

华中科技大学

# 课程实验报告

课程名称：Java 语言程序设计

实验名称：基于内存的搜索引擎设计和实现

院    系：计算机科学与技术

专业班级：CS2206

学    号：U202215543

姓    名：闫晓任

指导教师：纪俊文

2024 年 5 月 9 日

# 一、需求分析

## 1. 题目要求

实现一个基于内存的英文全文检索搜索引擎，需要完成以下功能：

**功能 1：**将指定目录下的一批.txt 格式的文本文件扫描并在内存里建立**倒排索引**，这里面包含必须的子功能包括：

- (1) 读取文本文件的内容；
- (2) 将内容切分成一个个的单词；
- (3) 过滤掉其中一些不需要的单词,例如数字、停用词（the, is and 这样的单词）、过短或过长的单词（例如长度小于 3 或长度大于 20 的单词）；
- (4) 利用 Java 的集合类在内存里建立过滤后剩下单词的倒排索引；
- (5) 内存里建立好的索引对象可以序列化到文件，同时可以从文件里反序列化成内存里的索引对象；
- (6) 可以在控制台输出索引的内容。

//完成功能 1 后用 experimenttest 测试之后

**功能 2：**基于构建好的索引，实现单个搜索关键词的全文检索，包含的子功能包括：

- (1) 根据搜索关键词得到命中的结果集合；
- (2) 可以计算每个命中的文档的得分，并根据文档得分对结果集排序；
- (3) 在控制台显示命中的文档的详细信息，如文档的路径、文档内容、命中的关键词信息（如在文档里出现次数）、文档得分；

**功能 3：**基于构建好的索引，实现二个搜索关键词的全文检索。包含的子功能包括：

- (1) 支持这二个关键词的与或查询。与关系必须返回同时包含这二个单词的文档集合，或关系返回包含这二个单词中的任何一个的文档集合；
- (2) 可以计算每个命中的文档的得分，并根据文档得分对结果集排序；
- (3) 在控制台显示命中的文档的详细信息，如文档的路径、文档内容、命中的关键词信息（如在文档里出现次数）、文档得分；

**功能 4：**基于构建好的索引，实现包含二个单词的短语检索，即这二个单词必须在作为短语文档里出现，它们的位置必须是相邻的。**这个功能为进阶功能。**

除了以上功能上的要求外，其他要求包括：

(1) 针对搜索引擎的倒排索引结构，已经定义好了创建索引和全文检索所需要的抽象类和接口。学生必须继承这些预定义的抽象类和实现预定义接口来完成实验的功能，不能修改抽象类和接口里规定好的数据成员、抽象方法；也不能在预定义抽象类和接口里添加自

已新的数据成员和方法。但是实现自己的子类和接口实现类则不作任何限定。

(2) 自己实现的抽象类子类和接口实现类里的关键代码必须加上注释，其中每个类、每个类里的公有方法要加上 **Javadoc** 注释，并自动生成 **Java API** 文档作为实验报告附件提交。

(3) 使用统一的测试文档集合、统一的搜索测试案例对代码进行功能测试，构建好的索引和基于统一的搜索测试案例的检索结果最后输出到文本文件里作为实验报告附件提交。

(4) 本实验只需要基于控制台实现，实验报告里需要提供运行时控制台输出截屏。

关于搜索引擎的倒排索引结构、相关的抽象类、接口定义、还有相关已经实现好的工具类会在单独的 **PPT** 文档里详细说明。同时也为学生提供了预定义抽象类和接口的 **Java API** 文档和 **UML** 模型图。

## 2. 需求分析

### (1) 预处理与索引构建

子功能 1.1（读取文件）：

确保程序能够遍历出指定目录下的所有.txt 文件，并将这些.txt 文件读取加载到倒排索引之中。

处理文件读取异常，如文件不存在、无权限访问等，需要抛出异常，并记录日志或在控制台提示。

子功能 1.2（内容切分）：

使用正则表达式或自然语言处理库进行单词分割，需要考虑标点符号分隔及大小写转换。

子功能 1.3（单词过滤）：

定义并维护一份停用词列表，支持动态加载或更新。

实现过滤逻辑，去除数字、停用词、过短或过长的单词等停用词，在单词载入时用停用词表进行过滤，让后续的倒排索引构建中不再出现这些单词。

子功能 1.4（倒排索引构建）：

对于每个文档，经过上面的单词过滤后，一个个地读取单词，为该文件出现的单词构建相应的位置链表。

对于每个单词，按照文档名进行区分，每个文档对应的链表记录该单词出现的位置，为每一个单词生成一个类似于二维数组的数据结构进行保存。

子功能 1.5（序列化/反序列化）：

序列化用于将生成的倒排索引保存在.dat 目标文件中，而反序列化则是将该目标文件进行解析，重新加载倒排索引到内存之中。这样，我们就可以避免重复构建倒排索引的过程，实现一次构建，多次搜索。甚至只要使用了相同的序列化和反序列化方式，就可以实现一次构建，到处搜索，不受其他限制，生成的.dat 文件可以作为倒排索引的核心进行传播。

为了实现序列化和反序列化，我们需要将倒排索引中存储索引数据的类实现 **FileSerializable** 接口，并覆盖相应的 **readObject** 和 **writeObject** 方法，通过调用这两个方法实

现序列化和反序列化。

子功能 1.6（输出索引内容）：

设计清晰的输出格式展示索引内容，包括但不限于单词、对应文档 ID 列表及其频率。

## **(2) 搜索功能**

### **2.1 功能 2 细化**

子功能 2.1（关键词匹配）：

需要对构建好的倒排索引实现搜索的方法，也即根据单词本身内容，用单词进行搜索。

子功能 2.2（文档打分与排序）：

设计文档评分算法，将搜索得到的各个文档，根据单词的匹配程度进行打分，并根据得分对结果进行降序排序。

子功能 2.3（结果展示）：

输出包括文档路径、内容摘要、得分等信息，便于用户快速浏览。

### **2.2 功能 3 细化**

子功能 3.1（与或查询）：

支持 AND/OR 逻辑运算符，实现布尔查询解析器，正确解析用户查询语句。

对于 AND 查询，交集两个单词的文档 ID 集合；对于 OR 查询，合并两个单词的文档 ID 集合。

子功能 3.2（得分计算与排序）：

针对与或查询，调整评分策略以反映逻辑关系的影响，如 AND 查询下文档得分可能更高。

### **2.3 功能 4 细化**

短语检索:利用功能 3 中的 AND 查询进行初步筛选，设计出比 AND 更高一级的词组查找。

## **(3) 异常处理与日志**

倒排索引的构建中可能出现很多异常，而一旦倒排索引构建完毕，进行搜索时，可能出现的异常就大大减少。

构建时，可能出现的异常如下

1. .txt 文档不存在，或者尝试打开.txt 文档的过程中出现了某些问题。
2. 构建倒排索引的过程中可能由于各种原因出现异常，如内存不足等原因。
3. 将倒排索引序列化写入到.dat 文件以及逆向解析的过程中可能出现异常，如文件读写异常等 IO 异常。
4. 搜索时，此时倒排索引已经够构建完毕，可能出现异常的机会大大减少。

