

深圳大学课程作业报告

课程名称：Java 程序设计

课程作业名称：结合大模型的 Java 知识问答机器人

学院：计算机与软件学院

专业：

指导教师：潘微科

报告人： 学号： 班级：

作业时间：2024 年 11 月 22 日（周五）~2024 年 12 月 25 日（周三）

课程作业报告提交时间：2024 年 12 月 13 日星期五

教务部制

作业目的与要求：

课程作业目的：Java 程序设计综合练习，含 GUI、多线程、I/O、网络通信等。

课程作业要求：

(1). 编写 Java 应用程序，实现“结合大模型的 Java 知识问答机器人”。

- 人机交互界面：使用命令行窗口（不得分）或图形用户界面（10 分）
- 大模型：允许用户从腾讯混元大模型、百度文心一言大模型、阿里通义千问大模型或其他大模型中（至少支持两个大模型）选择使用一个大模型（20 分）
- 多线程&网络通信：能够并行处理多个用户的网络请求，每个用户请求都应该在一个新的线程中处理，以避免阻塞主线程（20 分）
- 测试案例：要求测试 Java 程序设计方面的选择题、判断题、程序填空题和编程题（每个题型至少测试两个例子），并就测试效果做出简单分析（20 分）
- 日志记录：在本地文本文件（.txt）中记录人机对话过程（20 分）
- 特色功能：可以列出自己实现的其他特色功能（每个特色功能 5 分，最多记两个特色功能共 10 分）

在报告中附上程序截图、完整的运行结果截图和详细的文字说明。

报告写作。要求：主要思路有明确的说明，重点代码有详细的注释，行文逻辑清晰可读性强，报告整体写作较为专业。（20 分）

说明：

(1) 本次课程作业满分为 100 分，占总成绩的比例（待定）。

(2) 本次课程作业截至时间 2024 年 12 月 25 日（周**）21:59。

(3) 报告正文：请在**指定位置填写**，本次课程作业**需要单独提交源程序文件**（源程序单独打包在 Blackboard 中上传，不要包含外部导入的包）。

(4) 个人信息：WORD 文件名中的“姓名”、“学号”，请改为你的**姓名和学号**；课程作业报告的首页，**请准确填写“学院”、“专业”、“报告人”、“学号”、“班级”、“课程作业报告提交时间”**等信息。

(5) 提交方式：截至时间前，请在 Blackboard 平台中提交。

(6) 发现抄袭（包括复制&粘贴整句话、整张图），**抄袭者和被抄袭者的成绩记零分**。

(7) 延迟提交，不得分；如有特殊情况，请于截止日期之后的**48 小时内**发邮件到 panweike@szu.edu.cn，并在邮件中注明课程名称、作业名称、姓名、学号等信息，以及特殊情况的说明，我收到后会及时回复。

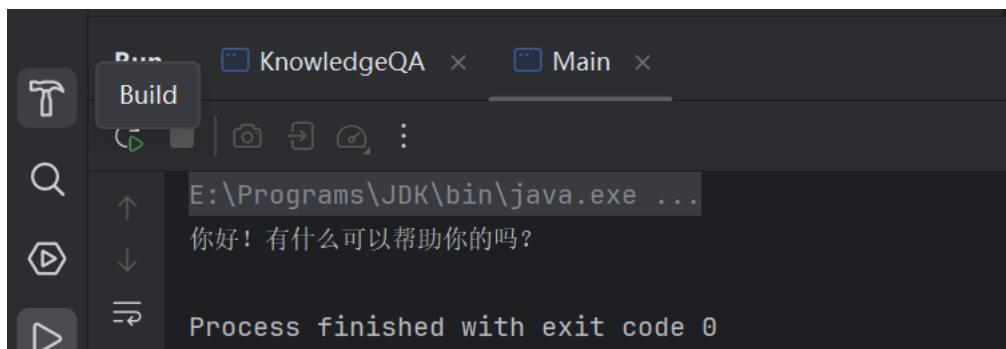
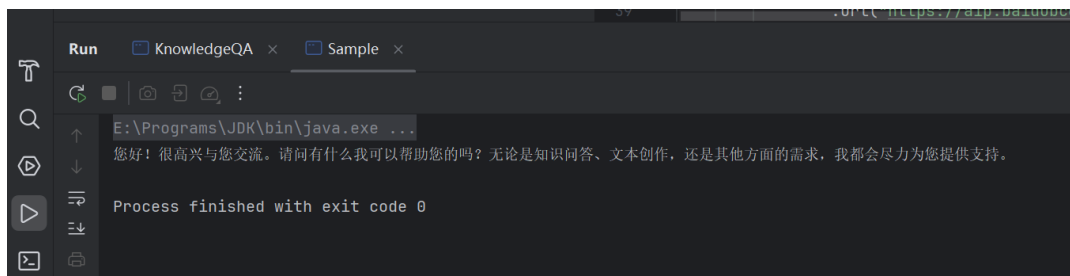
(8) 期末考试阶段补交无效。

