

Java 程序设计课程报告

2021-2022学年 秋冬学期

题目：用药助手信息检索系统

姓名：康雅琪

学院：计算机科学与技术学院

专业：计算机科学与技术

学号：3190106213

指导教师：鲁伟明

2021年11月30日

一、引言

本次作业完成的是一个用药助手信息检索系统，使用各种开源包，实现从丁香园用药助手网站 (<https://d rugs.dxy.cn/>) 爬取数据并储存到文件，再从文件获取信息创建索引，使用户可以在本地通过命令行使用查询关键词对丁香园用药助手的药品信息进行搜索。

1.1 设计目的

- 写一个Web爬虫，爬取网站的网页
- 解析网页内容，对内容进行结构化，并存储到文件中
- 为内容建立索引
- 通过命令行进行信息检索，并展示内容列表

1.2 设计说明

本次作业使用 Java 程序设计语言，在 Eclipse 下实现，引入并使用了 HttpClient 客户端编程工具包、Jsoup 解析器、Lucene 检索工具包、Fastjson 库。

- HttpClient 用于建立与网站建立连接获取数据
- Jsoup 用于实现对网站信息的解析
- Fastjson 用于获取 JSON 对象以便于获取网页的 url 信息
- Lucene 用于实现了索引的建立及信息检索

二、总体设计

2.1 功能模块设计

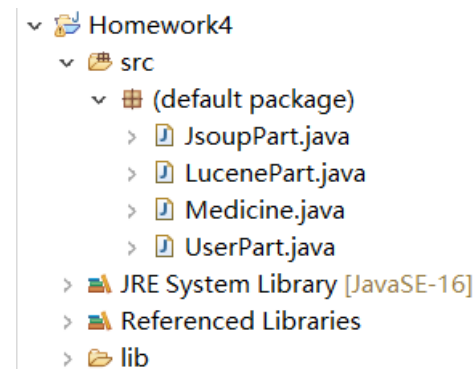
本次作业的工程 Homework4 的工程目录如下

`JsoupPart.java` 用于实现网站信息获取及解析，并将解析后的数据保存在文件中

`LucenePart.java` 用于实现根据文件中储存的信息建立索引

`Medicine.java` 用于根据药品信息建立 Document 以便建立索引

`UserPart` 为用户运行程序，可以在此进行信息检索。



- 网站信息获取

仅需运行 `JsoupPart.java` 中的 `main` 函数即可得到网站数据并保存在 .txt 文件中。

- 建立索引

仅需运行 `LucenePart.java` 中的 `main` 函数即可根据 .txt 文件数据创建索引。

- 用户运行

仅需运行 `User.java` 中的 `main` 函数即可进入用户程序，进行信息检索。

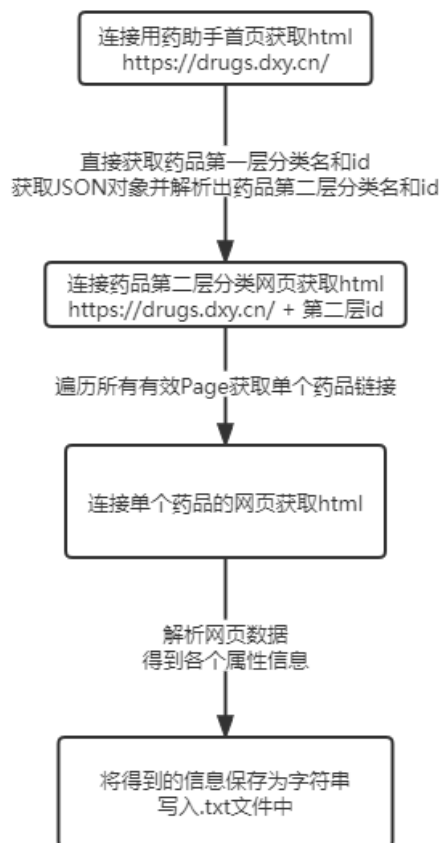
2.2 药品信息数据格式说明

`Medicine` 类包含的成员变量如下，均为 `String` 类型，储存每个药品的分类、网页地址、属性信息。

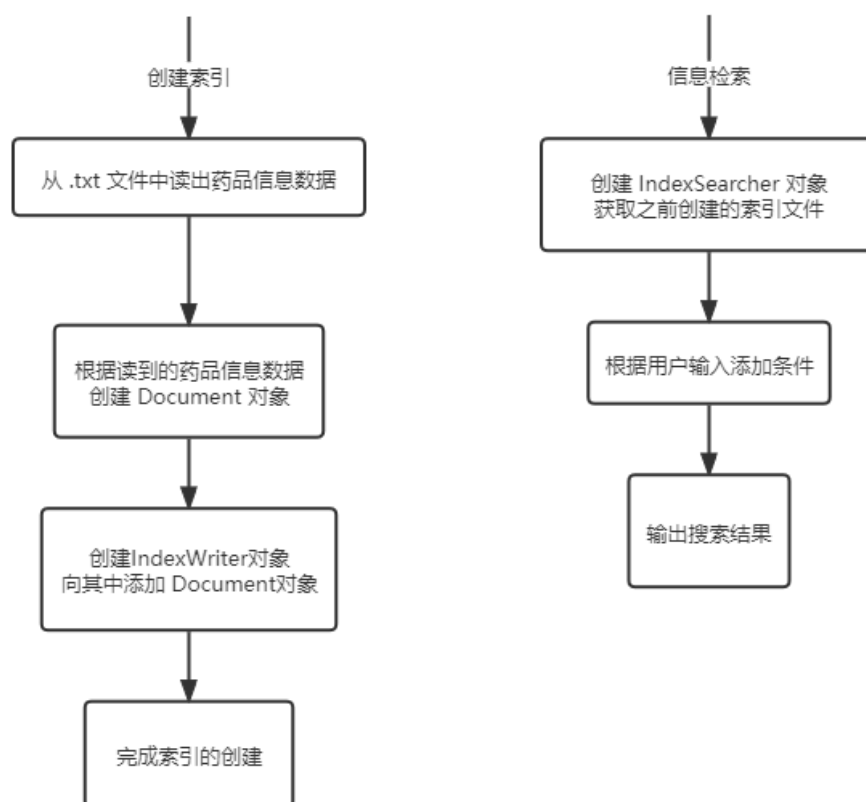
```
// 名称
String ChineseName;
String EnglishName;
String DrugName;
// 成份
String Component;
// 适应症
String Indication;
// 用法用量
String Usage;
// 注意事项
String Precautions;
// 禁忌
String Contraindication;
// 孕妇
String Gravida;
// 药理作用
String PharmacologicalAction;
// 药代动力学
String Pharmacokinetics;
// 化学成分
String ChemicalComposition;
// 是否OTC
String OTC;
// 具体网页地址
String Address;
// 打印信息
String PrintMedicine;
// 药品类别
String CategoryName;
```

2.3 流程图设计

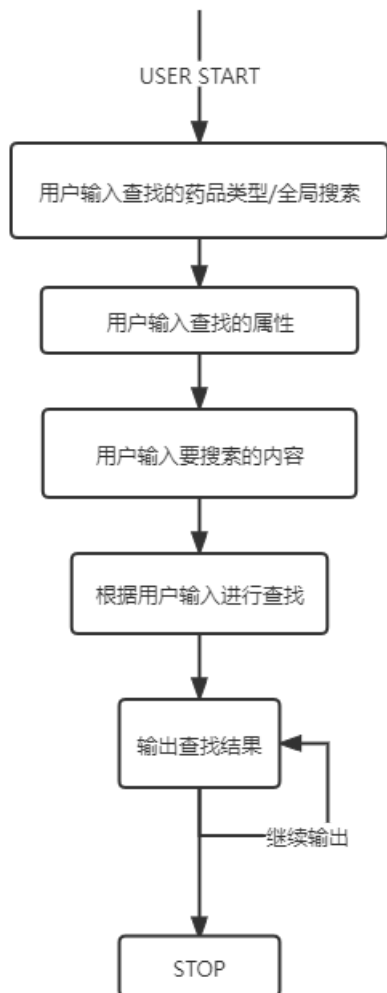
- JsoupPart



- LucenePart



- UserPart



三、详细设计

3.1 获取网站数据具体方法

- 使用 HttpClient 请求获取网页 html

```
// 获取 HttpGet 对象
CloseableHttpClient client = HttpClients.createDefault();
HttpGet get = new HttpGet("https://drugs.dxy.cn/");
// 执行 get 请求
CloseableHttpResponse response = client.execute(get);
String content = "";
// 判断响应状态是否正常
if(response.getStatusLine().getStatusCode() == 200){
    // 获取响应实体
    content = EntityUtils.toString(response.getEntity(), "UTF-8");
}
```

- 获取获取到的网页实体的 Document 对象

```
// 将前面获取到的 html 文本转换为 Document 对象
Document doc = Jsoup.parse(content);
```

- 获取 JSON 对象并逐层解析，得到下一跳的网页地址

```
// 由网页实体的字符串内容获取 JSONObject
JSONObject json = JSON.parseObject(tmpjson);
// 分层获取信息，直到得到 Category List
JSONObject props = json.getJSONObject("props");
JSONObject pageProps = props.getJSONObject("pageProps");
// 首页左边的 List (即药品分类信息)
JSONArray firstLevelCategoryList =
pageProps.getJSONArray("firstLevelCategoryList");
```

