**Department of Computing**

**CS 212: Object Oriented Programming**

**Lab 08: Function Overriding**

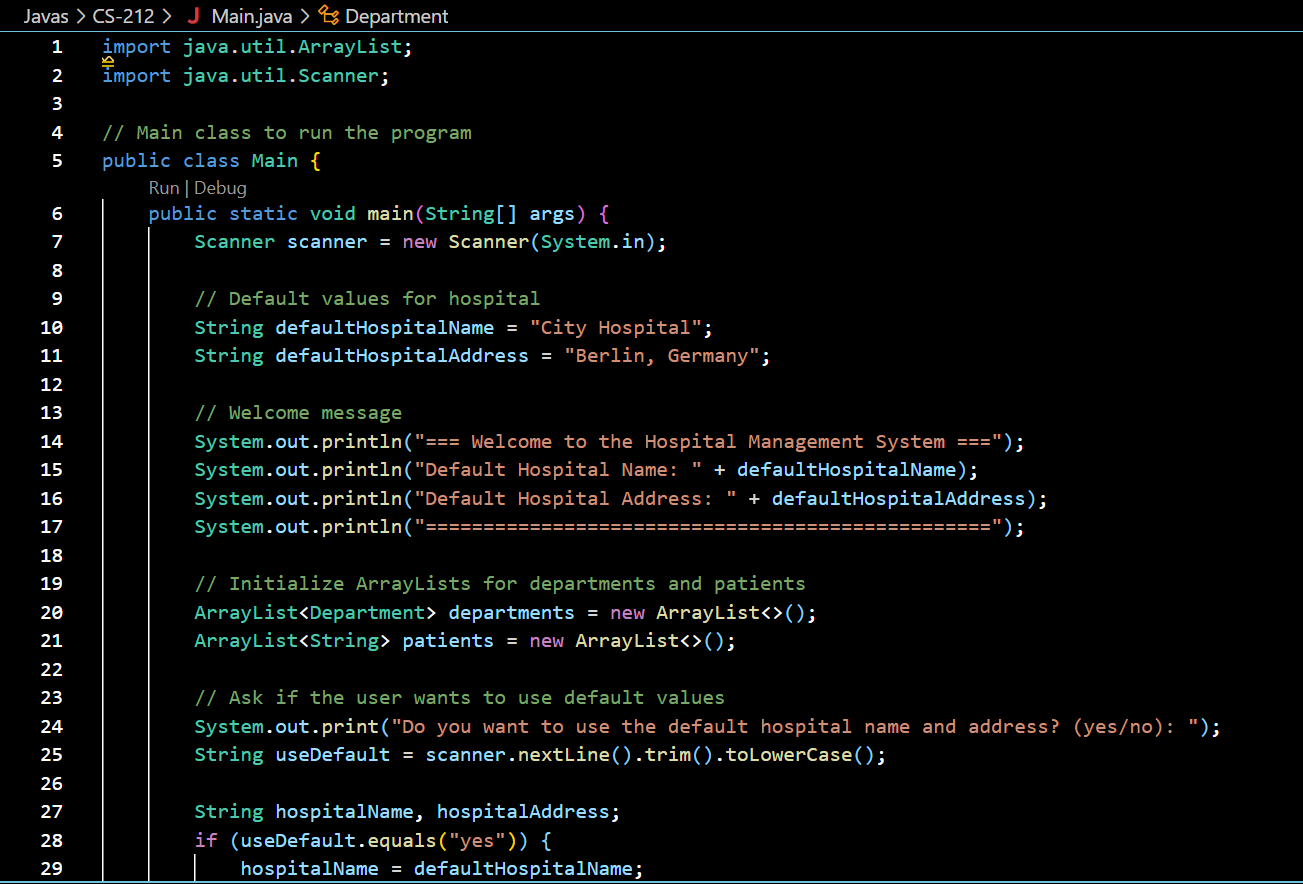
**Class: BSDS -02A**

**AILYA ZAINAB**

**( 5 2 3 5 0 6 )**

**HOSPITAL MANAGEMENT SYSTEM IN JAVA OOP**

**INPUT:**



            hospitalAddress = defaultHospitalAddress;

        } else {

            System.out.print("Enter Hospital Name: ");

            hospitalName = scanner.nextLine();

            System.out.print("Enter Hospital Address: ");

            hospitalAddress = scanner.nextLine();

