**华东师范大学数据科学与工程学院实验报告**

|  |  |  |
| --- | --- | --- |
| **课程名称**：信息检索 | **年级**：2019 | **上机实践成绩**： |
| **指导教师**：张蓉 | **姓名**：张雯怡 |  |
| **上机实践名称**：Search-RE | **学号**：10195501425 | **上机实践日期**： |
| **上机实践编号**：3 | **组号**： | **上机实践时间**： |

1. **实验目的**

学习构建支持通配查询的检索系统

1. **实验任务**

-输入：

—给定N个文档(实验2的文件)

—一个通配符查找Q

-输出：

—和Q匹配的词项列表

—Q的查询结果（界面输出文档名列表）

—也就是词项列表做OR操作

—为简化步骤，只要求输出第一项和第二项的OR结果

-要求：

—不要求做词条变化如friends -> friend等，直接用空格作为分割符

—都转成小写A->a

—能支持多次查找

1. **使用环境**

Python 3

1. **实验过程**

首先为词典中所有单词生成轮排索引，并将其存入b树。

def permute\_word(word):

    permute\_list = []

    word = word + '$'

    while word[0] != '$':

        permute\_list.append(word)

        word = word[1:] + word[0]

    permute\_list.append(word)

    return permute\_list

    for word in dict:

        for pm\_word in permute\_word(word):

            tree.insert(pm\_word)

然后输入的查询内容提取前缀，再在b树中进行查找，将所有符合通配查询内容的单词存入wordlist[]。

    while True:

        query = input('Query:')

        if query == 'q':

            print('————————END————————')

            break

        prefix = wildcard\_rotate(query)

        wordlist = []

        if prefix == '$':

            wlist = list(output\_sort.keys())

            for word in wlist:

                if wildcard\_match\_2(query, word):

                    wordlist.append(word)

            wordlist = list(set(wordlist))

            wordlist.sort()

        else:

            matchlist = tree.get\_suffix(prefix)

            wordlist = get\_match\_word(matchlist, query)

        print(wordlist)

最后通过构建倒排索引时生成的dict1{}得到wordlist中每个单词出现的文档编号集合，将其存入docID\_list[]。

        docID\_list = []

        for word in wordlist:

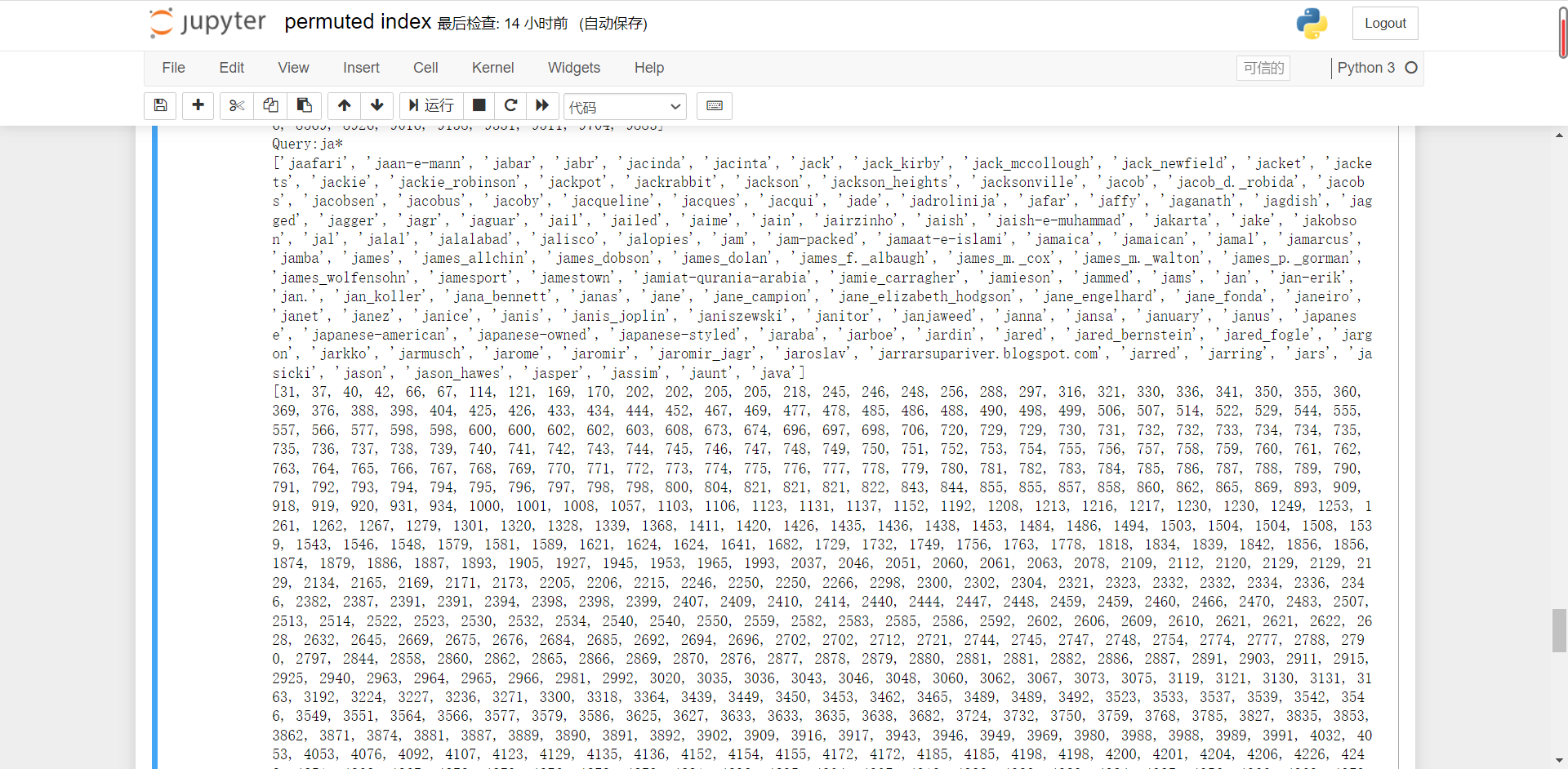
            docID = dict1[word]

