Contents

TRƯỜNG ĐẠI HỌC ĐÔNG Á **KHOA CÔNG NGHỆ THÔNG TIN**	**KỲ THI KẾT THÚC HỌC PHẦN** Học phần: **Lập trình hướng đối tượng** Hình thức thi: **Tự luận** Thời gian: **90 phút** *Không được sử dụng tài liệu.*

ĐỀ SỐ: 01

Phần I. LÝ THUYẾT VÀ ĐỌC HIỂU MÃ NGUỒN

Câu 1. (1.0 điểm) Để tạo một lớp các đối tượng trong java dùng từ khóa nào? Cho ví dụ code đơn giản tạo một lớp bất kỳ?

Câu 2. (1.0 điểm) Trong lập trình hướng đối tượng, hàm khởi (constructor) tạo là gì? Mục đích của hàm khởi tạo để làm gì? Có thể tạo nhiều hàm khởi tạo trong một lớp được không?

Câu 3. (1.0 điểm) **Nạp chồng** phương thức (Overloading) là gì? Nêu ý nghĩa của việc nạp chồng phương thức? Trong ví dụ bên dưới hãy chỉ ra hàm nào được được nạp chồng?

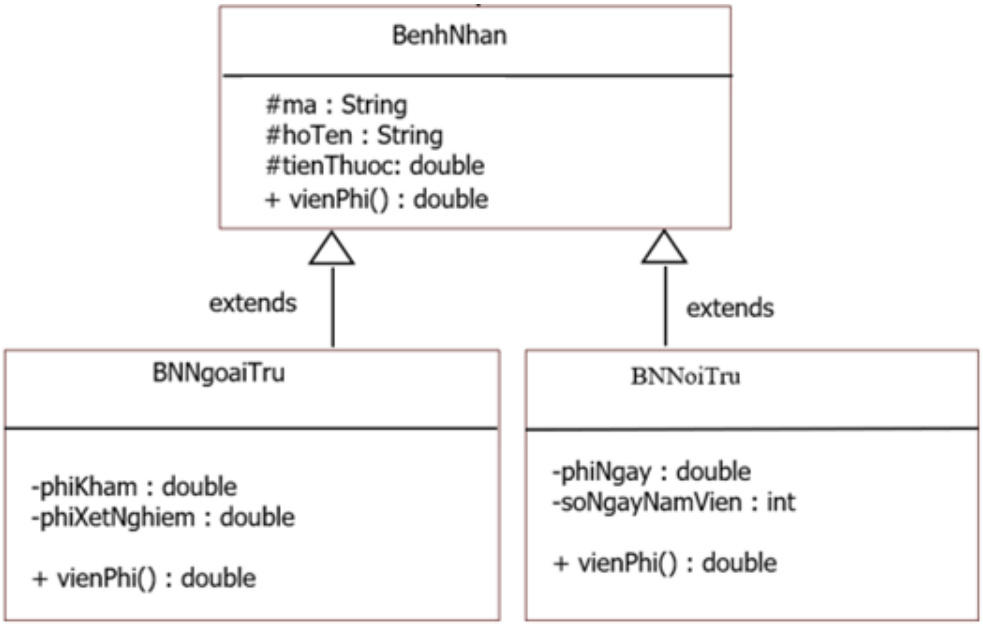
```
public class Calculator {
    // Overloaded methods
    public int add(int a, int b) {                  //Hàm 1
        return a + b;
    }

    public double add(double a, double b) { //Hàm 2
        return a + b;
    }
}
public class App {
    public static void main(String[] args) {
        Calculator calc = new Calculator();
        int sum1 = calc.add(5, 3);              // Sử dụng add(int, int)
        double sum2 = calc.add(2.5, 3.5);   // Sử dụng add(double, double)

        System.out.println("Sum1: " + sum1);
        System.out.println("Sum2: " + sum2);
    }
}
```

Phần II. VIẾT CHƯƠNG TRÌNH

Cho sơ đồ quan hệ giữa các lớp trong phần mềm quản lý bệnh viện như bên dưới:



Hãy thực hiện các yêu cầu sau:

Câu 4: (1.0 điểm) Định nghĩa lớp **BenhNhan** với các thuộc tính và phương thức theo sơ đồ trên.