(1). 编写 Java 应用程序，实现“结合大模型的 Java 知识问答机器人”。

- 人机交互界面：使用命令行窗口（不得分）或图形用户界面（10 分）
- 大模型：允许用户从腾讯混元大模型、百度文心一言大模型、阿里通义千问大模型或其他大模型中（至少支持两个大模型）选择使用一个大模型（20 分）
- 多线程&网络通信：能够并行处理多个用户的网络请求，每个用户请求都应该在一个新的线程中处理，以避免阻塞主线程（20 分）
- 测试案例：要求测试 Java 程序设计方面的选择题、判断题、程序填空题和编程题（每个题型至少测试两个例子），并就测试效果做出简单分析（20 分）
- 日志记录：在本地文本文件（.txt）中记录人机对话过程（20 分）
- 特色功能：可以列出自己实现的其他特色功能（每个特色功能 5 分，最多记两个特色功能共 10 分）

在报告中附上程序截图、完整的运行结果截图和详细的文字说明。

1.测试调用大模型 API



2.编写图形化界面

```
48
49     private void createComponents() { 1 usage
50         JPanel bottomPanel = new JPanel(new BorderLayout());
51         JPanel inputPanel = new JPanel(new FlowLayout(FlowLayout.LEFT, 5, 5));
52         JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT, 5, 5));
53
54         // 用户输入框
55         inputField = new JTextArea( rows: 5, columns: 50); // 设置行数和列数以确保可见大小
56         inputPanel.add(new JScrollPane(inputField)); // 添加滚动面板以支持多行文本区
57
58         // 模型选择器
59         modelSelector = new JComboBox<>(new String[]{"通义千问", "文心一言"});
60         inputPanel.add(modelSelector);
61
62         // 提问按钮
63         askButton = new JButton( text: "Ask");
64         askButton.addActionListener(this::onAskButtonClick);
65         buttonPanel.add(askButton);
66
67         // 查看历史按钮
68         JButton historyButton = new JButton( text: "查看历史");
69         historyButton.addActionListener(e -> new HistoryViewer(userId));
70         buttonPanel.add(historyButton);
71
72         bottomPanel.add(inputPanel, BorderLayout.CENTER);
73         bottomPanel.add(buttonPanel, BorderLayout.EAST);
74
75         // 输出显示区域
76         outputArea = new JTextArea( rows: 10, columns: 80); // 设置行数和列数以确保可见大小
77         outputArea.setEditable(false);
78         outputArea.setLineWrap(true); // 启用自动换行
79         outputArea.setWrapStyleWord(true); // 在单词边界处换行
80         add(new JScrollPane(outputArea), BorderLayout.CENTER);
81
82         add(bottomPanel, BorderLayout.SOUTH);
83     }
```

3.用户多线程

```
35     public UserChatWindow(int id) { 4 usages
36         this.userId = id;
37         this.LOG_FILE_PATH = "user_" + id + "_dialog_log.txt"; // 为每个用户创建独立的日志文件
38         setTitle("用户" + id + "的对话");
39         setSize( width: 800, height: 600);
40         setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE); // 关闭时只关闭当前窗口
41         setLocationByPlatform(true);
42
43         executorService = Executors.newFixedThreadPool( nThreads: 10); // 创建线程池
44
45         createComponents();
46         setVisible(true);
47     }
```

4.日志功能

```
85     private void log(String message) { 6 usages
86         try (BufferedWriter writer = new BufferedWriter(new FileWriter(LOG_FILE_PATH, append: true))) {
87             SimpleDateFormat sdf = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss");
88             String timestamp = sdf.format(new Date());
89             writer.write( str: timestamp + ": 用户" + userId + ": " + message);
90             writer.newLine();
91         } catch (IOException e) {
92             System.err.println("无法写入日志文件: " + e.getMessage());
93         }
94     }
```