- 遍历访问和获取药品的网页信息并保存到 .txt 文件中，默认将文件保存在 D 盘，按药品的一级分类储存，共有 14 个 .txt 文件

非性激素和胰岛素类的激素类系统用药.txt	2021/12/8 1:04	文本文档	516 KB
感觉器官.txt	2021/12/8 1:31	文本文档	311 KB
呼吸系统.txt	2021/12/8 1:30	文本文档	741 KB
肌肉-骨骼系统.txt	2021/12/8 1:16	文本文档	859 KB
抗寄生虫药、杀虫药和驱虫药.txt	2021/12/8 1:25	文本文档	319 KB
抗肿瘤药和免疫机能调节药.txt	2021/12/8 1:12	文本文档	661 KB
皮肤病用药.txt	2021/12/8 1:00	文本文档	1,306 KB
神经系统.txt	2021/12/8 1:22	文本文档	1,065 KB
生殖泌尿系统和性激素.txt	2021/12/8 1:02	文本文档	581 KB
系统用抗感染药.txt	2021/12/8 1:08	文本文档	959 KB
消化道及代谢.txt	2021/12/8 0:47	文本文档	1,882 KB
心血管系统.txt	2021/12/8 0:55	文本文档	1,453 KB
血液和造血器官.txt	2021/12/8 0:50	文本文档	689 KB
杂类.txt	2021/12/8 1:37	文本文档	913 KB

3.2 创建索引及信息检索方式

- 创建索引
 - 调用 LucenePart 的成员方法 `createIndexOuter()`，`createIndexOuter()` 将会调用 LucenePart 的成员方法 `createIndex()` 完成对索引的创建

```
public void createIndexOuter();
public void createIndex(String indexPath, String filePath, String
CategoryName);
```

- 创建 IndexWriter 对象

```
Directory directory = FSDirectory.open(indexfile);
Analyzer analyzer = new IKAnalyzer();
IndexWriterConfig indexConfig = new
IndexWriterConfig(Version.LUCENE_4_10_0, analyzer);
// 创建 IndexWriter
indexwriter = new IndexWriter(directory, indexConfig);
```

- 从文件中读取数据，当读到14个完整属性使创建 Medicine 对象并由此获得 Document 对象，将获得到的 Document 对象添加到 IndexWriter 对象中

```

while((whileTmpString = bufferReader.readLine()) != null) {
    tmpString[tmpStringIndex] = whileTmpString;
    //system.out.println(" ----- " + CategoryName + " ----- " +
tmpStringIndex + " ----- ");
    if(tmpStringIndex == 13) {
        Medicine tmpMedicine = new
Medicine(tmpString[0],tmpString[1],tmpString[2],tmpString[3],tmpString[4
],tmpString[5],tmpString[6],tmpString[7],tmpString[8],tmpString[9],tmpSt
ring[10],tmpString[11],tmpString[12],tmpString[13],file.getName().replac
e(".txt", ""));
        Document tmpDocument = tmpMedicine.toDocument(CategoryName);
        //添加 document, Lucene 的检索以 document 为基本单位
        indexWriter.addDocument(tmpDocument);
        tmpStringIndex = 0;
    }else {
        tmpStringIndex++;
    }
}
}

```

- 信息检索方法

- 调用 LucenePart 的成员方法 `searchIndexOuter()` , `searchIndexOuter()` 将会调用 LucenePart 的成员方法 `searchIndex()` 完成信息检索操作

```

public void searchIndexOuter(Integer SearchCategory, Integer SearchKey,
String SearchInput);
public void searchIndex(String indexPath, Integer CategoryKey, Integer
SearchKey, String SearchInput);

```

- 创建 IndexSearcher 对象

```

IndexSearcher searcher = new
IndexSearcher(DirectoryReader.open(FSDirectory.open(file)));

```

- 创建 BooleanQuery 对象并根据用户输入为其添加条件

```

// 添加条件
Term term1 = new Term("category",CategoryKeyString);
Term term2 = new Term(SearchKeyString, SearchInput);
WildcardQuery wildcardQuery1 = new WildcardQuery(term1);
WildcardQuery wildcardQuery2 = new WildcardQuery(term2);
BooleanQuery booleanQuery = new BooleanQuery();
if(CategoryKey < 13) {
    booleanQuery.add(wildcardQuery1, Occur.MUST);
}
booleanQuery.add(wildcardQuery2, Occur.MUST);

```

- 使用 IndexSearcher 对象及 BooleanQuery 对象进行搜索

```

// 1000 表示找到 1000 个匹配项
TopDocs topDocs = searcher.search(booleanQuery,null,1000);

```

- 使用 for 循环输出搜索结果, 每输出 10 个向用户确认要选择继续输出还是退出

```
// 输出搜索结果，每输出10个结果向用户确认是否需要继续输出
for(int i=0;i<docSize;i++) {
    if((i % 10 == 0)&&(i != 0)) {
        System.out.println("输入 “continue” 继续输出下面十个搜索结果");
        System.out.println("输入 “stop” 停止输出并结束此次搜索");
        Scanner sn = new Scanner(System.in);
        String GetInstruction = sn.nextLine();
        GetInstruction = GetInstruction.replaceAll("\n", "");
        while(
(GetInstruction.equals("continue")||GetInstruction.equals("stop"))) {
            System.out.println("输入 “continue” 继续输出下面十个搜索结果");
            System.out.println("输入 “stop” 停止输出并结束此次搜索");
            GetInstruction = sn.nextLine();
            GetInstruction = GetInstruction.replaceAll("\n", "");
        }
        sn.close();
        if(GetInstruction.equals("continue")) {
            ScoreDoc scoreDoc = topDocs.scoreDocs[i];
            int docIndex = scoreDoc.doc;
            Document document = searcher.doc(docIndex);
            System.out.println(document.get("print"));
        }else {
            break;
        }
    }else {
        ScoreDoc scoreDoc = topDocs.scoreDocs[i];
        int docIndex = scoreDoc.doc;
        Document document = searcher.doc(docIndex);
        System.out.println(document.get("print"));
    }
}
}
```

3.3 用户查找信息使用方法

- 打印 Welcome 信息及是否需要指定药品类别的提示，请用户输入代表对应药品类别的数字

```
System.out.println("  . . . . .");
System.out.println(" . . . . .");
System.out.println(" . . . . .");
System.out.println(" . . . . .");
System.out.println(" . . . . .");
System.out.println("welcome to kk的丁香园用药助手的药品搬运作业");
System.out.println("现在你可以选择是否要指定药品的类别,也可以不指定");
System.out.println("如果你想要指定药品的类别,请按照下文指示输入合法的数字");
System.out.println("0 - 非性激素和胰岛素的激素类系统用药");
System.out.println("1 - 感觉器官用药");
System.out.println("2 - 呼吸器官用药");
System.out.println("3 - 肌肉-骨骼系统用药");
System.out.println("4 - 抗寄生虫、杀虫药和驱虫药");
System.out.println("5 - 抗肿瘤药和免疫机能调节药");
System.out.println("6 - 皮肤病用药");
System.out.println("7 - 神经系统用药");
System.out.println("8 - 生殖泌尿系统和性激素用药");
System.out.println("9 - 系统用抗感染药");
System.out.println("10 - 消化道及代谢用药");
System.out.println("11 - 心血管系统用药");
System.out.println("12 - 血液和造血器官用药");
```



```
System.out.println("13 - 不指定药品类别,进行全局搜索");
System.out.println("14 - 退出程序");
```

- 检查用户输入的数字是否合法及用户是否想要退出程序

```
Integer CategoryType = Integer.parseInt(sn.nextLine());
while(CategoryType < 0 || CategoryType > 14) {
    System.out.println("!!请输入合法的数字(在 0 ~ 14 之间)");
    CategoryType = Integer.parseInt(sn.nextLine());
}
if(CategoryType.equals(14)) {
    break;
}
```

- 打印选择搜索内容的提示, 用户可选择搜索药品名称/成份/适应症

```
System.out.println("现在你可以选择用药品名称/成份/适应症的关键词来搜索药品,请按照下文指示输入合法的数字:");
System.out.println("0 - 药品名称");
System.out.println("1 - 成份");
System.out.println("2 - 适应症");
System.out.println("3 - 退出程序");
```

- 检查用户输入的数字是否合法及用户是否想要退出程序

```
Integer SearchType = Integer.parseInt(sn.nextLine());
while(SearchType < 0 || SearchType > 3) {
    System.out.println("!!请输入合法的数字(在 0 ~ 3 之间)");
    SearchType = Integer.parseInt(sn.nextLine());
}
if(SearchType.equals(3)) {
    break;
}
```

- 用户输入搜索内容, 调用 LucenePart 的成员函数进行信息检索, 完成后输出结果

```
System.out.println("现在你可以输入想要搜索的关键词:");
System.out.println("同时你也可以输入 “886” 以退出程序");
String SearchKey = sn.nextLine();
SearchKey = SearchKey.replaceAll("\n", "");

if(SearchKey.equals("886")) {
    break;
}

LucenePart lucenePart = new LucenePart();
lucenePart.searchIndexOuter(CategoryType, SearchType, SearchKey);

System.out.println("输入回车以继续");
sn.nextLine();
```

- 循环以上步骤, 直到用户选择结束程序

```
System.out.println("----- Byebye -----");
```

四、测试与运行

4.1 程序测试

经测试程序的各个部分均可正常运行，其中向网站获取数据的部分，最终共能够得到 8386 个药品信息，该过程大约需要花费一个半小时，程序的测试时间为 5669 秒，约为一个半小时。