Câu 5: (2.0 điểm) Định nghĩa lớp **BNNgoaiTru** kế thừa (extends) lớp **BenhNhan** với các thuộc tính và phương thức theo sơ đồ trên. Với *@override* phương thức **vienPhi()** được tính như sau: viện phí của bệnh nhân ngoại trú được tính bằng tổng: phiKham, phiXetNghiem và tienThuoc.

Câu 6: (2.0 điểm) Định nghĩa lớp **BNNoiTru** kế thừa (extends) lớp **BenhNhan** với các thuộc tính và phương thức theo sơ đồ trên. Với *@override* phương thức **vienPhi()** như sau:

- viện phí của bệnh nhân nội trú được tính bằng: **tienThuoc*soNgayNamVien** và

phiNgay* soNgayNamVien.

Câu 7: Trong hàm **main** chương trình chính (*Gợi ý*: tạo tệp tin App.java):

7.1 (1.0 điểm) Hãy tạo 2 đối tượng **BNNgoaiTru** và đưa vào ArrayList. Tính tổng tiền viện phí của bệnh nhân ngoại trú.

7.2 (1.0 điểm) Hãy tạo 2 đối tượng **BNNoiTru** và đưa vào ArrayList. Tính tổng tiền viện phí của bệnh nhân nội trú.

– Hết –

TRƯỜNG ĐẠI HỌC ĐÔNG Á KHOA CÔNG NGHỆ THÔNG TIN	**CẤU TRÚC ĐỀ THI KẾT THÚC HỌC PHẦN** Học phần: **Lập trình hướng đối tượng**. Hình thức thi: **Tự luận**. Thời gian thi: **90 phút** [ ] Được sử dụng tài liệu khi làm bài [X] Không được sử dụng tài liệu khi làm bài

PHẦN 1: CÁC YÊU CẦU CỦA CẤU TRÚC ĐỀ THI NHẰM ĐÁP ỨNG CÁC CHUẨN ĐẦU RA CỦA HỌC PHẦN.

Lưu ý: phần này phải thống nhất thông tin với nội dung đã nêu trong đề cương chi tiết của học phần

Ký hiệu CĐR	Nội dung CĐR	Hình thức kiểm tra đánh giá	Số lượng câu hỏi cần cho 1 đề thi	Điểm số tối thiểu (mức đạt CĐR)	Điểm số tối đa
(1)	(2)	(3)	(4)	(5)	(6)
CLO1	Vận dụng được các tính chất của lập trình hướng đối tượng: Tính kế thừa, tính đa hình, tính đóng gói, tính trừu tượng để giải quyết bài toán thực tế.	Bài tập lập trình cụ thể (viết chương trình Java)	1	0.5	1.0
CLO2	Vận dụng lập trình các ứng dụng Java với cấu trúc hướng đối tượng.	Đánh giá qua bài tập ứng dụng Java có cấu trúc hướng đối tượng OOP (Object-Oriented Programming).	3	1.0	3.0
CLO3	Vận dụng hướng đối tượng để phân tích và giải quyết các bài toán thực tế theo phương pháp hướng đối tượng.	Yêu cầu thiết kế và lập trình ứng dụng Java theo mô hình OOP.	2	2.0	4.0
CLO4	Vận dụng phát triển các giải pháp phần mềm Java, đảm bảo tuân thủ nguyên tắc lập trình tốt, dễ bảo trì.	Có khả năng viết mã code giải quyết các vấn đề kỹ thuật cụ thể trong Java.	1	0.5	2.0
Tổng			**7.0**	**4.0**	**10.0**

Chú thích các cột

1. Ký hiệu của các chuẩn đầu ra (CLO)

2. Nội dung của chuẩn đầu ra tương ứng

3. Hình thức kiểm tra, đánh giá có thể là: tự luận, trắc nghiệm, vấn đáp, thực hành trên máy tính, thực hành tại xưởng, thực hành tại phòng thí nghiệm, dự án, đố án, báo cáo, thuyết trình, ... Hình thức kiểm tra đánh giá phải thống nhất với đề cương chi tiết của học phần.

4. Số lượng câu hỏi cần thiết cho 1 đề thi để đảm bài đánh giá được CĐR đó.

5. Điểm số tối thiểu (mức đạt CĐR) là điểm số ít nhất người học cần đạt được để đảm bảo đạt được CĐR đó. Tổng cộng điểm số tối thiểu của một đề thi không được dưới 4 điểm.

6. Điểm số tối đa là điểm số tối đa của CĐR đó. Tổng cộng điểm số tối đa của một đề thi không được trên 10 điểm .

