# DATA STRUCTURE AND ALGORITHM

# (CSC-201)

# PROJECT TITLE: HOSPITAL MANAGEMENT SYSTEM

|  |
| --- |
| Student Name: Syed Saad Akhtar 22sp-029-cs |
| Student Name: Kinza Fatima 22sp-032-cs |
| Instructor name: Qazi Abdul Samad |

**Video link:** [**https://drive.google.com/file/d/1XtamKS7nTaZW7IK2M7-L\_8IItDuimFtm/view?usp=sharing**](https://drive.google.com/file/d/1XtamKS7nTaZW7IK2M7-L_8IItDuimFtm/view?usp=sharing)

# Introduction:

This program is called the Hospital Management System was created to simplify and automate hospital management. In a hospital setting, it offers functionality to manage doctors, patients, appointments, and other associated tasks. An overview of the Hospital Management System created with the Java programming language is provided in this project report.

# Objective:

The Hospital Management System project's goal is to provide a user-friendly and effective system that will allow hospital employees to successfully manage and organise patient and physician information. The system seeks to automate a number of processes, including adding new physicians and patients, examining physician and patient information, looking up patients by name, and categorising patient records according to various factors.

# Features:

The Hospital Management System includes the following features:

- Add new doctors: Allows the user to add details of new doctors, including name, age, gender, and specialization.

- Add new patients: Enables the user to add information about new patients, including name, age, gender, phone number, blood group, and disease.

- View all doctors: Displays a list of all doctors registered in the system, including their details.

- View all patients: Shows a list of all patients registered in the system, along with their details.

- Search patient by name: Allows the user to search for a patient by entering their name and displays matching patient records.

- Sort patients by name: Sorts the patient records in ascending order based on their names.

- Sort patients by age: Sorts the patient records in ascending order based on their ages.

- Sort patients by gender: Sorts the patient records in ascending order based on their genders.

- Sort patients by blood group: Sorts the patient records in ascending order based on their blood groups.

# Implementation:

The Hospital Management System is implemented using object-oriented programming principles in Java. The system consists of two main classes: "Doctor" and "Patient." The "Doctor" class represents a doctor and contains attributes such as name, age, gender, and specialization. The "Patient" class represents a patient and includes attributes like name, age, gender, phone number, blood group, and disease.

The system utilizes ArrayLists to store and manage collections of doctors and patients. It provides methods to add new doctors and patients, view all doctors and patients, search patients by name, and sort patients based on different criteria. The sorting algorithms used are insertion sort for sorting patients by age, gender, and blood group, and a custom comparison method for sorting patients by name.

The main class, "Hospital," acts as the entry point for the system. It contains the main menu and user interface for interacting with the system. Users can choose different options from the menu to perform various operations.

# Conclusion:

An efficient method of managing hospital-related activities is offered by the hospital management system. Searching for patients and sorting patient data are made easier, as well as adding and viewing doctor and patient information. The technology results in increased patient care and simplified administrative work by improving efficiency and accuracy in managing hospital information.

The project shows how to create a useful and user-friendly system using the Java programming language and object-oriented programming principles. It may be improved even more by adding extra functions like appointment scheduling, billing administration, and report production.

Overall, the Hospital Management System project provides hospital workers and managers with a useful tool for effectively organising and managing hospital activities.

# Code:

Doctor class:

import java.util.ArrayList;

public class Doctor {

    private String doctorName;

    private int doctorAge;

    private String doctorGender;

    private String doctorSpecialization;

    public Doctor(String doctorName, int doctorAge, String doctorGender, String doctorSpecialization) {

        this.doctorName = doctorName;

        this.doctorAge = doctorAge;

        this.doctorGender = doctorGender;

        this.doctorSpecialization = doctorSpecialization;

    }

   public static void viewAllDoctors(ArrayList<Doctor> doctors) {

    if (doctors.isEmpty()) {

        System.out.println("\n          No doctor records. The list is empty.");

    } else {

        System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        System.out.println("                                                List of All Doctors");

        System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        for (int i = 0; i < doctors.size(); i++) {

            Doctor doctor = doctors.get(i);

            System.out.println("\nDoctor Name: " + doctor.doctorName+ "    Age: " + doctor.doctorAge+ "    Gender: " + doctor.doctorGender+ "    Specialization: " + doctor.doctorSpecialization);

            System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        }

    }

}

}

Patient class:

import java.util.ArrayList;

public class patient {

    private String name;

    private int age;

    private String gender;

    private String phoneNumber;

    private String bloodGroup;

    private String disease;

    public patient(String name, int age, String gender, String phoneNumber, String bloodGroup, String disease) {

        this.name = name;

        this.age = age;

        this.gender = gender;

        this.phoneNumber = phoneNumber;

        this.bloodGroup = bloodGroup;

        this.disease = disease;

    }

    public static void patientdetail(ArrayList<patient> patients){

        for (int i = 0; i < patients.size(); i++) {

            patient patient = patients.get(i);