索引的建立和用户进行信息检索的部分也均可正常工作。


























- 数据获取部分

获得的 .txt 文件内容正确，每 14 行为一个药品信息。

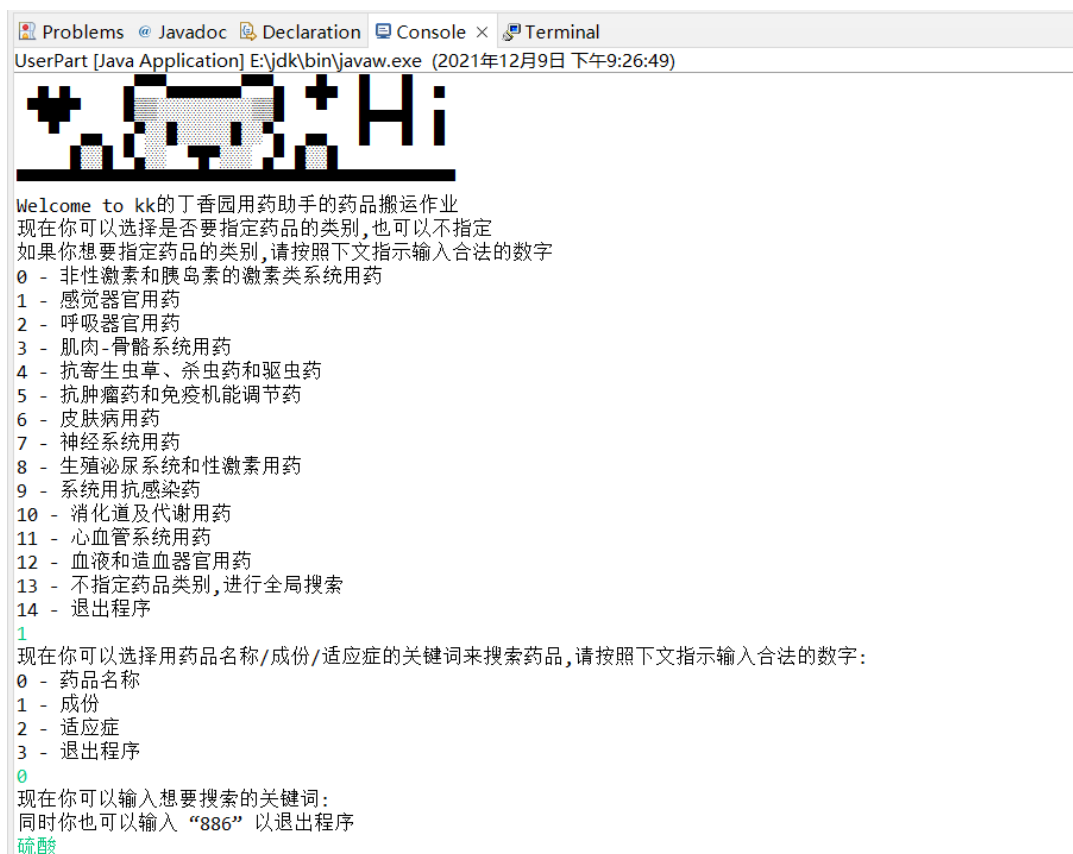


- 建立索引部分

建立索引、得到索引文件

名称	修改日期	类型	大小
 _a.fdt	2021/12/9 21:26	FDT 文件	4,535 KB
 _a.fdx	2021/12/9 21:26	FDX 文件	3 KB
 _a.fnm	2021/12/9 21:26	FNM 文件	1 KB
 _a.nvd	2021/12/9 21:26	NVD 文件	12 KB
 _a.nvm	2021/12/9 21:26	NVM 文件	1 KB
 _a.si	2021/12/9 21:26	SI 文件	1 KB
 _a_Lucene41_0.doc	2021/12/9 21:26	Microsoft Word ...	1,747 KB
 _a_Lucene41_0.pos	2021/12/9 21:26	POS 文件	2,660 KB
 _a_Lucene41_0.tim	2021/12/9 21:26	TIM 文件	600 KB
 _a_Lucene41_0.tip	2021/12/9 21:26	TIP 文件	12 KB
 _b.cfe	2021/12/9 21:26	CFE 文件	1 KB
 _b.cfs	2021/12/9 21:26	CFS 文件	2,549 KB
 _b.si	2021/12/9 21:26	SI 文件	1 KB
 _c.cfe	2021/12/9 21:26	CFE 文件	1 KB
 _c.cfs	2021/12/9 21:26	CFS 文件	1,980 KB
 _c.si	2021/12/9 21:26	SI 文件	1 KB
 _d.cfe	2021/12/9 21:26	CFE 文件	1 KB
 _d.cfs	2021/12/9 21:26	CFS 文件	1,006 KB
 _d.si	2021/12/9 21:26	SI 文件	1 KB
 _e.cfe	2021/12/9 21:26	CFE 文件	1 KB
 _e.cfs	2021/12/9 21:26	CFS 文件	1,177 KB
 _e.si	2021/12/9 21:26	SI 文件	1 KB
 segments.gen	2021/12/9 21:26	GEN 文件	1 KB
 segments_e	2021/12/9 21:26	文件	1 KB
 write.lock	2021/12/9 21:26	LOCK 文件	0 KB

- 输入：**



输出（仅截取部分）：

```
-----
【药品类别】
感觉器官
【药品名称】
通用名称： 注射用硫酸阿米卡星
英文名称： Amikacin Sulfate for Injection
商品名称： 注射用硫酸阿米卡星
【成份】
本品的主要成份为硫酸阿米卡星。化学名称为 0-3-氨基-3-脱氧-a-D-葡吡喃糖基-(1+6)-0-[（6-氨基-6-脱氧-a-D-葡吡喃糖基-(1+4)）]-N-(4-氨基-2-羟基-1-氧丁基)-
... 登录
【适应症】
本品适用于铜绿假单胞菌及其他假单胞菌、大肠埃希菌、变形杆菌属、克雷伯菌属、肠杆菌属、沙雷菌属、不动杆菌属等敏感革兰阴性杆菌与葡萄球菌属(甲氧西林敏感株)所致严重感染，
... 登录
【用法用量】
成人，肌肉注射或静脉滴注。单纯性尿路感染对常用抗菌药耐药者每 12 小时 0.2 g；用于其他全身感染每 12 小时 7.5 mg/kg，或每 24 小时 15 mg/kg。成人一日不超过 1.5 g
... 登录
【禁忌】
由于非肠道给予氨基糖苷类药物具有耳毒性和肾毒性，且治疗期超过 14 天的安全性尚未确定，所以对于接受此类药物治疗的患者要进行密切临床观察。 神经毒性 氨基糖苷类药物的神经
... 登录
【注意事项】
对阿米卡星或其他氨基糖苷类过敏的患者禁用。
【孕妇及哺乳期妇女用药】
本品属孕妇用药的 D 类，即对人类有危害，但用药后可能利大于弊。本品可穿过胎盘到达胎儿组织，可能引起胎儿听力损害。妊娠妇女使用本品前必须充分权衡利弊。哺乳期妇女用药时宜
【药理作用】
硫酸阿米卡星是一种氨基糖苷类抗生素。本品对多数肠杆菌科细菌，如大肠埃希菌、克雷伯菌属、肠杆菌属、变形杆菌属、志贺菌属、沙门菌属、枸橼酸杆菌属、沙雷菌属等均具良好作用
... 登录
【药代动力学】
肌/内注射后迅速被吸收。主要分布于细胞外液，部分药物可分布到各种组织，并可在肾脏皮质细胞和内耳液中积蓄；但在心脏心耳组织、心包液、肌肉、脂肪和间质液内的浓度很低。支气
... 登录
【化学成份】
C22H43N5O13·nH2SO4(n = 1.8)
【是否OTC】
否
【丁香园用药助手链接】
https://drugs.dxy.cn/drug/PI45QqDrYV8Hb527sHS8dg==
-----

【药品类别】
感觉器官
【药品名称】
通用名称： 硫酸庆大霉素滴眼液
英文名称： Gentamycin Sulfate Eye Drops
商品名称： 硫酸庆大霉素滴眼液
【成份】
本品主要成份为：硫酸庆大霉素，为一种多组分抗生素，含 C1、C1a、C2a、C2 等组分。
【适应症】
用于结膜炎、眼睑炎、睑板腺炎。
【用法用量】
滴眼。将本品滴入眼睑内，一次 1-2 滴，一日 3-5 次。
【禁忌】
滴眼时请勿使管口接触手和眼睛。本品不宜长期连续使用，使用 3-4 日症状未缓解时，应停药就医。若出现充血、眼痒、水肿等症状，应停药就医。使用后请拧紧瓶盖，以防污染。对本
... 登录
【注意事项】
对本品或其他氨基糖苷类抗生素过敏者禁用。
【孕妇及哺乳期妇女用药】
本品滴眼后虽极少吸收进入全身血液循环，但孕妇及哺乳期妇女仍应注意不可过量或长期使用，以免影响胎儿及婴儿的生长发育。
【药理作用】
本品为 氨基糖苷类广谱抗^生素，其作用机制主要是抑制细菌合成蛋白质。对眼部常见革兰阴性菌有抗菌作用。
【药代动力学】
本/品滴 眼后极少吸收进入^眼内组织或进入全身血液循环。
【化学成份】
暂无信息
【是否OTC】
甲类
【丁香园用药助手链接】
https://drugs.dxy.cn/drug/JCGpqCK8TLwte1qmepepmKgV5mQ==
-----

感觉器官
【药品名称】
通用名称： 注射用小诺霉素
英文名称： Micronomicin Sulfate For Injection
商品名称： 洛意
【成份】
小诺霉素
【适应症】
主要用于革兰阴性菌(如大肠杆菌、痢疾杆菌、变形杆菌、克雷白氏肺炎杆菌、绿脓杆菌等)感染引起的败血症、支气管炎、肺炎、腹膜炎、肾盂肾炎、膀胱炎等,对革兰阳性菌(如葡萄球菌
【用法用量】
本品用氯化钠注射液 1~2 ml 溶解后,肌肉注射或静脉滴注。肌肉注射:一次 6 万单位,必要时可用至 12 万单位,一日 2~3 次;静脉滴注:一次 6 万单位,加于 0.9% 氯化钠溶液 100
... 登录
【禁忌】
用药时间一般不宜超过 14 天,若必须继续用药时,应对听觉器官和肾功能进行严密监护。 本品稀释后可静脉滴注,不能作静脉推注用。 肝功能异常者慎用。
【注意事项】
对卡那霉素、链霉素、庆大霉素、妥布拉霉素、丁胺卡那霉素等氨基糖甙类抗生素及杆菌肽过敏史者禁用。 肾功能障碍者禁用。
【孕妇及哺乳期妇女用药】
目前尚不明确,应慎用。
【药理作用】
抗菌/作用:本品与其^它氨基糖甙类抗生素相同,阻碍蛋白质合成,为杀菌性抗生素,对革兰阴性菌和革兰阳性^菌显示广谱抗菌活性,抗菌作用较强、最低抑菌浓度为 0.1~6.25 毫微克/毫升
... 登录
【药代动力学】
肾功能正常的成人,肌^肉注射本品 60~120 mg。血药浓度达峰时间为 0.7&#177;0.17 小时,半衰期为 2.63&#177;0.98 小时,肌肉注射吸收良好,肾功能正常成^人肌肉注射本
... 登录
【化学成份】
化学名称为: 0-2-Amino-2,3,4,6-tetradecoxy-6-(methylamino)-a-D-erythro-hexopyranosyl-(1+4)-0-[3-deoxy-4-C-me
... 登录
【是否OTC】
否
【丁香园用药助手链接】
https://drugs.dxy.cn/drug/ZUjYdrth9PCifUrbQImFVA==
-----

输入 “continue” 继续输出下面十个搜索结果
输入 “stop” 停止输出并结束此次搜索
```