## 二、系统设计

### 1. 概要设计

根据题目要求中的功能需求，我们需要设计两个 **main** 函数接口，一个用来处理各种 **txt** 文件，生成倒排索引目录，将文件分析得到的数据存储在其中；另一个 **main** 函数接口用来加载倒排索引目录，并根据输入的单词以及模式进行搜索。因此，我们的主要的代码文件和数据结构都服务于第一部分，而第二部分就相对简单，只需要根据第一部分，逆向解析倒排索引目录，就可以直接检索到单词的相关信息。

总体流程图如下图所示。

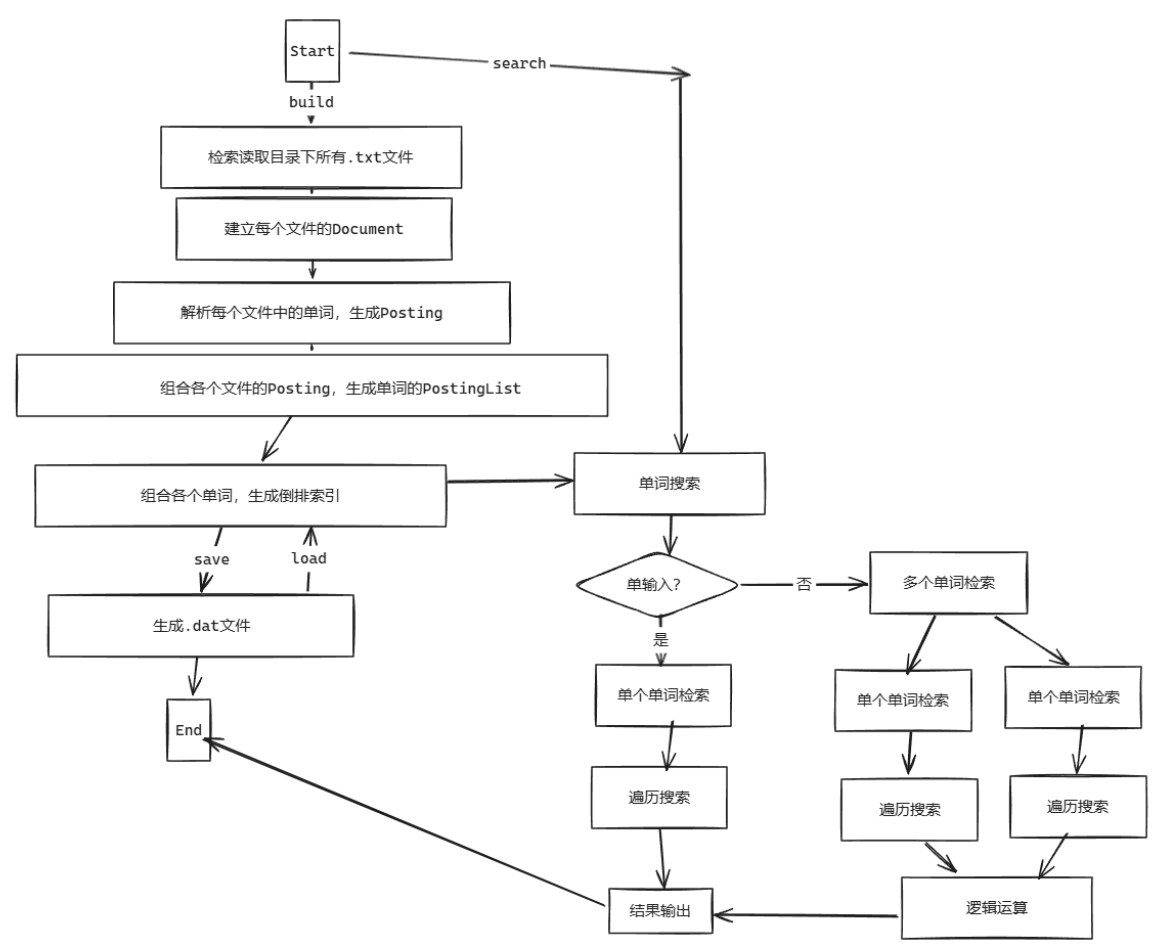


图 2.1

### (1) 生成倒排索引

倒排索引中，我们将单词和 **txt** 文档两个信息分别作为两项关键字。单词索引存储一个单词在所有 **txt** 文档中所有的信息；而文档索引用来存储一个文档的信息，也即文档内容。

在单词索引中，我们希望能够根据该单词的拼写获取该单词在所有提供的 **txt** 文件中的所有信息。由于一个单词可能在一个文件中出现多次，也有可能多个文件中出现，因此，我们需要构建如下图所示的数据结构。

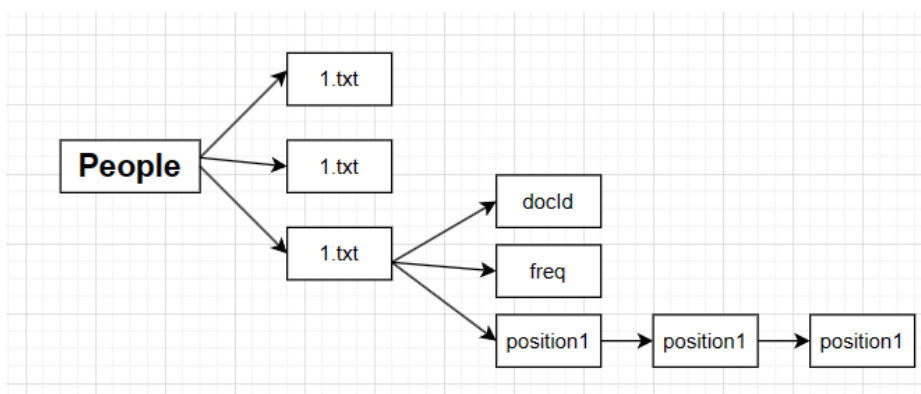


图 2.2

生成单词的倒排索引时，我们面对的首先是各个.txt 文本文件。我们首先需要记录这个.txt 文档的所有内容，然后分析文档中存在有哪些单词，生成每个单词在这个文档中的位置信息列表，插入到相应的单词索引中。