            System.out.println("\nPatient Name: " + patient.name + "    Age: " + patient.age+ "  Gender: " + patient.gender + "  Phone Number: " + patient.phoneNumber+"  Blood Group: " + patient.bloodGroup +"  Disease: " + patient.disease);

            System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        }

    }

 public static void viewAllPatients(ArrayList<patient> patients) {

    if (patients.isEmpty()) {

        System.out.println();

        System.out.println("            No patient records. The list is empty.");

    } else {

        System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        System.out.println("\t\t\t\t\t\t    List of All Patients");

        System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        for (int i = 0; i < patients.size(); i++) {

            patient patient = patients.get(i);

            System.out.println("\nPatient Name: " + patient.name + "    Age: " + patient.age+ "  Gender: " + patient.gender + "  Phone Number: " + patient.phoneNumber+"  Blood Group: " + patient.bloodGroup +"  Disease: " + patient.disease);

            System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        }

    }

}

    // SORT PATIENTS insertion sort

    public static void sortByName(ArrayList<patient> patients) {

        int n = patients.size();

        for (int i = 1; i < n; i++) {

            patient key = patients.get(i);

            int j = i - 1;

            while (j >= 0 && compareNames(patients.get(j).name, key.name) > 0) {

                patients.set(j + 1, patients.get(j));

                j--;

            }

            patients.set(j + 1, key);

        }

    }

     //print sorted patients by name

    public static void printSortedPatientsByName(ArrayList<patient> patients) {

        sortByName(patients);

        patientdetail(patients);

    }

    // Method to compare patient names

   private static int compareNames(String name1, String name2) {

    // Convert both names to lowercase to make the comparison case-insensitive

    String lowercaseName1 = name1.toLowerCase();

    String lowercaseName2 = name2.toLowerCase();

    // Compare the lowercase names character by character

    for (int i = 0; i < lowercaseName1.length() && i < lowercaseName2.length(); i++) {

        char char1 = lowercaseName1.charAt(i);

        char char2 = lowercaseName2.charAt(i);

        if (char1 < char2) {

            return -1; // name1 should come before name2

        } else if (char1 > char2) {

            return 1; // name1 should come after name2

        }

    }

    // If both names have common characters up to the length of the shorter name,

    // the shorter name should come before the longer name

    if (lowercaseName1.length() < lowercaseName2.length()) {

        return -1;

    } else if (lowercaseName1.length() > lowercaseName2.length()) {

        return 1;

    }

    return 0; // Both names are equal

}

    //sort patients by age using insertion sort

public static void sortPatientsByAge(ArrayList<patient> patients) {

    int n = patients.size();

    for (int i = 1; i < n; i++) {

        patient key = patients.get(i);

        int j = i - 1;

        while (j >= 0 && patients.get(j).age > key.age) {

            patients.set(j + 1, patients.get(j));

            j--;

        }

        patients.set(j + 1, key);

    }

    System.out.println("\nPatients sorted by age:");

    patientdetail(patients);

}

//sort patients by gender using insertion sort

public static void sortPatientsByGender(ArrayList<patient> patients) {

    int n = patients.size();

    for (int i = 1; i < n; i++) {

        patient key = patients.get(i);

        int j = i - 1;

        while (j >= 0 && compareNames(patients.get(j).gender, key.gender) > 0) {

            patients.set(j + 1, patients.get(j));

            j--;

        }

        patients.set(j + 1, key);

    }

    System.out.println("\nPatients sorted by gender:");

    patientdetail(patients);

}

    // Sort patients by blood group using insertion sort

    public static void sortPatientsByBloodGroup(ArrayList<patient> patients) {

        int n = patients.size();

        for (int i = 1; i < n; i++) {

            patient key = patients.get(i);

            int j = i - 1;

            while (j >= 0 && compareBloodGroups(patients.get(j).bloodGroup, key.bloodGroup) > 0) {

                patients.set(j + 1, patients.get(j));

                j--;

            }

            patients.set(j + 1, key);

        }

        patientdetail(patients);

    }

private static int compareBloodGroups(String bloodGroup1, String bloodGroup2) {

        return bloodGroup1.compareTo(bloodGroup2);

    }

    public static void searchPatientByName(ArrayList<patient> patients, String name) {

    ArrayList<patient> matchingPatients = new ArrayList<>();

    for (int i = 0; i < patients.size(); i++) {

        patient p = patients.get(i);

        if (p.name.equalsIgnoreCase(name)) {

            matchingPatients.add(p);

        }

    }

    if (matchingPatients.isEmpty()) {

        System.out.println("No matching patients found.");

    } else {

        System.out.println("Matching patients:");

        patientdetail(matchingPatients);

    }

}

}

Main Hospital class:

            // Syed saad akhtar 22sp-029-cs

            // Kinza Fatima     22sp-032-cs

import java.util.Scanner;

import java.util.ArrayList;

class Hospital {

    public static void main(String[] args) {

        boolean exit = false;

        boolean running = true;