5.实现功能连接

```
95
96 private void onAskButtonClick(ActionEvent e) { 1 usage
97     String question = inputField.getText().trim();
98     if (question.isEmpty()) {
99         JOptionPane.showMessageDialog( parentComponent: this, message: "请输入问题.", title: "输入错误", JOptionPane.ERROR_MESSAGE);
100         return;
101     }
102
103     String selectedModel = (String) modelSelector.getSelectedItem();
104     outputArea.append("用户" + userId + ": " + question + "\n");
105
106     // 记录用户的提问
107     log("用户" + userId + ": " + question);
108
109     if ("通义千问".equals(selectedModel)) {
110         // 使用线程池提交任务
111         executorService.submit(() -> callQwenApi(question));
112     } else if ("文心一言".equals(selectedModel)) {
113         // 使用线程池提交任务
114         executorService.submit(() -> callBaiduApi(question));
115     } else {
116         outputArea.append("所选模型不支持.\n");
117     }
118     inputField.setText("");
119 }
```

```
120
121 private void callQwenApi(String question) { 1 usage
122     Generation gen = new Generation();
123     Message systemMsg = Message.builder() MessageBuilder<capture of ?, capture of ?>
124         .role(Role.SYSTEM.getValue()) capture of ?
125         .content("You are a helpful assistant.")
126         .build();
127     Message userMsg = Message.builder() MessageBuilder<capture of ?, capture of ?>
128         .role(Role.USER.getValue()) capture of ?
129         .content(question)
130         .build();
131     GenerationParam param = GenerationParam.builder() GenerationParamBuilder<capture of ?, capture of ?>
132         .apiKey("sk-82b00fbaabd24f7893722ca8d7d9cc01") capture of ?
133         .model("qwen-plus")
134         .messages(Arrays.asList(systemMsg, userMsg))
135         .resultFormat(GenerationParam.ResultFormat.MESSAGE)
136         .build();
137
138     try {
139         GenerationResult result = gen.call(param);
140         final String response = result.getOutput().getChoices().get(0).getMessage().getContent();
141         SwingUtilities.invokeLater(() -> {
142             outputArea.append("通义千问: " + response + "\n\n");
143             // 记录AI的回答
144             log("通义千问: " + response);
145         });
146     } catch (ApiException | NoApiKeyException | InputRequiredException ex) {
147         SwingUtilities.invokeLater(() -> {
148             outputArea.append("错误: " + ex.getMessage() + "\n\n");
149             // 记录错误信息
150             log("错误: " + ex.getMessage());
151         });
152     }
```

```

154
155     private String getBaiduAccessToken() throws Exception { 1 usage
156         OkHttpClient client = new OkHttpClient().newBuilder().build();
157         final String API_KEY = "5m6s6YLMSuSUL6wM1mx0P7sn";
158         final String SECRET_KEY = "8IaRCIfXMvv6Z66YetQpYkFE74WIC1lH";
159         MediaType mediaType = MediaType.parse( $this$p$parse: "application/x-www-form-urlencoded");
160         RequestBody body = RequestBody.create(mediaType, content: "grant_type=client_credentials&client_id=" + API_KEY
161             + "&client_secret=" + SECRET_KEY);
162         Request request = new Request.Builder()
163             .url("https://aip.baidubce.com/oauth/2.0/token")
164             .method( method: "POST", body)
165             .addHeader( name: "Content-Type", value: "application/x-www-form-urlencoded")
166             .build();
167
168         try (Response response = client.newCall(request).execute()) {
169             if (!response.isSuccessful()) throw new IOException("意外的响应码 " + response);
170
171             return new JSONObject(response.body().string()).getString( key: "access_token");
172         } catch (IOException ex) {
173             throw new RuntimeException(ex);
174         }
175     }

```

```

177     private void callBaiduApi(String question) { 1 usage
178         try {
179             String accessToken = getBaiduAccessToken();
180             OkHttpClient client = new OkHttpClient().newBuilder().build();
181             MediaType mediaType = MediaType.parse( $this$p$parse: "application/json");
182
183             // 在用户的问题后附加要求简要回答的说明
184             String modifiedQuestion = question + " 请简要回答。";
185
186             JSONObject requestBody = new JSONObject();
187             JSONArray messagesArray = new JSONArray();
188             JSONObject userMessage = new JSONObject();
189             userMessage.put("role", "user");
190             userMessage.put("content", modifiedQuestion);
191             messagesArray.put(userMessage);
192             requestBody.put("messages", messagesArray);
193
194             RequestBody body = RequestBody.create(mediaType, requestBody.toString());
195             Request request = new Request.Builder()
196                 .url("https://aip.baidubce.com/rpc/2.0/ai_custom/v1/wenxinworkshop/chat/ernie_speed?access_token=" + accessToken)
197                 .method( method: "POST", body)
198                 .addHeader( name: "Content-Type", value: "application/json")
199                 .build();