创建好倒排索引后，我们调用倒排索引的 **save** 方法，将整个倒排索引序列化保存到.dat 数据文件中，以便进行解析。

## (2) 解析倒排索引和搜索实现

解析倒排索引起始就是生成倒排索引的逆过程。调用解析的 **load** 方法将倒排索引加载到内存之中，等待搜索。

由于我们有多种搜索需求，因此我们需要分别设计单个单词、OR、AND、词组的搜索方法。其中，单个单词搜索方法设计时终点，因为 OR 和 AND 设计都可以分别调用单个搜索两次，然后再用一些逻辑判断生成响应结果。对于词组搜索，可以调用 AND 搜索，以此为基础，再根据两个单词是否相邻进行进一步筛选。

## 2. 详细设计

设计每个模块的实现算法（处理流程）、所需的局部数据结构。具体介绍每个模块/子程序的功能、入口参数、出口参数、流程（图）等。

实验项目的 UML 图如下所示。

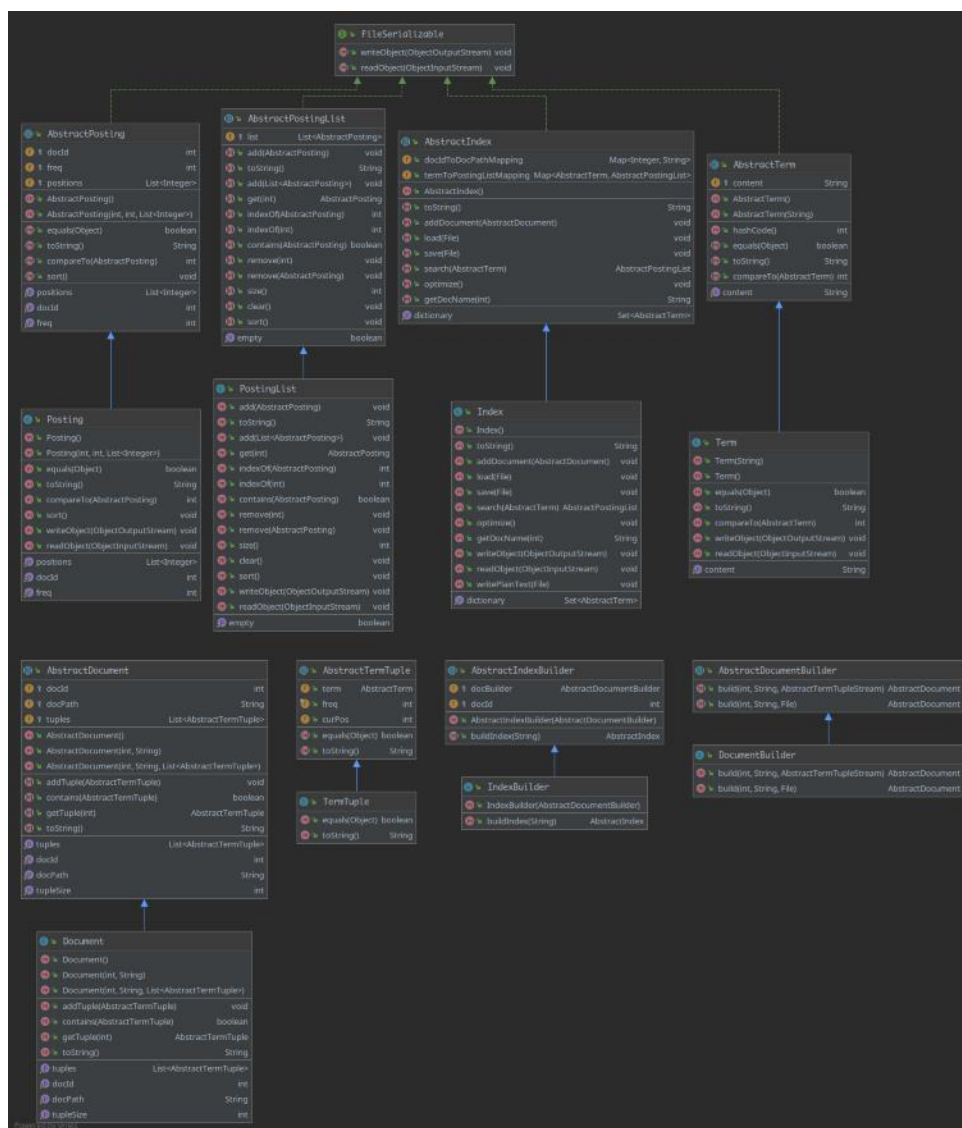


图 2.3

## index 模块

index 模块主要包含倒排索引的数据结构。

### Document:

文档对象是解析一个文本文件得到结果，文档对象里面包含： 文档 id. 文档的绝对路径. 文档包含的三元组对象列表，一个三元组对象是抽象类 AbstractTermTuple 的子类实例。

核心方法：public AbstractTermTuple getTuple(int index) 获得文档包含的三元组列表

public void addTuple(AbstractTermTuple tuple) 向文档对象里添加三元组，要求不能有内容重复的三元组

### DocumentBuilder:

Document 构造器。功能应该是由解析文本文档得到的 TermTupleStream，产生 Document 对象。

核心方法：public AbstractDocument build(int docId, String docPath, AbstractTermTupleStream termTupleStream) 由解析文本文档得到的 TermTupleStream,构造 Document 对象.解析流程如下