输入 continue 继续输出：

输入“continue”继续输出下面十个搜索结果

输入“stop”停止输出并结束此次搜索

continue

【药品类别】

感觉器官

【药品名称】

通用名称： 硫酸奈替米星氯化钠注射液

英文名称： Netilmicin Sulfate and Sodium Chloride Injection

商品名称： 君欣

【成份】

本品主要成份是硫酸奈替米星。辅料为氯化钠、亚硫酸钠、依地酸二钠。

【适应症】

本品适用于敏感细菌所引起的包括新生儿、婴儿、儿童等各年龄患者在内的严重或危及生命的细菌感染性疾病的短期治疗。这些感染性疾病包括：复杂性尿路感染；由埃希氏大肠杆菌、肺炎...

登录

【用法用量】

为了正确计算剂量，应在给药前获知病人的体重，对于肥胖病人的剂量应按瘦人的体重计算。肾功能状态可根据血清肌酐浓度的测定而估计，或根据内在的肌酐清除率而计算。另外，治疗...

登录

【禁忌】

为避免或减少耳、肾毒性反应的发生，治疗期间应定期进行尿常规、血尿素氮、血肌酐等检查，并应密切观察前庭功能及听力改变。有条件者应进行血药浓度监测，调整剂量使血药浓度在...

登录

【注意事项】

对奈替米星或任何一种氨基糖苷类抗生素有过敏或严重毒性反应者禁用。

【孕妇及哺乳期妇女用药】

氨基糖甙类药物可进入胎盘和乳汁，妊娠期或哺乳期患者应用此类药物应被告之对胎儿或婴儿有潜在的损伤，为安全起见，宜避免使用。

【药理作用】

暂无信息

【药代动力学】

暂无信息

【化学成份】

输入 stop 结束输出：

输入“continue”继续输出下面十个搜索结果

输入“stop”停止输出并结束此次搜索

stop

输入回车以继续

- 不指定药品类别的搜索，使用适应症搜索

12 - 血液和造血器官用药

13 - 不指定药品类别,进行全局搜索

14 - 退出程序

13

现在你可以选择用药品名称/成份/适应症的关键词来搜索药品,请按照下文指示输入合法的数字:

0 - 药品名称

1 - 成份

2 - 适应症

3 - 退出程序

2

现在你可以输入想要搜索的关键词:

同时你也可以输入“886”以退出程序

胃

共找到 161 个匹配药品

【药品类别】

神经系统

【药品名称】

通用名称： 丁溴东莨菪碱注射液

英文名称： Scopolamine Butylbromide Injection

商品名称： 解痉灵针

【成份】

本品主要成份为丁溴东莨菪碱。辅料为氯化钠、注射用水。 化学名称： 溴化 6β, 7β-环氧-3α-羟基-8-丁基-1αH, 5αH-托烷 (-)-托品酸酯。 分子式： C21H30BrNO4 分子量： 440.3

登录

【适应症】

于胃、十二指肠、结肠内窥镜检查的术前准备，内镜逆行胰胆管造影，和胃、十二指肠、结肠的气钡低张造影或腹部 CT 扫描的术前准备，可减少或抑制胃肠道蠕动； 用于各种病因引起的...

登录

【用法用量】

肌肉注射、静脉注射或溶于 5% 葡萄糖注射液、氯化钠注射液中静脉滴注。成人每次 1~2 支、或一次用 1 支间隔 20~30 分钟后再用 1 支。

【禁忌】

本品应用出现过过敏反应时应停药； 对于血压偏低者应用本品时，应注意防止产生体位性低血压； 皮下或肌注时要注意避开神经与血管，如需反复注射应不在同一部位，宜左右交替注射；

【注意事项】

严重心脏病、器质性幽门狭窄或麻痹性肠梗阻患者禁用；青光眼、前列腺肥大患者慎用。

【孕妇及哺乳期妇女用药】

尚不明确。

【药理作用】

本品为 M 胆/碱受体阻滞药。其外周作用与阿托品相似，仅在作用程度上略有不同； 本品对平滑肌解痉作用较阿托品为强，能选择性地缓解胃肠道、胆道及泌尿道平滑肌痉挛和抑制其蠕...

胃

- 使用成份搜索

10 - 消化道及代谢用药

11 - 心血管系统用药

12 - 血液和造血器官用药

13 - 不指定药品类别,进行全局搜索

14 - 退出程序

13

现在你可以选择用药品名称/成份/适应症的关键词来搜索药品,请按照下文指示输入合法的数字:

0 - 药品名称

1 - 成份

2 - 适应症

3 - 退出程序

1

现在你可以输入想要搜索的关键词:

同时你也可以输入“886”以退出程序

氯

共找到 510 个匹配药品

【药品类别】

皮肤病用药

【药品名称】

通用名称： 咪康唑氯倍他素乳膏

英文名称： Compound Miconazole Nitrate Cream

商品名称： 咪康唑氯倍他素乳膏

【成份】

本品为复方制剂，其组分为硝酸咪康唑、丙酸氯倍他素等。

【适应症】

用于真菌引起的皮炎、湿疹、手足癣、股癣及过敏性皮炎。

【用法用量】

孕妇及哺乳期妇女应在医师指导下，权衡利弊后使用。 孕妇禁止长期、大面积使用。

【禁忌】

本品含皮质类固醇外用制剂，若长期、大面积应用或采用封包治疗，因为全身性吸收作用，可造成可逆性下丘脑-垂体-肾上腺(PHA)轴的抑制，部分患者可出现柯兴综合症、高血糖及尿酸...

登录

【注意事项】

对皮质激素类药物及咪唑类药物过敏者禁用。 面部、眼部、腋部及腋窝等皮肤褶皱部位禁用。

【孕妇及哺乳期妇女用药】

暂无信息

【药理作用】

本品为/硝酸咪康唑与丙酸氯倍他素的复方制剂。硝酸咪康唑为咪唑类抗真菌药。具有抑菌作用、浓度度高时也可具杀菌作用；可抑制真菌麦角固醇等固醇的生物合成；作用于真菌细胞膜。

- 搜索不存在的内容



```
Welcome to kk的丁香园用药助手的药品搬运作业
现在你可以选择是否要指定药品的类别,也可以不指定
如果你想要指定药品的类别,请按照下文指示输入合法的数字
0 - 非性激素和胰岛素的激素类系统用药
1 - 感觉器官用药
2 - 呼吸器官用药
3 - 肌肉-骨骼系统用药
4 - 抗寄生虫、杀虫药和驱虫药
5 - 抗肿瘤药和免疫机能调节药
6 - 皮肤病用药
7 - 神经系统用药
8 - 生殖泌尿系统和性激素用药
9 - 系统用抗感染药
10 - 消化道及代谢用药
11 - 心血管系统用药
12 - 血液和造血器官用药
13 - 不指定药品类别,进行全局搜索
14 - 退出程序
13
现在你可以选择用药品名称/成份/适应症的关键词来搜索药品,请按照下文指示输入合法的数字:
0 - 药品名称
1 - 成份
2 - 适应症
3 - 退出程序
2
现在你可以输入想要搜索的关键词:
同时你也可以输入 "886" 以退出程序
kk肚子好饿
共找到 0 个匹配药品
输入回车以继续
```

◦ 退出程序

```
12 - 血液和造血器官用药
13 - 不指定药品类别,进行全局搜索
14 - 退出程序
14
|----- Byebye -----
```

4.2 程序运行

- 首先需要运行 `JsoupPart.java` 中的 `main` 函数以获取药品信息并存放在 `.txt` 文件中（仅需运行一次），由于获取 `.txt` 文件的时间较长，这些文件都随实验报告提交了，可以将这些 `.txt` 文件放在 D 盘后直接运行 `LucenePart.java` 的 `main` 函数而不需要运行这步提到的 `Jsoup.java` 的 `main` 函数。
- 然后需要运行 `LucenePart.java` 中的 `main` 函数以根据刚才获得的 `.txt` 文件建立索引（仅需运行一次）
- 运行 `UserPart.java` 中的 `main` 函数启动程序，进入用户的药品信息检索。

五、总结

一些遇到的问题：

- 应该避免引入过多未使用的包。
- 当 `Scanner` 调用 `close()` 方法后，对应的 `System.in` 流将会同时关闭，且无法再次打开，将导致无法从控制台输入内容的问题，采取的解决方法是不要关闭 `LucenePart` 中检索时使用的 `Scanner` 对象。
- 写入文件时注意设置参数使文件原本的内容不被覆盖
- `HttpClient` 连接超时的问题，虽然不是很明白为什么这样改就可以了，大概是在运行中只能存在一个 `HttpClient` 对象，不知道这样的猜想是不是对的，网上搜到的相关解释不太看得懂，不过现在应该是没有超时的问题了。