```

```

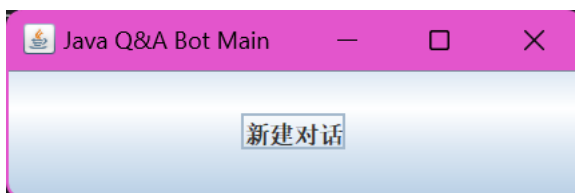
200
201         try (Response response = client.newCall(request).execute()) {
202             if (!response.isSuccessful()) throw new IOException("意外的响应码 " + response);
203
204             JSONObject jsonResponse = new JSONObject(response.body().string());
205             final String resultMessage = jsonResponse.getString( key: "result");
206             SwingUtilities.invokeLater(() -> {
207                 outputArea.append("文心一言: " + resultMessage + "\n\n");
208                 // 记录AI的回答
209                 log("文心一言: " + resultMessage);
210             });
211         } catch (IOException ex) {
212             SwingUtilities.invokeLater(() -> {
213                 outputArea.append("错误: " + ex.getMessage() + "\n\n");
214                 // 记录错误信息
215                 log("错误: " + ex.getMessage());
216             });
217         }
218     } catch (Exception ex) {
219         SwingUtilities.invokeLater(() -> {
220             outputArea.append("错误: " + ex.getMessage() + "\n\n");
221             // 记录错误信息
222             log("错误: " + ex.getMessage());
223         });
224     }
225 }

```

6.特色功能查看历史记录

```
13     public HistoryViewer(int userId) { no usages
14         setTitle("用户" + userId + "的对话历史");
15         setSize( width: 800, height: 600);
16         setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
17         setLocationByPlatform(true);
18
19         historyArea = new JTextArea( rows: 20, columns: 80);
20         historyArea.setEditable(false);
21         historyArea.setLineWrap(true);
22         historyArea.setWrapStyleWord(true);
23
24         add(new JScrollPane(historyArea), BorderLayout.CENTER);
25
26         JButton closeButton = new JButton( text: "关闭");
27         closeButton.addActionListener(this::onCloseClick);
28         add(closeButton, BorderLayout.SOUTH);
29
30         loadHistory( filePath: "user_" + userId + "_dialog_log.txt");
31
32         setVisible(true);
33     }
34
35     private void loadHistory(String filePath) { 1 usage
36         try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {
37             StringBuilder historyBuilder = new StringBuilder();
38             String line;
39             while ((line = reader.readLine()) != null) {
40                 historyBuilder.append(line).append("\n");
41             }
42             historyArea.setText(historyBuilder.toString());
43         } catch (IOException e) {
44             historyArea.setText("无法加载历史记录: " + e.getMessage());
45         }
46     }
47 }
```

使用效果:



点击“新建对话”后可以创建不同的进程用户

选择题测试:用户 1 通义千问

用户1的对话

用户1: 以下关于继承的叙述正确的是()。 A、在Java中类只允许单一继承 B、在Java中一个类只能实现一个接口 C、在Java中一个类不能同时继承一个类和实现一个接口 D、在Java中接口只允许单一继承
通义千问: 选项A是正确的。下面是对每个选项的具体解释:

A、在Java中类只允许单一继承: 这是正确的。Java 中的一个类只能继承自一个父类, 即不支持多继承。如果需要从多个类获取功能, 可以通过接口或者组合的方式来实现。

B、在Java中一个类只能实现一个接口: 这是错误的。Java 中的一个类可以实现多个接口。例如, `class MyClass implements Interface1, Interface2 {}` 是完全合法的。

C、在Java中一个类不能同时继承一个类和实现一个接口: 这是错误的。Java 中的一个类不仅可以继承一个父类, 还可以同时实现一个或多个接口。例如, `class MyClass extends ParentClass implements MyInterface {}` 是合法的。

D、在Java中接口只允许单一继承: 这是错误的。Java 中的接口可以继承多个其他接口。例如, `interface MyInterface extends Interface1, Interface2 {}` 是合法的。

因此, 正确答案是A。

用户1: 以下哪个不是Java的原始数据类型() A、int B、Boolean C、float D、char

通义千问: 选项B "Boolean" 不是Java的原始数据类型。在Java中, 原始数据类型是那些最基本的数据类型, 它们包括:

```
- 'byte'
- 'short'
- 'int'
- 'long'
- 'float'
- 'double'
- 'char'
- 'boolean'
```