图所示。

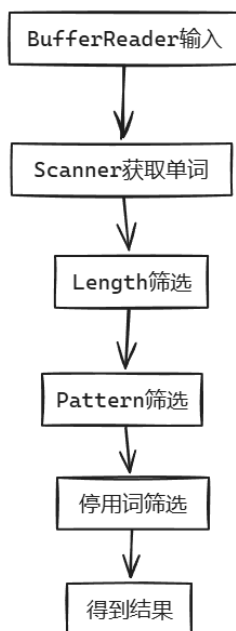


图 2.4

### Index:

**docIdToDocPathMapping:** 类型为 `Map<Integer, String>`，保存了文档 Id 和文档绝对路径之间的映射关系  
**termToPostingListMapping:** 类型为 `Map<AbstractTerm, AbstractPostingList>`，保存了每个单词与其对应的 `PostingList` 的映射关系。

这里没有必要用专门的数据结构来存放字典内容，我们直接通过 `termToPostingListMapping.keySet()` 方法就可以得到字典。

一个倒排索引对象包含了一个文档集合的倒排索引。内存中的倒排索引结构为 `HashMap`，key 为 `Term` 对象，value 为对应的 `PostingList` 对象。

`AbstractIndex` 包含了多个 `<AbstractTerm, AbstractPostingList>` 这样的 key-value 对，这些 key-value 对维护了单词到 `PostingList` 的链接关系。

`Index` 中含有这些文件的所有信息。

核心方法：`public void addDocument(AbstractDocument document)` 添加文档到索引，更新索引内部的 `HashMap`

`public void save(File file)` 将在内存里构建好的索引写入到文件。

`public void load(File file)` 从索引文件里加载已经构建好的索引。

### IndexBuilder:

`IndexBuilder` 是索引构造器的抽象父类 需要实例化一个具体子类对象完成索引构造的工作。

核心方法：`public AbstractIndex buildIndex(String rootDirectory)` 构建指定目录下的所有文本文件的倒排索引。需要遍历和解析目录下的每个文本文件，得到对应的 `Document` 对象，再依次加入到索引，并将索引保存到文件。



**Posting:**

Posting 对象代表倒排索引里每一项，是一个文档中所有关于某个单词的所有信息。一个 Posting 对象包括：包含单词的文档 id、单词在文档里出现的次数、单词在文档里出现的位置列表（位置下标不是以字符为编号，而是以单词为单位进行编号）。

核心方法：`public void sort()`对内部 positions 从小到大排序。

`public void setPositions(List<Integer> positions)` 设置单词在文档里出现的位置列表

**PostingList:**

PostingList 对象包含了各个文件中一个单词的 Posting 列表。也就是这些文件中单词的所有信息。

核心方法：`public void add(List<AbstractPosting> postings)` 添加 Posting 列表，要求不能有内容重复的 posting。

**Term:**

Term: 一个单词，是自主设计的 String 类。

**TermTuple:**

一个 TermTuple 对象为三元组(单词，出现频率，出现的当前位置)。当解析一个文档时，每解析到一个单词，应该产生一个三元组，其中 freq 始终为 1。最终应该将 TermTuple 合并成一个 Posting，组合成该文件关于这个单词的信息。

核心方法：`public TermTuple(int curPos, AbstractTerm term)`依据 Term 的内容构造三元组

**parse 模块**

parse 模块主要包含对单词过滤的方法。

**LengthTermTupleFilter:**

LengthTermTupleFilter 是抽象类 AbstractTermTupleFilter 的子类。将长度小于 3 或长度大于 20 的单词过滤掉。

核心方法：`public AbstractTermTuple next()`获得过滤后的下一个三元组。

**PatternTermTupleFilter:**

PatternTermTupleFilter 是抽象类 AbstractTermTupleFilter 的子类。选择英文字母组成的 Term 对应的三元组。

核心方法：`public AbstractTermTuple next()`获得过滤后的下一个三元组。

**StopWordTermTupleFilter:**

StopWordTermTupleFilter 是抽象类 AbstractTermTupleFilter 的子类。选用于过滤含 StopWords 类中定义的停用词的三元组。

核心方法：`public AbstractTermTuple next()`获得过滤后的下一个三元组。

**TermTupleScanner:**

TermTupleScanner 是抽象类 AbstractTermTupleScanner 的子类。读取所输入文档的每一行，并返回解析该行内容后得到的 Term 三元组。

核心方法：`public AbstractTermTuple next()`获得下一个三元组如果到了流的末尾，返回

null。

## query 模块

主要包含查询时用到的方法。

### Hit:

一个 Hit 表示搜索的单词在某一文件中的搜索结果。

核心方法: `public int compareTo(AbstractHit o)` 比较二个命中结果的大小, 根据 score 比较。

`public Hit(int docId, String docPath, Map<AbstractTerm, AbstractPosting> termPostingMapping)`构造函数。

### IndexSearcher:

用来存储对倒排索引的读取, 搜索方法。

核心方法: `public void open(String indexFile)` 从指定索引文件打开索引, 加载到 index 对象里. 一定要先打开索引, 才能执行 search 方法。

`public AbstractHit[] search(AbstractTerm queryTerm, Sort sorter)` 根据单个检索词进行搜索。

`public AbstractHit[] combineSearch(AbstractTerm queryTerm1, AbstractTerm queryTerm2, Sort sorter)` 根据输入的两个相邻的单词进行查询

`public AbstractHit[] search(AbstractTerm queryTerm1, AbstractTerm queryTerm2, Sort sorter, LogicalCombination combine)` 根据二个检索词进行搜索

Params:

queryTerm1 - : 第 1 个检索词 queryTerm2 - : 第 2 个检索词 sorter - : 排序器  
combine - : 多个检索词的逻辑组合方式

### SimpleSorter:

对搜索结果排序, 是 Sort 接口的具体实现。

核心方法: `public void sort(List<AbstractHit> hits)` 对命中结果集合根据文档得分排序

`public double score(AbstractHit hit)` 计算命中文档的得分, 作为命中结果排序的依据. 文档的得分值计算出来后要设置到 AbstractHit 子类对象里。

## run 模块

### TestBuildIndex:

用于进行索引构建。包括.txt 的载入, Document 和 Index 的构建, 以及倒排索引的保存, 可以参考概要设计中的流程图 2.1。

### TestSearchIndex:

是搜索程序入口。main 方法是一个恒为真的循环, 通过对输入查询的格式分析, 调用 query 模块中不同的查询方法。

## util 模块

### Config:

保存搜索引擎的配置信息，例如：

- 索引文件所在目录

- 要建立索引的文本文件所在目录

- 构建索引时是否忽略单词大小写

- 分词所需要的正则表达式

- 基于正则表达式的三元组过滤器所需的正则表达式

- 基于单词长度的三元组过滤器所需的最小单词长度和最大单词长度.