        }

        // Create a Hospital object

        Hospital hospital = new Hospital(hospitalName, hospitalAddress, departments, patients);

        // Add departments to the hospital

        System.out.println("\n=== Add Departments ===");

        while (true) {

            System.out.print("Enter Department Name (or type 'exit' to finish adding departments): ");

            String deptName = scanner.nextLine();

            if (deptName.equalsIgnoreCase("exit")) {

                break;

            }

            Department department = new Department(deptName);

            hospital.addDepartment(department);

            // Add team members to the department

            System.out.println("\n=== Add Team Members to " + deptName + " ===");

            while (true) {

                System.out.print("Enter Team Member Role (doctor/nurse) or type 'exit' to finish: ");

                String role = scanner.nextLine().trim().toLowerCase();

                if (role.equals("exit")) {

                    break;

                }

                System.out.print("Enter Team Member Name: ");

                String name = scanner.nextLine();

                System.out.print("Enter Team Member ID: ");

                String id = scanner.nextLine();

                System.out.print("Enter Team Member Gender (male/female): ");

                String gender = scanner.nextLine();

                System.out.print("Enter Team Member Join Date: ");

                String joinDate = scanner.nextLine();

                if (role.equals("doctor")) {

                    System.out.print("Enter Doctor Specialty (intern/senior/surgeon): ");

                    String specialty = scanner.nextLine();

                    Doctor doctor = new Doctor(name, id, gender, joinDate, specialty);

                    if (specialty.equals("intern")) {

                        System.out.print("Enter Supervisor's Name: ");

                        String supervisorName = scanner.nextLine();

                        doctor.setSupervisor(supervisorName);

                    }

                    department.addTeamMember(doctor);

                } else if (role.equals("nurse")) {

                    Nurse nurse = new Nurse(name, id, gender, joinDate);

                    department.addTeamMember(nurse);

                }

            }

        }

        // Add patients to the hospital

        System.out.println("\n=== Add Patients ===");

        while (true) {

            System.out.print("Enter Patient Name (or type 'exit' to finish adding patients): ");

            String name = scanner.nextLine();

            if (name.equalsIgnoreCase("exit")) {

                break;

            }

            System.out.print("Enter Patient Birth Date: ");

            String birthDate = scanner.nextLine();

            System.out.print("Enter Patient Gender (male/female): ");

            String gender = scanner.nextLine();

            System.out.print("Enter Patient Acceptance Date: ");

            String acceptanceDate = scanner.nextLine();

            System.out.print("Enter Patient Diagnosis: ");

            String diagnosis = scanner.nextLine();

            System.out.print("Enter Doctor Treating the Patient: ");

            String doctorName = scanner.nextLine();

            System.out.print("Enter Number of Days in Hospital: ");

            int daysInHospital = Integer.parseInt(scanner.nextLine());

            Patient patient = new Patient(name, birthDate, gender, acceptanceDate, diagnosis, doctorName, daysInHospital);

            hospital.addPatient(patient);

        }

        // Display all departments and their team members

        System.out.println("\n=== Hospital Details ===");

        hospital.displayDepartments();

        // Goodbye message

        System.out.println("\n=== Thank you for using the Hospital Management System! ===");

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

        scanner.close();

    }

}

// Hospital class

class Hospital {

    private String name;

    private String address;

    private ArrayList<Department> departments; // List of departments

    private ArrayList<Patient> patients; // List of patients

    // Constructor

    Hospital(String name, String address, ArrayList<Department> departments, ArrayList<Patient> patients) {

        this.name = name;

        this.address = address;

        this.departments = departments;

        this.patients = patients;

    }

    // Method to add a patient

    public void addPatient(Patient patient) {

        patients.add(patient);

        System.out.println("Patient added: " + patient.getName());

    }

    // Method to add a department

    public void addDepartment(Department department) {

        departments.add(department);

        System.out.println("Department added: " + department.getName());

    }

    // Method to display all departments and their staff

    public void displayDepartments() {

        System.out.println("Hospital Name: " + name);

        System.out.println("Hospital Address: " + address);

        System.out.println("\nDepartments and Their Team Members:");

        for (Department department : departments) {

            System.out.println("\nDepartment: " + department.getName());

            department.displayTeamMembers();

