

实验 05：类与对象

一、实验目的

1. 掌握面向对象程序设计的基本概念，包括类与对象的关系。
2. 理解功能驱动的设计方法，能够根据需求设计类和系统架构。
3. 掌握类的详细设计与实现方法，包括类的属性、方法和构造函数的定义。
4. 学会通过问题分解描述与类的初步设计，逐步实现复杂系统。
5. 综合运用面向对象的知识，实现一个医院患者管理系统。

二、实验学时

2 学时

三、实验类型

验证性

四、实验需求

1、硬件

每人配备计算机 1 台，建议优先使用个人计算机开展实验。

2、软件

Windows 10/11 操作系统

IntelliJ IDEA 版本号：2024.03

安装 IntelliJ IDEA，以及 Java 运行所需要的相关基础环境。

3、网络

本地主机能够访问互联网和实验中心网络。

4、工具

无。

五、实验任务

- 5.1 类的初步设计：设计一个 `Patient` 类，包含患者的基本信息。
- 5.2 类的详细设计与实现：为 `Patient` 类添加方法和构造函数。
- 5.3 功能驱动类设计：设计一个 `Doctor` 类，包含医生的基本信息和诊疗方法。
- 5.4 类的交互：实现 `Patient` 类和 `Doctor` 类的交互，模拟诊疗过程。
- 5.5 综合任务：设计一个 `Hospital` 类，管理患者和医生，并实现简单的挂号功能。

六、实验内容及步骤

6.1、任务 1：类的初步设计

步骤：

- (1) 设计一个 `Patient` 类，包含患者的姓名、年龄、性别和症状属性。

示例代码：

Java

```
1 // Patient.java
2 public class Patient {
3     // 属性
4     private String name;
5     private int age;
6     private String gender;
7     private String symptom;
8
9     // 构造函数
10    public Patient(String name, int age, String gender, String sympto
11        m) {
12        this.name = name;
13        this.age = age;
14        this.gender = gender;
15        this.symptom = symptom;
16    }
17    // Getter 方法
18    public String getName() {
19        return name;
20    }
21
22    public int getAge() {
23        return age;
24    }
25
26    public String getGender() {
27        return gender;
28    }
29
30    public String getSymptom() {
31        return symptom;
32    }
33
34    // 打印患者信息
35    public void printInfo() {
36        System.out.println("患者姓名: " + name);
37        System.out.println("患者年龄: " + age);
38        System.out.println("患者性别: " + gender);
39        System.out.println("患者症状: " + symptom);
40    }
41 }
```

6.2、任务 2：类的详细设计与实现

步骤：

(1) 为 `Patient` 类添加方法，如更新症状、打印详细信息等。

示例代码：

Java

```
1 // 在 Patient 类中添加以下方法
2 public void updateSymptom(String newSymptom) {
3     this.symptom = newSymptom;
4     System.out.println("更新后的症状: " + newSymptom);
5 }
6
7 // 测试类
8 public class PatientTest {
9     public static void main(String[] args) {
10         Patient patient = new Patient("张三", 35, "男", "发热");
11         patient.printInfo();
12         patient.updateSymptom("咳嗽");
13     }
14 }
```

6.3、任务 3：功能驱动的分类设计

步骤：

(1) 设计一个 `Doctor` 类，包含医生的姓名、科室和诊疗方法。

示例代码：

```
1 // Doctor.java
2 public class Doctor {
3     // 属性
4     private String name;
5     private String department;
6
7     // 构造函数
8     public Doctor(String name, String department) {
9         this.name = name;
10        this.department = department;
11    }
12
13    // 诊疗方法
14    public void diagnose(Patient patient) {
15        System.out.println("医生 " + name + " 正在为患者 " + patient.getName() + " 进行诊疗。");
16        System.out.println("患者症状: " + patient.getSymptom());
17        System.out.println("建议前往 " + department + " 进一步检查。");
18    }
19
20    // 打印医生信息
21    public void printInfo() {
22        System.out.println("医生姓名: " + name);
23        System.out.println("所属科室: " + department);
24    }
25 }
```

6.4、任务 4：类的交互

步骤：

(1) 实现 `Patient` 类和 `Doctor` 类的交互，模拟诊疗过程。

示例代码：

Java

```
1 public class HospitalSimulation {
2     public static void main(String[] args) {
3         // 创建患者和医生对象
4         Patient patient = new Patient("李四", 28, "女", "头痛");
5         Doctor doctor = new Doctor("王医生", "神经科");
6
7         // 打印信息
8         patient.printInfo();
9         doctor.printInfo();
10
11        // 诊疗过程
12        doctor.diagnose(patient);
13    }
14 }
```

6.5、任务 5：综合任务 - 医院管理系统

步骤：

- (1) 设计一个 `Hospital` 类，管理患者和医生。
- (2) 实现患者挂号功能，将患者分配给医生。

示例代码：

Java

```
1 import java.util.ArrayList;
2 import java.util.List;
3
4 // Hospital.java
5 public class Hospital {
6     // 属性
7     private List<Patient> patients;
8     private List<Doctor> doctors;
9
10    // 构造函数
11    public Hospital() {
12        patients = new ArrayList<>();
13        doctors = new ArrayList<>();
14    }
15
16    // 添加患者
17    public void addPatient(Patient patient) {
18        patients.add(patient);
19        System.out.println("患者 " + patient.getName() + " 已挂号。");
20    }
21
22    // 添加医生
23    public void addDoctor(Doctor doctor) {
24        doctors.add(doctor);
25        System.out.println("医生 " + doctor.getName() + " 已加入医院。");
26    }
27
28    // 分配医生给患者
29    public void assignDoctor(Patient patient, Doctor doctor) {
30        System.out.println("患者 " + patient.getName() + " 被分配给医生
31    " + doctor.getName() + "。");
32        doctor.diagnose(patient);
33    }
34
35    // 测试类
36    public static void main(String[] args) {
37        // 创建医院对象
38        Hospital hospital = new Hospital();
39
40        // 添加患者和医生
41        Patient patient1 = new Patient("王五", 40, "男", "发热");
42        Patient patient2 = new Patient("赵六", 25, "女", "咳嗽");
```

```
42         Doctor doctor1 = new Doctor("张医生", "发热门诊");
43         Doctor doctor2 = new Doctor("李医生", "呼吸科");
44
45         hospital.addPatient(patient1);
46         hospital.addPatient(patient2);
47         hospital.addDoctor(doctor1);
48         hospital.addDoctor(doctor2);
49
50         // 分配医生
51         hospital.assignDoctor(patient1, doctor1);
52         hospital.assignDoctor(patient2, doctor2);
53     }
54 }
```

关于测试

- (1) 编写测试用例，对系统的关键功能进行测试。
- (2) 调试代码，修复可能存在的 BUG。

七、实验考核

本实验考核采用【实验随堂查】方式开展。

每个实验完成后，在实验课上通过现场演示的方式向实验指导教师进行汇报，并完成现场问答交流。

每个实验考核满分 100 分，其中实验成果汇报 60 分，现场提问交流 40 分。

实验考核流程：

- (1) 学生演示汇报实验内容的完成情况，实验指导老师现场打分。
- (2) 指导老师结合实验内容进行提问，每位学生提问 2-3 个问题，根据回答的情况现场打分。
- (3) 实验考核结束后，进行公布成绩。