Data Structures and Algorithms

Theory/Lab

CSC/CSL-221

Semester Project

TO

Mr Sikander Hayat

BY

Osama Mustafa *-01-134191-060*
Sulman Ahmed Satti *-01-134192-079*
BSCS-3B

Submitted on ***January 11,2020***

Department of Computer Science
BAHRIA UNIVERSITY, ISLAMABAD

Acknowledgement:

We would like to express thanks and gratitude to Mr Sikander Hayat for considering us worthy of solving a real-time problem and providing us with the opportunity of using our programming skills to built a real-time application.

We would like to express special thanks to instructors

- Sir Sikander Hayat for OOP and DSA
- Mam Momina Moetesum for DSA
- Mam Saima Javed for OOP
- Mam Mehwish Pervaiz for CP
- Sir Saqlain for CP

And Friends like M Wajahat(01-134192-102), Jahanzeb Naeem(01-134192-027), M Osama(01-134192-110), Shaheer Khan Niazi(01-134192-077), Malick Zohaib Mustafa((01-134192-030) and Tauheed Ejaz Khan((01-134192-108)

Who have been supporting and helping in enhancing our programming skills.

Websites like ***youtube.com***, ***stackoverflow.com***, ***cplusplus.com***, ***geeksforgeeks.com*** etc. have been playing their important role.

In this project, no specific help was taken from these sources. Development of console app and it's features were designed in the light of group discussion and prior knowledge.

Regards

Team Members

Sulman Ahmed Satti

01-134192-079

BSCS-3B

Osama Mustafa

01-134191-060

BSCS-3B

Problem Statement:

Design and develop an application to automate a **Corona Virus Patient Management System**. The detail of the application is as below:

- Application should have a proper login.
- Functionality to Add, Delete, Search and Modify records of the patient.
- A detail file (txt or binary) that includes the record of the patient along with all the attributes, some of them are symptoms, immune level, severity, city name etc.
- A function that computes which city of the Pakistan contains more patient than other cities.
- A person must be declared Corona Virus Patient if the immune level is below the minimum level and symptoms are fever, dry cough, tiredness etc.
- Application code must be divided as separate (.h, .cpp) files.

Objectives:

Following are some of the objectives of the to whom Project has been intended to

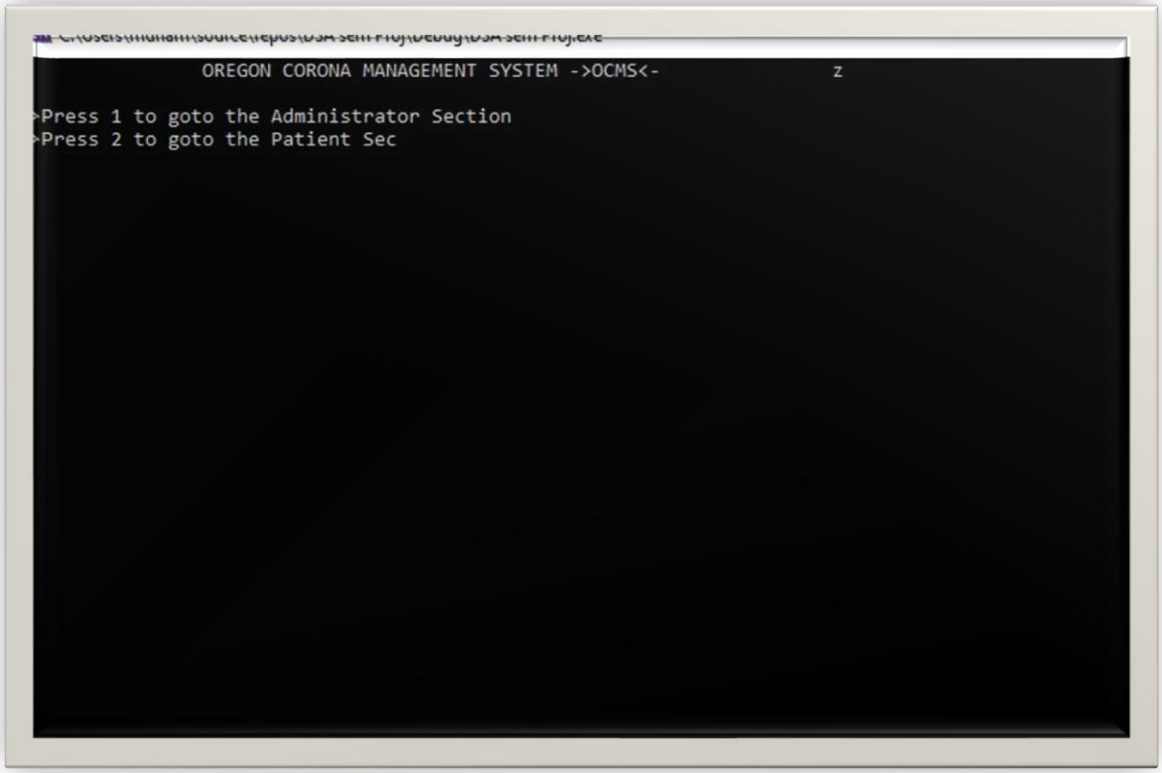
- To recap the concepts of Object-Oriented Programming
- To know how to solve real-world problems through programming
- To know how to apply Data Structures and Algorithms to applications
- To know how to build Efficient applications

Salient Features of Application:

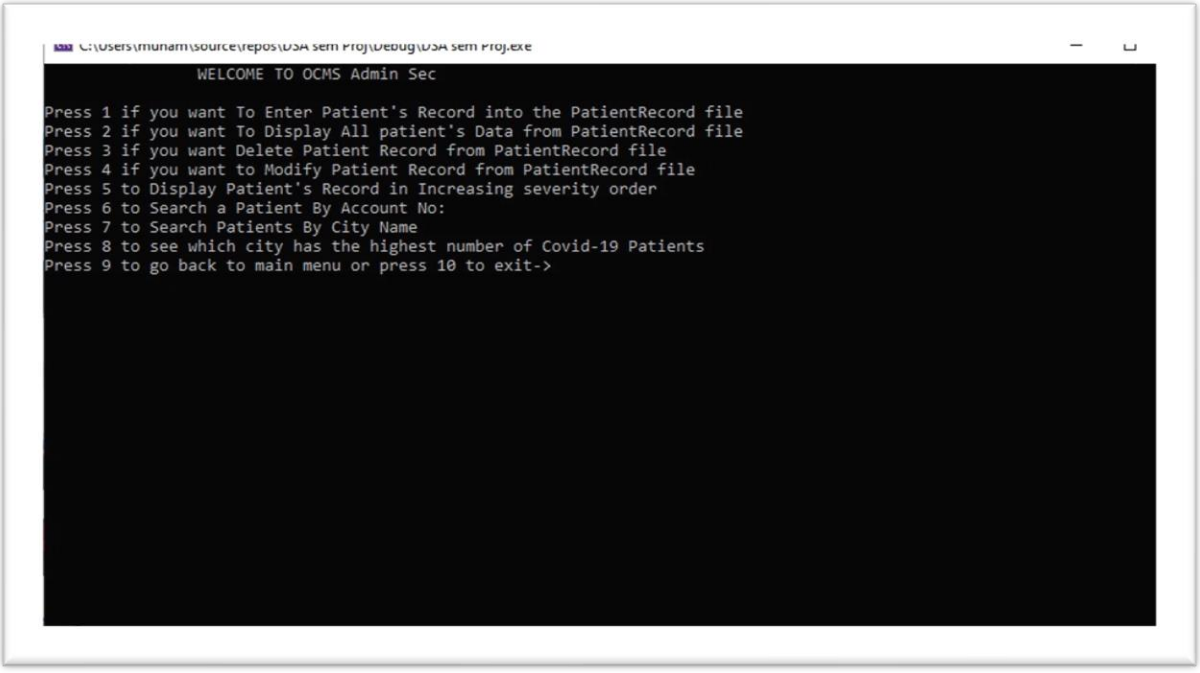
➢ Interface:

Application starts with Menu which provides two logins.