六、参考内容

- Java - Lucene 常用类代码示例 <https://vimsky.com/article/4608.html>
- Apache Lucene Core <https://lucene.apache.org/core/>
- Java - 爬虫教程 JSOUP <http://xiaolongonly.cn/2016/05/06/Reptile1/>
- Jsoup 的 Select 选择器语法 <http://t.zoukankan.com/onblog-p-13036308.html>
- 关于 System.in 流的异常 <https://blog.csdn.net/hsee2006/article/details/105030439>

附加

数据丰富程度

从丁香园用药助手获取的数据包含了该网站上所有的有效药品资料中的：

药品类别、药品名称、成份、适应症、用法用量、禁忌、注意事项、孕妇及哺乳期妇女用药、药理作用、化学成份、OTC属性内容及该药品的具体网页链接。

可以获取到共 14 个 .txt 文件如下图

■ 消化道及代谢.txt	2021/12/9 16:34	文本文档	1,883 KB
■ 血液和造血器官.txt	2021/12/9 16:41	文本文档	689 KB
■ 心血管系统.txt	2021/12/9 16:52	文本文档	1,453 KB
■ 皮肤病用药.txt	2021/12/9 17:03	文本文档	1,313 KB
■ 生殖泌尿系统和性激素.txt	2021/12/9 17:07	文本文档	580 KB
■ 非性激素和胰岛素的激素类系统用药.txt	2021/12/9 17:10	文本文档	516 KB
■ 系统用抗感染药.txt	2021/12/9 17:17	文本文档	959 KB
■ 抗肿瘤药和免疫机能调节药.txt	2021/12/9 17:21	文本文档	661 KB
■ 肌肉-骨骼系统.txt	2021/12/9 17:26	文本文档	868 KB
■ 神经系统.txt	2021/12/9 17:33	文本文档	1,065 KB
■ 抗寄生虫药、杀虫药和驱虫药.txt	2021/12/9 17:36	文本文档	320 KB
■ 呼吸系统.txt	2021/12/9 17:42	文本文档	741 KB
■ 感觉器官.txt	2021/12/9 17:44	文本文档	311 KB
■ 杂类.txt	2021/12/9 17:51	文本文档	913 KB

其中消化道及代谢中包含 1370 个药品资料，血液和造血器官中包含 486 个药品资料，心血管系统中包含 900 个药品资料，皮肤病用药中包含 968 个药品资料，生殖泌尿系统和性激素中包含 400 个药品资料，非性激素和胰岛素的激素类系统用药中包含 292 个药品资料，系统用抗感染药中包含 598 个药品资料，抗肿瘤药和免疫机能调节药中包含 400 个药品资料，肌肉-骨骼系统中包含 594 个药品资料，神经系统中包含 700 个药品资料，抗寄生虫药、杀虫药和驱虫药中包含 239 个药品资料，呼吸系统中包含 579 个药品资料，感觉器官中包含 200 个药品资料，杂类中包含 660 个药品资料。

以上共计 8386 个药品资料。

获取以上全部资料大约需要一个半小时，所以上述获得的 .txt 文件将一同上交qwq，如果仅测试用户程序及索引情况就不需要再花费获取资料的时间，将上述 .txt 文件存放在 D 盘就可。

源代码

出于对可能出现的编码格式导致的注释及输出中的中文乱码情况的考虑，在这里贴贴所有的源代码

- JsoupPart.java

```
// TODO 关于close client，增加 a b c 的关闭，是否可以调整关闭位置

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileOutputStream;
```



```

import java.io.OutputStreamWriter;

import org.apache.http.client.methods.CloseableHttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.util.EntityUtils;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.select.Elements;

import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;

public class JsoupPart {

    // 此 MAIN 函数用于获取药品数据，只需要运行一次获得文件就可
    public static void main(String[] args) throws Exception{

        //long startTime = System.currentTimeMillis();
        // HttpClient 请求获取 https://drugs.dxy.cn/ 首页 html
        CloseableHttpClient client = HttpClients.createDefault();
        HttpGet get = new HttpGet("https://drugs.dxy.cn/");
        CloseableHttpResponse response = client.execute(get);
        String content = "";
        if(response.getStatusLine().getStatusCode() == 200){
            content = EntityUtils.toString(response.getEntity(), "UTF-8");
        }

        //关闭链接
        //client.close();

        // 将前面获取到的 html 文本转换为 Document 对象
        Document doc = Jsoup.parse(content);

        // 首页左边分类名称
        String FirstLevelName[] = new String[50];

        // 首页左边分类id
        String FirstLevelId[] = new String[50];

        // 获取 json 对象(为了得到全部下一级链接)
        String tmpjson =
doc.select("script[type=application/json]").toString();
        int tmpj1 = tmpjson.lastIndexOf("<");
        int tmpj2 = tmpjson.indexOf(">");
        tmpjson = tmpjson.substring(tmpj2+1, tmpj1);
        JSONObject json = JSON.parseObject(tmpjson);

        // 分层获取信息，直到得到 Category List
        JSONObject props = json.getJSONObject("props");
        JSONObject pageProps = props.getJSONObject("pageProps");

        // 首页左边的 List (即药品分类信息)
        JSONArray firstLevelCategoryList =
pageProps.getJSONArray("firstLevelCategoryList");

```



```

// 遍历 firstLevelCategoryList 将左边 List 的 name 和 id 存在数组中
int FirstLevelSize = firstLevelCategoryList.size();
for(int i=0;i<FirstLevelSize;i++) {
    FirstLevelName[i] =
(firstLevelCategoryList.getJSONObject(i)).get("name").toString();
    FirstLevelId[i] =
(firstLevelCategoryList.getJSONObject(i)).get("id").toString();
}

// 首页右边的 List (左边的下一层
JSONArray secondLevelCategoryList =
pageProps.getJSONArray("secondLevelCategoryList");
int SecondLevelSize = secondLevelCategoryList.size();

// 遍历访问首页右边的所有 List
for(int i=0;i<SecondLevelSize;i++) {

    // FirstName 为药品类别名，即首页左边栏中的药品一层分类
    String FirstName = "";

    // SecondName 为首页右边栏中的药品二级分类，用处不大所以未做处理
    //String SecondName =
secondLevelCategoryList.getJSONObject(i).getString("name").toString();

    // FirstId 为首页左边栏中的药品分类id
    String FirstId =
secondLevelCategoryList.getJSONObject(i).getString("supId").toString();

    // SecondId 为首页右边栏中的药品二层分类id，用于获取右边链接的 url，即
SecondUrl
    String SecondId =
secondLevelCategoryList.getJSONObject(i).getString("id").toString();
    String SecondUrl = "https://drugs.dxy.cn/category/" + SecondId;

    // 遍历药品分类的 Id，用于确定右边的药品二级分类属于哪一个一级分类
    for(int j=0;j<FirstLevelSize;j++) {
        if(FirstLevelId[j].equals(FirstId)) {
            FirstName = FirstLevelName[j];
            break;
        }
    }
}

// HttpClient 请求获取网页 html
HttpGet get_c = new HttpGet(SecondUrl);
CloseableHttpResponse response_c = client.execute(get_c);
content = "";
if(response_c.getStatusLine().getStatusCode()==200){
    content = EntityUtils.toString(response_c.getEntity(), "UTF-
8");
}

// 将前面获取到的 html 文本转换为Document对象
doc = Jsoup.parse(content);

// 获取当前页面的 PageSize
String tmpstring =
doc.select("script[id=__NEXT_DATA_]").toString();
int tmp1 = tmpstring.indexOf("pageSize");

```

```

int tmp2 = tmpstring.indexOf("total");
int PageSize = Integer.valueOf(tmpstring.substring(tmp1+10,
tmp2-2));

// 遍历当前页面的所有有效 Page
for(int j = 0;j < PageSize;j++) {
    // HttpClient 请求获取网页 html
    HttpGet get_b = new HttpGet(SecondUrl + "?page=" + j);
    CloseableHttpResponse response_b = client.execute(get_b);
    content = "";
    if(response_b.getStatusLine().getStatusCode()==200){
        content = EntityUtils.toString(response_b.getEntity(),
"UTF-8");
    }

    // 将前面获取到的 html 文本转换为Document对象
    doc = Jsoup.parse(content);

    // 选择以 /drug 开头的链接的 Elements
    Elements DrugsElements = doc.select("a[href^=/drug]");
    //System.out.println(DrugsElements);

    // 遍历访问以上获得的 Elements
    int NumberElements = DrugsElements.size();
    for(int k = 0;k < NumberElements;k++) {

        // 获取具体药品链接地址
        String ElementTmp = DrugsElements.get(k).toString();
        int elementtmp1 = ElementTmp.indexOf("\\");
        int elementtmp2 = ElementTmp.lastIndexOf("\\");
        ElementTmp = ElementTmp.substring(elementtmp1+1,
elementtmp2);

        ElementTmp = "https://drugs.dxy.cn" + ElementTmp;
        System.out.println(ElementTmp);

        // HttpClient 请求获取网页 html
        HttpGet get_a = new HttpGet(ElementTmp);
        CloseableHttpResponse response_a =
client.execute(get_a);
        content = "";
        if(response_a.getStatusLine().getStatusCode() == 200){
            content =
EntityUtils.toString(response_a.getEntity(), "UTF-8");
        }

        // 将前面获取到的 html 文本转换为Document对象
        doc = Jsoup.parse(content);

        // 获取文本
        String OneOfDrugs = doc.text();
        //System.out.println(doc);
        //System.out.println("-----
-----");

        // 获得药品的必要属性
        // 每行最后注释的数字为搜索标记的长度
        int MedicineNameIndex1 = OneOfDrugs.indexOf("通用名称");
        // + 5

