Exercise 1

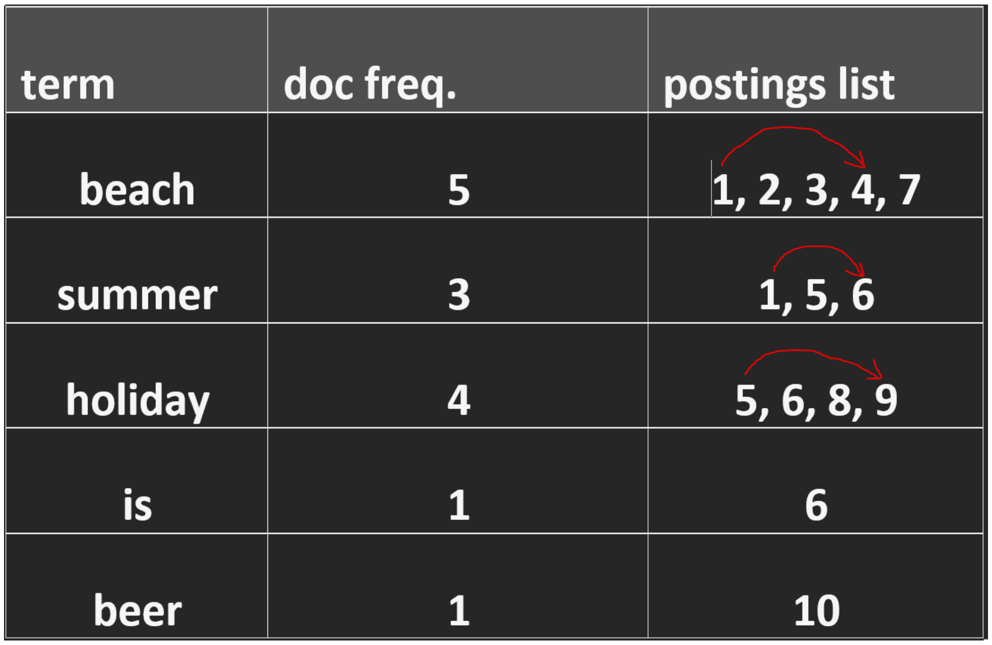
Task 1

Subtask A

term	doc freq.	postings list
beach	5	1, 2, 3, 4, 7
summer	3	1, 5, 6
holiday	4	5, 6, 8, 9
is	1	6
beer	1	10

Subtask B

Skip pointers added:



term	doc freq.	postings list
beach	5	1, 2, 3, 4, 7
summer	3	1, 5, 6
holiday	4	5, 6, 8, 9
is	1	6
beer	1	10

Example:

- Add skip pointer for postings list for the term beach from 1 to 4
- The query:
    - **beach AND holiday**
- We start at the postings lists with the entries **beach = 1** and **holiday = 5**
- So we can take the skip pointer at the term beach from 1 to 4, so we are at beach = 4
- Next we compare beach = 7 → Now we know that there is no match
- **The skip pointer saved us the comparisons ob 1,2,3 for the term beach.**
    - Without skip pointers we need more comparisons, thus this query can be answered in a more efficient way with these skip pointers.

Task 2

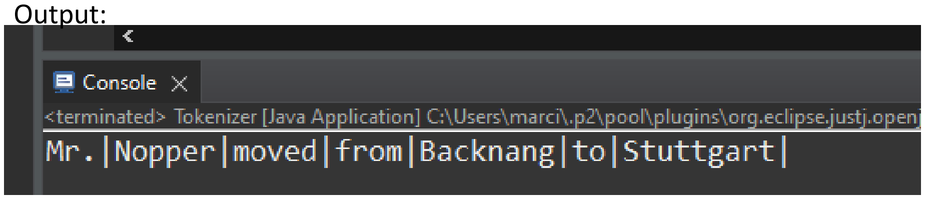
Pseudo-Code:

```
1. Initialize currentChar = 0
2. Initialize lastStop = 0
3. Initialize resultList = []
4. Iterate over inputString at position currentChar with condition currentChar < inputString.length
    a. If inputString[currentChar] is Whitespace add substring of inputString starting from
       index lastStop, ending at index currentChar to resultList. Set lastStop = currentChar+1
    b. If currentChar equals inputString.length – 1 add substring of inputString starting from
       index lastStop, ending at index currentChar+1 to resultList.
    c. currentChar++ and continue iteration at 4
5. return resultList
```

Java-Code implementation:

```java
public class Tokenizer {
	public static String example = "Mr. Nopper moved from Backnang to Stuttgart";

	static List<String> tokenize(String input) {
		List<String> result = new ArrayList();

		int lastStop = 0;

		for(int currentChar=0; currentChar<input.length(); currentChar++) {
			if(Character.isWhitespace(input.charAt(currentChar))) {
				result.add(input.substring(lastStop, currentChar));
				lastStop = currentChar+1;
			}
			if(currentChar == input.length()-1) {
				result.add(input.substring(lastStop, currentChar+1));
			}
		}
		return result;
	}

	public static void main(String[] args) {
		List<String> test = tokenize(example);
		test.forEach(t -> System.out.print(t+ "|"));
	}
}
```

Output:



```
<terminated> Tokenizer [Java Application] C:\Users\marci\.p2\pool\plugins\org.eclipse.justj.openj
Mr.|Nopper|moved|from|Backnang|to|Stuttgart|
```

The presented algorithm has in general a linear runtime, because it only iterates the inputString once with the for-Loop and the loop variable currentChar is never changed/reset inside the loop. Therefore the for-loop is executing exactly n-times for an inputString of length n. All other operations (variable initialization) have a runtime of O(1). This results in a O(n) runtime for the algorithm.