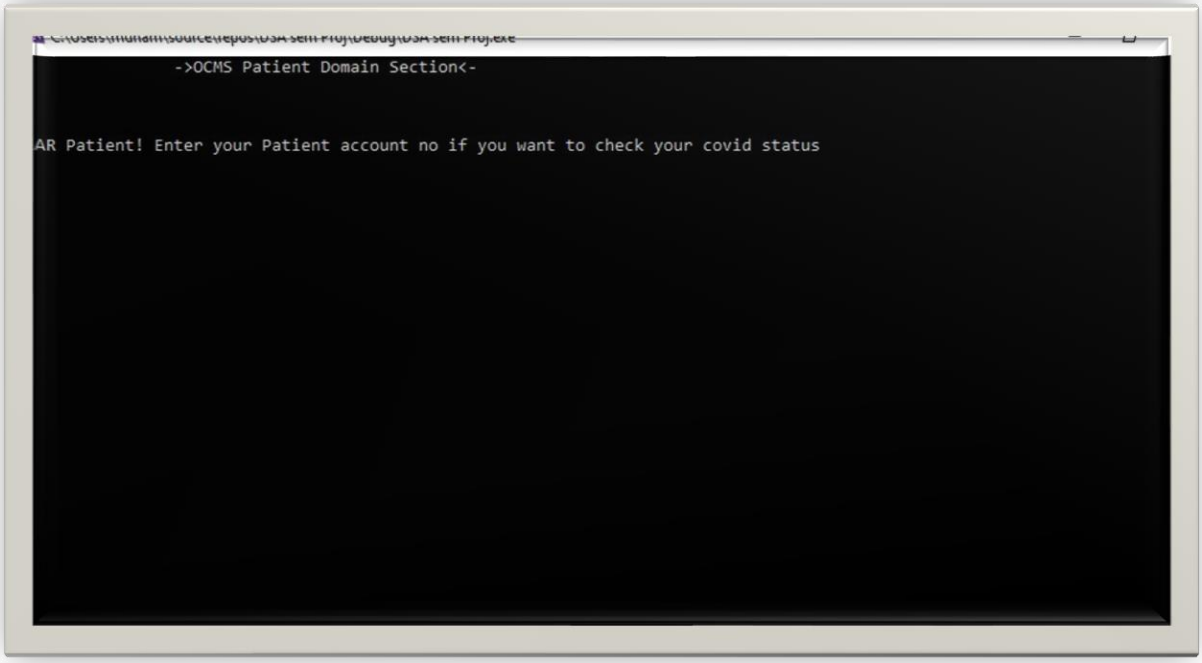
- Admin
- Patient


OREGON CORONA MANAGEMENT SYSTEM ->OCMS<-
z
>Press 1 to goto the Administrator Section
>Press 2 to goto the Patient Sec

- *Admin Login:*


WELCOME TO OCMS Admin Sec
Press 1 if you want To Enter Patient's Record into the PatientRecord file
Press 2 if you want To Display All patient's Data from PatientRecord file
Press 3 if you want Delete Patient Record from PatientRecord file
Press 4 if you want to Modify Patient Record from PatientRecord file
Press 5 to Display Patient's Record in Increasing severity order
Press 6 to Search a Patient By Account No:
Press 7 to Search Patients By City Name
Press 8 to see which city has the highest number of Covid-19 Patients
Press 9 to go back to main menu or press 10 to exit->

- Student Login:



Source Code:

```
int main()
{
a:      system("cls");
        int choice1;

        cout << "          OREGON CORONA MANAGEMENT SYSTEM ->OCMS<-          z\n\n";
        cout << "->Press 1 to goto the Administrator Section\n";
        cout << "->Press 2 to goto the Patient Sec\n\n";
        cin >> choice1;
        switch (choice1)
        {
        case 1:
        {
        b:              system("cls");
                int choice2;
                        cout << "          WELCOME TO OCMS Admin Sec\n\n";
                cout << "Press 1 if you want To Enter Patient's Record into the PatientRecord file\n";
                cout << "Press 2 if you want To Display All patient's Data from PatientRecord file\n";
                cout << "Press 3 if you want Delete Patient Record from PatientRecord file\n";
                cout << "Press 4 if you want to Modify Patient Record from PatientRecord file\n";
                cout << "Press 5 to Display Patient's Record in Increasing severity order\n";
```

```
		cout << "Press 6 to Search a Patient By Account No: \n";
		cout << "Press 7 to Search Patients By City Name\n";
		cout << "Press 8 to see which city has the highest number of Covid-19 Patients\n";
		cout << "Press 9 to go back to main menu or press 10 to exit->\n\n";
		cin >> choice2;
		switch (choice2)
		{
		case 1:
		{
			system("cls");
			CoronaManagementSystem CMS;
			int choice3 = 1;
			int choice4;
			while (choice3 != 0)
			{
				CMS.write_into_file();
				cout << "Press 0 if you donot want to enter another record\n\n";
				cin >> choice3;
			}
			cout << "Press 1 if you want to goto back menu else any other key to
exit\n";
			cin >> choice4;
			if (choice4 == 1)
			{
				goto b;
			}
			Else
			{
				return(0);
			}

		}
		case 2:
		{

			system("cls");
			CoronaManagementSystem CMS;
			CMS.read_from_file();
			int choice5;
			cout << "Press 1 if you want to goto the main menu or any other key to
exit\n\n";
			cin >> choice5;
			if (choice5 == 1)
			{
				goto b;
			}
```

```
                    Else
                    {
                            return(0);
                    }

            }
            case 3:
            {
                    system("cls");
                    int key;
                    cout << "Enter Account no in which you want to perform deletion of
record\n\n";
                    cin >> key;
                    CoronaManagementSystem CMS;
                    CMS.delete_patient_record(key);
                    int choice6;
                    cout << "\nPress 1 if you want to return to main menu or any other key to
exit\n\n";
                    cin >> choice6;
                    if (choice6 == 1)
                    {
                            goto b;
                    }
                    Else
                    {
                            return(0);
                    }

            }
            case 4:
            {
                    system("cls");
                    int key;
                    cout << "Enter Account no in which you want to perform Modification of
record\n\n";
                    cin >> key;
                    CoronaManagementSystem CMS;
                    CMS.Modify_Patient_Record(key);
                    int choice7;
                    cout << "\nPress 1 if you want to return to main menu or any other key to
exit\n\n";
                    cin >> choice7;
                    if (choice7 == 1)
                    {
                            goto b;
                    }
```

```
                    else
                    {
                            return(0);
                    }


            }
            case 5:
            {
                    system("cls");
                    CoronaManagementSystem CMS;
                    Dlist list;
                    list = CMS.load_fromfile_to_Dlist();
                    CMS.display_data_incorder(list.head);
                    int choice8;
                    cout << "\nPress 1 if you want to goto main menu or any other key to
exit\n";
                    cin >> choice8;
                    if (choice8 == 1)
                    {
                            goto b;
                    }
                    else
                    {
                            return(0);
                    }

            }
            case 6:
            {
                    system("cls");
                    CoronaManagementSystem CMS;
                    int dkey;
                    cout << "Enter key no that you want to search from the record\n\n";
                    cin >> dkey;
                    Dlist list;
                    list = CMS.load_fromfile_to_Dlist();
                    CMS.search_by_key(list.head, dkey);
                    int choice9;
                    cout << "\nPress 1 if you want ot goto main menu or any other key to
exit\n\n";
                    cin >> choice9;
                    if (choice9 == 1)
                    {
                            goto b;
                    }
```

```
                else
                {
                        return(0);
                }


        }
        case 7:
        {
                system("cls");
                CoronaManagementSystem CMS;
                string dcity;
                cout << "Enter key no that you want to search from the record\n\n";
                cin >> dcity;
                Dlist list;
                list = CMS.load_fromfile_to_Dlist();
                CMS.search_by_city_name(list.head, dcity);
                int choice10;
                cout << "\nPress 1 if you want ot goto main menu or any other key to
exit\n\n";
                cin >> choice10;
                if (choice10 == 1)
                {
                        goto b;
                }
                else
                {
                        return(0);
                }
        }
        case 8:
        {
                system("cls");
                CoronaManagementSystem CMS;
                Dlist list = CMS.load_fromfile_to_Dlist();
                list.ccount();
                int choice11;
                cout << "\nPress 1 if you want ot goto main menu or any other key to
exit\n\n";
                cin >> choice11;
                if (choice11 == 1)
                {
                        goto b;
                }
                else
                {
```

```
                                        return(0);
                                }
                        }
                        case 9:
                        {
                                goto a;

                        }
                        case 10:
                        {
                                return(0);
                        }
                        }

        }
        case 2:
        {
                        system("cls");
                        cout << "              ->OCMS Patient Domain Section<-\n\n\n\n";
                        CoronaManagementSystem CMS;
                        int schoice;
                        cout << "DEAR Patient! Enter your Patient account no if you want to check your
covid status\n";
                        cin >> schoice;
                        Dlist list = CMS.load_fromfile_to_Dlist();
                        CMS.check_patient(list.head, schoice);
                        int choice12;
                        cout << "\nPress 1 if you want to goto Main Menu or any other key to exit\n\n";
                        cin >> choice12;
                        if (choice12 == 1)
                        {
                                goto a;
                        }
                        else
                        {
                                return(0);
                        }
        }
        }



        system("pause");
        return 0;
}
```

➢ Efficiency :

There were mainly two data structures used.

