一、总体设计

1.1 系统概述

本系统旨在构建一个本地文档搜索引擎，能够处理多种格式的文档（如PDF、Word、HTML等）。系统的核心功能包括：提取文档内容、建立索引、执行搜索等。用户通过输入搜索关键词，能够快速地从文档中获取相关内容。

1.2 系统架构

本系统主要由以下模块组成：

1. **文档内容提取模块**：使用Apache Tika和Jsoup库从不同格式的文档中提取文本内容。
2. **Lucene索引模块**：通过Lucene对文档进行索引处理，以便在搜索时高效地查找相关文档。
3. **搜索模块**：通过Lucene提供的查询功能进行关键词搜索，并返回匹配的文档路径。
4. **用户交互模块**：提供命令行界面（CLI），允许用户输入搜索关键词并查看搜索结果。

1.3 技术栈

- **Java**：主要开发语言。
- **Apache Tika**：用于解析PDF、Word等格式的文档并提取文本内容。
- **Jsoup**：用于解析HTML文档。
- **Lucene**：用于创建文档索引并实现高效搜索。
- **Maven**：项目构建和依赖管理工具。

二、详细设计

2.1 模块划分

2.1.1 文档内容提取模块

- **功能**：该模块的核心功能是从PDF、Word、HTML等文件中提取文本内容。对于PDF和Word格式，使用Apache Tika进行提取；对于HTML格式，使用Jsoup解析并提取页面内容（如标题和正文）。
- **关键类**：
  - `DocumentExtractor`：负责提取PDF、Word等格式的文档内容，依赖Apache Tika。
  - `HtmlExtractor`：负责提取HTML格式文档的内容，依赖Jsoup。

2.1.2 Lucene索引模块

- **功能**：通过Lucene的`IndexWriter`将文档的内容及元数据（如文档路径）写入索引文件。该模块还负责在索引中添加新文档，并将文档的内容进行索引以便快速检索。
- **关键类**：
  - `Indexer`：创建Lucene索引，负责将每个文档的内容和路径索引到Lucene索引中。
  - `Searcher`：提供搜索功能，接受用户的查询，并根据Lucene索引返回搜索结果。

2.1.3 搜索模块

- **功能**：根据用户输入的搜索关键词，通过Lucene的查询接口进行检索，返回匹配的文档路径。

- **关键类**：
  - `Searcher`：使用Lucene的`QueryParser`解析用户的查询关键词，然后执行搜索，返回相关文档路径。

2.1.4 用户交互模块

- **功能**：通过命令行界面与用户交互，允许用户输入查询关键词，并显示搜索结果。
- **关键类**：
  - `SimpleSearchEngine`：负责管理整个搜索引擎的流程，包括文档索引的创建和用户搜索。

2.2 数据结构设计

1. **Lucene索引结构**：每个文档通过Lucene的`Document`对象进行表示，包含两个字段：
   - `path`：存储文档的绝对路径，作为文档的唯一标识。
   - `content`：存储文档的文本内容，作为搜索的索引内容。
2. **查询结果**：Lucene返回一个`TopDocs`对象，包含多个`ScoreDoc`对象，每个`ScoreDoc`对象表示一个文档的搜索结果。每个结果都包括文档的ID（用于索引查找）和该文档与查询的匹配度（得分）。

2.3 系统流程

1. **文档索引构建流程**：
   - 用户将待索引的文档放入指定目录（如`docs/`目录）。
   - 系统遍历该目录，识别每个文件的类型（PDF、Word、HTML等），并使用Tika或Jsoup提取文本内容。
   - 系统将文档的内容与路径通过Lucene索引存储在指定目录（如`index/`目录）。
2. **搜索流程**：
   - 用户输入搜索关键词。
   - 系统通过Lucene的`QueryParser`解析用户的查询，并进行搜索。
   - 系统根据匹配度返回最相关的文档路径。

三、测试与运行

3.1 测试目标

本系统的测试目标是验证文档提取、索引构建、搜索功能的正确性与效率。但目前整个项目尚未debug完成。

3.2 单元测试

1. **文档提取测试**：
   - 测试各种格式文档（PDF、Word、HTML）的提取功能，确保Tika和Jsoup能够正确解析并提取内容。
   - 断言提取的文本内容不为空，且与原文档内容一致。
2. **Lucene索引测试**：
   - 测试Lucene索引的正确性，确保每个文档的路径和内容都被正确地添加到索引中。
   - 断言索引中包含所有预期的文档。
3. **搜索测试**：

- 测试搜索功能，确保根据关键词能够找到相关的文档路径。
- 断言搜索结果返回的文档路径正确，并且符合预期。

3.3 运行步骤

1. **构建索引**：
   - 将待索引的文档（PDF、Word、HTML等）放入`docs/`目录中。
   - 运行`SimpleSearchEngine`类，系统将遍历文档并建立索引。
2. **搜索功能**：
   - 运行`SimpleSearchEngine`类后，系统将提示输入搜索关键词。
   - 输入关键词后，系统会返回匹配的文档路径。

四、总结

4.1 系统优势

- **简洁高效**：系统通过使用Apache Tika和Jsoup，能够支持多种文档格式的文本提取。使用Lucene建立高效的索引，支持快速搜索。
- **可扩展性强**：系统可以轻松地扩展支持更多文档格式、优化搜索算法、添加用户界面等。
- **灵活性**：可以根据用户需求调整搜索逻辑，支持多种文档类型的定制化处理。

4.2 存在的不足

- **性能问题**：当文档数量非常大时，Lucene的搜索性能可能会受到影响，可以考虑使用分布式搜索引擎（如Elasticsearch）进行扩展。
- **用户界面**：目前仅提供命令行交互方式，未来可以增加图形用户界面（GUI）或Web接口来增强用户体验。
- **全文索引优化**：Lucene的默认配置可能未达到最佳的索引性能，未来可以对Lucene索引进行优化（如使用自定义分词器、压缩索引等）。

4.3 下一步工作

- **支持更多格式**：除了PDF、Word、HTML之外，未来可以支持更多的文件格式（如Excel、PPT、Markdown等）。
- **图形化界面**：为系统增加一个图形用户界面，使得用户能够更加方便地上传文档、执行搜索。
- **性能优化**：对Lucene索引和搜索的性能进行进一步优化，特别是对大规模文档库的支持。