### **FileUtil:**

文件操作的工具类

核心方法： `public static String read(String filePath)` 一次读取指定文本文件的所有内容

`public static List<String> list(String dirPath, String suffix)` 列出指定目录下的匹配指定后缀名的所有文件的绝对路径(不递归)。

### **StopWords:**

停用词表类

### **StringSplitter:**

字符串分割类，根据标点符号和空白符将字符串分成一个个单词.

核心方法： `public List<String> splitByRegex(String input)` 将字符串分割成单词列表。

### 三、软件开发

简单介绍采用什么开发环境，如何编译、连接生成可执行文件。使用了什么调试工具。  
篇幅不要长。

开发环境：

java 版本

```
C:\Users\icyyoung\Desktop\JAVA\Experiment_file\Experiment1Test(JDK1)\Experiment1Test(JDK17)>java -version
openjdk version "17.0.7" 2023-04-18
OpenJDK Runtime Environment Temurin-17.0.7+7 (build 17.0.7+7)
OpenJDK 64-Bit Server VM Temurin-17.0.7+7 (build 17.0.7+7, mixed mode, sharing)
```

图 3.1

计算机软硬件配置：

操作系统: Windows 11 家庭中文版 64 位 (10.0, 版本 22631)  
语言: 中文(简体) (区域设置: 中文(简体))  
系统制造商: LENOVO  
系统型号: 82JW  
BIOS: HHCN29WW  
处理器: AMD Ryzen 7 5800H with Radeon Graphics (16 CPUs), ~3.2GHz  
内存: 16384MB RAM  
页面文件: 8483MB 已用, 10696MB 可用  
DirectX 版本: DirectX 12

图 3.2

IDE: IntelliJIDEA 版本信息：

IntelliJ IDEA 2023.1.6 (Community Edition)

Build #IC-231.9414.13, built on February 14, 2024

Runtime version: 17.0.10+10-b829.26 amd64

VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.

Windows 11.0

GC: G1 Young Generation, G1 Old Generation

Memory: 2030M

Cores: 16

Non-Bundled Plugins:

intellij-theme-relax-your-eyes-green (1.0.3)

com.alibaba.p3c.xenoamess (2.1.1.6x-SNAPSHOT)

PlantUML integration (6.5.0-IJ2022.2)

Kotlin: 231-1.8.21-IJ9414.13

如何编译生成可执行文件：利用 IntelliJIDEA 的 Ctrl+F9 的 Build Project 进行编译程序。

调试工具：IntelliJIDEA 的 Shift+F9 的调试功能。

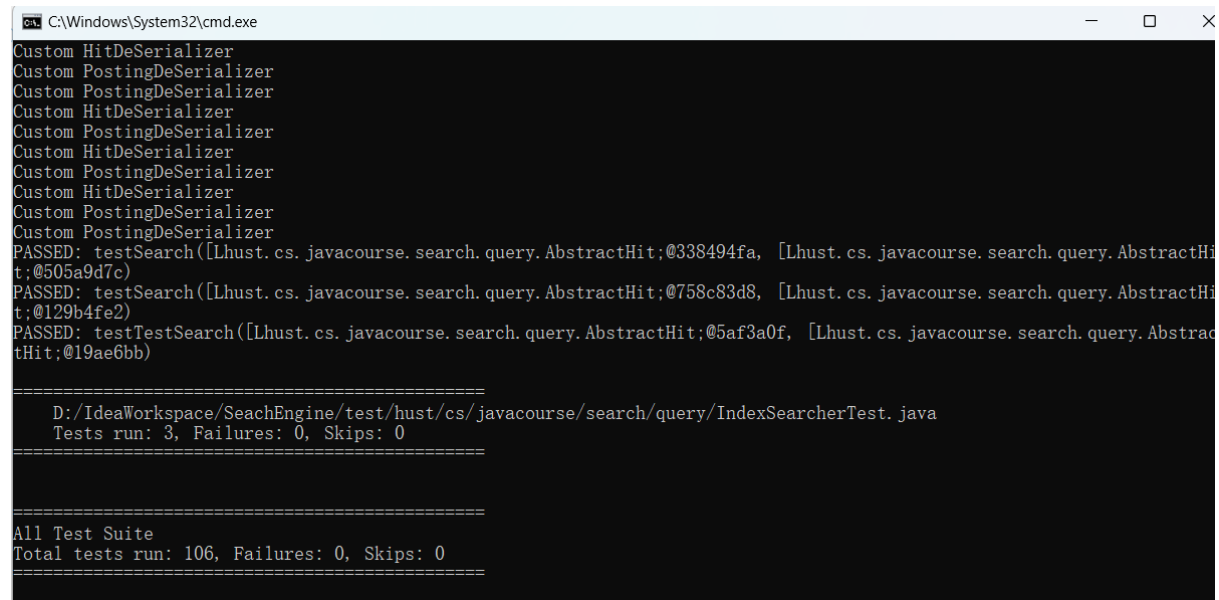
## 四、软件测试

对照题目要求，构造测试例，给出程序界面截图，举证题目要求的功能（以及自行补充的功能）已实现。

分析测试效果。

注意：已实现但未在报告中主动举证的功能可能被当作没有实现。

如下图为自动化测试的结果。106 个测试项目均通过，没有跳过项。



```
C:\Windows\System32\cmd.exe
Custom HitDeSerializer
Custom PostingDeSerializer
Custom PostingDeSerializer
Custom HitDeSerializer
Custom PostingDeSerializer
Custom HitDeSerializer
Custom PostingDeSerializer
Custom HitDeSerializer
Custom PostingDeSerializer
Custom PostingDeSerializer
Custom PostingDeSerializer
PASSED: testSearch([Lhust.cs.javacourse.search.query.AbstractHit;@338494fa, [Lhust.cs.javacourse.search.query.AbstractHit;@505a9d7c)
PASSED: testSearch([Lhust.cs.javacourse.search.query.AbstractHit;@758c83d8, [Lhust.cs.javacourse.search.query.AbstractHit;@129b4fe2)
PASSED: testSearch([Lhust.cs.javacourse.search.query.AbstractHit;@5af3a0f, [Lhust.cs.javacourse.search.query.AbstractHit;@19ae6bb)

=====
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/query/IndexSearcherTest.java
Tests run: 3, Failures: 0, Skips: 0
=====

All Test Suite
Total tests run: 106, Failures: 0, Skips: 0
=====
```