- Binary Search Tree

Patients data was stored in BST.

Binary search tree brings efficiency in searching as in case of hierarichal data structures.

Also it works on pointer hence not only efficient but also conserves unnecessary space.

- Doubly-link List

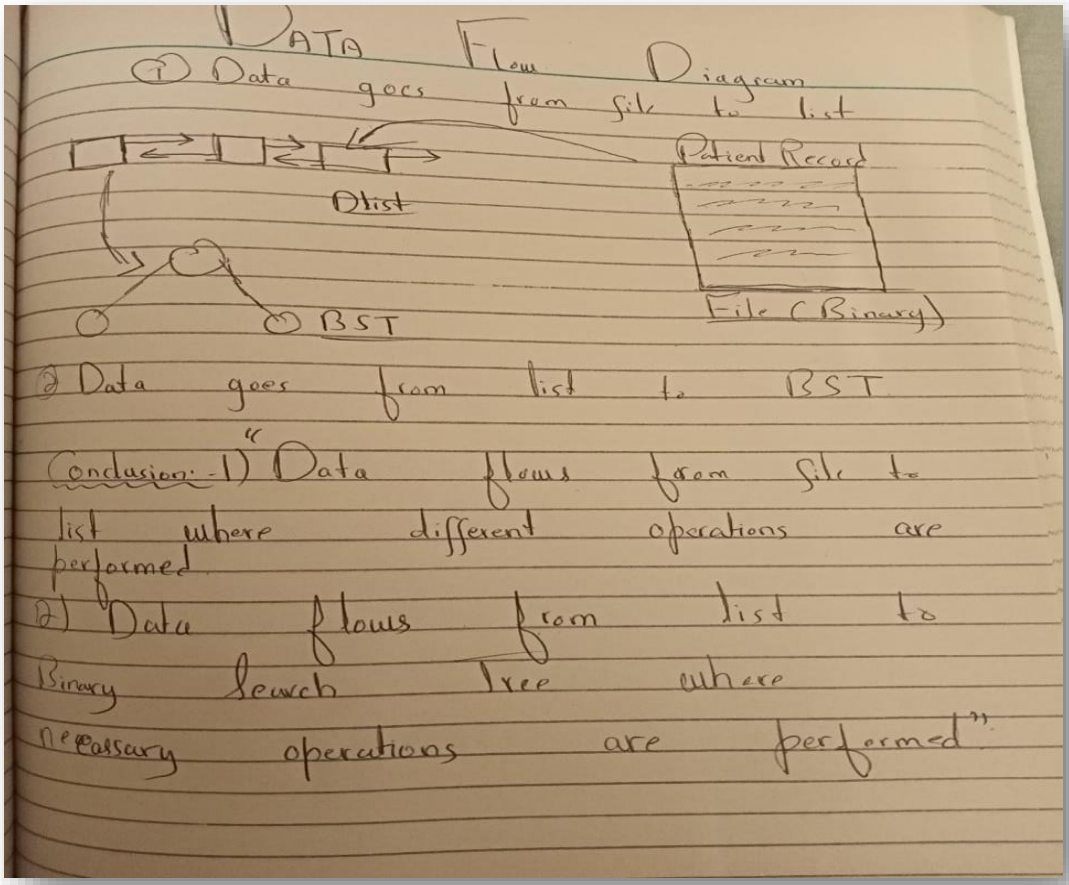
City data was stored in Doubly-linked list

As there are not many entities to be filled in case of cities as compared to patients.

Hense searching is not using linear data structure isn't low in efficiency relatively.

Usage of Doubly-link list rather than arrays/vectors not only brings overall time-efficiency but also saves a lot of space.

➢ Data Flow:



Modules with details:

Following major functions are involved in this program included under implementation.cpp. ADT functions are not included in this list to avoid complexity as they are generic.

- CoronaManagementSystem(): Default Contructor for Main Class Corona Mangement System.
- Write_into_file(): This function is used to write complete record of Patients in Binary File.
- Read_from_file(): This functions is used to read Patients complete portfolio fom binary file and then display it. Mainly for data entry verification purpose
- Delete_patient_record(int): This function is used to delete patient record from the Binary file by going into the specific key no
- Modify_patient_record(int): This function is used to modify patient record in the binary file by going into the specific key no
- Load_from_file_to_Dlist(): This function is used to load Patients data from file to Nodes of the doubly linked list.
- Search_by_key(int): This function is used to search patient portfolio by going onto the specific key no in the Doubly List
- Search_by_city_name(string): This function is used to search patients data from Doubly List by City Name
- Display_data_incorder(Node*): This function take head of a list as parameter. Than it passes the data from List to a binary search tree. And then by in order traversal of the BST it displays the data of patients from BST in an increasing order by severity level of the disease.
- Check_patient(Node*,int): This function takes two parameters. First is the head of the list and the second parameter as the key no of patient to check whether the patient is a covid patient or not based on the details of the patient. If any ssymptoms, severity level is above 2 and immune level is below 4 it says it as covid positive.
- Ccount(): Checks which city has highest no of Patients.

➢ CoronaManagementSystem()

Source Code:

```
class Dlist;

                keyno = 0;
                password = 0;
                for (int i = 0; i < 100; i++)
                {
                        userName[i] = NULL;
                        cityName[i] = NULL;
                }
                symptoms = 0;
                immuneLevel = 0;

                severity = 0;
```

➢ Write_into_file() and Read_from_file()

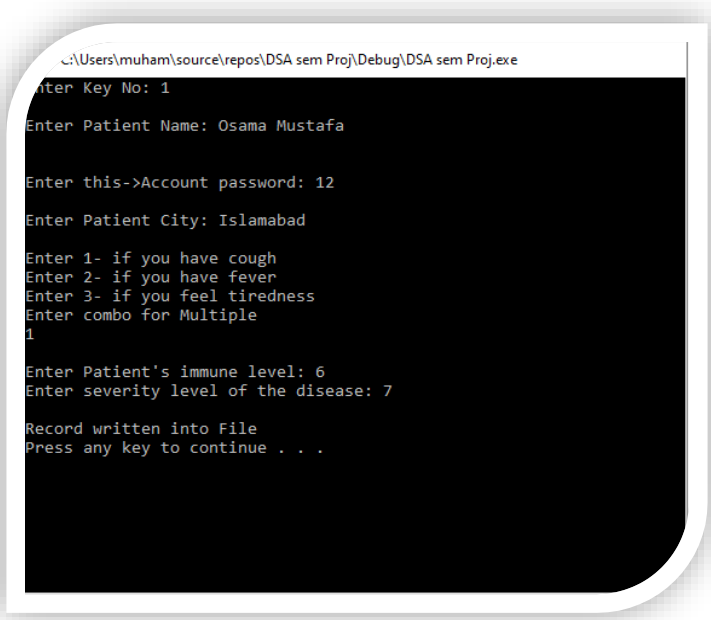
Source Code:

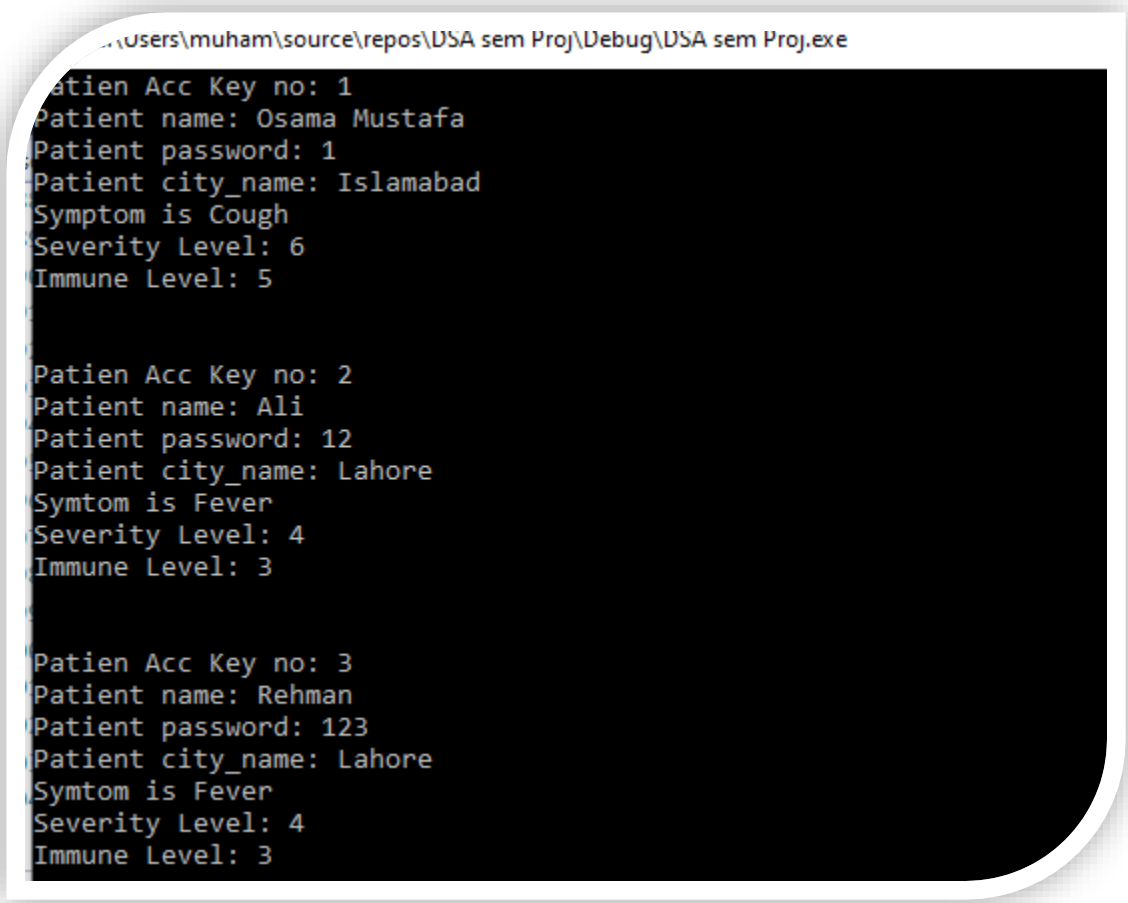
```
fstream myfile;
		myfile.open("PatientRecord.dat", ios::binary | ios::app | ios::in | ios::out);
		if (!myfile)
		{
			cout << "Error! -Opening Main File\n";
		}
		else
		{
			CoronaManagementSystem system1;
			cout << "Enter Key No: ";
			cin >> system1.keyno;
			cin.ignore();
			cout << "\nEnter Patient Name: ";
			cin.getline(system1.userName, 100);
			cin.ignore();
			cout << "\nEnter this->Account password: ";
			cin >> system1.password;
			cin.ignore();
			cout << "\nEnter Patient City: ";
			cin.getline(system1.cityName, 100);
			cin.ignore();
			cout << "Enter 1- if you have cough\nEnter 2- if you have fever\nEnter 3- if you feel tiredness\nEnter combo for Multiple\n";
			cin >> system1.symptoms;
			cin.ignore();
			cout << "\nEnter Patient's immune level: ";
			cin >> system1.immuneLevel;
			cin.ignore();
			cout << "Enter severity level of the disease: ";
			cin >> system1.severity;
			cin.ignore();


			myfile.write((char*)&system1, sizeof(system1));
			myfile.close();
			cout << "\nRecord written into File\n";

		}
```

```
CoronaManagementSystem system2;
		fstream rfile;
		rfile.open("PatientRecord.dat", ios::in | ios::out);
		if (!rfile)
		{
			cout << "Error! -Opening File\n";
		}
		else
		{
```

```
                while (rfile.read((char*)&system2, sizeof(system2)))
                {
                        cout << "Patien Acc Key no: " << system2.keyno << endl;
                        cout << "Patient name: " << system2.userName << endl;
                        cout << "Patient password: " << system2.password << endl;
                        cout << "Patient city_name: " << system2.cityName << endl;
                        if (system2.symptoms == 1)
                        {
                                cout << "Symptom is Cough\n";
                        }
                        if (system2.symptoms == 2)
                        {
                                cout << "Symtom is Fever\n";
                        }
                        if (system2.symptoms == 3)
                        {
                                cout << "Symptom is Tiredness\n";
                        }
                        cout << "Severity Level: " << system2.severity << endl;
                        cout << "Immune Level: " << system2.immuneLevel << endl;
                        cout << endl << endl;

                }
                rfile.close();

        }
```

Output:





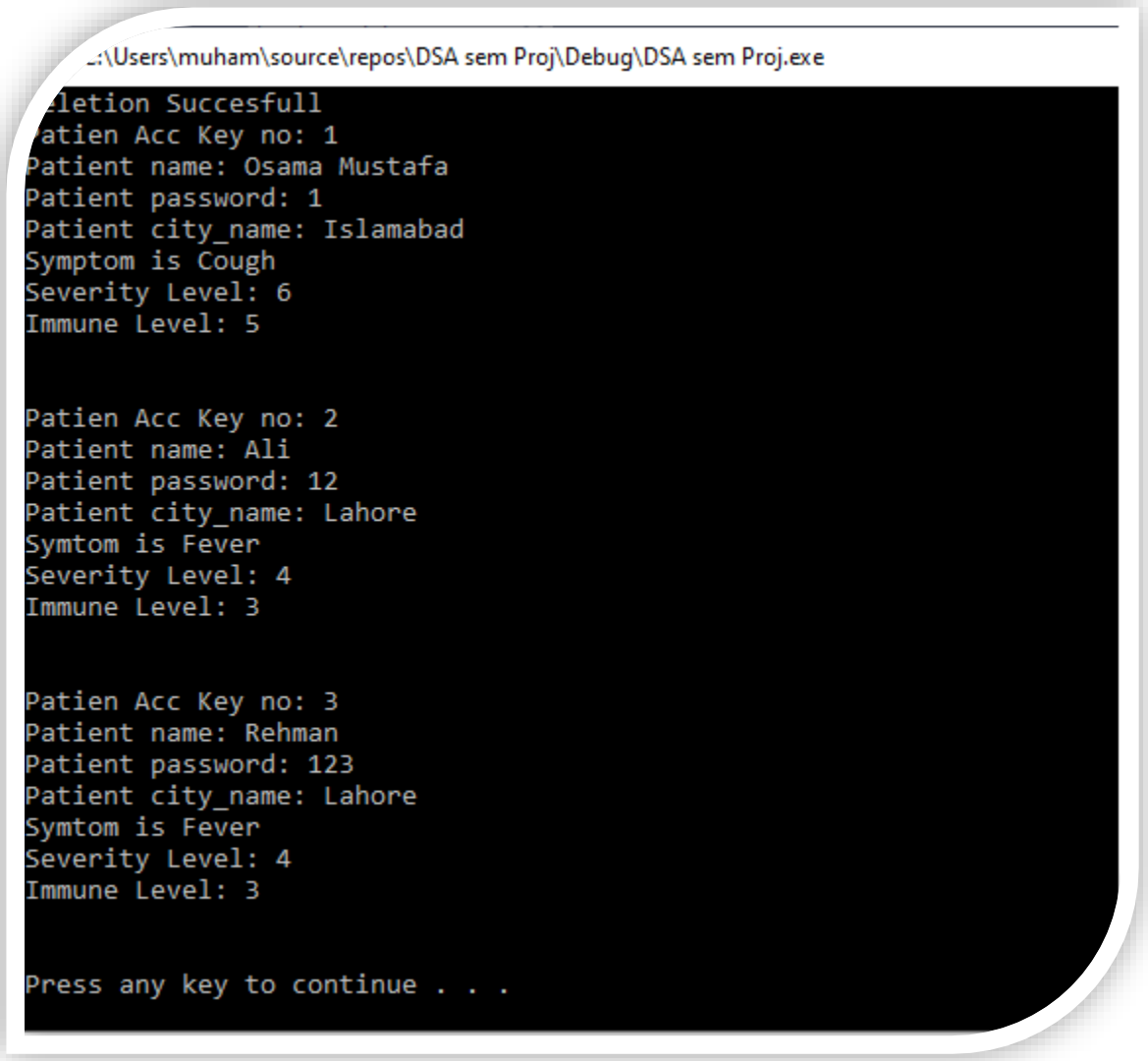
➢ Delete_patient_record(int)

