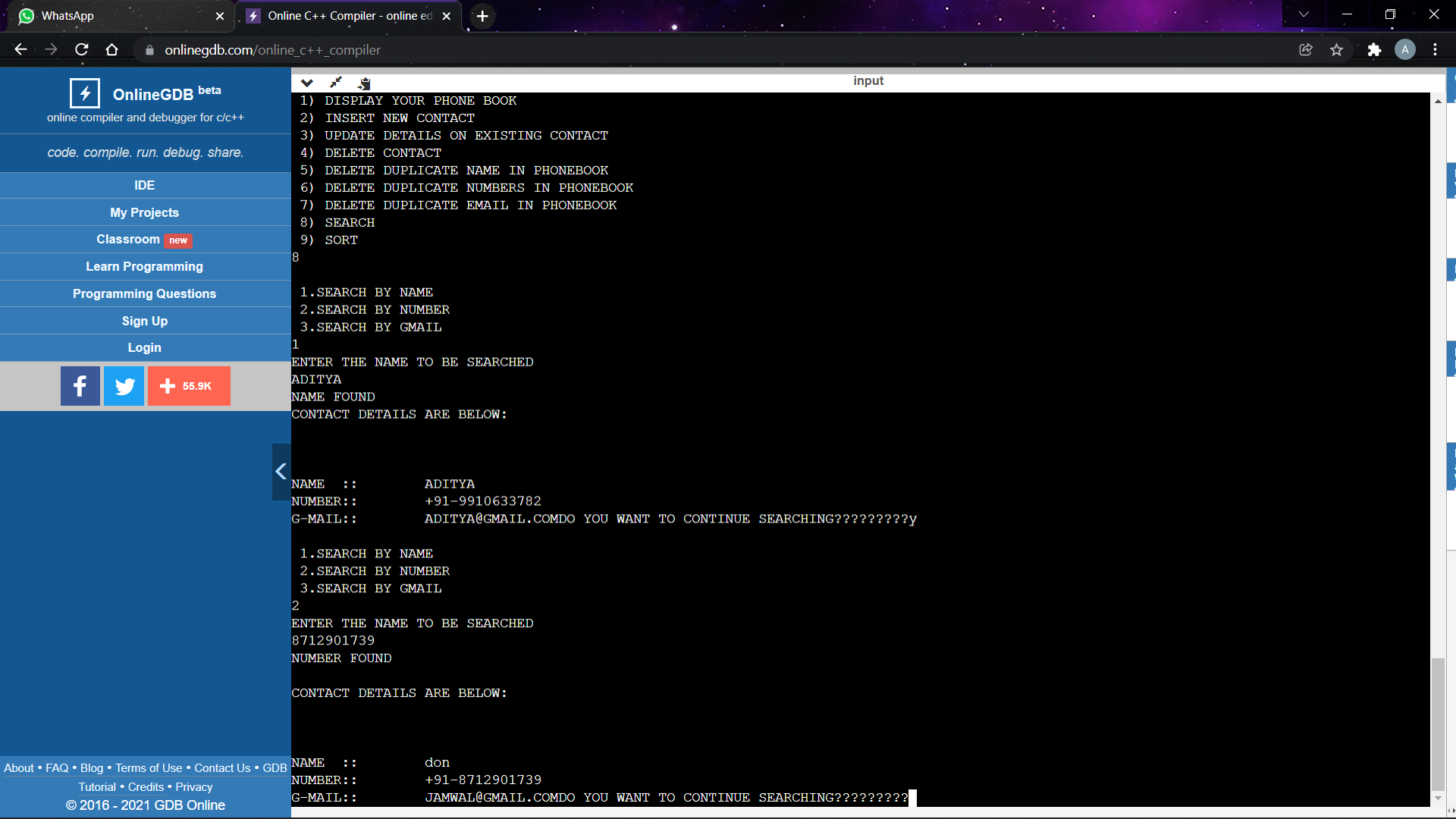
**DELETING DUPLICATE NAMES**



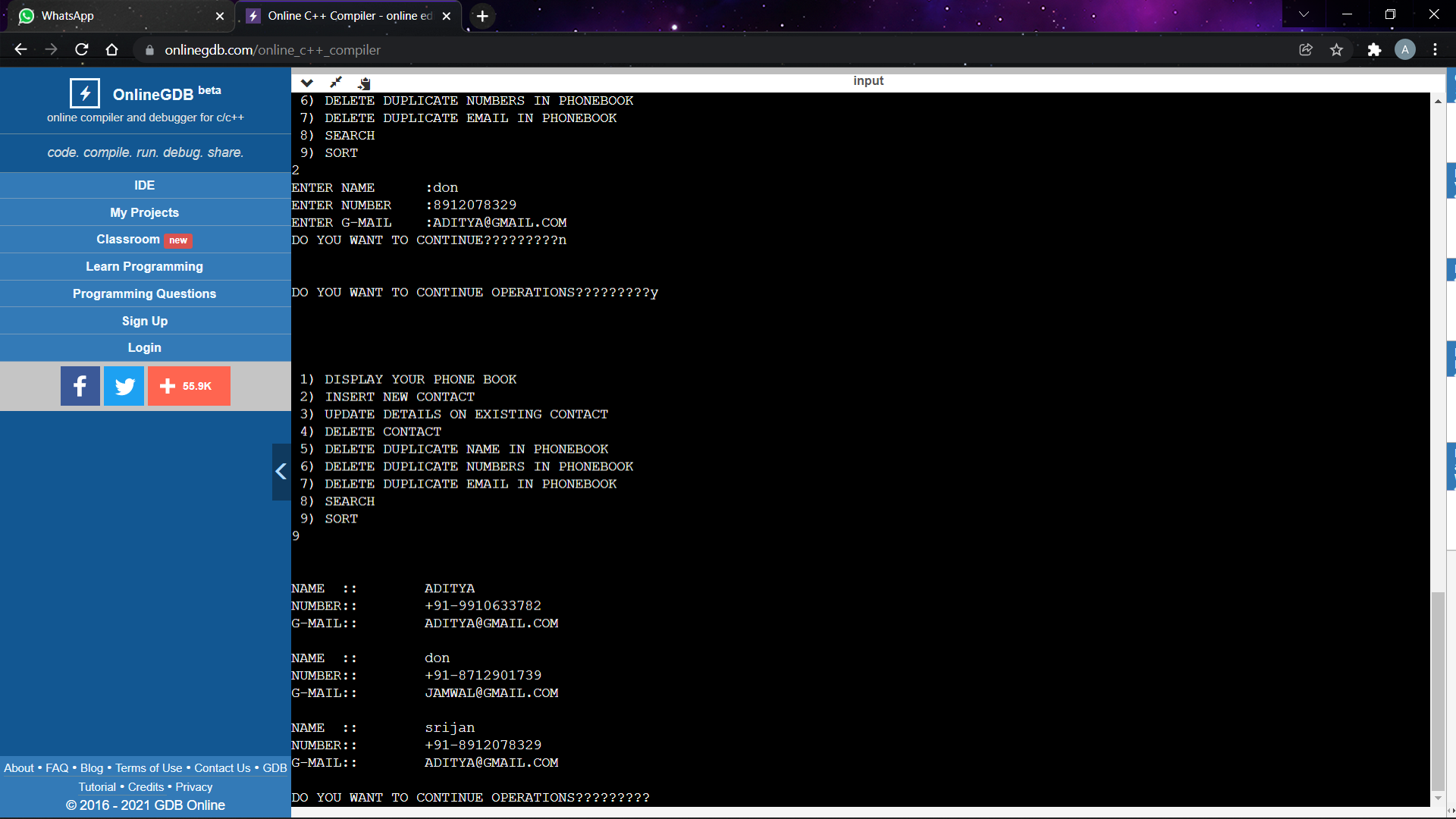
**DELETING DUPLICATE EMAIL**



**SEARCH**



**SORTING**



**THANK YOU**