        }

        System.out.println("\nPatients in the Hospital:");

        for (Patient patient : patients) {

            System.out.println(patient.getName() + " - Diagnosis: " + patient.getDiagnosis() + " - Doctor: " + patient.getDoctorName());

        }

    }

}

// Department class

class Department {

    private String name;

    private ArrayList<TeamMember> staff; // List of team members

    // Constructor

    Department(String name) {

        this.name = name;

        this.staff = new ArrayList<>(); // Initialize the staff list

    }

    // Method to add a team member

    public void addTeamMember(TeamMember member) {

        if (staff.size() < 5) { // Check if staff limit is not exceeded

            staff.add(member);

            System.out.println("Team member added: " + member.getName());

        } else {

            System.out.println("Cannot add more than 5 team members to " + name);

        }

    }

    // Method to display all team members

    public void displayTeamMembers() {

        System.out.println("Team Members:");

        for (TeamMember member : staff) {

            System.out.println(member.getDetails());

        }

    }

    public String getName() {

        return name;

    }

}

// TeamMember class (Superclass for Doctor and Nurse)

abstract class TeamMember {

    private String name;

    private String id;

    private String gender;

    private String joinDate;

    // Constructor

    TeamMember(String name, String id, String gender, String joinDate) {

        this.name = name;

        this.id = id;

        this.gender = gender;

        this.joinDate = joinDate;

    }

    public String getName() {

        return name;

    }

    public String getDetails() {

        return "Name: " + name + ", ID: " + id + ", Gender: " + gender + ", Join Date: " + joinDate;

    }

    public abstract void treatPatient(Patient patient);

}

// Doctor class (Subclass of TeamMember)

class Doctor extends TeamMember {

    private String specialty;

    private String supervisor; // Only for interns

    private ArrayList<Patient> patients; // List of patients treated by the doctor

    // Constructor

    Doctor(String name, String id, String gender, String joinDate, String specialty) {

        super(name, id, gender, joinDate);

        this.specialty = specialty;

        this.patients = new ArrayList<>();

    }

    public void setSupervisor(String supervisor) {

        this.supervisor = supervisor;

    }

    public void addPatient(Patient patient) {

        patients.add(patient);

    }

    public void checkPatientReport(Patient patient) {

        System.out.println("Patient: " + patient.getName() + ", Diagnosis: " + patient.getDiagnosis());

    }

    @Override

    public void treatPatient(Patient patient) {

        if (specialty.equals("intern")) {

            System.out.println("Intern " + getName() + " is treating patient " + patient.getName() + " under supervision of " + supervisor);

        } else if (specialty.equals("senior")) {

            System.out.println("Senior Doctor " + getName() + " is treating patient " + patient.getName());

        } else if (specialty.equals("surgeon")) {

            System.out.println("Surgeon " + getName() + " is performing surgery on patient " + patient.getName());

        }

    }

    @Override

    public String getDetails() {

        String details = super.getDetails() + ", Specialty: " + specialty;

        if (specialty.equals("intern")) {

            details += ", Supervisor: " + supervisor;

        }

        return details;

    }

}

// Nurse class (Subclass of TeamMember)

class Nurse extends TeamMember {

    // Constructor

    Nurse(String name, String id, String gender, String joinDate) {

        super(name, id, gender, joinDate);

    }

    @Override

    public void treatPatient(Patient patient) {

        System.out.println("Nurse " + getName() + " is assisting in the treatment of patient " + patient.getName());

    }

}

// Patient class

class Patient {

    private String name;

    private String birthDate;

    private String gender;

    private String acceptanceDate;

    private String diagnosis;

    private String doctorName;

    private int daysInHospital;

    // Constructor

    Patient(String name, String birthDate, String gender, String acceptanceDate, String diagnosis, String doctorName, int daysInHospital) {

        this.name = name;

        this.birthDate = birthDate;

        this.gender = gender;

        this.acceptanceDate = acceptanceDate;

        this.diagnosis = diagnosis;

        this.doctorName = doctorName;

        this.daysInHospital = daysInHospital;

    }

    public String getName() {

        return name;

    }

    public String getDiagnosis() {

        return diagnosis;