Source Code:

```
CoronaManagementSystem system4;
		fstream my_file;
		fstream my_file1;
		my_file1.open("New.dat", ios::in | ios::app | ios::out);
		my_file.open("PatientRecord.dat", ios::in | ios::out | ios::app);
		if (!my_file)
		{
			cout << "Error- Opening File\n";
		}
		else
		{
			while (my_file.read((char*)&system4, sizeof(system4)))
			{
				if (system4.keyno != dkeyno)
				{
					my_file1.write((char*)&system4, sizeof(system4));
				}
			}
			my_file.close();
```

```
                    my_file1.close();
                    cout << "Deletion Succesfull\n";
                    remove("PatientRecord.dat");
                    rename("New.dat", "PatientRecord.dat");

                }
```

Output:



➢ Modify_patient_record(int)

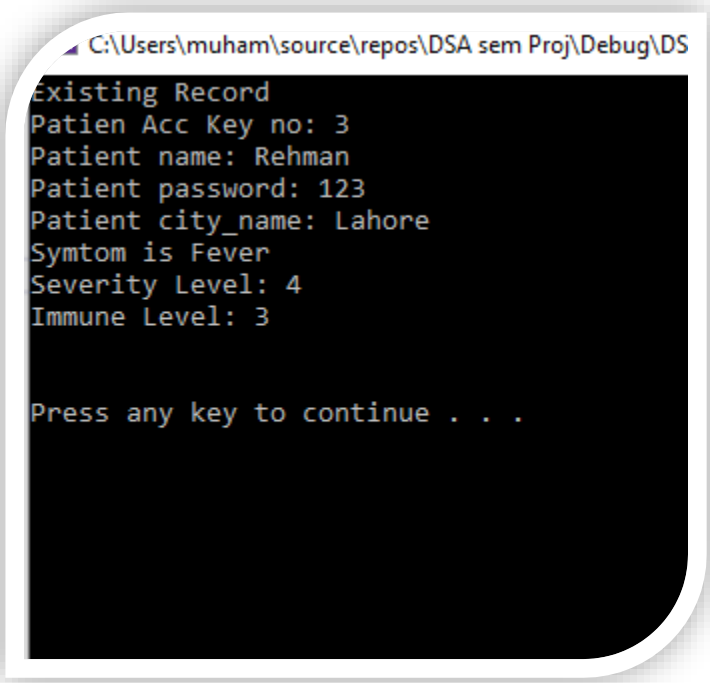
Source Code:

```
CoronaManagementSystem system5,system6;
                fstream my_file;
                my_file.open("PatientRecord.dat", ios::in | ios::out);
                while (my_file.read((char*)&system5, sizeof(system5)))
                {
```

```
                        if (system5.keyno == dkeyno)
                        {
                                cout << "Existing Record\n";
                                cout << "Patien Acc Key no: " << system5.keyno << endl;
                                cout << "Patient name: " << system5.userName << endl;
                                cout << "Patient password: " << system5.password << endl;
                                cout << "Patient city_name: " << system5.cityName << endl;
                                if (system5.symptoms == 1)
                                {
                                        cout << "Symptom is Cough\n";
                                }
                                if (system5.symptoms == 2)
                                {
                                        cout << "Symtom is Fever\n";
                                }
                                if (system5.symptoms == 3)
                                {
                                        cout << "Symptom is Tiredness\n";
                                }
                                cout << "Severity Level: " << system5.severity << endl;
                                cout << "Immune Level: " << system5.immuneLevel << endl;
                                cout << endl << endl;
                                system("pause");
                                system("cls");
                                cout << "Enter New Record/Modified\n";
                                system("pause");
                                system("cls");
                                cout << "Enter Key No: ";
                                cin >> system6.keyno;
                                cin.ignore();
                                cout << "\nEnter Patient Name: ";
                                cin.getline(system6.userName, 100);
                                cin.ignore();
                                cout << "\nEnter this->Account password: ";
                                cin >> system6.password;
                                cin.ignore();
                                cout << "\nEnter Patient City: ";
                                cin.getline(system6.cityName, 100);
                                cin.ignore();
                                cout << "Enter 1- if you have cough\nEnter 2- if you have
fever\nEnter 3- if you feel tiredness\nEnter combo for Multiple\n";
                                cin >> system6.symptoms;
                                cin.ignore();
                                cout << "\nEnter Patient's immune level: ";
                                cin >> system6.immuneLevel;
                                cin.ignore();
                                cout << "Enter severity level of the disease(1-3: ";
                                cin >> system6.severity;
                                cin.ignore();
                                cout << endl;
                                my_file.seekp(-(long int)sizeof(system5),ios::cur);
                                my_file.write((char*)&system6, sizeof(system6));
                                cout << "Patient Record Updated\n";
                                break;
```

```
                    }
                }
                my_file.close();
```

Output:



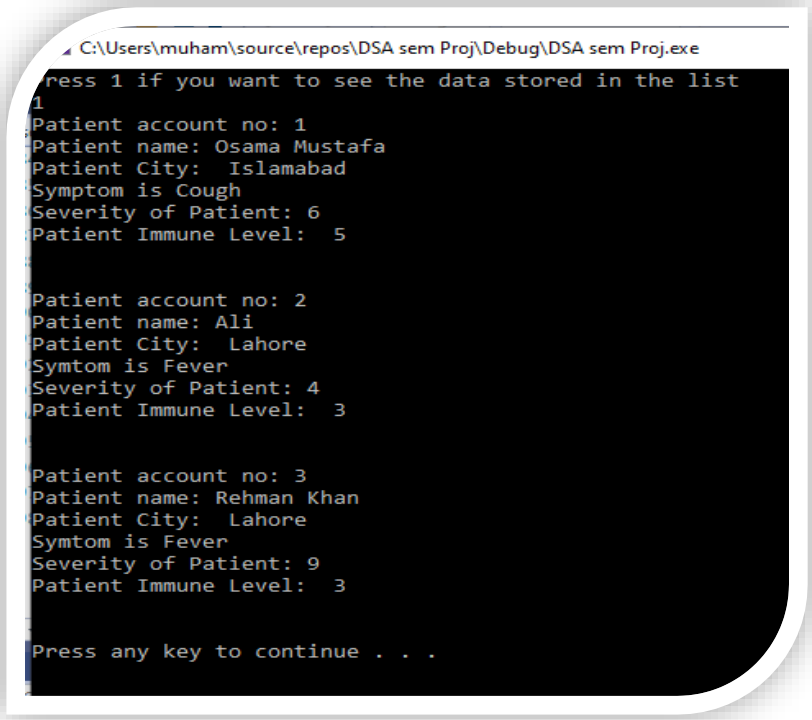
➢ Load_from_file_to_Dlist()