```

```

        int MedicineNameIndex2 = OneOfDrugs.indexOf("英文名称");
        // + 5
        int MedicineNameIndex3 = OneOfDrugs.indexOf("商品名称");
        // + 5
        int MedicineComponentIndex = OneOfDrugs.indexOf("【成份】");
        // + 5
        int MedicineIndicationIndex = OneOfDrugs.indexOf("【适应症】");
        // + 6
        int MedicineUsageIndex = OneOfDrugs.indexOf("【用法用量】");
        // + 7
        int MedicinePrecautionsIndex = OneOfDrugs.indexOf("【注意事项】");
        // + 7
        int MedicineContraindicationsIndex =
OneOfDrugs.indexOf("【禁忌】");
        // + 5
        int MedicineGravidaIndex = OneOfDrugs.indexOf("【孕妇及哺乳期妇女用药】");
        // + 13
        int MedicinePharmacologicalActionIndex =
OneOfDrugs.indexOf("【药理作用】");
        // + 7
        int MedicinePharmacokineticsIndex =
OneOfDrugs.indexOf("【药代动力学】");
        // + 8
        int MedicineChemicalCompositionIndex =
OneOfDrugs.indexOf("【化学成份】");
        // + 7
        int MedicineOTCIndex = OneOfDrugs.indexOf("【是否OTC】");
        // + 8

        // 获取各个属性的内容
        // 获取通用名称
        String MedicineNameString1 = "";
        if(MedicineNameIndex1 >= 0) {
            MedicineNameString1 =
OneOfDrugs.substring(MedicineNameIndex1 + 5);
            if(MedicineNameIndex2 >= 0) {
                MedicineNameString1 =
MedicineNameString1.substring(0, MedicineNameIndex2 - MedicineNameIndex1 -
5);
            } else if(MedicineNameIndex3 >= 0) {
                MedicineNameString1 =
MedicineNameString1.substring(0, MedicineNameIndex3);
            } else {
                int NameIndexTmp =
MedicineNameString1.indexOf("【");
                if(NameIndexTmp < 0) {
                    NameIndexTmp =
MedicineNameString1.indexOf("药品资讯");
                }
                if(NameIndexTmp < 0) {
                    NameIndexTmp =
MedicineNameString1.indexOf("同类型药品");
                }
                if(NameIndexTmp < 0) {
                    NameIndexTmp =
MedicineNameString1.indexOf("。");
                }
                if(NameIndexTmp < 0) {
                    NameIndexTmp = MedicineNameString1.indexOf("
");
                }
                if(NameIndexTmp < 0) {

```

```

NameIndexTmp = MedicineNameString1.length()
/ 2;
    }
    MedicineNameString1 =
MedicineNameString1.substring(0,NameIndexTmp);
    }
    MedicineNameString1 =
MedicineNameString1.replace('\n', '. ');
    }

    // 获取英文名称
    String MedicineNameString2 = "";
    if(MedicineNameIndex2 >= 0) {
        MedicineNameString2 =
OneOfDrugs.substring(MedicineNameIndex2 + 5);
        if(MedicineNameIndex3 >= 0) {
            MedicineNameString2 =
MedicineNameString2.substring(0,MedicineNameIndex3 - MedicineNameIndex2 -
5);
        }else {
            int NameIndexTmp =
MedicineNameString2.indexOf("【");
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineNameString2.indexOf("药品资讯");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineNameString2.indexOf("同类型药品");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineNameString2.indexOf(". ");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp = MedicineNameString2.indexOf("
");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp = MedicineNameString2.length()
/ 2;
            }
            MedicineNameString2 =
MedicineNameString2.substring(0,NameIndexTmp);
        }
        MedicineNameString2 =
MedicineNameString2.replace('\n', '. ');
    }

    // 获取商品名称
    String MedicineNameString3 = "";
    if(MedicineNameIndex3 >= 0) {
        MedicineNameString3 =
OneOfDrugs.substring(MedicineNameIndex3 + 5);
        int NameIndexTmp = MedicineNameString3.indexOf("【");
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineNameString3.indexOf("药品
资讯");

```

```

    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineNameString3.indexOf("同类
型药品");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineNameString3.indexOf("。");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineNameString3.indexOf(" ");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineNameString3.length() / 2;
    }
    MedicineNameString3 =
MedicineNameString3.substring(0,NameIndexTmp);
    MedicineNameString3 =
MedicineNameString3.replace('\n', '。');
}

// 获取成份
String MedicineComponentString = "";
if(MedicineComponentIndex >= 0) {
    MedicineComponentString =
OneOfDrugs.substring(MedicineComponentIndex + 5);
    int NameIndexTmp =
MedicineComponentString.indexOf("【");
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineComponentString.indexOf("药品资讯");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineComponentString.indexOf("同类型药品");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineComponentString.indexOf("。");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineComponentString.indexOf("
");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineComponentString.length()
/ 2;
    }
    MedicineComponentString =
MedicineComponentString.substring(0,NameIndexTmp);
    MedicineComponentString =
MedicineComponentString.replace('\n', '。');
}

// 获取适应症
String MedicineIndicationString = "";
if(MedicineIndicationIndex >= 0) {
    MedicineIndicationString =
OneOfDrugs.substring(MedicineIndicationIndex + 6);

```

```

        int NameIndexTmp =
MedicineIndicationString.indexOf("【");
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicineIndicationString.indexOf("药品资讯");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicineIndicationString.indexOf("同类型药品");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicineIndicationString.indexOf("。");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicineIndicationString.indexOf(" ");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineIndicationString.length()
/ 2;
        }
        MedicineIndicationString =
MedicineIndicationString.substring(0,NameIndexTmp);
        MedicineIndicationString =
MedicineIndicationString.replace('\n', '. ');
    }

    // 获取用法用量
    String MedicineUsageString = "";
    if(MedicineUsageIndex >= 0) {
        MedicineUsageString =
OneOfDrugs.substring(MedicineUsageIndex + 7);
        int NameIndexTmp = MedicineUsageString.indexOf("【");
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineUsageString.indexOf("药品
资讯");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineUsageString.indexOf("同类
型药品");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineUsageString.indexOf("。");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineUsageString.indexOf(" ");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineUsageString.length() / 2;
        }
        MedicineUsageString =
MedicineUsageString.substring(0,NameIndexTmp);
        MedicineUsageString =
MedicineUsageString.replace('\n', '. ');
    }

    // 获取注意事项

```

```

        String MedicinePrecautionsString = "";
        if(MedicinePrecautionsIndex >= 0) {
            MedicinePrecautionsString =
OneOfDrugs.substring(MedicinePrecautionsIndex + 7);
            int NameIndexTmp =
MedicinePrecautionsString.indexOf("【");
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicinePrecautionsString.indexOf("药品资讯");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicinePrecautionsString.indexOf("同类型药品");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicinePrecautionsString.indexOf("。");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicinePrecautionsString.indexOf(" ");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicinePrecautionsString.length() / 2;
            }
            MedicinePrecautionsString =
MedicinePrecautionsString.substring(0,NameIndexTmp);
            MedicinePrecautionsString =
MedicinePrecautionsString.replace('\n', '. ');
        }

        // 获取用药禁忌
        String MedicineContraindicationString = "";
        if(MedicineContraindicationsIndex >= 0) {
            MedicineContraindicationString =
OneOfDrugs.substring(MedicineContraindicationsIndex + 5);
            int NameIndexTmp =
MedicineContraindicationString.indexOf("【");
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineContraindicationString.indexOf("药品资讯");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineContraindicationString.indexOf("同类型药品");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineContraindicationString.indexOf("。");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineContraindicationString.indexOf(" ");
            }
            if(NameIndexTmp < 0) {
                NameIndexTmp =
MedicineContraindicationString.length() / 2;
            }

```

```

    }
    MedicineContraindicationString =
MedicineContraindicationString.substring(0,NameIndexTmp);
    MedicineContraindicationString =
MedicineContraindicationString.replace('\n', '. ');
    }

    // 获取孕妇及哺乳期妇女用药
    String MedicineGravidaString = "";
    if(MedicineGravidaIndex >= 0) {
        MedicineGravidaString =
OneOfDrugs.substring(MedicineGravidaIndex + 13);
        int NameIndexTmp =
MedicineGravidaString.indexOf("【");
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineGravidaString.indexOf("药
品资讯");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineGravidaString.indexOf("同
类型药品");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicineGravidaString.indexOf(". ");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineGravidaString.indexOf("
");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp = MedicineGravidaString.length() /
2;
        }
        MedicineGravidaString =
MedicineGravidaString.substring(0,NameIndexTmp);
        MedicineGravidaString =
MedicineGravidaString.replace('\n', '. ');
    }

    // 获取药理作用
    String MedicinePharmacologicalActionString = "";
    if(MedicinePharmacologicalActionIndex >= 0) {
        MedicinePharmacologicalActionString =
OneOfDrugs.substring(MedicinePharmacologicalActionIndex + 7);
        int NameIndexTmp =
MedicinePharmacologicalActionString.indexOf("【");
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicinePharmacologicalActionString.indexOf("药品资讯");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicinePharmacologicalActionString.indexOf("同类型药品");
        }
        if(NameIndexTmp < 0) {
            NameIndexTmp =
MedicinePharmacologicalActionString.indexOf(". ");