Ask

查看历史

通义千问

用户1的对话

接口或者组合的方式来实现。

B、在Java中一个类只能实现一个接口: 这是错误的。Java 中的一个类可以实现多个接口。例如, `class MyClass implements Interface1, Interface2 {}` 是完全合法的。

C、在Java中一个类不能同时继承一个类和实现一个接口: 这是错误的。Java 中的一个类不仅可以继承一个父类, 还可以同时实现一个或多个接口。例如, `class MyClass extends ParentClass implements MyInterface {}` 是合法的。

D、在Java中接口只允许单一继承: 这是错误的。Java 中的接口可以继承多个其他接口。例如, `interface MyInterface extends Interface1, Interface2 {}` 是合法的。

因此, 正确答案是A。

用户1: 以下哪个不是Java的原始数据类型() A、int B、Boolean C、float D、char

通义千问: 选项B "Boolean" 不是Java的原始数据类型。在Java中, 原始数据类型是那些最基本的数据类型, 它们包括:

```
- 'byte'
- 'short'
- 'int'
- 'long'
- 'float'
- 'double'
- 'char'
- 'boolean'
```


注意, 虽然 "Boolean" 看起来与 "boolean" 相似, 但在这里 "Boolean" 是以大写字母开头的, 实际上指的是 'Boolean' 类, 它是 'boolean' 原始类型的包装类。因此, 正确答案是 B、Boolean。

Ask

查看历史

通义千问

判断题测试：用户 2 文心一言

 用户2的对话

—

□

×

用户2: 判断题: Java语言有三种技术平台, 分别是JavaSE、JavaME、JavaEE

文心一言: 题目陈述正确。Java语言确实有三种主要的技术平台, 分别是:

1. **JavaSE (Java Standard Edition)**: 这是Java的标准版, 主要用于桌面应用程序的开发, 如开发图形用户界面(GUI)应用程序等。JavaSE包含了Java的核心类库和API, 是所有Java开发的基础。
2. **JavaME (Java Platform, Micro Edition)**: 这是Java的微版, 主要用于嵌入式设备和移动设备的开发, 如手机、PDA等。JavaME为这些设备提供了丰富的API支持, 包括用户界面、网络连接、数据持久化等。不过, 近年来随着移动应用的主流趋势转向原生应用和混合应用, JavaME的重要性已经降低。
3. **JavaEE (Java Platform, Enterprise Edition)**: 这是Java的企业版, 主要用于企业级应用的开发, 如大型的网络应用程序、企业级服务器应用程序等。JavaEE提供了一套完整的解决方案, 包括Web服务、企业应用框架、安全服务等, 可以帮助开发者快速构建复杂的企业级应用。

这三种技术平台共同构成了Java语言的全貌, 为开发者提供了丰富的选择和可能。

用户2: 判断题: continue 语句只用于循环语句中, 它的作用是跳出循环


文心一言: 题目的判断是正确的。`continue`语句确实只在循环语句(如`for`循环或`while`循环)中使用, 其主要作用是跳过当前循环的剩余部分并直接进入下一次循环。换句话说, 它不会执行当前循环中的剩余代码, 而是直接回到循环的开始并检查条件以决定是否进行下一次迭代。然而, `continue`并不会完全跳出整个循环, 只是跳过当前循环的剩余部分。如果循环条件仍然满足, 循环会继续进行。因此, 题目的描述“它的作用是跳出循环”不完全准确, 更准确的说法是跳过当前循环的剩余部分。

Ask

查看历史

文心一言 ▼

填空题测试：用户 3 通义千问、文心一言

 用户3的对话

—

□

×

用户3: 填空题: Java是从____语言改进重新设计的

通义千问: Java 是从 Oak 语言改进重新设计的。Oak 是由 Sun Microsystems (太阳微系统公司, 后被 Oracle 收购) 的 James Gosling 在 1991 年左右开发的一种编程语言, 最初是为消费类电子产品设计的。后来由于 Oak 名称的版权问题以及对互联网应用的适应性考虑, Oak 被重新命名为 Java, 并在 1995 年正式发布。Java 的设计目标之一是“一次编写, 到处运行”, 这使得它成为了广泛使用的编程语言之一。

用户3: 填空题: Java程序的基本组成单元是____

文心一言: Java程序的基本组成单元是“类(Class)和接口(Interface)”。这是因为Java是一种面向对象的编程语言, 所有的代码都围绕类和接口进行组织和设计。类是创建对象的模板, 定义了对对象的属性和行为。接口则定义了一组方法的契约, 可以被多个类实现以实现多态性。除此之外, Java程序还包括其他组成部分, 如变量、常量、数据类型、运算符、控制结构(如条件语句和循环语句)等。这些组成部分共同构成了Java程序的完整结构。