Source Code:

```
Dlist loader_list;
		fstream my_file;
		my_file.open("PatientRecord.dat", ios::in | ios::out);
		if (!my_file)
		{
			cout << "Error- Opening File\n";
		}
		else
		{
			CoronaManagementSystem system3;
			Dlist loader_list;
			while (my_file.read((char*)&system3, sizeof(system3)))
			{
				loader_list.insertend(system3.keyno, system3.password,
system3.userName, system3.cityName, system3.severity, system3.symptoms,
system3.immuneLevel);
```

```
                        }
                        int choice;
                        system("cls");
                        cout << "Press 1 if you want to see the data stored in the
list\n";
                        cin>>choice;
                        if (choice == 1)
                        {
                                loader_list.traverse();
                        }
                        return loader_list;

                }
                my_file.close();
                if (loader_list.head == NULL)
                {
                        cout << "Ok";
                }
```

Output:



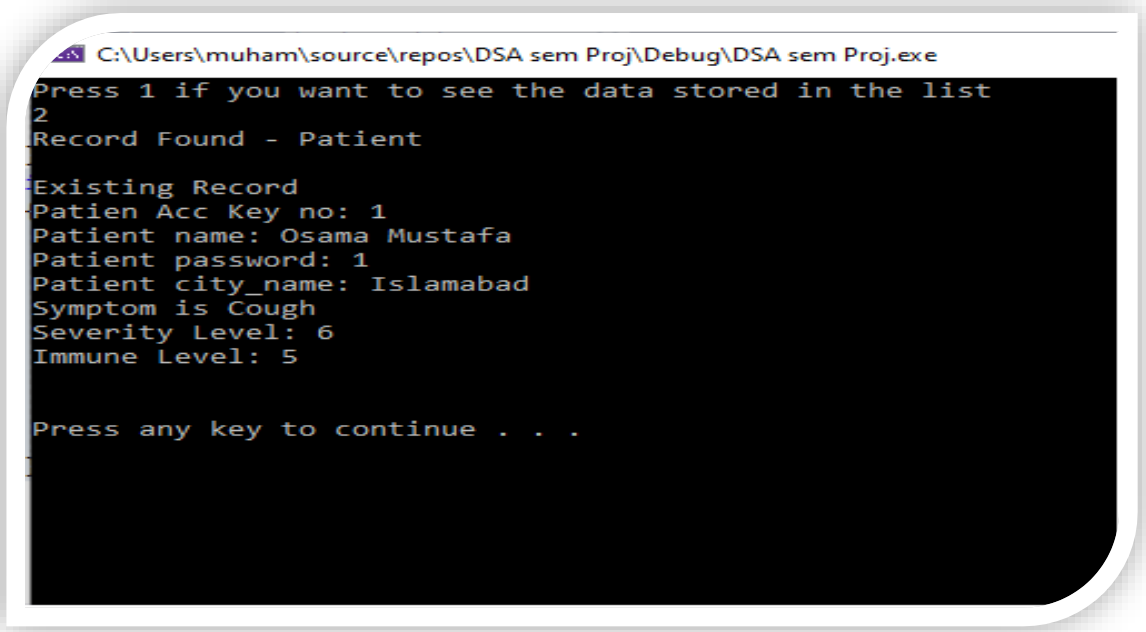
➢ Search_by_key(int)

Source Code:

```
Node* ptr = head;
                bool found = false;
```

```
		while (ptr != NULL)
		{
			if (ptr->keyno == dkeyno)
			{
				cout << "Record Found - Patient\n\n";
				cout << "Existing Record\n";
				cout << "Patien Acc Key no: " << ptr->keyno << endl;
				cout << "Patient name: " << ptr->userName << endl;
				cout << "Patient password: " << ptr->password << endl;
				cout << "Patient city_name: " << ptr->cityName << endl;
				if (ptr->symptoms == 1)
				{
					cout << "Symptom is Cough\n";
				}
				if (ptr->symptoms == 2)
				{
					cout << "Symtom is Fever\n";
				}
				if (ptr->symptoms == 3)
				{
					cout << "Symptom is Tiredness\n";
				}
				cout << "Severity Level: " << ptr->severity << endl;

				cout << "Immune Level: " << ptr->immuneLevel << endl;
				cout << endl << endl;
				found = true;

			}
			ptr = ptr->next;
		}
		if (found == false)
		{
			cout << "Record of patient->key->" << dkeyno << " donot exists in
database\n\n";
		}
```

Output:



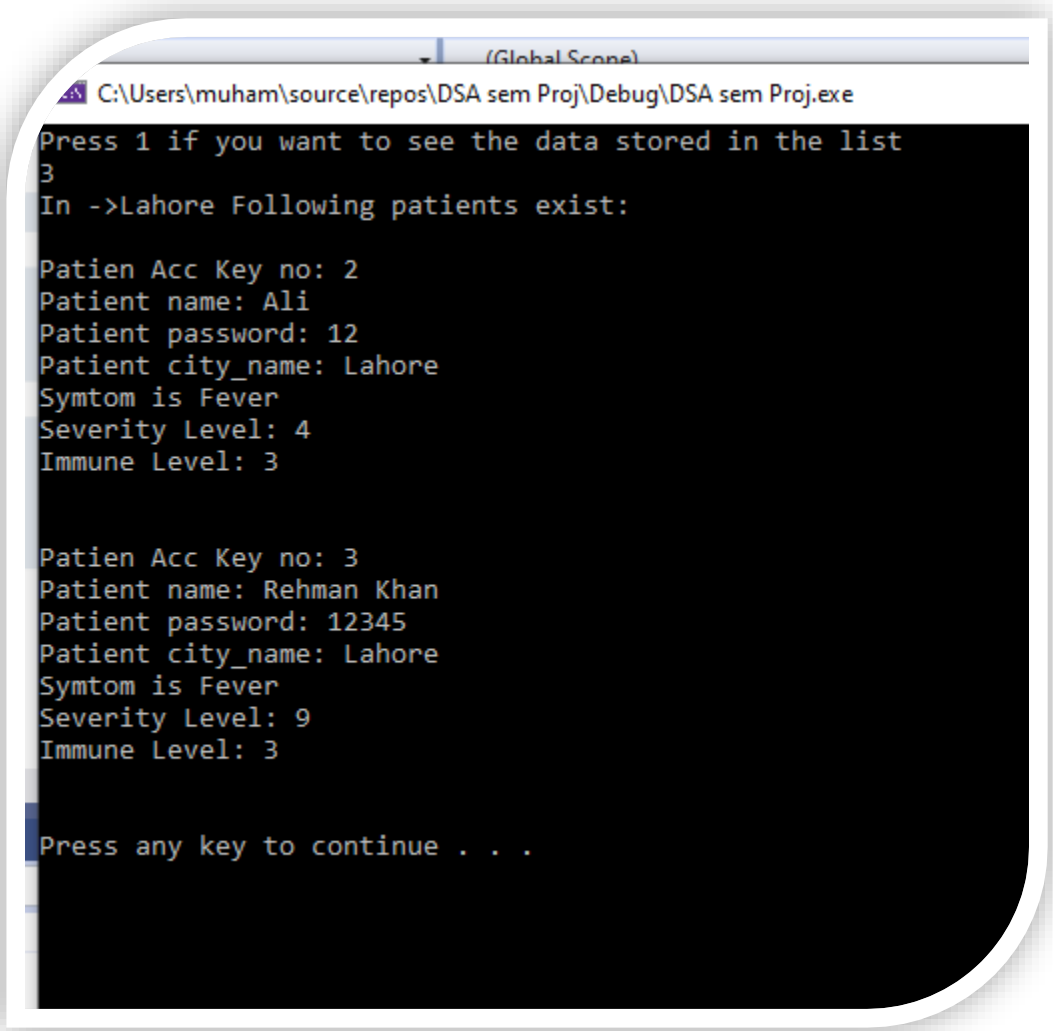
➢ Search_by_city_name(string)

Source Code:

```
Node* ptr = head;
		bool found = false;
		cout << "In ->" << city << " Following patients exist: \n\n";
		while (ptr != NULL)
		{
			if (ptr->cityName==city)
			{

				cout << "Patien Acc Key no: " << ptr->keyno << endl;
				cout << "Patient name: " << ptr->userName << endl;
				cout << "Patient password: " << ptr->password << endl;
				cout << "Patient city_name: " << ptr->cityName << endl;
				if (ptr->symptoms == 1)
				{
					cout << "Symptom is Cough\n";
				}
				if (ptr->symptoms == 2)
				{
					cout << "Symtom is Fever\n";
				}
				if (ptr->symptoms == 3)
				{
					cout << "Symptom is Tiredness\n";
				}
				cout << "Severity Level: " << ptr->severity << endl;