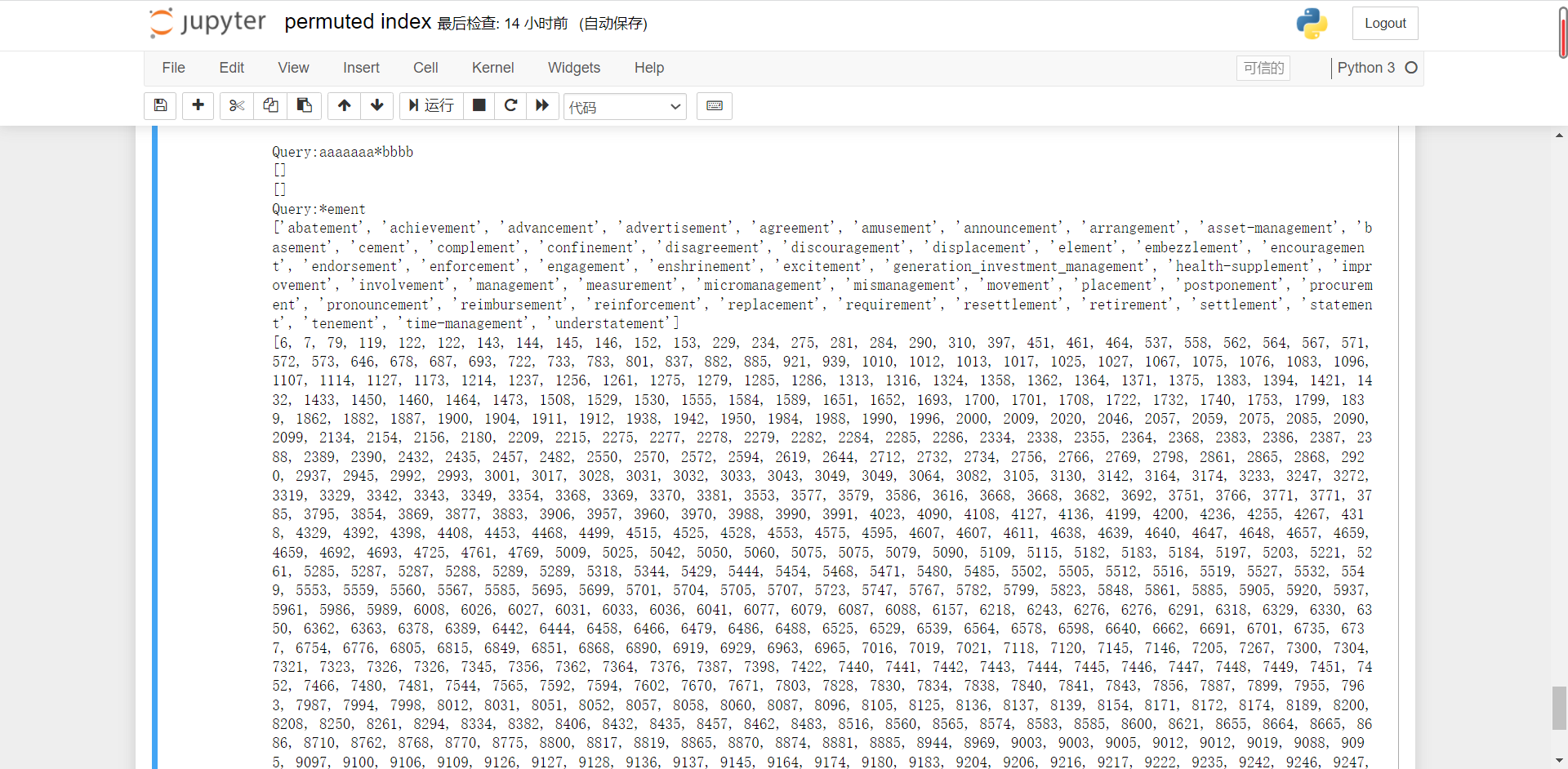
            docID\_list += docID

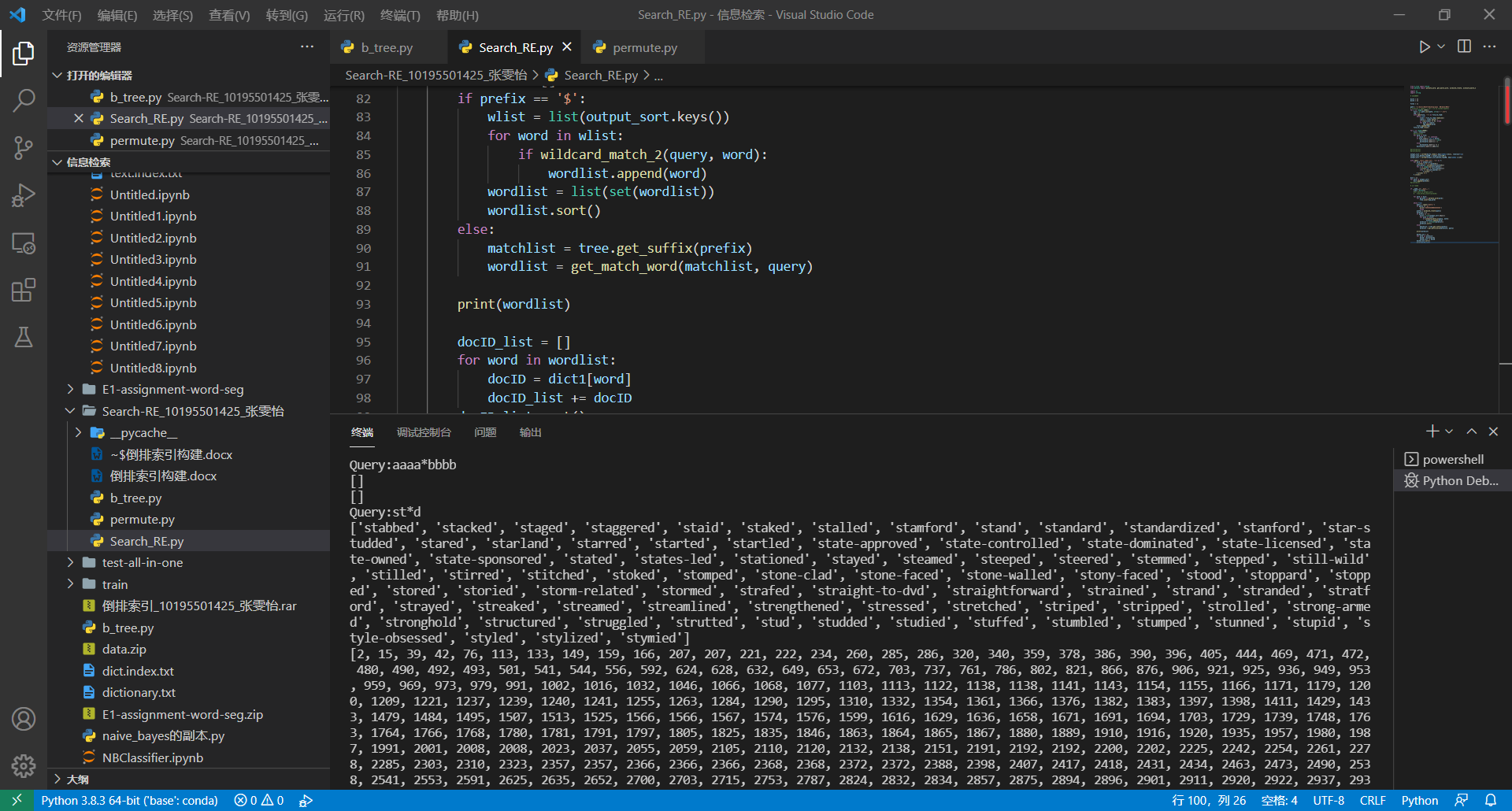
        docID\_list.sort()

        print(docID\_list)

运行结果：







1. **总结**

轮排索引：

* 对于词项hello: 将 *hello$, ello$h, llo$he, lo$hel,* 和 *o$hell* 加入到 B-树中，其中 $ 是一个特殊符号
* 即在词项前面再加一层索引
  + 对于hello，已经存储了 *hello$, ello$h, llo$he, lo$hel,* 和*o$hell*
  + 查询
    - 对于 X, 查询 X$
    - 对于 X\*, 查询 $X\*
    - 对于 \*X, 查询 X$\*
    - 对于 \*X\*, 查询 X\*
    - 对于 X\*Y, 查询 Y$X\*
    - 例子: 假定通配查询为 hel\*o, 那么相当于要查询o$hel\*
  + 轮排索引称为轮排树更恰当