Ask

查看历史

文心一言 ▼

编程题测试：用户 5 通义千问、文心一言

用户的问题

用户5 编写一个Java程序在屏幕上输出1! + 2! + 3! + ... + 10! 的和。

提示千问: 编写一个Java程序来计算并输出1! + 2! + ... + 10! 的和，可以按照以下步骤进行：

1. 创建一个方法来计算各个数字的阶乘。
2. 在方法中，使用一个循环来累加从1到10的所有数字的阶乘。
3. 最后，输出累加的结果。

下面是完整的Java程序代码：

```
import java.util.Scanner;

public class FactorialSum {
    public static void main(String[] args) {
        int n = 10; // 计算1! + 2! + ... + 10!
        long sum = 0;

        for (int i = 1; i <= n; i++) {
            sum += factorial(i);
        }

        System.out.println("1! + 2! + ... + 10! 的和是: " + sum);
    }

    // 计算阶乘的方法
    public static long factorial(int number) {
        if (number == 0 || number == 1) {
            return 1;
        }
        long result = 1;
        for (int i = 2; i <= number; i++) {
            result *= i;
        }
        return result;
    }
}
```

代码解释

1. "主方法" `main`：

- 定义了一个变量 `n` 来表示计算的范围（1到10）。
- 初始化了一个变量 `sum` 用于存储阶乘的累加和。
- 使用一个 `for` 循环从1到10，每次调用 `factorial` 方法计算当前数字的阶乘，并将其加到 `sum` 中。
- 最后，使用 `System.out.println` 输出结果。

2. "阶乘方法" `factorial`：

- 接受一个整数参数 `number`。
- 如果 `number` 是0或1，直接返回1（因为0! 和 1! 都等于1）。
- 否则，初始化一个变量 `result` 为1，然后使用一个 `for` 循环从2到 `number`，将每个数字乘到 `result` 上。
- 返回计算后的 `result`。

Ask 查看历史

用户的问题

用户5 编写一个Java程序在屏幕上输出"你好！"

文心一言: 以下是一个简单的Java程序，可以在屏幕上输出"你好！"：

```
import java.util.Scanner;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("你好！");
    }
}
```

简而言之，这是一个基础的Java程序，其中定义了一个名为 `HelloWorld` 的公共类。在这个类中，有一个名为 `main` 的静态方法。它是Java程序的入口点。在 `main` 方法中，使用了 `System.out.println` 语句向标准输出"你好！"输出到控制台。当执行这个程序时，它会在屏幕上显示"你好！"这句话。

Ask 查看历史

在测试问题的时候可以看到，程序获得的返回结果和正常使用大模型进行提问效果大差不差，可以说明我们的 API 调用效果良好。

特色功能 1：查看历史对话（查看用户日志）

The screenshot shows a Java IDE with a file named "用户代码清单7-18.java". The code calculates the sum of factorials from 1! to 10!.

```
//Java  
public class FactorialSum {  
    public static void main(String[] args) {  
        int n = 10; // 计算1! + 2! + ... + 10!  
        long sum = 0;  
  
        for (int i = 1; i <= n; i++) {  
            sum += factorial(i);  
        }  
  
        System.out.println("1! + 2! + ... + 10! 的和是:" + sum);  
    }  
  
    // 计算阶乘的方法  
    public static long factorial(int number) {  
        if (number == 0 || number == 1) {  
            return 1;  
        }  
  
        long result = 1;  
        for (int i = 2; i <= number; i++) {  
            result *= i;  
        }  
        return result;  
    }  
}
```