Task 3

i → j ↓ 0	H 1	u 2	a 3	s 4
H 1	0 stay	1 delete	2 delete	3 delete
a 2	1 insert	1 replace	1 stay	2 delete
u 3	2 insert	1 stay	1 transpose	2 delete, replace
s 4	3 insert	2 insert	2 insert, replace	1 stay

Operations		**add**	**condition**
stay	Di-1, j-1	0	if ui = vj
replace	Di-1, j-1	1	
insert	Di,j-1	1	
delete	Di-1,j	1	
transpose	Di-2,j-2	1	ui = vj-1 && ui-1 = vj

Explanation

- The Levenshtein distance between „Haus“ and „Huas“ is 2, because we need to perform two times the replace operation, replacing a with u and the u with a.
- The Damerau-Levenshtein distance between „Haus“ and „Huas“ is 1, because we need to perform one transpose operation, which switches the letters u and a.
- The operations that produced the value in the matrix are described in the table. If there is more than one possible operation these are mentioned in a comma separated list.
- The final distance can be retrieved from the bottom right corner of the matrix.
  We can backtrack the steps starting from the bottom right corner:
  - Stay → step -1, -1 in matrix
  - Transpose → step -2,-2 in matrix
  - Stay → We reached the the top left corner of the matrix

Task 4

Export Tweets to disk → createTweetDB.py

```python
import pandas as pd
import numpy as np
# this script exports all tweets as .txt files to disk. The filename is the tweet handle

nrows = 0
folder = "E:/tweets/"

# read csv
print("reading csv...")
if(nrows>0):
    # only read nrows number of tweets
    data = pd.read_csv('twitter.csv',   sep="\t",
                                        names=["handle", "userid", "username", "tweet"],
                                        dtype={"handle": np.int64, "userid": str, "username": str, "tweet": str},
                                        header=None, nrows=nrows,
                                        quoting=3)
else:
    # read all tweets
    data = pd.read_csv('twitter.csv',   sep="\t",
                                        names=["handle", "userid", "username", "tweet"],
                                        dtype={"handle": np.int64, "userid": str, "username": str, "tweet": str},
                                        header=None,
                                        quoting=3)

# export tweets to txt files where filname is equal to the tweet handle
for item in data.iloc:
    with open(folder+str(item.handle)+'.txt', 'w', encoding='utf-8') as f:
        f.write(item.tweet)
```

