基于文本内容的销售线索检索

实验目的：对网页进行信息提取；对字符串进行中文分词

实验环境：Mac的Vmare虚拟机下， Windows10 Visual Studio 2012

抽象数据结构说明：

|  |  |  |
| --- | --- | --- |
| 数据结构 | 函数名称 | 函数功能 |
| 栈Stack | push | 压栈 |
| pop | 退栈 |
| top | 返回栈顶元素 |
| empty | 判断栈是否为空 |
| 字符串CharString | indexOf | 查找子串的位置 |
| subString | 截取字符串，返回一个新串 |
| concat | 将另外一个字符串接到当前字符串的末尾 |
| operator= | 重载赋值操作符 |
| insert | 在字符串末尾插入字符 |
| remove | 删除第i个字符 |
| operator== | 判断操作符 |
| operator[] | 返回第i个位置元素的引用 |
| operator<<, >> | 输出流、文件输入流 |
| removeSpace | 删除字符中的所有空格 |
| size | 返回字符串的规模 |
| data | 将字符串转化为字符数组 |
| 字符串链表CharStringLink | add | 添加元素（字符串、字符数组、另外一个字符串链表） |
| remove | 删除元素（字符串、字符数组），删除第i个位置的元素 |
| search | 查找某元素的位置（字符串、字符数组） |
| size | 字符串链表的规模 |
| operator<< | 重载输出流（逆序输出） |
| 字典Dictionary | init | 字典的初始化 |
| put | 插入词条 |
| search | 查询某个词是否在词典中 |
| size | 查询词条的数目 |
| dividePhrase | 对短语进行分词（短语中只有中文字符） |
| divideSentence | 对句子进行分词（句子中可以有标点、数字、英文） |
| 网页处理WebsiteProcessor | extractInfo | 处理文件中存储的网页url，将处理结果存储到另外一个文件 |
| initDictionary | 初始化词库 |
| divideWords | 对某个句子进行分词 |
| 编码转换Converter | UnicodeToChinese | 将unicode编码转换为中文 |

*算法说明*

**网页文件的解析和提取**

1. 首先从头扫描整个网页源码，转1。
2. 如果找到了<\*\*\*>, </\*\*\*>, <\*\*\*/>形式的子串，则生成一个节点，节点分别标记为左界、右界、自匹配，转3。如果没有找到，则转4。

Note : 左界、右界、自匹配的说明例子

1. 如果当前节点是左界，则将结点压入栈中；

<a href="[http://dealer.cehome.com/](view-source:http://dealer.cehome.com/)" target="\_blank">代理商</a>

<meta name="keywords" content="" />

（红色部分是左界，绿色部分是右界，蓝色部分自匹配）

如果当前节点是右界，则将结点弹出，同时根据节点的标签分别处理；

如果当前节点是自匹配，则不作处理。转2。

1. 如果已经搜集到所有的信息，则将信息写入文件；否则，则向文件输出空。

Note 1: 此处采取的策略是搜集所有可能满足如下条件的模式串，然后在搜集完成后，挑选其中特定位置的模式串中的内容，即为所要提取的信息。

|  |  |
| --- | --- |
| 模式串 | 包含的文本类型 |
| <div class=”z”><a>\*\*\*</a></div> | 发帖大类、小类、标题 |
| <td class=”t\_f”>\*\*\*</td> | 发帖内容 |
| <div class=”pi”><div class=”authi”><a>\*\*\*</a></div></div> | 发帖人 |
| <div class=”pti”><div class=”authi”><a>\*\*\*</a></div></div> | 发帖日期 |
| <div class=”ts z h1”><a>\*\*\*</a></div> | 发帖分类、标题 |

Note 2: 由于<td></td>中除了发帖内容外，还可能出现其他的标签，使用getText(“WebsiteProcessor.h”)对一个字符串去除标签内容，保留文本内容。