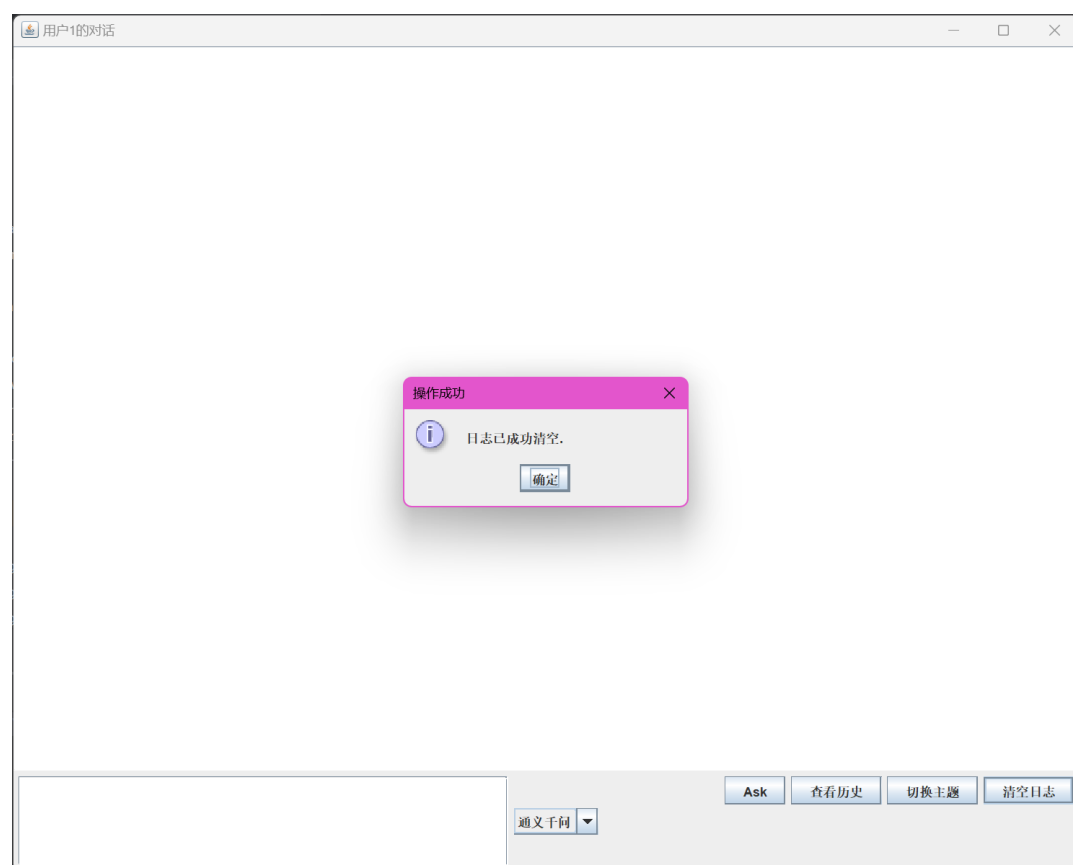
程序代码解析

- 主方法 main**:
 - 定义了一个变量 `n` 来表示计算的范围 (1到10)。
 - 初始化一个变量 `sum` 用于存储阶乘的累加和。
 - 使用一个 `for` 循环从1到10，每次调用 `factorial` 方法计算当前数字的阶乘，并将其加到 `sum` 中。
 - 最后，使用 `System.out.println` 输出结果。
- 阶乘方法 factorial**:
 - 接受一个整数参数 `number`。
 - 如果 `number` 是0或1，直接返回1 (因为0!和1!都等于1)。
 - 否则，初始化一个变量 `result` 为1，然后使用一个 `for` 循环从2到 `number`，将每个数字乘到 `result` 上。
 - 循环计算后的 `result`。

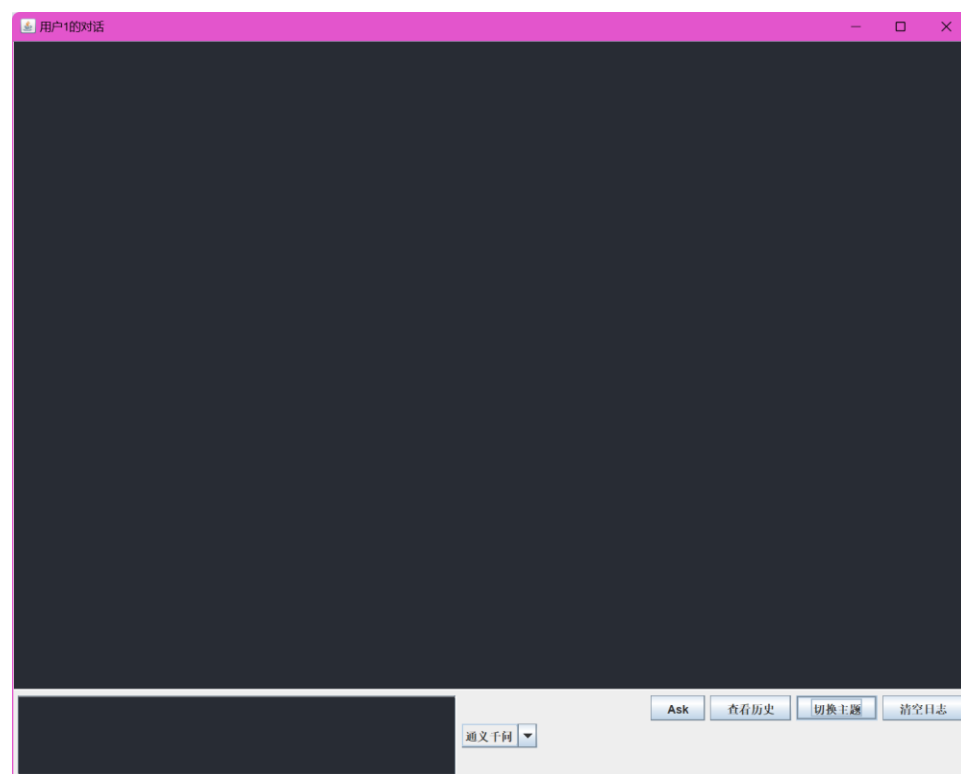
运行这个程序，你将看到输出：

```
1! + 2! + ... + 10! 的和是: 4037913
```