    }

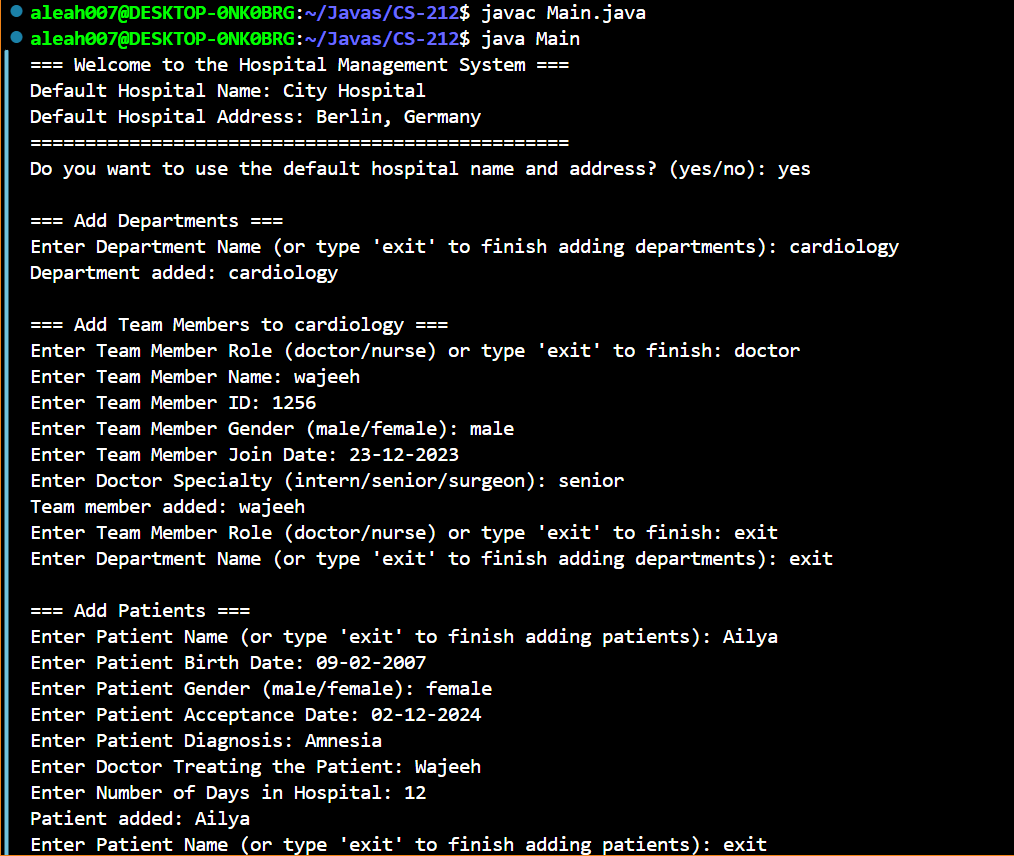
    public String getDoctorName() {

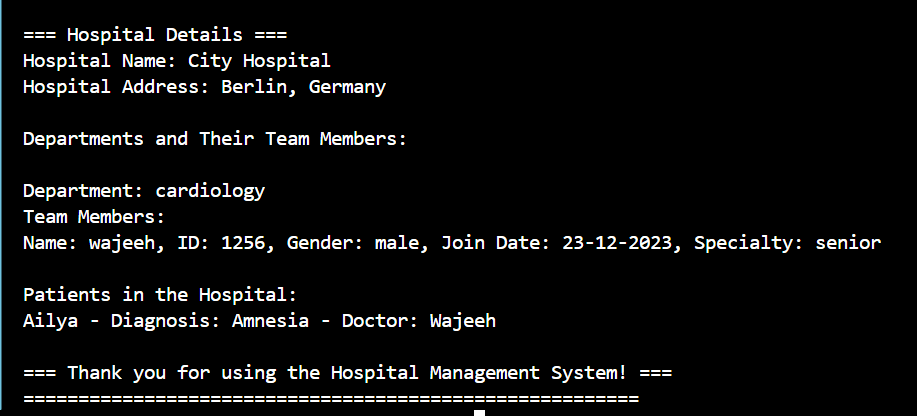
        return doctorName;

    }

}

**OUTPUT:**





**CODE:**

import java.util.ArrayList;

import java.util.Scanner;

// Main class to run the program

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Default values for hospital

String defaultHospitalName = "City Hospital";

String defaultHospitalAddress = "Berlin, Germany";

// Welcome message

System.out.println("=== Welcome to the Hospital Management System ===");

