

Information Retrieval

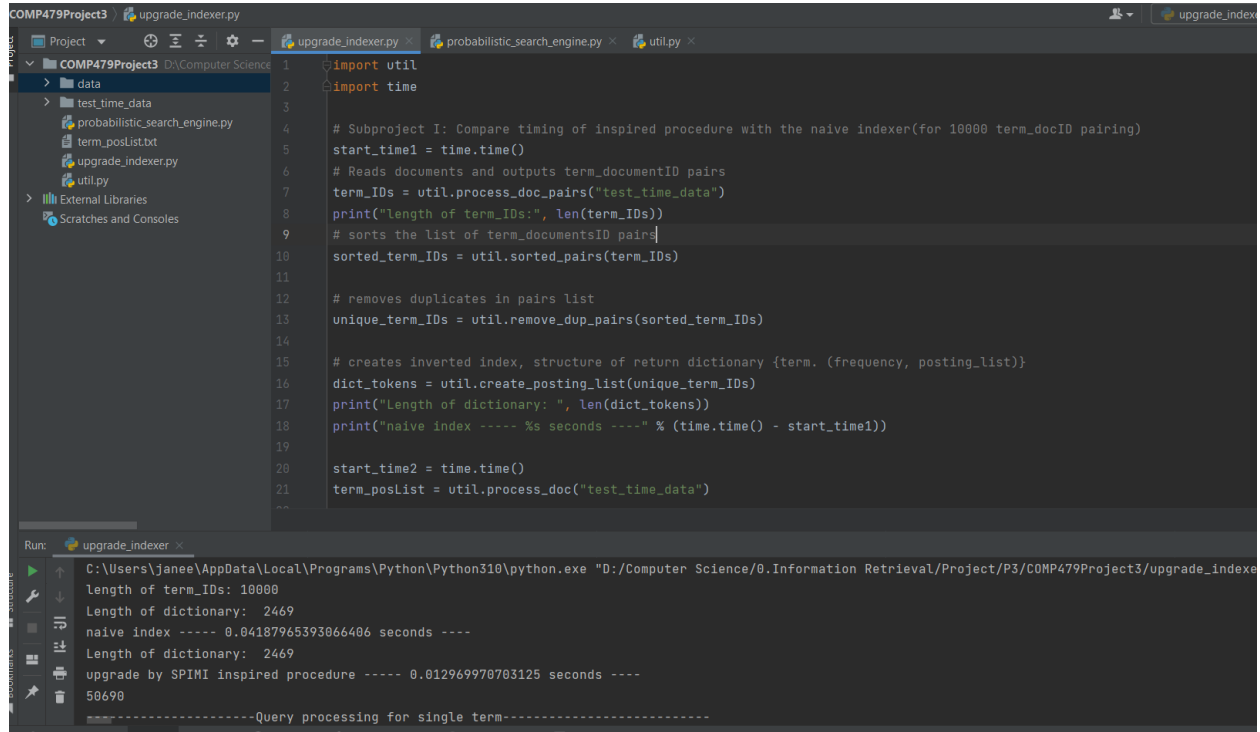
COMP 479 Project 3 DEMO

Lin Ling (40153877)

A report submitted in partial fulfilment of the requirements of Comp479.

Concordia University

If you run the `upgrade_indexer.py`, it will print out the result of subproject 1 needed.



```
1 import util
2 import time
3
4 # Subproject I: Compare timing of inspired procedure with the naive indexer(for 10000 term_docID pairing)
5 start_time1 = time.time()
6 # Reads documents and outputs term_documentID pairs
7 term_IDs = util.process_doc_pairs("test_time_data")
8 print("Length of term_IDs:", len(term_IDs))
9 # sorts the list of term_documentsID pairs
10 sorted_term_IDs = util.sorted_pairs(term_IDs)
11
12 # removes duplicates in pairs list
13 unique_term_IDs = util.remove_dup_pairs(sorted_term_IDs)
14
15 # creates inverted index, structure of return dictionary {term: (frequency, posting_list)}
16 dict_tokens = util.create_posting_list(unique_term_IDs)
17 print("Length of dictionary: ", len(dict_tokens))
18 print("naive index ----- %s seconds ----" % (time.time() - start_time1))
19
20 start_time2 = time.time()
21 term_posList = util.process_doc("test_time_data")
```

Run: upgrade_indexer x

C:\Users\jane\AppData\Local\Programs\Python\Python310\python.exe "D:/Computer Science/0.Information Retrieval/Project/P3/COMP479Project3/upgrade_indexer.py"