```



```

    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicinePharmacologicalActionString.indexOf(" ");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicinePharmacologicalActionString.length() / 2;
    }
    MedicinePharmacologicalActionString =
MedicinePharmacologicalActionString.substring(0,NameIndexTmp);
    MedicinePharmacologicalActionString =
MedicinePharmacologicalActionString.replace('\n', '. ');
}

// 获取药代动力学
String MedicinePharmacokineticsString = "";
if(MedicinePharmacokineticsIndex >= 0) {
    MedicinePharmacokineticsString =
OneOfDrugs.substring(MedicinePharmacokineticsIndex + 8);
    int NameIndexTmp =
MedicinePharmacokineticsString.indexOf("【");
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicinePharmacokineticsString.indexOf("药品资讯");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicinePharmacokineticsString.indexOf("同类型药品");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicinePharmacokineticsString.indexOf(". ");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicinePharmacokineticsString.indexOf(" ");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicinePharmacokineticsString.length() / 2;
    }
    MedicinePharmacokineticsString =
MedicinePharmacokineticsString.substring(0,NameIndexTmp);
    MedicinePharmacokineticsString =
MedicinePharmacokineticsString.replace('\n', '. ');
}

// 获取化学成份
String MedicineChemicalCompositionString = "";
if(MedicineChemicalCompositionIndex >= 0) {
    MedicineChemicalCompositionString =
OneOfDrugs.substring(MedicineChemicalCompositionIndex + 7);
    int NameIndexTmp =
MedicineChemicalCompositionString.indexOf("【");
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineChemicalCompositionString.indexOf("药品资讯");
    }

```

```

    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineChemicalCompositionString.indexOf("同类型药品");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineChemicalCompositionString.indexOf("。");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineChemicalCompositionString.indexOf(" ");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp =
MedicineChemicalCompositionString.length() / 2;
    }
    MedicineChemicalCompositionString =
MedicineChemicalCompositionString.substring(0,NameIndexTmp);
    MedicineChemicalCompositionString =
MedicineChemicalCompositionString.replace('\n', '. ');
}

// 获取 OTC 信息
String MedicineOTCString = "";
if(MedicineOTCIndex >= 0) {
    MedicineOTCString =
OneOfDrugs.substring(MedicineOTCIndex + 8);
    int NameIndexTmp = MedicineOTCString.indexOf(" ");
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineOTCString.indexOf("。");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineOTCString.indexOf("药品资
讯");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineOTCString.indexOf("同类型药
品");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = MedicineOTCString.indexOf("【");
    }
    if(NameIndexTmp < 0) {
        NameIndexTmp = 0;
    }
    MedicineOTCString =
MedicineOTCString.substring(0,NameIndexTmp);
    MedicineOTCString = MedicineOTCString.replace('\n',
'. ');
}

// 该药品的准确网页地址
String MedicineAddressString = ElementTmp;

// 在这里进行一个文件的写入
// 默认写入D盘，每个药品类别为一个.txt文件，共 14 个，文件名为药
品分类名

```

```

String FilePath = "D:\\\" + FirstName + ".txt";
File file = new File(FilePath);
// 若文件不存在则创建文件
if(!file.exists()) {
    file.createNewFile();
}
OutputStreamWriter outputStreamWriter = new
OutputStreamWriter(new FileOutputStream(file,true),"UTF-8");
BufferedWriter bufferWriter = new
BufferedWriter(outputStreamWriter);

// 写入文件
bufferWriter.write(MedicineNameString1+"\n");
bufferWriter.write(MedicineNameString2+"\n");
bufferWriter.write(MedicineNameString3+"\n");
bufferWriter.write(MedicineComponentString+"\n");
bufferWriter.write(MedicineIndicationString+"\n");
bufferWriter.write(MedicineUsageString+"\n");
bufferWriter.write(MedicinePrecautionsString+"\n");
bufferWriter.write(MedicineContraindicationString+"\n");
bufferWriter.write(MedicineGravidaString+"\n");

bufferWriter.write(MedicinePharmacologicalActionString+"\n");
bufferWriter.write(MedicinePharmacokineticsString+"\n");

bufferWriter.write(MedicineChemicalCompositionString+"\n");
bufferWriter.write(MedicineOTCString+"\n");
bufferWriter.write(MedicineAddressString+"\n");

bufferWriter.close();
    }
}
//关闭链接
client.close();
//long endTime = System.currentTimeMillis();
//long usedTime = (endTime-startTime)/1000;

//System.out.print(usedTime);
}
}

```

- LucenePart.java

```

import java.io.*;
import java.util.Scanner;

import org.apache.lucene.analysis.Analyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.index.DirectoryReader;
import org.apache.lucene.index.IndexWriter;
import org.apache.lucene.index.IndexWriterConfig;
import org.apache.lucene.index.Term;
import org.apache.lucene.search.BooleanClause.Occur;
import org.apache.lucene.search.BooleanQuery;
import org.apache.lucene.search.IndexSearcher;
import org.apache.lucene.search.ScoreDoc;

```

```

import org.apache.lucene.search.TopDocs;
import org.apache.lucene.search.WildcardQuery;
import org.apache.lucene.store.Directory;
import org.apache.lucene.store.FSDirectory;
import org.apache.lucene.util.Version;
import org.wltea.analyzer.lucene.IKAnalyzer;

public class LucenePart {

    // 索引文件默认存放在 D 盘的 LuceneIndex 文件夹中
    public static final String luceneIndexPath = "D:\\LuceneIndex";
    // 存放 txt 文件目录
    public static final String luceneFilePath = "D:\\";

    public static void main(String[] args) {
        // 此 MAIN 函数用于创建索引，只需要运行一次，创建索引完成后不需要再重复创建
        LucenePart lucene = new LucenePart();
        lucene.createIndexOuter();
    }

    public void createIndexOuter() {
        // 共14个文件
        // 0 非性激素和胰岛素的激素类系统用药.txt
        // 1 感觉器官.txt
        // 2 呼吸器官.txt
        // 3 肌肉-骨骼系统.txt
        // 4 抗寄生虫草、杀虫药和驱虫药.txt
        // 5 抗肿瘤药和免疫机能调节药.txt
        // 6 皮肤病用药.txt
        // 7 神经系统.txt
        // 8 生殖泌尿系统和性激素.txt
        // 9 系统用抗感染药.txt
        // 10 消化道及代谢.txt
        // 11 心血管系统.txt
        // 12 血液和造血器官.txt
        // 13 杂类.txt
        LucenePart lucenePart=new LucenePart();
        // 药品信息文件名
        String fileName[] = {"非性激素和胰岛素的激素类系统用药.txt","感觉器
        官.txt","呼吸器官.txt","肌肉-骨骼系统.txt","抗寄生虫草、杀虫药和驱虫药.txt","抗肿瘤药
        和免疫机能调节药.txt","皮肤病用药.txt","神经系统.txt","生殖泌尿系统和性激素.txt","系统
        用抗感染药.txt","消化道及代谢.txt","心血管系统.txt","血液和造血器官.txt","杂类.txt"};
        // 药品分类名
        String categoryName[] = {"非性激素和胰岛素的激素类系统用药","感觉器官","呼吸
        器官","肌肉-骨骼系统","抗寄生虫草、杀虫药和驱虫药","抗肿瘤药和免疫机能调节药","皮肤
        病用药","神经系统","生殖泌尿系统和性激素","系统用抗感染药","消化道及代谢","心血管系统","血
        液和造血器官","杂类"};
        // 遍历 14 个文件创建索引
        for(int i=0;i<14;i++) {
            lucenePart.createIndex(luceneIndexPath, luceneFilePath +
            fileName[i], categoryName[i]);
        }
    }

    public void searchIndexOuter(Integer searchCategory, Integer searchKey,
    String searchInput) {

```

```

        LucenePart luceneSearch = new LucenePart();
        luceneSearch.searchIndex(luceneIndexPath, SearchCategory, SearchKey,
        SearchInput);
    }

    public void createIndex(String indexPath, String filePath, String
    CategoryName){
        File indexfile = new File(indexPath);
        IndexWriter indexwriter = null;
        try {
            Directory directory = FSDirectory.open(indexfile);
            Analyzer analyzer = new IKAnalyzer();

            IndexWriterConfig indexConfig = new
            IndexWriterConfig(Version.LUCENE_4_10_0,analyzer);
            // 创建 IndexWriter
            indexwriter = new IndexWriter(directory,indexConfig);

            File file = new File(filePath);
            InputStreamReader inputStreamReader = new InputStreamReader(new
            FileInputStream(file),"UTF-8");
            BufferedReader bufferReader = null;

            bufferReader = new BufferedReader(inputStreamReader);

            String whileTmpString = "";
            String tmpString[] = new String[20];
            int tmpStringIndex = 0;
            // 从文件中按行读出内容
            while((whileTmpString = bufferReader.readLine()) != null) {
                tmpString[tmpStringIndex] = whileTmpString;
                //system.out.println(" ----- " + CategoryName + " ----- "
                + tmpStringIndex + " ----- ");
                if(tmpStringIndex == 13) {
                    Medicine tmpMedicine = new
                    Medicine(tmpString[0],tmpString[1],tmpString[2],tmpString[3],tmpString[4],tm
                    pString[5],tmpString[6],tmpString[7],tmpString[8],tmpString[9],tmpString[10]
                    ,tmpString[11],tmpString[12],tmpString[13],file.getName().replace(".txt",
                    ""));

                    Document tmpDocument =
                    tmpMedicine.toDocument(CategoryName);
                    //添加 document, Lucene 的检索以 document 为基本单位
                    indexwriter.addDocument(tmpDocument);
                    tmpStringIndex = 0;
                }else {
                    tmpStringIndex++;
                }
            }
            bufferReader.close();
            indexwriter.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void searchIndex(String indexPath, Integer CategoryKey, Integer
    SearchKey, String SearchInput){
        // 药品类别名称