System.out.println("Default Hospital Name: " + defaultHospitalName);

System.out.println("Default Hospital Address: " + defaultHospitalAddress);

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

// Initialize ArrayLists for departments and patients

ArrayList<Department> departments = new ArrayList<>();

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

// Ask if the user wants to use default values

System.out.print("Do you want to use the default hospital name and address? (yes/no): ");

String useDefault = scanner.nextLine().trim().toLowerCase();

String hospitalName, hospitalAddress;

if (useDefault.equals("yes")) {

hospitalName = defaultHospitalName;

hospitalAddress = defaultHospitalAddress;

} else {

System.out.print("Enter Hospital Name: ");

hospitalName = scanner.nextLine();

System.out.print("Enter Hospital Address: ");

hospitalAddress = scanner.nextLine();

}

// Create a Hospital object

Hospital hospital = new Hospital(hospitalName, hospitalAddress, departments, patients);

// Add departments to the hospital

System.out.println("\n=== Add Departments ===");

while (true) {

System.out.print("Enter Department Name (or type 'exit' to finish adding departments): ");

String deptName = scanner.nextLine();

if (deptName.equalsIgnoreCase("exit")) {

break;

}

Department department = new Department(deptName);

hospital.addDepartment(department);

// Add team members to the department

System.out.println("\n=== Add Team Members to " + deptName + " ===");

while (true) {

System.out.print("Enter Team Member Role (doctor/nurse) or type 'exit' to finish: ");

String role = scanner.nextLine().trim().toLowerCase();

if (role.equals("exit")) {

break;

}

System.out.print("Enter Team Member Name: ");

String name = scanner.nextLine();

System.out.print("Enter Team Member ID: ");

String id = scanner.nextLine();

System.out.print("Enter Team Member Gender (male/female): ");

String gender = scanner.nextLine();

System.out.print("Enter Team Member Join Date: ");

String joinDate = scanner.nextLine();

if (role.equals("doctor")) {

System.out.print("Enter Doctor Specialty (intern/senior/surgeon): ");

String specialty = scanner.nextLine();

Doctor doctor = new Doctor(name, id, gender, joinDate, specialty);

if (specialty.equals("intern")) {

System.out.print("Enter Supervisor's Name: ");

String supervisorName = scanner.nextLine();

doctor.setSupervisor(supervisorName);

}

department.addTeamMember(doctor);

} else if (role.equals("nurse")) {

Nurse nurse = new Nurse(name, id, gender, joinDate);

department.addTeamMember(nurse);

}

}

}

// Add patients to the hospital

System.out.println("\n=== Add Patients ===");

while (true) {

System.out.print("Enter Patient Name (or type 'exit' to finish adding patients): ");

String name = scanner.nextLine();

if (name.equalsIgnoreCase("exit")) {

break;

}

System.out.print("Enter Patient Birth Date: ");

String birthDate = scanner.nextLine();

System.out.print("Enter Patient Gender (male/female): ");

String gender = scanner.nextLine();

System.out.print("Enter Patient Acceptance Date: ");

String acceptanceDate = scanner.nextLine();

System.out.print("Enter Patient Diagnosis: ");

String diagnosis = scanner.nextLine();

System.out.print("Enter Doctor Treating the Patient: ");

String doctorName = scanner.nextLine();

System.out.print("Enter Number of Days in Hospital: ");

int daysInHospital = Integer.parseInt(scanner.nextLine());

Patient patient = new Patient(name, birthDate, gender, acceptanceDate, diagnosis, doctorName, daysInHospital);

hospital.addPatient(patient);

}

// Display all departments and their team members

System.out.println("\n=== Hospital Details ===");

hospital.displayDepartments();

// Goodbye message

System.out.println("\n=== Thank you for using the Hospital Management System! ===");

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

scanner.close();

}

}

// Hospital class

class Hospital {

private String name;

private String address;

private ArrayList<Department> departments; // List of departments

private ArrayList<Patient> patients; // List of patients

// Constructor

Hospital(String name, String address, ArrayList<Department> departments, ArrayList<Patient> patients) {

this.name = name;

this.address = address;

this.departments = departments;

this.patients = patients;

}

// Method to add a patient

public void addPatient(Patient patient) {

patients.add(patient);

System.out.println("Patient added: " + patient.getName());