Đà Nẵng, ngày ... tháng ... năm ...

NGƯỜI DUYỆT **GIẢNG VIÊN/NHÓM BIÊN SOẠN**

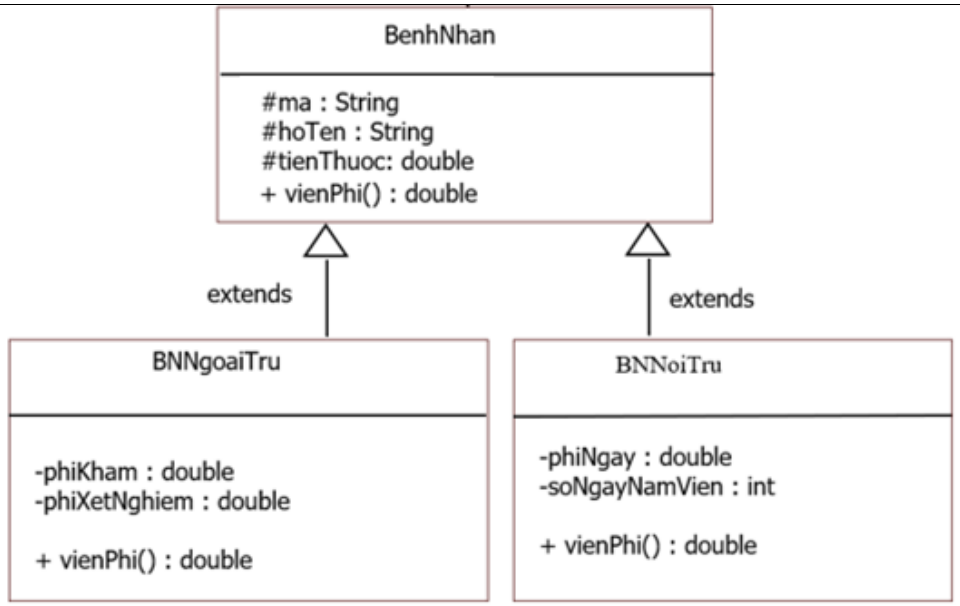
PHẦN 2: NGÂN HÀNG CÂU HỎI ĐÁNH GIÁ CHUẨN ĐẦU RA CỦA HỌC PHẦN

TRƯỜNG ĐẠI HỌC ĐÔNG Á KHOA CÔNG NGHỆ THÔNG TIN	**NGÂN HÀNG CÂU HỎI THI KẾT THÚC HỌC PHẦN** Học phần: **Kỹ thuật lập trình** Hình thức thi: **Tự luận**

TT	Nội dung câu hỏi	Điểm	Cấp độ
(1)	(2)	(3)	(4)
Phần I	**LÝ THUYẾT VÀ ĐỌC HIỂU MÃ NGUỒN** **Chuẩn đầu ra CLO2:** Hiểu được các khái niệm, phương pháp về lập trình hướng đối tượng.		
Câu 1	Để tạo một lớp các đối tượng trong java dùng từ khóa nào? Cho ví dụ code đơn giản tạo một lớp bất kỳ?	**1.0**	**A**
	Đáp án:		
	Ý 1: Trong Java, để tạo một lớp đối tượng, bạn sử dụng từ khóa "class"	0.5	
	Ý 2: Cho ví dụ đúng về một lớp: có thuộc tính/hàm khởi tạo	0.5	
Câu 2	Trong lập trình hướng đối tượng, hàm khởi tạo (constructor) là gì? Mục đích của hàm khởi tạo để làm gì? Có thể tạo nhiều hàm khởi tạo trong một lớp được không?	**1.0**	**A**
	Đáp án:		
	Ý 1: Trong lập trình hướng đối tượng, hàm khởi tạo (constructor) là một phương thức đặc biệt được gọi khi một đối tượng của lớp được tạo ra. Hàm khởi tạo có tên trùng với tên lớp và không có kiểu trả về (không kể void).	0.5	
	Ý 2: **trả lời được ít nhất 1 ý** - Khởi tạo đối tượng: Hàm khởi tạo được sử dụng để thiết lập các giá trị ban đầu cho các thuộc tính của đối tượng. - Thiết lập trạng thái ban đầu/hoặc các thao tác cần thiết: để đối tượng được tạo ra ở một trạng thái hợp lệ và sẵn sàng để sử dụng/hoặc chẳng hạn như mở tệp, thiết lập kết nối cơ sở dữ liệu.	0.25	
	Ý 3: Có thể tạo nhiều hàm khởi tạo trong một lớp.	0.25	

