作业1

https://api-docs.deepseek.com/zh-cn/

```
from openai import OpenAI

client = OpenAI(api_key="<DeepSeek API Key>", base_url="https://api.deepseek.com")

response = client.chat.completions.create(
    model="deepseek-chat",
    messages=[
        {"role": "system", "content": "You are a helpful assistant"},
        {"role": "user", "content": "Hello"},
    ],
    stream=False
)

print(response.choices[0].message.content)
```

作业2

ollama 问答：https://blog.csdn.net/weixin_49938804/article/details/138320957

作业 3

检索增强生成（RAG）作为一种创新的AI框架，通过将生成式大语言模型（LLM）与外部可更新、可信赖的知识源相结合，从根本上解决了LLM固有的“幻觉”、知识时效性滞后以及不可控性等核心挑战。

检索增强生成（RAG）是一种将传统信息检索系统（如搜索和数据库）的优势与生成式大语言模型（LLM）的能力相结合的AI框架 。其核心思想是，在LLM生成回答之前，首先从外部知识库中检索与用户查询相关的实时信息，并将这些信息作为额外的上下文提供给LLM，从而引导LLM生成更准确、更具事实依据的回答 。这种方法将LLM的语言技能与其内部预训练数据之外的外部数据和世界知识相结合，能够使输出更准确、更及时，并且与特定需求高度相关 。

RAG系统的工作流程通常包含几个关键步骤：首先是检索和预处理，系统利用强大的搜索算法查询外部数据源，如网页、知识库或数据库，并对检索到的信息进行预处理。

RAG系统通常由三个主要模块构成：知识构建与索引、检索和生成。这些模块协同工作，以确保最终输出的准确性和上下文相关性 。

- **数据准备**：首先，需要识别、获取和加载要与LLM共享的源文档，这些文档可以是文本文件、数据库表或PDF格式 。在此阶段，进行数据清理与标注至关重要。收集到的数据可能存在错误、冗余或不一致等问题，需要进行清理，并添加元数据信息以提高索引数据的质量 。
- **文本切分策略（Chunking）**：将文档切分成可管理的小块是RAG工作流中的一个关键预处理步骤 。不当的切分可能会导致检索到不相关或不完整的响应，从而降低用户体验 。目前，业界主要采用以下几种主流策略：
    - **固定大小/递归切分**：这是最常见的切分方法，基于字符或Token进行硬切，可以指定前后重叠的大小来保留上下文语义 。

- **页面级切分**：这种策略将文档的每一页都作为一个单独的块 。有研究表明，页面级切分是RAG系统最有效的默认策略，因为它在不同文档类型中提供了最高的平均准确率和最一致的性能 。页面的静态边界也使引用和追溯功能更易于实现，因为每个块都对应一个固定的页面来源 。
- **语义切分**：旨在确保每个块包含一个完整的语义单元。这种方法可以基于不同段落的嵌入差异或使用LLM辅助进行切分，以达到更“智能”的效果 。