**分词算法**

1. 首先正向扫描一个句子sentence，根据其中的英文标点符号（单字节）、数字（单字节）和中文标点符号（双字节）将其分成多个短语（phrase）。
2. 对每个短语分别进行分词，采用逆向最大匹配法。
3. 从后往前取一个固定长度的子串。若子串长度大于0，转b；否则，结束。
4. 如果子串在词库中，则将其加入分词结果，将子串从短语中删除，转a。
5. 如果子串不在词库中，则长度减少1，若子串长度大于0，则转b；否则，取出末尾的汉字，加入分词结果，短语中删除该汉字，转a。

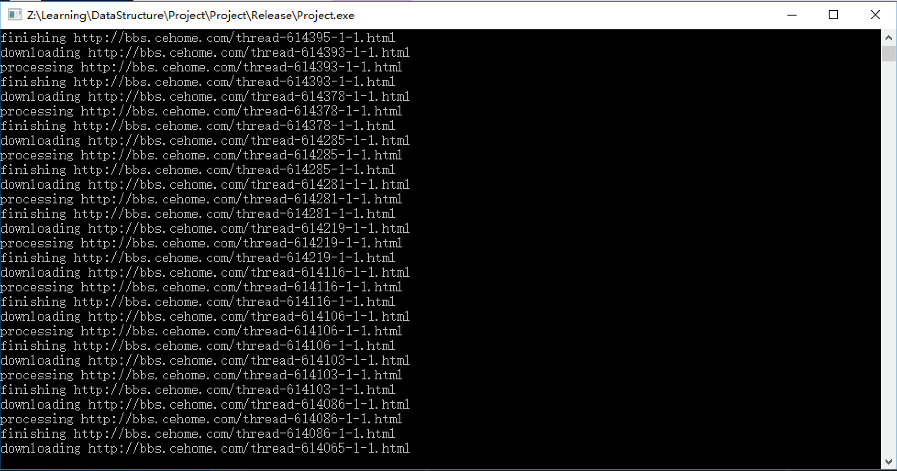
Note 1 : 分词算法可以选择删除还是保留无用词。无用词词库从/Dictionary/uselessWord.dic加载，其中包括中文中的连词、感叹词、谓词等。

Note 2 : 此处采用了hash算法来实现对词库中词语的快速查找，每个桶都是一个字符串链表，假如两个字符的hash码相同，则将它们放入同一个桶。桶容量为素数10133, 总共的词条数为275909 条，平均意义下的每个字符串链表的长度为27, 实际测得的字符串链表最长为48。因此这种策略实现了空间和时间之间较好的折中，如果希望更加节省空间或者更加节省时间，只需要调整桶容量即可。

***流程:*** [载入词库]->[读取网页到本地文件]->[从本地文件中提取关键信息]->[中文分词]->[输入结果]

***遇到的问题及解决：***下载下来的网页源码中，有形如“&#25105;”的字符串，没能找到相关的函数去解析这些字符串。于是自行写了一个函数UnicodeToChinese(“Convert.h”)，通过将“&#25105;”转化为整数25105，然后再转化为wchar\_t，然后调用wstring\_convert，转化为对应的中文字符串。

***操作说明：*打开Project.exe，出现如图所示界面，显示当前正在下载或者处理的网页信息。**



***实验结果：***100个网页中有3个无法获取信息（网页本身无法访问）。



***功能亮点***

1. 自行设计和改进了hash函数。具体参见算法说明。
2. 分词算法能够过滤掉英文、标点、数字等，同时还能去除中文中的无用词，保留信息量更大的词语，为后续的搜索做好准备。具体算法参加算法说明。

***实验体会***

在建立一个较大的系统时，选取效率较高的数据结构就显得至关重要，比如这次实验中使用的——字符串的KMP算法、使用hash的字典。

此外，自己在不同的编码之间的转换问题上还需要更多的学习，这次在编码转换问题上花费了较多时间，采用的策略也不够优雅。