Câu 3	Nạp chồng phương thức (Overloading) là gì? Nêu ý nghĩa của việc nạp chồng phương thức? Trong ví dụ bên dưới hãy chỉ ra hàm nào được nạp chồng?
	public class **Calculator** {
	// Overloaded methods
	public int add(int a, int b) { //Hàm 1
	return a + b;
	}
	
	public double add(double a, double b) { //Hàm 2
	return a + b;
	}
	}
	public class **App** {
	public static void **main**(String[] args) {
	Calculator calc = new Calculator();
	int sum1 = calc.add(5, 3); // Sử dụng add(int, int)
	double sum2 = calc.add(2.5, 3.5); // Sử dụng add(double, double)
	
	System.out.println("Sum1: " + sum1);
	System.out.println("Sum2: " + sum2);
	}
	}
		1.0	**A**

	Đáp án:		
	Ý 1: Nạp chồng phương thức (*Method Overloading*) trong lập trình hướng đối tượng là khả năng của một lớp để **có nhiều phương thức** cùng tên nhưng khác nhau về danh sách tham số (số lượng, kiểu dữ liệu hoặc thứ tự của tham số).	0.5	
	Ý 2: Trả lời được ít nhất 1 ý sau: - Tính linh hoạt: Cho phép sử dụng cùng một tên phương thức cho các hoạt động tương tự nhưng với các loại dữ liệu hoặc số lượng tham số khác nhau. - Tính dễ đọc và dễ bảo trì: Người dùng không cần nhớ nhiều tên phương thức khác nhau cho các nhiệm vụ tương tự. - Khả năng mở rộng: Dễ dàng thêm chức năng mới mà không ảnh hưởng đến mã hiện tại.	0.25	
	Ý 3: Phương thức nạp chồng (Overloading) ở ví dụ là phương thức: - public double add(double a, double b): kiểu dữ liệu trả về và 2 tham số truyền vào kiểu double. - public int add(int a, int b): kiểu dữ liệu trả về và 2 tham số truyền vào kiểu số nguyên int.	0.25	
Phần II	**VIẾT CHƯƠNG TRÌNH** **Chuẩn đầu ra CLO1:** Vận dụng kiến thức về ngôn ngữ lập trình để lập trình được các bài toán thực tế. **Chuẩn đầu ra CLO3:** Vận dụng kỹ thuật lập trình hướng đối tượng để giải quyết một số bài toán trong thực tế. **Chuẩn đầu ra CLO4:** Có khả năng giải quyết vấn đề kỹ thuật về lập trình hướng đối tượng.		
	Cho sơ đồ quan hệ giữa các lớp trong phần mềm quản lý bệnh viện như bên dưới:		**B**