特色功能 2：清空历史记录（日志）

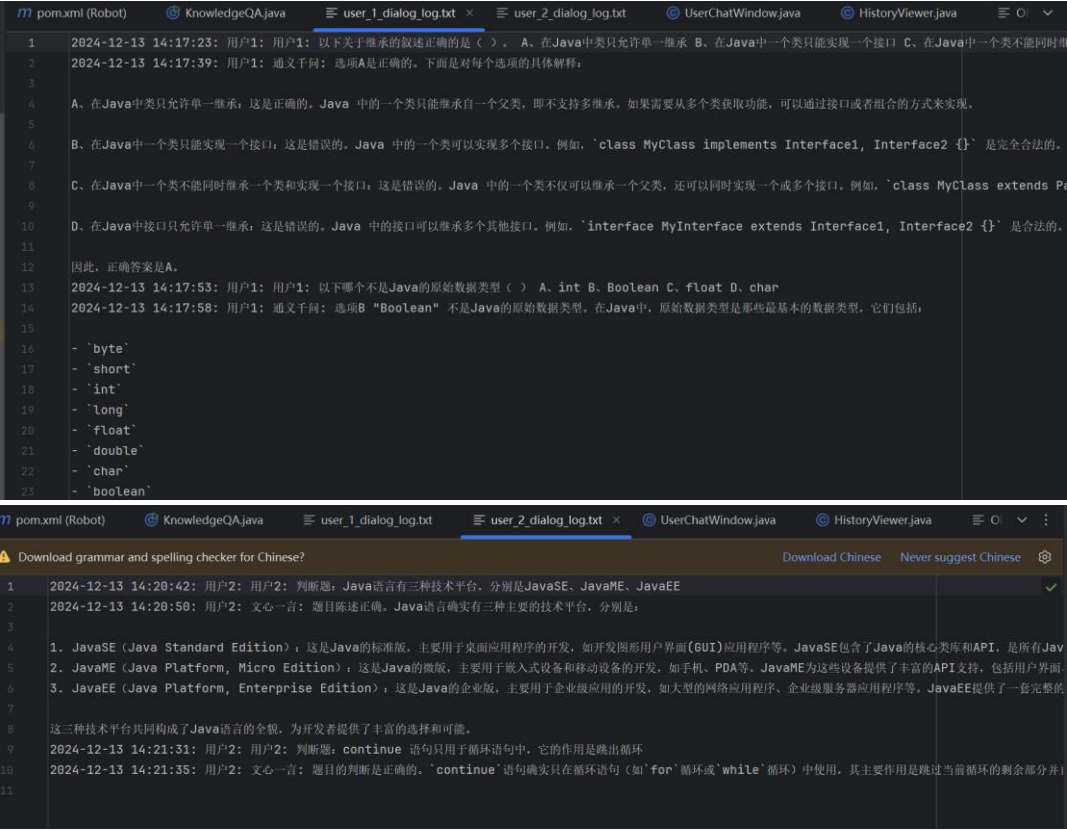


特色功能 3：切换主题色



日志文件：

```
≡ user_1_dialog_log.txt
≡ user_2_dialog_log.txt
≡ user_3_dialog_log.txt
≡ user_5_dialog_log.txt
```



+++++

其他（例如感想、建议等等）。

学了好多调 API 的东西，还有给大模型提示词。

感觉作业难度还是挺高的

深圳大学学生课程作业报告用纸

指导教师批阅意见：

成绩评定：

指导教师签字：

2024 年 月 日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生课程作业报告时间应在学生提交课程作业报告时间后 10 日内。