}

// Method to add a department

public void addDepartment(Department department) {

departments.add(department);

System.out.println("Department added: " + department.getName());

}

// Method to display all departments and their staff

public void displayDepartments() {

System.out.println("Hospital Name: " + name);

System.out.println("Hospital Address: " + address);

System.out.println("\nDepartments and Their Team Members:");

for (Department department : departments) {

System.out.println("\nDepartment: " + department.getName());

department.displayTeamMembers();

}

System.out.println("\nPatients in the Hospital:");

for (Patient patient : patients) {

System.out.println(patient.getName() + " - Diagnosis: " + patient.getDiagnosis() + " - Doctor: " + patient.getDoctorName());

}

}

}

// Department class

class Department {

private String name;

private ArrayList<TeamMember> staff; // List of team members

// Constructor

Department(String name) {

this.name = name;

this.staff = new ArrayList<>(); // Initialize the staff list

}

// Method to add a team member

public void addTeamMember(TeamMember member) {

if (staff.size() < 5) { // Check if staff limit is not exceeded

staff.add(member);

System.out.println("Team member added: " + member.getName());

} else {

System.out.println("Cannot add more than 5 team members to " + name);

}

}

// Method to display all team members

public void displayTeamMembers() {

System.out.println("Team Members:");

for (TeamMember member : staff) {

System.out.println(member.getDetails());

}

}

public String getName() {

return name;

}

}

// TeamMember class (Superclass for Doctor and Nurse)

abstract class TeamMember {

private String name;

private String id;

private String gender;

private String joinDate;

// Constructor

TeamMember(String name, String id, String gender, String joinDate) {

this.name = name;

this.id = id;

this.gender = gender;

this.joinDate = joinDate;

}

public String getName() {

return name;

}

public String getDetails() {

return "Name: " + name + ", ID: " + id + ", Gender: " + gender + ", Join Date: " + joinDate;

}

public abstract void treatPatient(Patient patient);

}

// Doctor class (Subclass of TeamMember)

class Doctor extends TeamMember {

private String specialty;

private String supervisor; // Only for interns

private ArrayList<Patient> patients; // List of patients treated by the doctor

// Constructor

Doctor(String name, String id, String gender, String joinDate, String specialty) {

super(name, id, gender, joinDate);

this.specialty = specialty;

this.patients = new ArrayList<>();

}

public void setSupervisor(String supervisor) {

this.supervisor = supervisor;

}

public void addPatient(Patient patient) {

patients.add(patient);

}

public void checkPatientReport(Patient patient) {

System.out.println("Patient: " + patient.getName() + ", Diagnosis: " + patient.getDiagnosis());

}

@Override

public void treatPatient(Patient patient) {

if (specialty.equals("intern")) {

System.out.println("Intern " + getName() + " is treating patient " + patient.getName() + " under supervision of " + supervisor);

} else if (specialty.equals("senior")) {

System.out.println("Senior Doctor " + getName() + " is treating patient " + patient.getName());

} else if (specialty.equals("surgeon")) {

System.out.println("Surgeon " + getName() + " is performing surgery on patient " + patient.getName());

}

}

@Override

public String getDetails() {

String details = super.getDetails() + ", Specialty: " + specialty;

if (specialty.equals("intern")) {

details += ", Supervisor: " + supervisor;

}

return details;

}

}

// Nurse class (Subclass of TeamMember)

class Nurse extends TeamMember {

// Constructor

Nurse(String name, String id, String gender, String joinDate) {

super(name, id, gender, joinDate);

}

@Override

public void treatPatient(Patient patient) {

System.out.println("Nurse " + getName() + " is assisting in the treatment of patient " + patient.getName());

}

}

// Patient class

class Patient {

private String name;

private String birthDate;

private String gender;

private String acceptanceDate;

private String diagnosis;

private String doctorName;

private int daysInHospital;

// Constructor

Patient(String name, String birthDate, String gender, String acceptanceDate, String diagnosis, String doctorName, int daysInHospital) {

this.name = name;

this.birthDate = birthDate;

this.gender = gender;

this.acceptanceDate = acceptanceDate;

this.diagnosis = diagnosis;

this.doctorName = doctorName;

this.daysInHospital = daysInHospital;

}

public String getName() {

return name;

}

public String getDiagnosis() {

return diagnosis;

}

public String getDoctorName() {

return doctorName;

}

}