图 4.1

Tests for All Test Suite
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/TermTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/TermTupleTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/PostingTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/PostingListTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/DocumentTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/parse/TermTupleScannerTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/parse/StopWordTermTupleFilterTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/parse/PatternTermTupleFilterTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/parse/LengthTermTupleFilterTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/parse/ScannerFilterAllInOneTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/DocumentBuilderTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/IndexTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/index/IndexBuilderTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/query/HitTest.java (1 class)
D:/IdeaWorkspace/SeachEngine/test/hust/cs/javacourse/search/query/IndexSearcherTest.java (1 class)

图 4.2

测试构建倒排索引时出现下面错误：

PostingList:

```
java.io.FileNotFoundException Create breakpoint : C:\Users\icyyoung\Desktop\JAVA\Experiment_file\index\index.dat (系统找不到指定的路径。)  
at java.base/java.io.FileOutputStream.open0(Native Method)  
at java.base/java.io.FileOutputStream.open(FileOutputStream.java:293)  
at java.base/java.io.FileOutputStream.<init>(FileOutputStream.java:235)  
at java.base/java.io.FileOutputStream.<init>(FileOutputStream.java:184)  
at hust.cs.javacourse.search.index.impl.Index.save(Index.java:131)  
at hust.cs.javacourse.search.run.TestBuildIndex.main(TestBuildIndex.java:34)
```

图 4.3

分析原因：没有 index 文件夹，无法正常保存到其中。应该在项目文件的目录下，创建一个空的 index 文件夹，这样就可以正常保存倒排索引了。

测试构建倒排索引，程序正常运行，如下图所示。

```
D:\A_MODdevelop\JDK\bin\java.exe --enable-preview "-javaagent:D:\A_MODdevelop\IDEA\IntelliJ IDEA Community Edition 2023.1.3\lib\idea_rt.jar=62410:D:\A_MODdevelop\IDEA\Inte  
Start build index ...  
dictionary: [accord, according, action, activity, aged, agree, agreed, agreement, alleviate, announced, announcement, announcing, antarctic, antarctica, anthony, anymore,  
  
docPath mapping:  
0 => C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\1.txt  
1 => C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\10.txt  
2 => C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\11.txt  
3 => C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\12.txt  
4 => C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\13.txt  
5 => C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\14.txt
```

图 4.4

测试搜索一个单词：

```
input:  
according  
Document Number:4  
docId:0 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\1.txt score:3.0 times:2.0  
contentThe novel coronavirus death toll has reached 21 as of Saturday in Britain as the number of confirmed cases totalled 1,140, according to the latest figures released by th  
  
The new figures showed an increase of 342 confirmed COVID-19 cases in Britain, the largest rise on a single day since the start of the outbreak in the country. Ten more patient  
  
All the 10 patients who died were aged over 60 and had underlying health conditions, said Chris Whitty, chief medical officer for England.  
  
According to health authorities, most of the cases are in England. There have been 121 confirmed cases in Scotland, 60 in Wales and 34 in Northern Ireland.
```

图 4.5

```
input:  
people  
Document Number:5  
docId:4 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\13.txt score:3.0 times:2.0  
contentAs Valentine's Day approaches, many people with dogs will not care one bit whether they get a card.  
A survey by the Kennel Club has found almost one in five would rather have a dog than be in a relationship.  
Almost half said they were so attached to their pet because dogs don't let you down in the same way that other people might.  
Although a dog might not buy you roses or take you out for dinner, it seems unconditional love and a wagging tail can make being single a less lonely prospect.  
The research, which asked 2,612 dog owners for their thoughts on romance, found just under a quarter agreed that having a dog would make them feel less concerned about r
```

图 4.6

测试 AND 搜索：

```
according and people  
Document Number:1  
docId:0 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\1.txt score:4.5 times:3.0  
contentThe novel coronavirus death toll has reached 21 as of Saturday in Britain as the number of confirmed cases totalled 1,140, according to the latest figures released by  
  
The new figures showed an increase of 342 confirmed COVID-19 cases in Britain, the largest rise on a single day since the start of the outbreak in the country. Ten more patie  
  
All the 10 patients who died were aged over 60 and had underlying health conditions, said Chris Whitty, chief medical officer for England.  
  
According to health authorities, most of the cases are in England. There have been 121 confirmed cases in Scotland, 60 in Wales and 34 in Northern Ireland.
```

图 4.7

测试 OR 搜索：

```
D:\A_MODdevelop\JDK\bin\java.exe --enable-preview "-javaagent:D:\A_MODdevelop\IDEA\IntelliJ IDEA Community Edition 2023.1.3\lib\idea_rt.jar=62438:D:\A_MODdevelop\IDEA\IntelliJ
input:
according or people
Document Number:8
docId:8 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\1.txt score:4.5 times:3.0
contentThe novel coronavirus death toll has reached 21 as of Saturday in Britain as the number of confirmed cases totalled 1,140, according to the latest figures released by th

The new figures showed an increase of 342 confirmed COVID-19 cases in Britain, the largest rise on a single day since the start of the outbreak in the country. Ten more patient

All the 10 patients who died were aged over 60 and had underlying health conditions, said Chris Whitty, chief medical officer for England.

According to health authorities, most of the cases are in England. There have been 121 confirmed cases in Scotland, 60 in Wales and 34 in Northern Ireland.
```

图 4.8

## 测试词组搜索：

```
input:
coffee shops
Document Number:2
docId:7 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\16.txt score:4.5 times:3.0
contentSome coffee shops decorates so well, which attrated my eyes. Some coffee shops decorates so well, which attrated my eyes. Some coffee shops decorates so well, which attr
Term:coffee shops,Posting:Posting{docId:7, freq:3, positions: 1 11 21}
docId:9 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\3.txt score:1.5 times:1.0
contentPeople always say that we are lacking of the eyes of realizing the beauty in life. I can not agree with it anymore. Last week, I woke up very early in the morning, so I de
Term:coffee shops,Posting:Posting{docId:9, freq:1, positions: 68}
input:
```