	 Hãy thực hiện các yêu cầu sau:		
Câu 4	Định nghĩa lớp **BenhNhan** với các thuộc tính và phương thức theo sơ đồ trên.	**1.0**	**B**
	Đáp án:		
	Ý 1: Tạo được lớp **BenhNhan** gồm: - Tên lớp & thuộc tính (0.25đ) - Hàm khởi tạo cho lớp (0.25đ)	0.5	
	Ý 2: - định nghĩa đúng hàm viện phí **vienPhi()** và trả về kiểu dữ liệu đúng.	0.25	
	Ý 3: - Có hàm các hàm get và set cho từng tham số	0.25	
Câu 5	Định nghĩa lớp **BNNgoaiTru** kế thừa (extends) lớp **BenhNhan** với các thuộc tính và phương thức theo sơ đồ trên. Với *@override* phương thức **vienPhi()** được tính như sau: viện phí của bệnh nhân ngoại trú được tính bằng tổng: phiKham, phiXetNghiem và tienThuoc.	**2.0**	**B**
	Đáp án:		
	Ý 1: - tạo lớp **BNNgoaiTru** kế thừa extends từ lớp cha **BenhNhan**	0.5	
	Ý 2: - Có hàm khởi tạo cho lớp và có dùng hàm **super** để gọi hàm khởi tạo lớp cha.	0.5	
	Ý 3: - Có hàm tính toán viện phí, ghi đè (*overwrite*) lại hàm **vienPhi()** ở lớp cha	0.5	
	Ý 4: - Có hàm get/set cho các tham số.	0.5	
Câu 6	Định nghĩa lớp **BNNoiTru** kế thừa (extends) lớp **BenhNhan** với các thuộc tính và phương thức theo sơ đồ trên. Với *@override* phương thức **vienPhi()** như sau: -viện phí của bệnh nhân nội trú được tính bằng: **tienThuoc*soNgayNamVien** và **phiNgay* soNgayNamVien**.	**2.0**	**B**
	Đáp án:		
	Ý 1: - tạo lớp **BNNoiTru** kế thừa extends từ lớp cha **BenhNhan**	0.5	
	Ý 2: - Có hàm khởi tạo cho lớp và có dùng hàm **super** để gọi hàm khởi tạo lớp cha.	0.5	
	Ý 3: - Có hàm tính toán viện phí, ghi đè (*override*) lại hàm **vienPhi()** ở lớp cha	0.5	
	Ý 4: - Có hàm get/set cho các tham số.	0.5	
Câu 7	Trong hàm **main** chương trình chính (*Gợi ý*: tạo tệp tin App.java): 7.1 (1.0 điểm) Hãy tạo 2 đối tượng **BNNgoaiTru** và đưa vào ArrayList. Tính tổng tiền viện phí của bệnh nhân ngoại trú. 7.2 (1.0 điểm) Hãy tạo 2 đối tượng **BNNoiTru** và đưa vào ArrayList. Tính tổng tiền viện phí của bệnh nhân nội trú.	**2.0**	**C**
	Đáp án:		

	Ý 7.1: - Tạo được 2 đối tượng từ lớp **BNNgoaiTru** (0.25đ) - add được 2 đối tượng vào arraylist của lớp đối tượng **BNNgoaiTru** (0.25đ) - Tạo được vòng lặp for tính tổng: tiền viện phí của các đối tượng từ arraylist ở trên (0.5đ)	1.0	
	Ý 7.2: - Tạo được 2 đối tượng từ lớp **BNNoiTru** (0.25đ) - add được 2 đối tượng vào arraylist của lớp **BNNoiTru** (0.25đ) - Tạo được vòng lặp for tính tổng: tiền viện phí của các đối tượng từ arraylist ở trên (0.5đ)	1.0	

Chú thích các cột

1. Số thứ tự của CĐR và câu hỏi trong CĐR

2. Nội dung câu hỏi và đáp án:

- Nội dung câu hỏi phải đầy đủ, dễ hiểu, rõ ràng các ý. Nếu có hình ảnh, bảng biểu, đồ thị phải định dạng bằng hình ảnh có màu sắc tương phản để dễ dàng nhìn thấy khi in trắng đen.

- Nội dung đáp án phải tương ứng với các tiêu chí đánh giá chuẩn đầu ra đó. Đáp án phải đánh giá đầy đủ các ý của câu hỏi và đảm bảo đo lường được.

3. Điểm: là điểm tối đa của câu hỏi/ đáp án

4. Cấp độ: là cấp độ đánh giá của câu hỏi: câu hỏi cấp độ Nhận biết/Thông hiểu ký hiệu: A. Câu hỏi cấp độ Vận dụng, Phân tích:ký hiệu B. Câu hỏi cấp độ Đánh giá, Sáng tạo: ký hiệu C

Đà Nẵng, ngày ... tháng ... năm ...

