

实验 11：使用 AI 辅助编程

一、实验目的

1. 掌握如何使用 AI 工具（如 DeepSeek、文心智能体平台、ChatGPT 等）辅助编写 Java 代码。
2. 学会利用 AI 工具生成代码片段、优化代码结构、调试代码错误。
3. 理解 AI 辅助编程的局限性，学会在 AI 生成的代码基础上进行手动优化。
4. 综合运用 AI 辅助编程的知识，设计并实现一个较为复杂的医院患者管理系统。
5. 提升学生的代码编写效率和问题解决能力。

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

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

文心智能体平台链接：<https://ilql1j.smartapps.baidu.com>

五、实验任务

- 1. **AI 生成基础代码**：使用 AI 工具生成医院患者管理系统的核心类（如 `Patient`、`Doctor` 等）。
- 2. **AI 优化代码结构**：使用 AI 工具优化生成的代码，提高代码的可读性和可维护性。
- 3. **AI 调试代码错误**：使用 AI 工具调试代码中的错误，并修复问题。
- 4. **AI 生成测试用例**：使用 AI 工具生成单元测试用例，验证代码的正确性。
- 5. **综合任务**：结合 AI 辅助编程，实现一个完整的医院患者管理系统。

六、实验内容及步骤

任务 1：AI 生成基础代码

步骤：

- 1. 使用 AI 工具生成 `Patient` 类和 `Doctor` 类的代码。
- 2. 确保生成的代码包含必要的属性和方法。

向 AI 提问：



AI 生成的代码：

Java

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

```

43     }
44
45     public String getDepartment() {
46         return department;
47     }
48
49     public void diagnose(Patient patient) {
50         System.out.println("医生 " + name + " 正在为患者 " + patient.getName() + " 进行诊断。");
51         System.out.println("患者症状: " + patient.getSymptom());
52         System.out.println("建议前往 " + department + " 进一步检查。");
53     }
54 }

```

任务 2: AI 优化代码结构

步骤:

1. 使用 AI 工具优化生成的代码，提高代码的可读性和可维护性。
2. 例如，将重复代码提取为方法，或使用设计模式优化代码结构。

向 AI 提问:



文心智能体平台
AgentBuilder



AIJAVA编程
开发者 @huangkai8686
591人聊过

更多智能体

分享智能体

分享

手机版

🔖 📄

使用 AI 工具优化生成的代码，提高代码的可读性和可维护性。

例如，将重复代码提取为方法，或使用设计模式优化代码结构。

好的，以下是针对您提供的 Patient 类和 Doctor 类代码的优化建议，旨在提高代码的可读性和可维护性。

优化后的 Patient 类

```

java
1  public class Patient {
2      // 属性
3      private String name;
4      private int age;
5      private String medicalHistory;

```

Copy Code

🗣️

可以问我任何问题...

内容由AI生成，仅供参考

AI 优化后的代码:

Java

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

```
43         return name;
44     }
45
46     public String getDepartment() {
47         return department;
48     }
49
50     public void diagnose(Patient patient) {
51         printDiagnosisInfo(patient);
52         printTreatmentAdvice(patient);
53     }
54
55     private void printDiagnosisInfo(Patient patient) {
56         System.out.println("医生 " + name + " 正在为患者 " + patient.getName() + " 进行诊断。");
57         System.out.println("患者症状: " + patient.getSymptom());
58     }
59
60     private void printTreatmentAdvice(Patient patient) {
61         System.out.println("建议前往 " + department + " 进一步检查。");
62     }
63 }
```

任务 3: AI 调试代码错误

步骤:

1. 在代码中故意引入一些错误（如空指针异常、逻辑错误等）。
2. 使用 AI 工具调试代码，并修复错误。

示例错误代码:

Java

```
1 // 错误示例: 未初始化 symptom
2 Patient patient = new Patient("张三", 35, null);
3 patient.printInfo();
```

修复后的代码:

Java

```
1 // 修复示例：检查 symptom 是否为 null
2 if (patient.getSymptom() == null) {
3     System.out.println("患者症状未填写。");
4 } else {
5     patient.printInfo();
6 }
```

任务 4：AI 生成测试用例

步骤：

1. 使用 AI 工具生成单元测试用例，验证代码的正确性。
2. 使用 JUnit 框架编写测试用例。

测试用例代码：

Java

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 class PatientTest {
5     @Test
6     void testPatientInfo() {
7         Patient patient = new Patient("张三", 35, "发热");
8         assertEquals("张三", patient.getName());
9         assertEquals(35, patient.getAge());
10        assertEquals("发热", patient.getSymptom());
11    }
12 }
13
14 class DoctorTest {
15     @Test
16     void testDoctorDiagnose() {
17         Doctor doctor = new Doctor("王医生", "发热门诊");
18         Patient patient = new Patient("张三", 35, "发热");
19
20         // 测试诊断方法是否正确
21         doctor.diagnose(patient);
22         // 可以通过日志或输出验证
23     }
24 }
```

任务 5：综合任务 - 医院患者管理系统

步骤：

1. 结合 AI 辅助编程，实现一个完整的医院患者管理系统。
2. 系统功能包括患者挂号、医生接诊、患者信息查询等。

AI 生成的代码：

Java

```
1 import java.util.ArrayList;
2 import java.util.List;
3 import java.util.Scanner;
4
5 public class HospitalSystem {
6     private List<Patient> patients = new ArrayList<>();
7     private List<Doctor> doctors = new ArrayList<>();
8
9     public void addPatient(Patient patient) {
10         patients.add(patient);
11         System.out.println("患者 " + patient.getName() + " 已挂号。");
12     }
13
14     public void addDoctor(Doctor doctor) {
15         doctors.add(doctor);
16         System.out.println("医生 " + doctor.getName() + " 已加入医院。");
17     }
18
19     public void diagnosePatient(String patientName, String doctorName) {
20         Patient patient = patients.stream()
21             .filter(p -> p.getName().equals(patientName))
22             .findFirst()
23             .orElse(null);
24
25         Doctor doctor = doctors.stream()
26             .filter(d -> d.getName().equals(doctorName))
27             .findFirst()
28             .orElse(null);
29
30         if (patient != null && doctor != null) {
31             doctor.diagnose(patient);
32         } else {
33             System.out.println("患者或医生未找到。");
34         }
35     }
36
37     public static void main(String[] args) {
38         HospitalSystem system = new HospitalSystem();
39         Scanner scanner = new Scanner(System.in);
40
41         while (true) {
```

```

42         System.out.println("请选择操作：1. 添加患者 2. 添加医生 3. 诊断
    患者 4. 退出");
43         int choice = scanner.nextInt();
44         scanner.nextLine(); // 清除缓冲区
45
46         switch (choice) {
47             case 1:
48                 System.out.print("请输入患者姓名：");
49                 String patientName = scanner.nextLine();
50                 System.out.print("请输入患者年龄：");
51                 int age = scanner.nextInt();
52                 scanner.nextLine(); // 清除缓冲区
53                 System.out.print("请输入患者症状：");
54                 String symptom = scanner.nextLine();
55                 system.addPatient(new Patient(patientName, age, sym
ptom));
56                 break;
57             case 2:
58                 System.out.print("请输入医生姓名：");
59                 String doctorName = scanner.nextLine();
60                 System.out.print("请输入医生科室：");
61                 String department = scanner.nextLine();
62                 system.addDoctor(new Doctor(doctorName, departmen
t));
63                 break;
64             case 3:
65                 System.out.print("请输入患者姓名：");
66                 patientName = scanner.nextLine();
67                 System.out.print("请输入医生姓名：");
68                 doctorName = scanner.nextLine();
69                 system.diagnosePatient(patientName, doctorName);
70                 break;
71             case 4:
72                 System.out.println("退出系统。");
73                 return;
74             default:
75                 System.out.println("无效选择，请重新输入。");
76         }
77     }
78 }
79 }

```

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。