```

```

String CategoryName[] = {"非性激素和胰岛素的激素类系统用药","感觉器官","呼吸器官","肌肉-骨骼系统","抗寄生虫草、杀虫药和驱虫药","抗肿瘤药和免疫机能调节药","皮肤病用药","神经系统","生殖泌尿系统和性激素","系统用抗感染药","消化道及代谢","心血管系统","血液和造血器官","杂类"};

// 搜索的属性
String SearchName[] = {"name","component","indication"};

String CategoryKeyString = "";
String SearchKeyString = "";
if(CategoryKey < 13) {
    CategoryKeyString = CategoryName[CategoryKey];
}
SearchKeyString = SearchName[SearchKey];

File file = new File(indexPath);
try {
    IndexSearcher searcher = new
IndexSearcher(DirectoryReader.open(FSDirectory.open(file)));

// 添加条件
Term term1 = new Term("category",CategoryKeyString);
Term term2 = new Term(SearchKeyString, SearchInput);

WildcardQuery wildcardQuery1 = new WildcardQuery(term1);
WildcardQuery wildcardQuery2 = new WildcardQuery(term2);

BooleanQuery booleanQuery = new BooleanQuery();
if(CategoryKey < 13) {
    booleanQuery.add(wildcardQuery1, Occur.MUST);
}
booleanQuery.add(wildcardQuery2, Occur.MUST);

TopDocs topDocs = searcher.search(booleanQuery,null,1000);
Integer docSize = topDocs.scoreDocs.length;

// 输出搜索结果，每输出10个结果向用户确认是否需要继续输出
for(int i=0;i<docSize;i++) {
    if((i % 10 == 0)&&(i != 0)) {
        System.out.println("输入 “continue” 继续输出下面十个搜索结果");
        System.out.println("输入 “stop” 停止输出并结束此次搜索");
        Scanner sn = new Scanner(System.in);
        String GetInstruction = sn.nextLine();
        GetInstruction = GetInstruction.replaceAll("\n", "");
        while(!
(GetInstruction.equals("continue")||GetInstruction.equals("stop"))) {
            System.out.println("输入 “continue” 继续输出下面十个搜索结果");
            System.out.println("输入 “stop” 停止输出并结束此次搜索");
            GetInstruction = sn.nextLine();
            GetInstruction = GetInstruction.replaceAll("\n",
""");
        }
        //sn.close();
        if(GetInstruction.equals("continue")) {
            ScoreDoc scoreDoc = topDocs.scoreDocs[i];
            int docIndex = scoreDoc.doc;

```

```

        Document document = searcher.doc(docIndex);
        System.out.println(document.get("print"));
    }else {
        break;
    }
}
}else {
    ScoreDoc scoreDoc = topDocs.scoreDocs[i];
    int docIndex = scoreDoc.doc;
    Document document = searcher.doc(docIndex);
    System.out.println(document.get("print"));
}
}
}
} catch (IOException e) {
    e.printStackTrace();
}
}
}

```

- Medicine.java

```

import org.apache.lucene.document.Document;
import org.apache.lucene.document.Field;
import org.apache.lucene.document.TextField;

public class Medicine {
    // 名称
    String ChineseName;
    String EnglishName;
    String DrugName;

    // 成份
    String Component;
    // 适应症
    String Indication;
    // 用法用量
    String Usage;
    // 注意事项
    String Precautions;

    // 禁忌
    String Contraindication;

    // 孕妇
    String Gravida;

    // 药理作用
    String PharmacologicalAction;

    // 药代动力学
    String Pharmacokinetics;

    // 化学成分
    String ChemicalComposition;

    // 是否OTC
    String OTC;
}

```

```
// 具体网页地址
String Address;

// 打印信息
String PrintMedicine;

// 药品类别
String CategoryName;

Medicine(String a, String b, String c, String d, String e, String f,
String g, String h, String i, String j, String k, String l, String m, String
n, String o){
    ChineseName = a;
    if(a.equals("")) {
        ChineseName = "暂无信息";
    }
    EnglishName = b;
    if(b.equals("")) {
        EnglishName = "暂无信息";
    }
    DrugName = c;
    if(c.equals("")) {
        DrugName = "暂无信息";
    }
    Component = d;
    if(d.equals("")) {
        Component = "暂无信息";
    }
    Indication = e;
    if(e.equals("")) {
        Indication = "暂无信息";
    }
    Usage = f;
    if(f.equals("")) {
        Usage = "暂无信息";
    }
    Precautions = g;
    if(g.equals("")) {
        Precautions = "暂无信息";
    }
    Contraindication = h;
    if(h.equals("")) {
        Contraindication = "暂无信息";
    }
    Gravida = i;
    if(i.equals("")) {
        Gravida = "暂无信息";
    }
    PharmacologicalAction = j;
    if(j.equals("")) {
        PharmacologicalAction = "暂无信息";
    }
    Pharmacokinetics = k;
    if(k.equals("")) {
        Pharmacokinetics = "暂无信息";
    }
    ChemicalComposition = l;
    if(l.equals("")) {
```



```

        ChemicalComposition = "暂无信息";
    }
    OTC = m;
    if(m.equals("")) {
        OTC = "暂无信息";
    }
    Address = n;
    if(n.equals("")) {
        Address = "暂无信息";
    }
    CategoryName = o;
    if(o.equals("")) {
        CategoryName = "暂无信息";
    }
    PrintMedicine = "-----
-----\n"
        + "【药品类别】\n" + CategoryName + "\n"
        + "【药品名称】\n通用名称: " + ChineseName + "\n英文名
称: " + EnglishName + "\n商品名称: " + DrugName + "\n"
        + "【成份】\n" + Component + "\n"
        + "【适应症】\n" + Indication + "\n"
        + "【用法用量】\n" + Usage + "\n"
        + "【禁忌】\n" + Precautions + "\n"
        + "【注意事项】\n" + Contraindication + "\n"
        + "【孕妇及哺乳期妇女用药】\n" + Gravida + "\n"
        + "【药理作用】\n" + PharmacologicalAction + "\n"
        + "【药代动力学】\n" + Pharmacokinetics + "\n"
        + "【化学成份】\n" + ChemicalComposition + "\n"
        + "【是否OTC】\n" + OTC + "\n"
        + "【丁香园用药助手链接】\n" + Address + "\n"
        + "-----\n";
    }

    Document toDocument(String Category) {

        Document doc = new Document();

        doc.add(new TextField("category",Category,Field.Store.YES));
        doc.add(new TextField("name",ChineseName + " " + EnglishName + " " +
DrugName, Field.Store.YES));
        doc.add(new TextField("component",Component,Field.Store.YES));
        doc.add(new TextField("indication",Indication,Field.Store.YES));
        doc.add(new TextField("print",PrintMedicine,Field.Store.YES));

        return doc;
    }
}

```

- UserPart.java

```

import java.util.Scanner;

public class UserPart {
    public static void main(String[] args) {
        Scanner sn = new Scanner(System.in);
    }
}

```

```
while(true) {
    System.out.println("      ████ ██████████ ██████████ ██████████ ");
    System.out.println("      ████ ██████████ ██████████ ██████████ ");
    System.out.println("      ████ ██████████ ██████████ ██████████ ");
    System.out.println("      ████ ██████████ ██████████ ██████████ ");
    System.out.println("████████████████████████████████████████████");
    System.out.println("Welcome to kk的丁香园用药助手的药品搬运作业");
    System.out.println("现在你可以选择是否要指定药品的类别,也可以不指定");
    System.out.println("如果你想要指定药品的类别,请按照下文指示输入合法的数字");

    System.out.println("0 - 非性激素和胰岛素的激素类系统用药");
    System.out.println("1 - 感觉器官用药");
    System.out.println("2 - 呼吸器官用药");
    System.out.println("3 - 肌肉-骨骼系统用药");
    System.out.println("4 - 抗寄生虫草、杀虫药和驱虫药");
    System.out.println("5 - 抗肿瘤药和免疫机能调节药");
    System.out.println("6 - 皮肤病用药");
    System.out.println("7 - 神经系统用药");
    System.out.println("8 - 生殖泌尿系统和性激素用药");
    System.out.println("9 - 系统用抗感染药");
    System.out.println("10 - 消化道及代谢用药");
    System.out.println("11 - 心血管系统用药");
    System.out.println("12 - 血液和造血器官用药");
    System.out.println("13 - 不指定药品类别,进行全局搜索");
    System.out.println("14 - 退出程序");

    Integer CategoryType = Integer.parseInt(sn.nextLine());
    while(CategoryType < 0 || CategoryType > 14) {
        System.out.println("!!请输入合法的数字(在 0 ~ 14 之间)");
        CategoryType = Integer.parseInt(sn.nextLine());
    }
    if(CategoryType.equals(14)) {
        break;
    }

    System.out.println("现在你可以选择用药品名称/成份/适应症的关键词来搜索药品,请按照下文指示输入合法的数字:");
    System.out.println("0 - 药品名称");
    System.out.println("1 - 成份");
    System.out.println("2 - 适应症");
    System.out.println("3 - 退出程序");
    Integer SearchType = Integer.parseInt(sn.nextLine());
    while(SearchType < 0 || SearchType > 3) {
        System.out.println("!!请输入合法的数字(在 0 ~ 3 之间)");
        SearchType = Integer.parseInt(sn.nextLine());
    }
    if(SearchType.equals(3)) {
        break;
    }
    System.out.println("现在你可以输入想要搜索的关键词:");
    System.out.println("同时你也可以输入 “886” 以退出程序");
    String SearchKey = sn.nextLine();
    SearchKey = SearchKey.replaceAll("\n", "");

    if(SearchKey.equals("886")) {
        break;
    }
}
```

```
        LucenePart lucenePart = new LucenePart();
        lucenePart.searchIndexOuter(CategoryType, SearchType,
SearchKey);

        System.out.println("输入回车以继续");
        sn.nextLine();
    }
    sn.close();
    System.out.println("----- Byebye -----");
    System.exit(0);
}
}
```