NGƯỜI DUYỆT **GIẢNG VIÊN/NHÓM BIÊN SOẠN**

```
import java.util.ArrayList;
import java.util.Scanner;

// Định nghĩa giao diện IBenhNhan
public interface IBenhNhan {
    double vienphi();
}

// Lớp BenhNhan triển khai giao diện IBenhNhan
public class BenhNhan implements IBenhNhan {
    protected int ma;
    protected String hoTen;
    protected double tienThuoc;

    public BenhNhan(int ma, String hoTen, double tienThuoc) {
        this.ma = ma;
        this.hoTen = hoTen;
        this.tienThuoc = tienThuoc;
    }

    @Override
    public double vienphi() {
        return tienThuoc;
    }

    public int getMa() {
        return ma;
    }

    public String getHoTen() {
        return hoTen;
    }
}

// Lớp BNNgoaiTru kế thừa từ BenhNhan
public class BNNgoaiTru extends BenhNhan {
    private double phiKham;
    private double phiXetNghiem;

    public BNNgoaiTru(int ma, String hoTen, double tienThuoc, double phiKham,
double phiXetNghiem) {
        super(ma, hoTen, tienThuoc);
        this.phiKham = phiKham;
```

```
        this.phiXetNghiem = phiXetNghiem;
    }

    @Override
    public double vienphi() {
        return phiKham + phiXetNghiem + tienThuoc;
    }
}

// Lớp BNNoiTru kế thừa từ BenhNhan
public class BNNoiTru extends BenhNhan {
    private double phiNgay;
    private int soNgayNamVien;

    public BNNoiTru(int ma, String hoTen, double tienThuoc, double phiNgay, int soNgayNamVien) {
        super(ma, hoTen, tienThuoc);
        this.phiNgay = phiNgay;
        this.soNgayNamVien = soNgayNamVien;
    }

    @Override
    public double vienphi() {
        double phuPhi = (soNgayNamVien < 10) ? 50 : 100;
        return (tienThuoc * soNgayNamVien) + (phiNgay * soNgayNamVien) + phuPhi;
    }
}

// Lớp DSBenhNhan
public class DSBenhNhan {
    private ArrayList<BenhNhan> danhSach;
    private Scanner scanner;

    public DSBenhNhan() {
        danhSach = new ArrayList<>();
        scanner = new Scanner(System.in);
    }

    public void themBenhNhan() {
        System.out.println("Chọn loại bệnh nhân: 1. Nội trú  2. Ngoại trú");
        int choice = scanner.nextInt();
        scanner.nextLine();
```

```
        System.out.print("Nhập mã: ");
        int ma = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Nhập họ tên: ");
        String hoTen = scanner.nextLine();
        System.out.print("Nhập tiền thuốc: ");
        double tienThuoc = scanner.nextDouble();

        if (choice == 1) {
            System.out.print("Nhập phí ngày: ");
            double phiNgay = scanner.nextDouble();
            System.out.print("Nhập số ngày nằm viện: ");
            int soNgayNamVien = scanner.nextInt();
            danhSach.add(new BNNoiTru(ma, hoTen, tienThuoc, phiNgay,
soNgayNamVien));
        } else {
            System.out.print("Nhập phí khám: ");
            double phiKham = scanner.nextDouble();
            System.out.print("Nhập phí xét nghiệm: ");
            double phiXetNghiem = scanner.nextDouble();
            danhSach.add(new BNNgoaiTru(ma, hoTen, tienThuoc, phiKham,
phiXetNghiem));
        }
    }

    public void inDanhSach() {
        for (BenhNhan bn : danhSach) {
            System.out.println("Mã: " + bn.getMa() + ", Họ tên: " + bn.getHoTen() + ", Viện
phí: " + bn.vienphi());
        }
    }

    public void timBenhNhanNoiTru() {
        for (BenhNhan bn : danhSach) {
            if (bn instanceof BNNoiTru && bn.vienphi() >= 3000) {
                System.out.println("Mã: " + bn.getMa() + ", Họ tên: " + bn.getHoTen() + ",
Viện phí: " + bn.vienphi());
            }
        }
    }

    public void timBenhNhanTheoMa() {
        System.out.print("Nhập mã bệnh nhân cần tìm: ");
```

```
        int maTim = scanner.nextInt();
        for (BenhNhan bn : danhSach) {
            if (bn.getMa() == maTim) {
                System.out.println("Mã: " + bn.getMa() + ", Họ tên: " + bn.getHoTen() + ",
Viện phí: " + bn.vienphi());
                return;
            }
        }
        System.out.println("Không tìm thấy bệnh nhân!");
    }

    public void menu() {
        while (true) {
            System.out.println("1. Thêm bệnh nhân");
            System.out.println("2. In danh sách bệnh nhân");
            System.out.println("3. Tìm bệnh nhân nội trú có viện phí >= 3000");
            System.out.println("4. Tìm bệnh nhân theo mã");
            System.out.println("5. Thoát");
            System.out.print("Chọn: ");
            int choice = scanner.nextInt();
            switch (choice) {
                case 1: themBenhNhan(); break;
                case 2: inDanhSach(); break;
                case 3: timBenhNhanNoiTru(); break;
                case 4: timBenhNhanTheoMa(); break;
                case 5: return;
                default: System.out.println("Lựa chọn không hợp lệ!");
            }
        }
    }
}
```