图 4.9

可以注意到，according 搜索结果出现在 4 个文件中，people 为 5，according and people 为 1，according or people 为 8，满足  $1+8=4+5$ ，可进一步证明程序的正确性。



## 五、特点与不足

### 1. 技术特点

**方法复用性强：**在实现倒排索引的过程中，通过精心设计模块化的函数，使得单个单词的检索逻辑能够被有效复用于多单词的 OR/AND 复合检索场景中，这不仅体现了面向对象设计原则中的“重用性”原则，还大幅提升了代码的可维护性和扩展性。同样，对于词组检索，通过复用 AND 检索逻辑，减少了代码冗余，确保了逻辑的一致性。此外，利用对象自身的 readObject、writeObject 和 toString 等方法，进一步增强了代码的自解释性和一致性，减少了开发工作量，提高了开发效率。

**高效的循环机制：**在搜索功能的设计上，采用永真循环（如 while(true)）包裹主要的搜索逻辑，这种设计模式使得程序能够持续响应用户的查询请求，直到接收到明确的退出指令。这样的机制避免了因重复调用主函数导致的资源浪费，提高了程序的响应速度和用户体验。通过在循环内动态分析用户输入，实现了灵活的退出机制，确保了程序的健壮性。

### 2. 不足和改进的建议

**哈希映射优化：**当前系统在进行元素对比时，由于未对数据结构实现定制化的哈希函数，导致在进行元素查找时无法充分利用哈希表的高效特性。改进策略是为涉及频繁比较操作的自定义类实现 hashCode 和 equals 方法，确保哈希值能准确反映对象的逻辑等价性，从而大大提高集合操作的性能。

**并发编程：**针对大数据量下的索引构建耗时问题，引入多线程或并发编程框架将是显著提升性能的有效途径。通过并行处理文本数据，可以将任务拆分成多个子任务，实现并行处理，从而显著缩短索引构建时间。需要注意的是，在设计并发方案时，要合理控制线程数量，避免过多线程导致的上下文切换开销，并确保数据访问的线程安全性。

**序列化压缩机制：**序列化过程中加入数据压缩机制，或者在序列化前对数据进行特定算法的压缩处理，能够有效减少生成的.dat 文件体积，进而节省存储空间。这一步骤虽然会增加序列化和反序列化的计算成本，但在磁盘空间受限或需要传输大量数据的场景下，其带来的长期效益远远超过额外的计算开销。

## 六、过程和体会

### 1. 遇到的主要问题和解决方法

课程中，我遇到的第一个问题就是没有正确理解倒排索引整体结构框架，比如 Posting 和 PostingList 等数据结构如何存储倒排索引的信息，这对我后续的代码编写产生了极大的负面影响。经过反反复复阅读实验要求文档以及多次尝试之后，才正确的理解项目框架。

第二个问题就是由于对 java 和序列化了解的不够充分，对于接口 `FileSerializable` 的方法 `readObject` 和 `writeObject` 方法的实现不够熟练。因此我参考了有关教材和网络上的博客讲解并仔细阅读了 `ObjectInputStream` 和 `ObjectOutputStream` 的文档，了解如何使用它们来读写序列化对象。

在自动化测试的过程中，有的方法是由其他方法的测试代码调用的，因此会出现修改一个 bug，出现更多方法不通过测试的情况，非常考验我们的思维严密性。

正如上面四种的自动化测试遇到的问题，首次构建倒排索引时始终无法正常生成相应的 .dat 文件，通过分析错误信息得知是由于使用了 `INDEX_DIR` 常量，而这个常量又依赖于项目的根目录，需要在项目的根目录下创建 index 文件夹

### 2. 课程设计的体会

构建一个倒排索引的过程是一段既充满挑战又极具启发性的旅程，它不仅深化了我对信息检索底层机制的理解，也锻炼了我的编程实践能力和问题解决技巧。

理论知识是构建倒排索引的基础，但只有通过亲手编码实现，才能真正理解运用在理论课程中我们学习到的各种知识，如序列化、内部类、`lambda` 表达式等内容。这些内容在平时的练手测验中并不多见，但却是 java 的重要组成部分。

在构建倒排索引时，我深入体验到了 Java 面向对象编程的精髓。通过类的定义、继承、封装和多态等特性，能够构建出清晰、可维护的代码结构。特别是虚基类和接口的使用，不仅简化了代码设计，还提高了代码的复用性和可扩展性。同时，工具类和配置类的应用也让我意识到代码组织的重要性，加强了对 java 类这一核心概念的认知。类之间相互 `import`，有一些工具类专门存放项目的常量，还有一些类可以存放项目的配置信息等。

虽然实验中构建的倒排索引是简化版的，但它已经涵盖了搜索引擎技术的核心思想。通过这个过程，我对搜索引擎的工作原理有了初步的认识，包括文档的预处理、索引的构建、查询的处理和结果的排序等。这对于未来进一步学习和研究搜索引擎技术打下了坚实的基础。

## 七、源码和说明

### 1. 文件清单及其功能说明

根目录 Experiment\_File 下的文件如下：










 .idea	2024/5/27 21:06	文件夹	
 experiment_javadoc	2024/5/20 23:07	文件夹	
 Experiment1Test(JDK1	2024/4/10 14:03	文件夹	
 index	2024/5/27 21:04	文件夹	
 out	2024/5/27 20:58	文件夹	
 SearchEngineForStudent	2024/5/27 21:03	文件夹	
 text	2024/5/20 23:12	文件夹	
 CS2206-闫晓任-U202215543.doc	2020/2/25 3:05	Microsoft Word ...	148 KB
 Experiment_file.iml	2024/5/27 20:55	IML 文件	1 KB