        ArrayList<patient> patients = new ArrayList<>();

        ArrayList<Doctor> doctors = new ArrayList<>();

        Scanner input = new Scanner(System.in); // object of scanner to take inputs

        System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        System.out.println("\t\t\t\t\t\t    \*WELCOME TO HOSPITAL MANAGEMENT SYSTEM\*  ");

        System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

        while (!exit) {

            System.out.println("\n                    1.   Add New Doctor");

            System.out.println("                    2.   Add New Patient");

            System.out.println("                    3.   View All Doctors");

            System.out.println("                    4.   View All Patients");

            System.out.println("                    5.   Search Patient by Name");

            System.out.println("                    6.   Sort Patients by Names");

            System.out.println("                    7.   Sort Patients by Age");

            System.out.println("                    8.   Sort Patients by Gender");

            System.out.println("                    9.   Sort Patients by Blood Group");

            System.out.println("                    10.  Exit");

            System.out.print("\n                Please Enter your choice: ");

            int choice = input.nextInt();

            switch (choice) {

                case 1:

                    // Add new Doctor

                    do {

                        System.out.print("\nEnter Doctor's Name: ");

                        String doctorName = input.next();

                        System.out.print("Enter Doctor's Age: ");

                        int doctorAge = input.nextInt();

                        System.out.print("Enter Doctor's Gender (M/F): ");

                        String doctorGender = input.next();

                        System.out.print("Enter Doctor's Specialization: ");

                        String doctorSpecialization = input.next();

                        Doctor newDoctor = new Doctor(doctorName, doctorAge, doctorGender, doctorSpecialization);

                        doctors.add(newDoctor);

                        System.out.println("\nDoctor added successfully!");

                        System.out.print("\nDo you want to add more doctors? (yes/no): ");

                        String continueChoice = input.next();

                        if (continueChoice.equalsIgnoreCase("no")) {

                            System.out.println("\nReturning to the main menu.");

                            break; // Exit the loop if the user chooses not to add more doctors

                        } else {

                            System.out.println("Continuing to add new doctors.");

                        }

                    } while (true);

                    break;

                case 2:

                    // Add new Patient

                    do {

                        System.out.print("\nEnter Patient's Name: ");

                        String name = input.next();

                        System.out.print("Enter Patient's Age: ");

                        int age = input.nextInt();

                        System.out.print("Enter Patient's Gender (M/F): ");

                        String gender = input.next();

                        System.out.print("Enter Patient's Phone Number: ");

                        String phoneNumber = input.next();

                        System.out.print("Enter Patient's Blood Group: ");

                        String bloodGroup = input.next();

                        System.out.print("Enter Patient's Disease: ");

                        String disease = input.next();

                        patient newPatient = new patient(name, age, gender, phoneNumber, bloodGroup, disease);

                        patients.add(newPatient);

                        System.out.println("\nPatient added successfully!");

                        System.out.print("\nDo you want to add more patients? (yes/no): ");

                        String ch = input.next();

                        if (ch.equalsIgnoreCase("no")) {

                            running = false;

                            System.out.println("\nReturning to the main menu.");

                            break; // Exit the loop if the user chooses not to add more patients

                        } else {

                            System.out.println("Continuing to add new patients.");

                        }

                    } while (running);

                    break;

                case 3:

                    // View all Doctors

                    Doctor.viewAllDoctors(doctors);

                    break;

                case 4:

                    // List all Patients

                    patient.viewAllPatients(patients);

                    break;

                case 5:

                    // Search Patient by Name

                    System.out.print("\nEnter the name of the patient to search: ");

                    String searchName = input.next();

                    patient.searchPatientByName(patients, searchName);

                    break;

                case 6:

                    // Sort Patients by Names

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    System.out.println("\t\t\t\t\t\t    List of Patients Sorted by Names");

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    patient.printSortedPatientsByName(patients);

                    break;

                case 7:

                    // Sort Patients by Age

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    System.out.println("\t\t\t\t\t\t    List of Patients Sorted by Age");

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    patient.sortPatientsByAge(patients);

                    break;

                case 8:

                    // Sort Patients by Gender

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    System.out.println("\t\t\t\t\t\t    List of Patients Sorted by Gender");

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    patient.sortPatientsByGender(patients);

                    break;

                case 9:

                    // Sort Patients by Blood Group

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    System.out.println("\t\t\t\t\t\t    List of Patients Sorted by Blood Group");

                    System.out.println("-------------------------------------------------------------------------------------------------------------------------------------");

                    patient.sortPatientsByBloodGroup(patients);

                    break;

                case 10:

                    // Exit

                    System.out.println("\nExiting...");

                    exit = true;

                    break;

                default:

                    System.out.println("Invalid choice");

            }

        }

    }

}

https://drive.google.com/file/d/1XtamKS7nTaZW7IK2M7-L\_8IItDuimFtm/view?usp=sharing