

**KIET Group of Institutions, Ghaziabad**

***COMPUTER SCIENCE AND INFORMATION TECHNOLOGY***

**on**

**HOSPITAL MANAGEMENT SYSTEM**

**PROJECT BASED LEARNING**

**SUBJECT: DATA STRUCTURE USING C LAB**

**Submitted By:**

**ABHISHEK KUMAR- 2100290110006**

**ABHISHEK 2100290110005**

**ANSHU KUMAR KUSHWAHA - 2100290110030**

**ACKNOWLEDGEMENT**

## I’ve got this golden opportunity to express my kind gratitude and sincere thanks to my subject faculty **“Mr. VINAY KUMAR”**, Computer Science and Information Technology Department, **KIET GROUP OF INSTITUTIONS** for their kind support and necessary counselling in the preparation of this project report. I’m also indebted to each and every person responsible for the making up of this project directly or indirectly.

I must also acknowledge or deep debt of gratitude each one of my colleague who led this project come out in the way it is. It’s my hard work and untiring sincere efforts and mutual cooperation to bring out the project work. Last but not the least, I would like to thank my parents for their sound counselling and cheerful support. They have always inspired us and kept our spirit up.

**ABSTRACT:**

Hospital Management System is an organized computerized system designed and programmed to deal with day-to-day operations and management of hospital activities. The program can look after inpatients, outpatients, records, database treatments, status illness, billings in the pharmacy, and labs

**INTRODUCTION:**

Hospital Management System is a system enabling hospitals to manage information and data related to all aspects of healthcare – processes, providers, patients, and more, which in turn ensures that processes are completed swiftly and effectively.

**AIM :**

To allow the administrator of any organization to edit and find out the personal details of a patient and allows the patient to keep up to date his profile.

**BASIC REQUIREMENTS:**

Basic concept of c++ has been used in this implementation.

Especially using of linked list played a vital role to create HMS.

VS CODE is required for this implementation.

# CODE IMPLEMENTATION :

#include <iostream>

#include <cstdlib>

#include <string>

using namespace std;

struct Data {

    string name, address, disease, gender, description;

    int specialRoomNo, age;

};

struct Node {

    Data data;

    Node\* next;

};

Node\* insertFront(Node\* head, Data data)

{

    Node\* temp = new Node;

    temp->data = data;

    temp->next = head;

    head = temp;

    return head;

}

Node\* append(Node\* head, Data data) {

    Node\* temp = new Node;

    temp->data = data;

    temp->next = NULL;

    if (head == NULL)

    {

        head = temp;

        return head;

    }

    Node\* last = head;

    while (last->next != NULL)

    {

        last = last->next;

    }

    last->next = temp;

    clog << "Append completed" << endl;

    return head;

}

void changeNode(Node\* head, Data data, Data newData)

{

    while (head != NULL)

    {

        if (head->data.name == data.name)

        {

            head->data = newData;

            break;

        }

        head = head->next;

    }

}

void changeNode(Node\* head, string data, string newData)

{

    while (head != NULL)

    {

        if (head->data.name == data)

        {

            head->data.name = newData;

            break;

        }

        head = head->next;

    }

}

void printLinkedList(Node\* head)

{

    if (head == NULL)

    {

        cout << "Head is null" << endl;

        return;

    }

    while (head->next != NULL)

    {

        cout << "Name: " << head->data.name << ", Address: " << head->data.address

            << ", Gender: " << head->data.gender << ", Disease: " << head->data.disease

            << ", Description: " << head->data.description << ", Age: " << head->data.age

            << ", Specialist No: " << head->data.specialRoomNo << endl;

        head = head->next;

    }

    cout << "Name: " << head->data.name << ", Address: " << head->data.address

        << ", Gender: " << head->data.gender << ", Disease: " << head->data.disease

        << ", Description: " << head->data.description << ", Age: " << head->data.age

        << ", Specialist No: " << head->data.specialRoomNo << endl;

}

int listLength(Node\* head) {

    int temp = 0;

    if (head == NULL)

    {

        cout << "Node is empty" << endl;

        return 0;

    }

    while (head->next != NULL)

    {

        ++temp;

        head = head->next;

    }

    return temp + 1;

}

void delElement(Node\* head, int loc)

{

    Node\* temp = new Node;

    temp = head;

    if (head == NULL) {

        cout << "Nod is null" << endl;

        return;

    }

    for (int i = 1; i <= loc; ++i)

    {

        temp = temp->next;

        if (i < loc) {

            head = head->next;

        }//end of if statement

    }//end of for statement

    head->next = temp->next;

}

Node\* searchList(Node\* head, Data v) {

    if (head == NULL)

    {

        cout << "Node is empty returning null" << endl;

        return NULL;

    }

    int l = 1;

    while (head->next != NULL && head->data.name != v.name)

    {

        head = head->next;

        ++l;

    }

    cout << "Element found at location " << l << endl;

    return head;

}

Node\* searchList(Node\* head, string v) {

    if (head == NULL)

    {

        cout << "Node is empty returning null" << endl;

        return NULL;

    }

    int l = 1;

    while (head->next != NULL && head->data.name != v)

    {

        head = head->next;

        ++l;

    }

    cout << "Element found at location " << l << endl;

    return head;

}

Data inputPatients()

{

    string name, address, disease, gender, description;

    int specialRoomNo, age;

