### 小组成员：

### 输入文件：

包含两个部分。dataset下预先放置的是\allen-p\all\_documents下的所有文件。

searching\_list.txt为自行输入的查询词表（若是验证布尔检索尽量少用点词，因为用的 是and，词多了可能没有任何一个文件包含查询词表中的所有词）

### 算法描述：

预处理（公共部分）：

将所有文件都读出来，并进行词根化，去掉停用词的处理。最后得到一个二重列表，第一维相当于是第几个文件，第二维的列表中以单词的形式显示预处理之后文件的所有内容。

布尔检索：

先构建倒排表。这个倒排表是二维字典的形式。第一维是单词，当遍历文件出现新的单词时应该建立第二维的字典。遍历文件出现原来有的词时，如果该文件的序号已经在这个单词对应的字典中出现过直接+1，否则在二维字典上建立该字典项。

倒排表构建完成后，直接采用and的方式进行布尔检索。遍历所有查询词表中的单词，让包含某个单词的所有文件序号构成一个列表，然后将所有列表取交集，得到的最终列表就是搜索结果。（得到的是序号，但是已经做好了文件序号对应到文件名的工作）

语义检索：

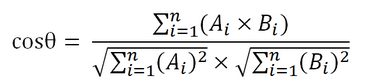
1. 构建一个文件中单词出现的频率的表
2. 构建TF-IDF矩阵，这是一个文件数\*出现过的不同的单词总数 矩阵。

TF=当前文件当前单词出现的次数/当前文件长度

IDF=log[文章总数/(包含该词的文章数目+1)]

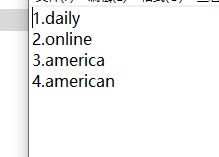
TF\*IDF即可。

1. 搜索，利用余弦相似度。

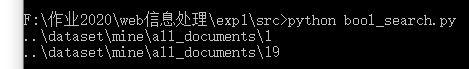


### 运行结果：

初始查询词表：

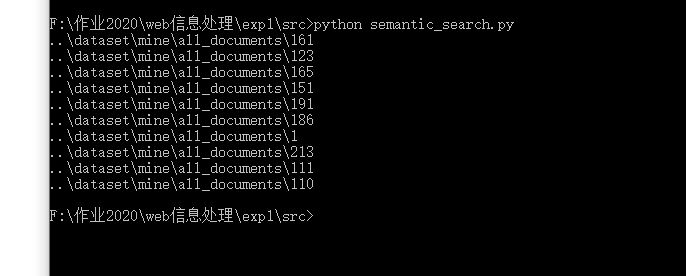


布尔检索运行结果：



可以查到\allen-p\all\_documents下的1和19号文件都包含查询词表中的所有词。

语义检索结果：



因为america和american在许多文件中都出现，所以权重较小，故19号文件被筛了下去（可能还因为它比较长）。

其他的10个文件都含有其他两个查询词，且要么文件较短要么出现次数较多。