```

```
                cout << "Immune Level: " << ptr->immuneLevel << endl;
                cout << endl << endl;
                found = true;

            }
            ptr = ptr->next;
        }
        if (found == false)
        {
            cout << "No patient from ->"<<city<<"\n\n";
        }
```

Output:

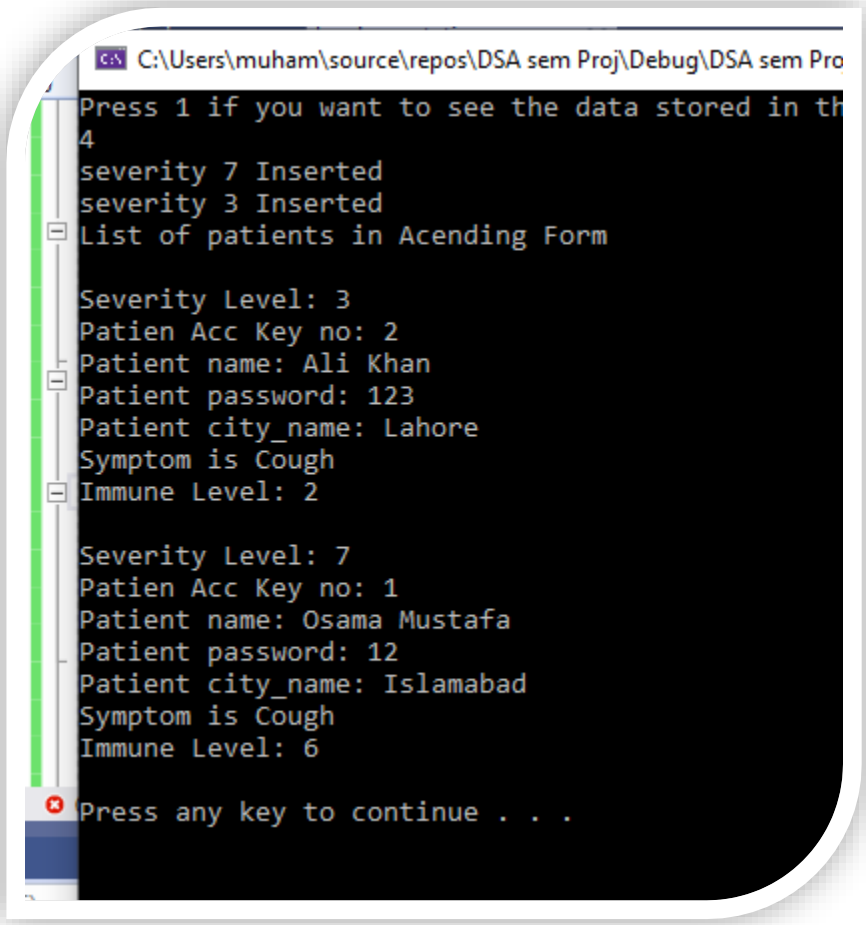


- Display_data_incorder(Node*)

Source Code:

```
BST b;
		Node* ptr1 = head;
		if (ptr1 == NULL)
		{
			cout << "Passed List is Empty\n";
		}
		else
		{
			while (ptr1->next != NULL)
			{
				b.insert(b.root, ptr1->keyno, ptr1->password, ptr1->userName, ptr1->cityName, ptr1->symptoms, ptr1->immuneLevel, ptr1->severity);
				ptr1 = ptr1->next;
			}
			cout << "List of patients in Acending Form\n\n";
			b.in_orderTraverse(b.root);

		}
```

Output:

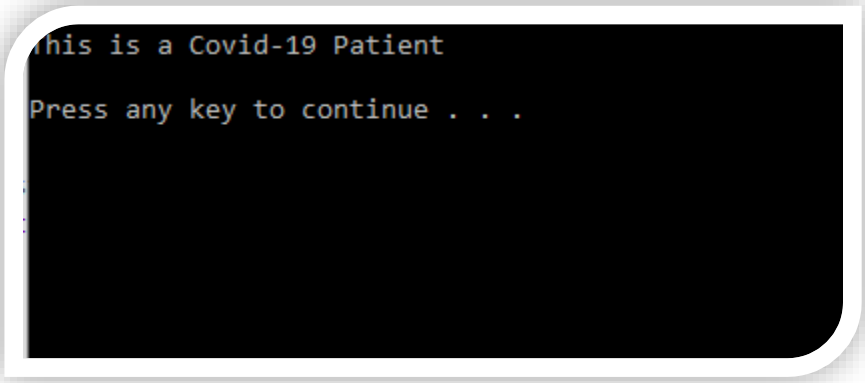


➢ Check_patient(Node*,int)

Source Code:

```
Node* ptr = head;
		while (ptr != NULL&&ptr->keyno!=dkeyno)
		{
			ptr = ptr->next;
		}
		if (ptr == NULL)
		{
			cout << "This patient with account no: " << dkeyno << " does not exist\n";
		}
		else if (ptr->keyno == dkeyno)
		{
			if (ptr->symptoms > 0 && ptr->severity > 2 && ptr->immuneLevel < 4)
			{
				cout << "This is a Covid-19 Patient\n\n";
			}
			else
			{
				cout << "The patient is not yet a potential Covid Patient\n\n";
			}

		}
```

Output:



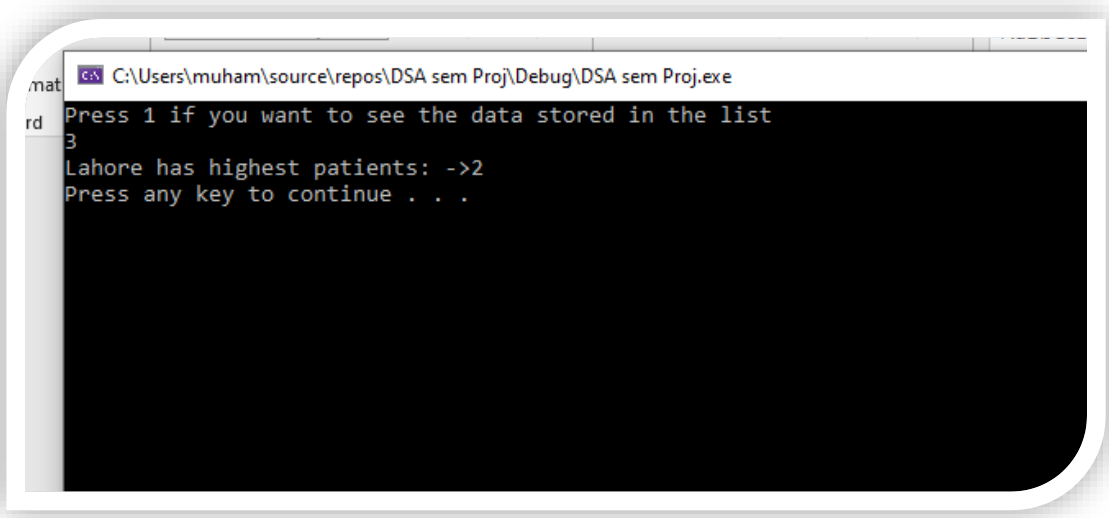
➢ CCount():

Source Code:

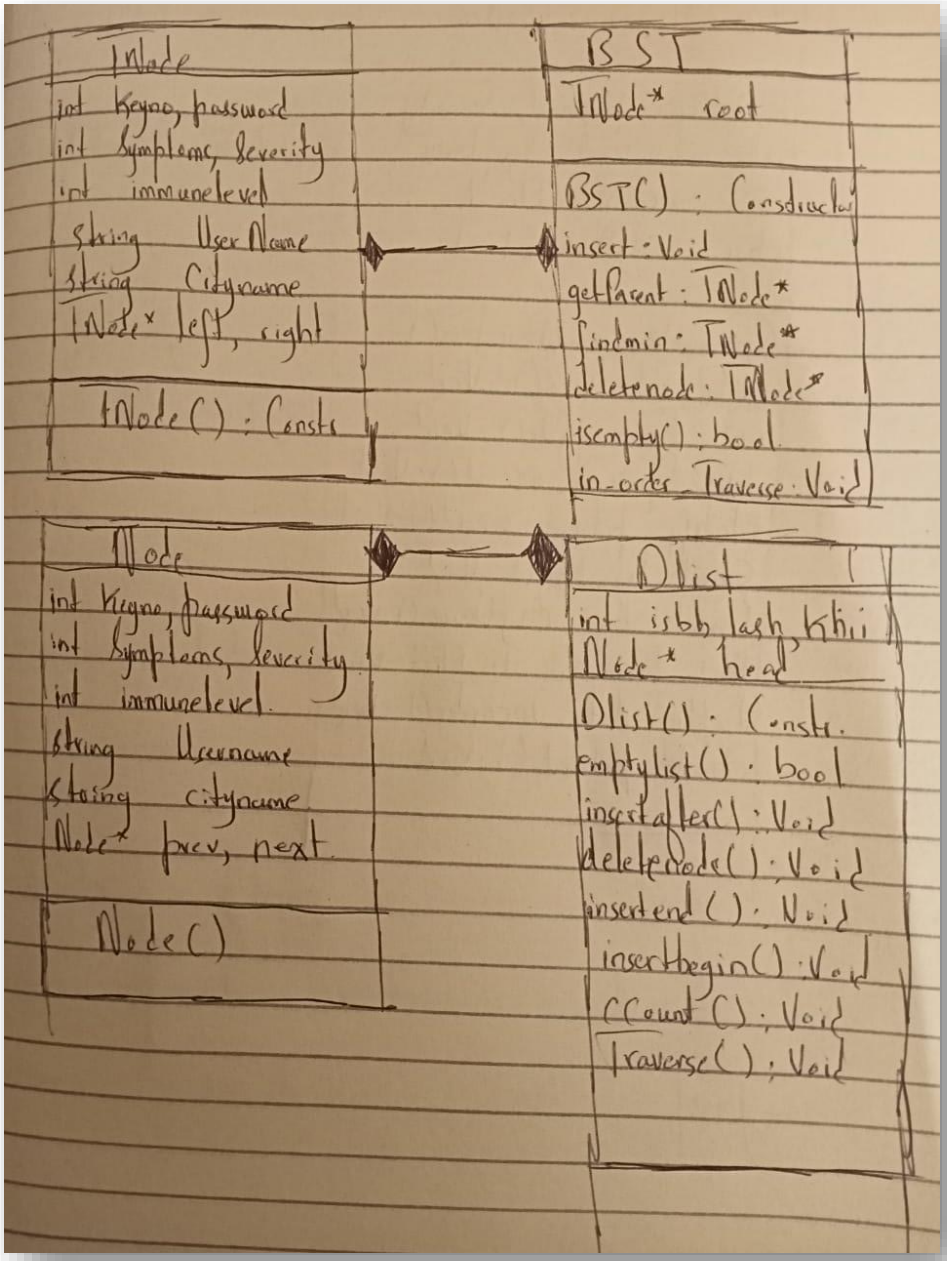
```
void ccount()
	{
		int max = isbb;
		int dec = lash;
		if (lash < khii)
		{
			dec = khii;
```

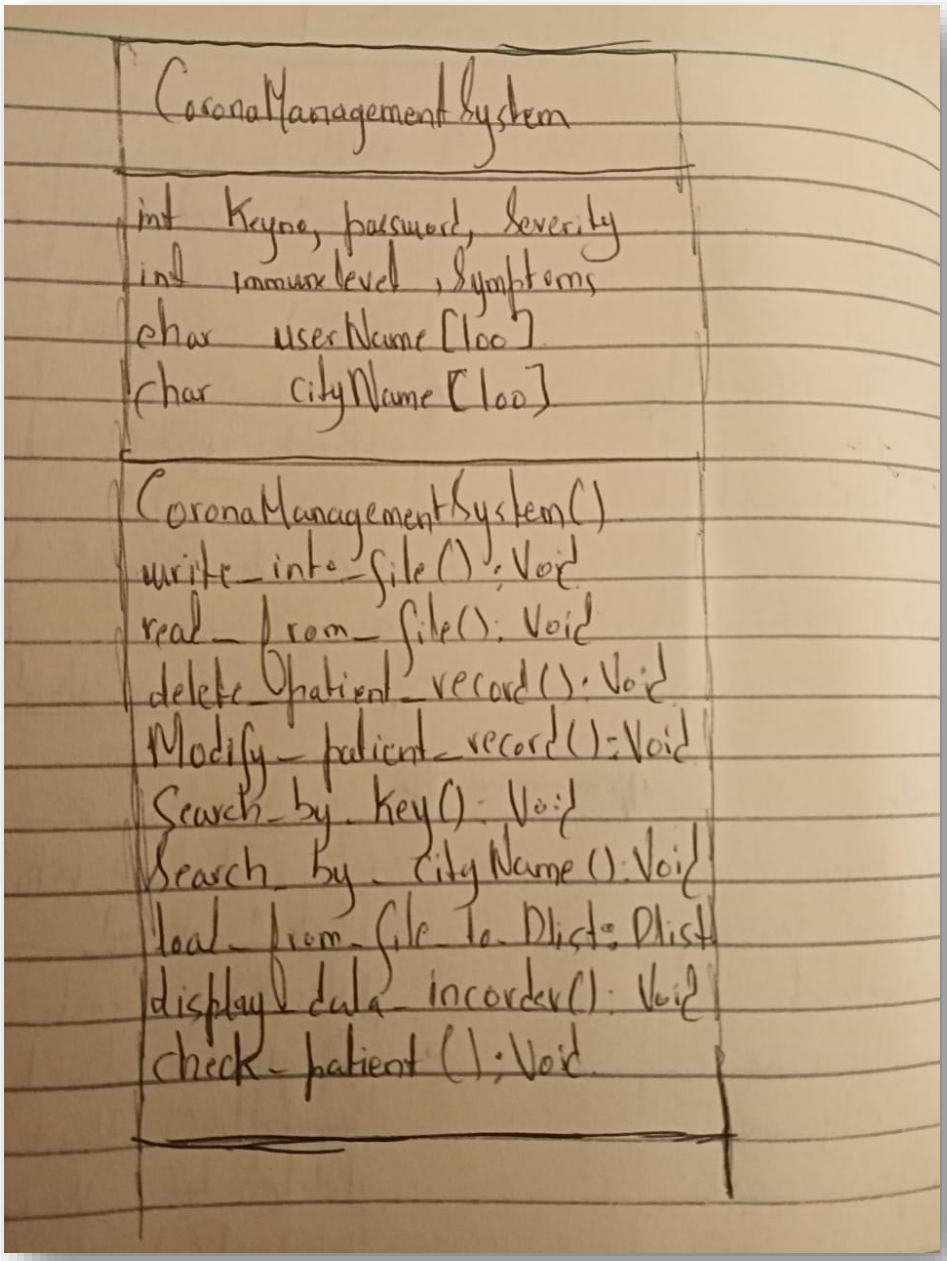
```
		}
		if (max < dec)
		{
			max = dec;
		}
		if (max == isbb)
		{
			cout << "Islamabad has highest patients: ->" << max << endl;
		}
		else if (max == lash)
		{
			cout << "Lahore has highest patients: ->" << max << endl;
		}
		else if (max == isbb)
		{
			cout << "Karachi has highest patients: ->" << max << endl;
		}

	}
```

Output:



UML:


TNode
int keyno, password
int Symptoms, Severity
int immunelevel
string UserName
string Cityname
TNode * left, right
tNode() : Constr
BST
TNode* root
BST() : Constructor
insert : Void
getParent : TNode*
findmin : TNode*
deletenode : TNode*
isempty() : bool
in-order_Traverse : Void
Node
int keyno, password
int Symptoms, Severity
int immunelevel
string Username
string cityname
Node* prev, next
Node()
Dlist
int isbb lash, khii
Node * head
Dlist() : Constr.
emptylist() : bool
insertafter() : Void
deletenode() : Void
insertend() : Void
insertbegin() : Void
rCount() : Void
Traverse() : Void



Conclusion:

Using data structures, real-time console application was successfully made using multiple data structures and numerous efficient algorithms. All objectives were achieved.