

**Name: Fabliha Faiza Maliha**

**ID:500910780**

### **Assignment 1:**

There are two files in this assignment: invert.py and test.py. For implementing both the functions I used Python as a programming language. The invert.py processes the given cacm.all data based on given stopwords and saves it to a dictionary.csv and posting\_list.txt file. Then I implemented the test function which takes in a query term as input and search in the saved dictionary to check if it has appeared in the document. Note that the main function in the test file first runs the invert.py and checks the dictionary with query term. I will provide details of each step in invert.py followed by test.py.

#### **invert.py:**

As stated in the assignment, the input to the program is a collection of documents. The output will be two files: dictionary.csv and postings\_list.txt file.

**Invert.py implementation:** I first read the given cacm.all file and process it through multiple helper functions and save the output to a dictionary.csv and postings\_list.csv. The structure of my dictionary is implemented based on the following keys:

Dictionary\_list = {I, 'T', 'W', 'B', 'A'} where I, T, W, B, and A stands for document\_ID, Title, Abstract, Bibliography, and Authors name respectively.

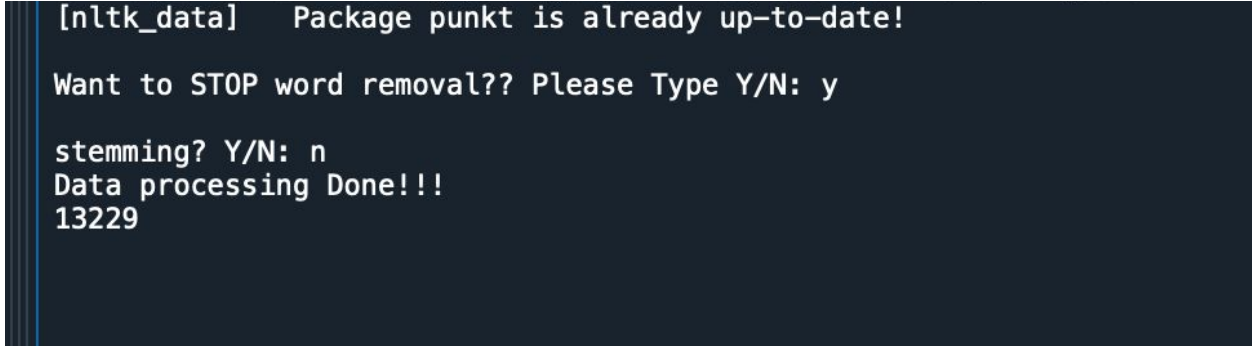
Then I used a *list of lists* data structures to implement the posting list where each term has an associated index (sorted based on alphabetic order). Since it's a list of lists, each item in the postings\_list is also a list which contains the postings of that term. Each posting is associated with a positing class object which is defined by the *Posting* class in the code.

I have considered the given *stopwords* while creating the dictionary.

Next, I created the dictionary and the frequency of each term and saved it as a dictionary.csv which has two columns [index\_term, frequency] along with the postings\_list.txt file.

**Test.py implementation:** Now I will describe how I implemented the test.py. First, the run the invert.py inside the main function of test.py which creates the dictionary and postings\_list file. Since the code is first running the invert file it does not require to separately load any files. Then the code will ask for a query term from the user. If the file appeared in the dictionary then it will print how many times and where the term appeared in terms of the term\_frequency (TF), document ID, and Title. I also print the processing time at the end of the program to show how long it took to process the query term.

**Screenshot:**



```
[nltk_data] Package punkt is already up-to-date!  
Want to STOP word removal?? Please Type Y/N: y  
stemming? Y/N: n  
Data processing Done!!!  
13229
```

lincoln laboratory under leap language data associative processing line drawings (  
's lincoln laboratory under leap language data structure , comparison hardware approach  
's lincoln laboratory under leap language data structure , comparison hardware approach  
positions : 5,54

---

Document number 2969 :

Given query term is seen in the document 2 times.

Title:

optimal program data locations computer networks optimization procedure allocation program data  
files computer network presented . algorithm takes into account dependencies between files  
programs such occur real heterogeneous computer networks . insights into whether not convert  
programs one computer another can also gained model . search procedure file location problem  
described , along example possible application model . morgan h. l. levin k. d

optimal program data locations computer networks optimization procedure  
optimization procedure allocation program data files computer network presented .  
optimization procedure allocation program data files computer network presented .  
positions : 2,10

---

Document number 2971 :

Given query term is seen in the document 1 times.

Title:

sp/k : system teaching computer programming sp/k compatible subset pl/i language has been  
designed teaching programming . features sp/k language were chosen encourage structured problem  
solving computers , make language easy learn use , eliminate confusing redundant constructs ,  
make language easy compile . resulting language suitable troduding programming concepts used  
various applications , including business data processing , scientific calculations non-numeric  
computation . sp/k actually sequence language subsets called sp/1 , sp/2 , ... sp/8 . each  
subset trodudes new programming language constructs while retaining all constructs preceding  
subsets . each subset precisely defined can learned implemented without following subsets . hol  
r. c. wortman d. b. barnard d. t. cordy j. r

used various applications , including business data processing , scientific calculations non-  
numeric  
used various applications , including business data processing , scientific calculations non-  
numeric  
used various applications , including business data processing , scientific calculations non-  
numeric  
positions : 56

---

accurate difficult problems than other two . chan t.f lewis j.g

standard deviation ( unweighted ) sampled data analyzed . two algorithms well-known  
standard deviation ( unweighted ) sampled data analyzed . two algorithms well-known  
standard deviation ( unweighted ) sampled data analyzed . two algorithms well-known  
positions : 15

---

Document number 3167 :

Given query term is seen in the document 2 times.

Title:

updating mean variance estimates : improved method method improved efficiency given updating  
mean variance weighted sampled data additional data value included set . evidence presented  
method stable least accurate best existing updating method . west d.h.d

given updating mean variance weighted sampled data additional data value included set  
given updating mean variance weighted sampled data additional data value included set  
given updating mean variance weighted sampled data additional data value included set  
positions : 16,18

---

Document number 3177 :

Given query term is seen in the document 1 times.

Title:

share secret paper we show divide data d into n pieces such way d easily reconstructable any k  
pieces , but even complete knowledge k - 1 pieces reveals olutely no information d. technique  
enables construction robust key management schemes cryptographic systems can function securely  
reliably even misfortunes destroy half pieces security breaches expose all but one remaining  
pieces . shamir a

share secret paper we show divide data d into n pieces such  
paper we show divide data d into n pieces such  
paper we show divide data d into n pieces such  
positions : 6

---

Total time spent to search this query term: 0.39813648000017565

Please enter a word as Query: |

Document number 3177 :

Given query term is seen in the document 1 times.

Title:

share secret paper we show divide data d into n pieces such way d easily reconstructable any k  
pieces , but even complete knowledge k - 1 pieces reveals olutely no information d. technique  
enables construction robust key management schemes cryptographic systems can function securely  
reliably even misfortunes destroy half pieces security breaches expose all but one remaining  
pieces . shamir a

share secret paper we show divide data d into n pieces such  
paper we show divide data d into n pieces such  
paper we show divide data d into n pieces such  
positions : 6

---

Total time spent to search this query term: 0.39813648000017565

Please enter a word as Query: ZZEND