    Data p;

    cout << "Enter Patient Name: ";

    cin.ignore();//clearing buffer

    getline(cin, name);

    cout << "Enter Patient Address: ";

    getline(cin, address);

    cout << "Enter Patient Disease: ";

    getline(cin, disease);

    cout << "Enter Patient Gender: ";

    getline(cin, gender);

    cout << "Enter Disease Description: ";

    getline(cin, description);

    cout << "Enter Patient Special Room No.: ";

    cin >> specialRoomNo;

    cout << "Enter Patient Age: ";

    cin >> age;

    p.name = name;

    p.address = address;

    p.gender = gender;

    p.description = description;

    p.specialRoomNo = specialRoomNo;

    p.age = age;

    clog << "Completed input operation" << endl;

    return p;

}//end of inputPatients function

//============================================================================

int main()

{

    Node\* head = NULL;

    Data patient;

    string nameToSearch;

    string oldName, newName;

    int op;

    cout << "\nCTRL + Z-To exit\n1-Add Patient\n2-Del Patient\n3-Search by Name\n4-List Length"

        << "\n5-print List\n6-Change Patient Name\n7-Insert new at front" << endl;

    while (cin >> op)

    {

        switch (op) {

        case 1:

            cout << "Enter Patient Details Below" << endl;

            patient = inputPatients();

            head = append(head, patient);

            break;

        case 2:

            if (listLength(head) < 2)

            {

                cout << "Length is less then two.\nTerminating program" << endl;

                exit(1);

            }

            else {

                cout << "Enter location where you want to delete a patient, at least three patients must be in list? ";

                int l;

                cin >> l;

                delElement(head, l - 1);

            }//end of else of case 2

            break;

        case 3:

            cout << "Enter name to search patient: ";

            cin.ignore();

            getline(cin, nameToSearch);

            searchList(head, nameToSearch);

            break;

        case 4:

            cout << "You have " << listLength(head) << " Patients in your Hospital." << endl;

            break;

        case 5:

            printLinkedList(head);

            break;

        case 6:

            cin.ignore();

            cout << "Enter old name ";

            getline(cin, oldName);

            cout << "Enter New Name ";

            getline(cin, newName);

            changeNode(head, oldName, newName);

            break;

        case 7:

            patient = inputPatients();

            head = insertFront(head, patient);

            break;

        default:

            cout << "Wrong option Selected" << endl;

        }

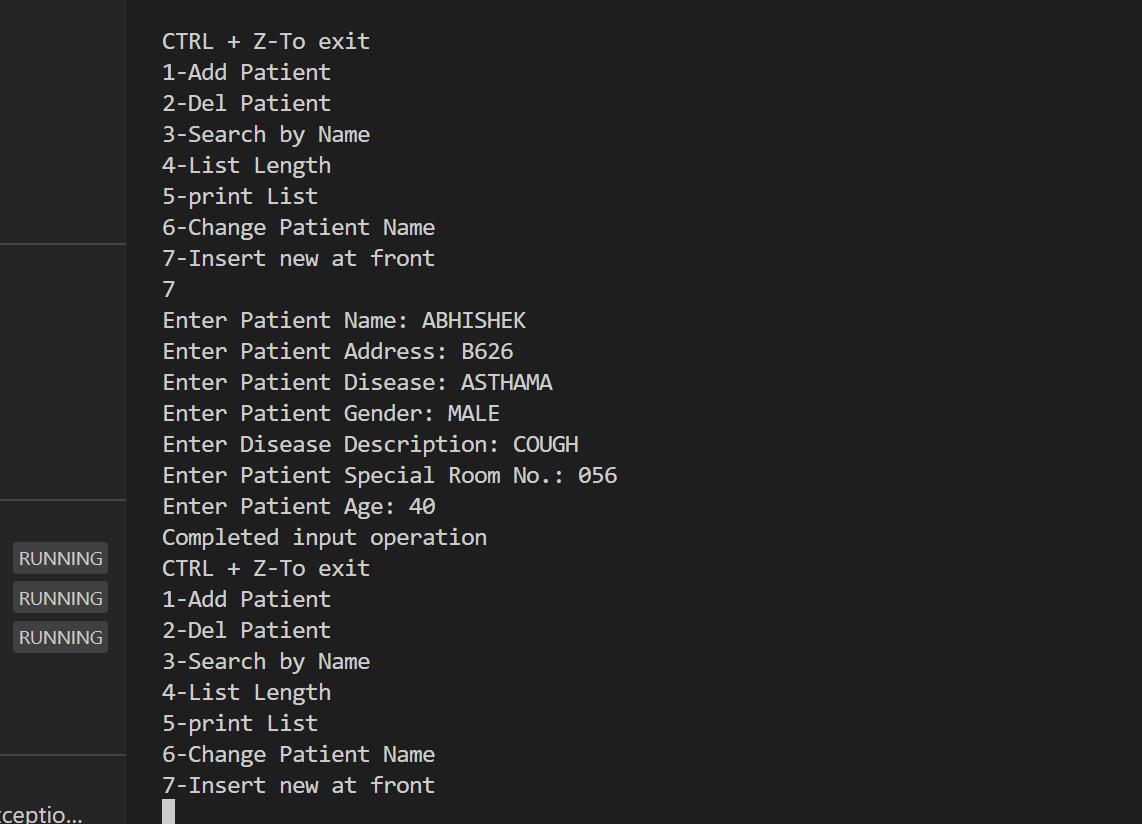
        cout << "CTRL + Z-To exit\n1-Add Patient\n2-Del Patient\n3-Search by Name\n4-List Length"

            << "\n5-print List\n6-Change Patient Name\n7-Insert new at front" << endl;

    }

}

# RESULT:



**TIME COMPLEXITY:** O(logn)

# SPACE COMPLEXITY: O(1)

**REFERENCE:**

1.https://www.youtube.com/watch?v=EnN6zQ0fbBk&ab\_channel=AdroitInfosystems