public abstract class Person {

private int id;

private String name;

private int age;

public Person(int id, String name, int age) {

this.id = id;

this.name = name;

this.age = age;

}

public int getId() {

return id;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

@Override

public String toString() {

return "ID: " + id + ", Name: " + name + ", Age: " + age;

}

}

public class Patient extends Person {

private String ailment;

public Patient(int id, String name, int age, String ailment) {

super(id, name, age);

this.ailment = ailment;

}

public String getAilment() {

return ailment;

}

@Override

public String toString() {

return super.toString() + ", Ailment: " + ailment;

}

}

public class Doctor extends Person {

private String specialty;

public Doctor(int id, String name, int age, String specialty) {

super(id, name, age);

this.specialty = specialty;

}

public String getSpecialty() {

return specialty;

}

@Override

public String toString() {

return super.toString() + ", Specialty: " + specialty;

}

}

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class MedicalManagement {

private List<Patient> patients;

private List<Doctor> doctors;

private int nextPersonId;

public MedicalManagement() {

patients = new ArrayList<>();

doctors = new ArrayList<>();

nextPersonId = 1;

}

public void addPatient(String name, int age, String ailment) {

Patient patient = new Patient(nextPersonId++, name, age, ailment);

patients.add(patient);

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

}

public void addDoctor(String name, int age, String specialty) {

Doctor doctor = new Doctor(nextPersonId++, name, age, specialty);

doctors.add(doctor);

System.out.println("Doctor added: " + doctor);

}

public void listPatients() {

if (patients.isEmpty()) {

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

} else {

System.out.println("Patient List:");

for (Patient patient : patients) {

System.out.println(patient);

}

}

}

public void listDoctors() {

if (doctors.isEmpty()) {

System.out.println("No doctors found.");

} else {

System.out.println("Doctor List:");

for (Doctor doctor : doctors) {

System.out.println(doctor);

}

}

}

public void removePatient(int id) {

Patient toRemove = null;

for (Patient patient : patients) {

if (patient.getId() == id) {

toRemove = patient;

break;

}

}

if (toRemove != null) {

patients.remove(toRemove);

System.out.println("Patient removed: " + toRemove);

} else {

System.out.println("Patient with id " + id + " not found.");

}

}

public void removeDoctor(int id) {

Doctor toRemove = null;

for (Doctor doctor : doctors) {

if (doctor.getId() == id) {

toRemove = doctor;

break;

}

}

if (toRemove != null) {

doctors.remove(toRemove);

System.out.println("Doctor removed: " + toRemove);

} else {

System.out.println("Doctor with id " + id + " not found.");

}

}

public static void main(String[] args) {

MedicalManagement management = new MedicalManagement();

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("\nMedical Management System");

System.out.println("1. Add Patient");

System.out.println("2. List Patients");

System.out.println("3. Remove Patient");

System.out.println("4. Add Doctor");

System.out.println("5. List Doctors");

System.out.println("6. Remove Doctor");

System.out.println("7. Exit");

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

int choice = scanner.nextInt();

scanner.nextLine(); // consume the newline

switch (choice) {

case 1:

System.out.print("Enter patient name: ");

String patientName = scanner.nextLine();

System.out.print("Enter patient age: ");

int patientAge = scanner.nextInt();

scanner.nextLine(); // consume the newline

System.out.print("Enter patient ailment: ");

String ailment = scanner.nextLine();

management.addPatient(patientName, patientAge, ailment);

break;

case 2:

management.listPatients();

break;

case 3:

System.out.print("Enter patient ID to remove: ");

int patientId = scanner.nextInt();

management.removePatient(patientId);

break;

case 4:

System.out.print("Enter doctor name: ");

String doctorName = scanner.nextLine();

System.out.print("Enter doctor age: ");

int doctorAge = scanner.nextInt();

scanner.nextLine(); // consume the newline

System.out.print("Enter doctor specialty: ");

String specialty = scanner.nextLine();

management.addDoctor(doctorName, doctorAge, specialty);

break;

case 5:

management.listDoctors();

break;

case 6:

System.out.print("Enter doctor ID to remove: ");

int doctorId = scanner.nextInt();

management.removeDoctor(doctorId);

break;

case 7:

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

scanner.close();

System.exit(0);

default:

System.out.println("Invalid choice. Please try again.");

}

}

}

}