图 5

.idea 及 Experiment\_file.iml：IntelliJIDEA 生成的配置文件信息，可忽略。

experiment\_javadoc：实验项目的 javadoc，其中有 index.html，可打开查看项目的 javadoc 信息。

Experiment1Test(JDK1：自动化测试文件 Experiment1Test(JDK1\Experiment1Test(JDK17) 下可以用 cmd 运行 test.bat，进行自动化测试（需要配置 test.bat 下的 JAVA\_HOME 为本机的 JAVA\_HOME）。

index：生成的倒排索引文件。

out：编译项目产生的.class 字节码文件。

SearchEngineForStudent：项目源码。

text：构建倒排索引用到的文本，可以按照需求加入文本。

CS2206-闫晓任-U202215543：本报告。

### 2. 用户使用说明书

1. 用 IntelliJIDEA 打开文件的根目录 Experiment\_File，作为项目的根目录。
2. 为项目配置 JDK 环境，如下图所示。

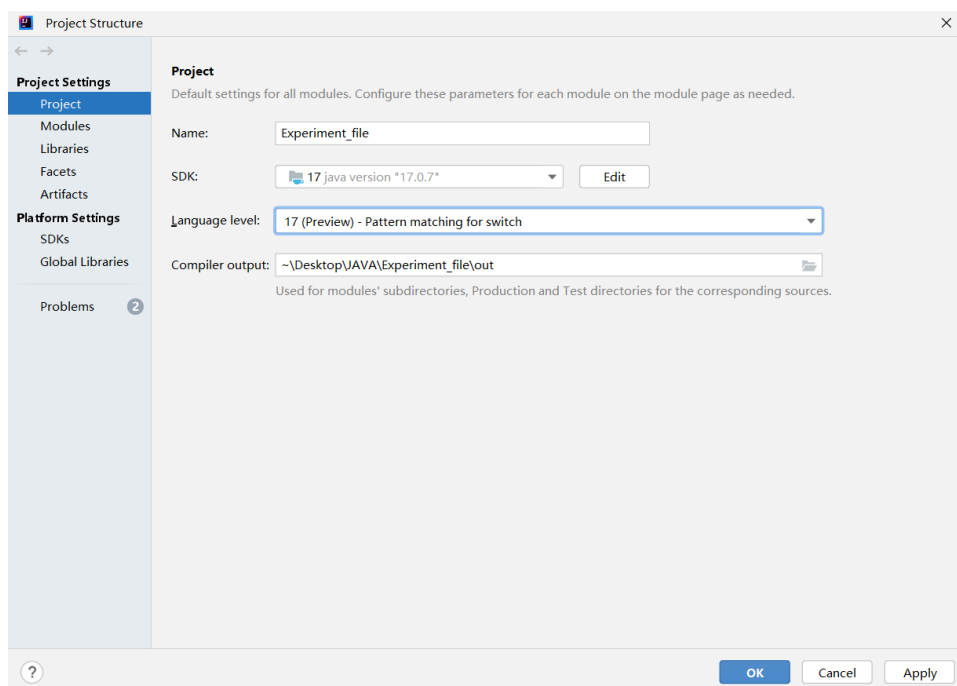


图 7.2

3. 运行 package hust.cs.javacourse.search.run 下的 TestBuildIndex 类，构建倒排索引。
4. 运行 package hust.cs.javacourse.search.run 下的 TestSearchIndex 类，进行搜索。搜索方式如下：

单个单词搜索：直接在输入中输入搜索的单词，如下图所示，包含文档数目、文档得分 score，并根据文档得分对结果集排序，以及文档内容、文档路径、命中次数 times。

```
D:\A_MODdevelop\JDK\bin\java.exe --enable-preview "-javaagent:D:\A_MODdevelop\IDEA\IntelliJ IDEA Community Edition 2023.1.3\lib\idea_rt.jar=50889:D:\A_MODdevelop\IDEA\IntelliJ I
input:
home
Document Number:2
docId:1 docPath:C:\Users\icvyyoung\Desktop\JAVA\Experiment_file\text\10.txt score:1.5 times:1.0
contentThe ongoing bushfires that have devastated Australia closed in on the country's capital, Canberra, last Friday as it declared its first state of emergency since 2003.
The Rural Fire Service of New South Wales, the state surrounding the capital, warned of severe fire danger for the Australian Capital Territory, with flames and embers projected
Term:home,Posting:Posting{docId:1, freq:1, positions: 65}
docId:13 docPath:C:\Users\icvyyoung\Desktop\JAVA\Experiment_file\text\7.txt score:1.5 times:1.0
contentClimate experts stationed at an Argentine research base in a northern part of Antarctica have just logged the highest temperature on record for the typically frozen conti
According to preliminary data from the U.N.'s World Meteorological Organization, temperatures on the Antarctic Peninsula just hit 64.9 degrees Fahrenheit, or 18.3 degrees Celsius
```

图 7.3

AND/OR 搜索：在输入中输入搜索的单词，中间用空格和 and/And/Or（不区分大小写）等分隔开，如下为搜索”according and people”的结果。

```
according and people
Document Number:1
docId:0 docPath:C:\Users\icvyyoung\Desktop\JAVA\Experiment_file\text\1.txt score:4.5 times:3.0
contentThe novel coronavirus death toll has reached 21 as of Saturday in Britain as the number of confirmed cases totalled 1,140, according to the latest figures released by
The new figures showed an increase of 342 confirmed COVID-19 cases in Britain, the largest rise on a single day since the start of the outbreak in the country. Ten more patie
All the 10 patients who died were aged over 60 and had underlying health conditions, said Chris Whitty, chief medical officer for England.
According to health authorities, most of the cases are in England. There have been 121 confirmed cases in Scotland, 60 in Wales and 34 in Northern Ireland.
```

图 7.4

两个单词构成的短语搜索：输入短语，中间用空格分隔，如下图所示为搜索”coffee shops”的结果。

```
input:
coffee shops
Document Number:2
docId:7 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\16.txt score:4.5 times:3.0
contentSome coffee shops decorates so well, which attrated my eyes. Some coffee shops decorates so well, which attrated my eyes. Some coffee shops decorates so well, which attr
Term:coffee shops,Posting:Posting{docId:7, freq:3, positions: 1 11 21}
docId:9 docPath:C:\Users\icyyoung\Desktop\JAVA\Experiment_file\text\3.txt score:1.5 times:1.0
contentPeople always say that we are lacking of the eyes of realizing the beauty in life. I can not agree with it anymore. Last week, I woke up very early in the morning, so I de
Term:coffee shops,Posting:Posting{docId:9, freq:1, positions: 68}
input:
```

图 7.5

退出程序：输入”q”或”quit”即可退出程序，如下图。

```
input:
```

```
q
```

```
Process finished with exit code 0
```

图 7.6

### 3. 源代码

源代码位于根目录\SearchEngineForStudent\src\hust\cs\javacourse\search 之中。