Length of term_IDs: 10000

Length of dictionary: 2469

naive index ----- 0.04187965393066406 seconds ----

Length of dictionary: 2469

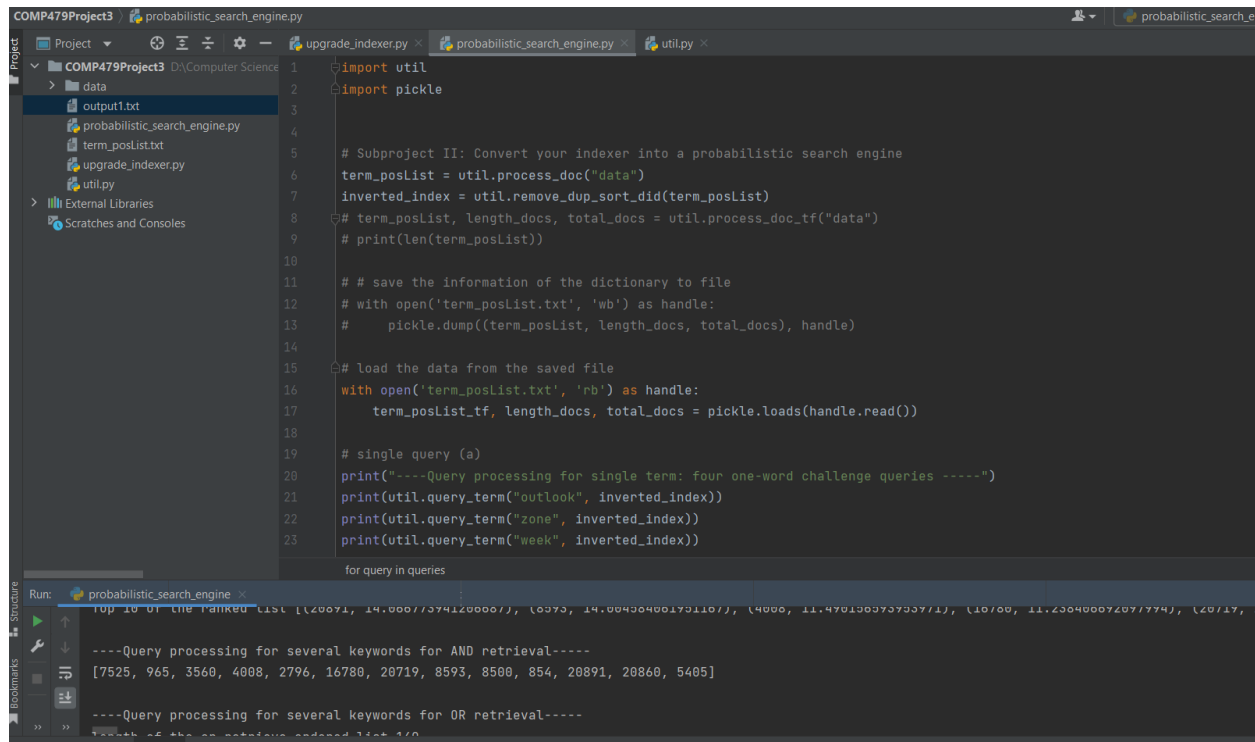
upgrade by SPIMI inspired procedure ----- 0.012969970703125 seconds ----

50690

-----Query processing for single term-----

The detailed explanation of the code implementation and the comparing result includes in [report](#). And if you want to check out the output of this module's execution, you could refer to [output1.txt](#).

If you run the `probabilistic_search_engine.py`, it will print out the result of subproject 2 needed.



```
1 import util
2 import pickle
3
4
5 # Subproject II: Convert your indexer into a probabilistic search engine
6 term_posList = util.process_doc("data")
7 inverted_index = util.remove_dup_sort_did(term_posList)
8 # term_posList, length_docs, total_docs = util.process_doc_tf("data")
9 # print(len(term_posList))
10
11 # # save the information of the dictionary to file
12 # with open('term_posList.txt', 'wb') as handle:
13 #     pickle.dump((term_posList, length_docs, total_docs), handle)
14
15 # load the data from the saved file
16 with open('term_posList.txt', 'rb') as handle:
17     term_posList_tf, length_docs, total_docs = pickle.loads(handle.read())
18
19 # single query (a)
20 print("----Query processing for single term: four one-word challenge queries ----")
21 print(util.query_term("outlook", inverted_index))
22 print(util.query_term("zone", inverted_index))
23 print(util.query_term("week", inverted_index))
24
25 for query in queries:
```

Run: probabilistic_search_engine

```
Top 10 of the ranked list [(20071, 14.00073741200007), (6373, 14.004304001731107), (4000, 11.470130373733771), (10700, 11.230400072077774), (2017, ...
----Query processing for several keywords for AND retrieval----
[7525, 965, 3560, 4008, 2796, 16780, 20719, 8593, 8500, 854, 20891, 20860, 5405]
----Query processing for several keywords for OR retrieval----
Rank of the retrieved ordered list 10
```

Note: If you want to run the code, just be noticed that the code from line 8 to line 13 which is dumped the `term_postList`, `length_docs`, and `total_docs` information to a file and then read from the file for future search using, as generated this information a little bit time-consuming.

The detailed explanation of the code implementation and the comparing result and analysis of three multiplies words queries includes in [report](#). And if you want to check out the output of this module's execution, you could refer to [output2.txt](#). This output just including the top10 of the ranked list of BM25 and Top 10 of the ordered list of OR retrieve.

If you want to check all information of the ranked list of BM25 and all ordered list of OR retrieve, please refer to [output3.txt](#)