Build index file → createIndex.py

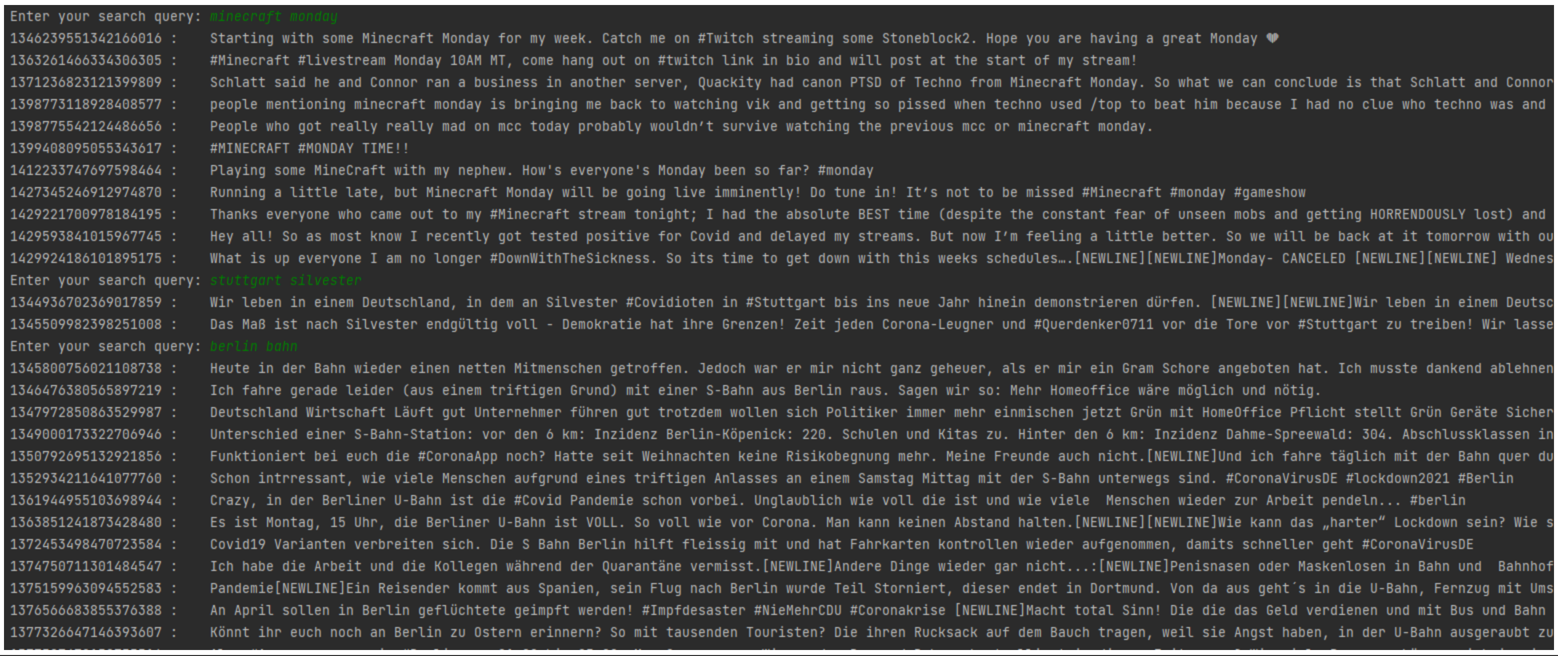
```python
import csv
from datetime import datetime
from itertools import chain
import numpy as np
import nltk
import pandas as pd
import pickle
# This script builds a non-positional inverted index. Postings lists are stored as .csv files on disk.
# The index is stored as pickled python object on disk and can be loaded by the queryEngine later.
# Stopwords of the 8 most common languages of the tweets are filtered, as well as words with length = 1.

# The index is a dictionary with the term as key and the values docFrequency and postings
# docFrequency is an integer which counts how often the term occurs in all documents
# postings is an integer with the file ID for the postings list.
# E.g. postings=100 means, the posting list is stored in the file p100.csv

# config
indexName = datetime.today().strftime('%Y%m%d-%H_%M_%S') + "_index.pickle"
nrows = 0


# token normalization
def normalize(line):
    # this function removes special characters, newlines and tab from tweets, tokenizes the text and sets all terms
    # to lowercase
    text = line.tweet
    if (type(text) != str):
        return ""

    for ch in ['[NEWLINE]', '[TAB]', '#', '\\', '`', '*', '_', '{', '}', '[', ']',
               '(', ')', '>', '+', '-', '.', '!', '?', '$', '\'', '"', '/']:
        if ch in text:
            text = text.replace(ch, " ")
    return text.lower().split()


# index function
def index(filename):
    # read csv
    print("reading csv...")
    if (nrows > 0):
        data = pd.read_csv(filename, sep="\t",
                           names=["handle", "userid", "username", "tweet"],
                           dtype={"handle": np.int64, "userid": str, "username": str, "tweet": str},
                           header=None,
                           nrows=nrows,
                           quoting=3)
    else:
        data = pd.read_csv(filename, sep="\t",
                           names=["handle", "userid", "username", "tweet"],
                           dtype={"handle": np.int64, "userid": str, "username": str, "tweet": str},
                           header=None,
                           quoting=3)
    # get stopwords
    nltk.download('stopwords')
    languages = ['english', 'german', 'spanish', 'portuguese', 'italian', 'french', 'turkish', 'dutch']
    stopwords = dict.fromkeys([i for i in chain.from_iterable([nltk.corpus.stopwords.words(l) for l in languages])])


    # gather data
    progress = 0;
    print("building inverted index ...")
    results = {}
    for item in data.iloc:

        progress = progress + 1;
        if (progress % 10000 == 0):
            print("processing line " + str(progress) + " ...")

        # tokenize tweet
        terms = normalize(item)

        for term in terms:

            # ignore stopwords and single characters
            if term in stopwords or len(term) < 2:
                continue

            if term in results:
                entry = results[term]
                entry["docFreq"] = entry["docFreq"] + 1
                entry["postings"].add(item.handle)
            else:
                results[term] = {"docFreq": 1, "postings": set([item.handle])}

    # save postings lists as file
    print("create postings lists files...")
    fileId = 0

    for term in results:
        # for each term write postings lists to csv file on disk
        entry = results[term]
        fileName = str(fileId) + '.csv'
        file = open('E:/postings_lists/p' + fileName, 'w', newline='')
        writer = csv.writer(file)
        postings = sorted(list(entry['postings']))
        writer.writerows([[handle] for handle in postings])
        entry['postings'] = fileId
        fileId = fileId + 1

        if fileId % 10000 == 0:
            print("writing file " + str(fileId))

    # pickle index as file on disk
    print("pickling results...")
    f = open(indexName, 'wb')
    pickle.dump(results, f)
    f.close()
    print("done.")
    return results


# run
indexData = index("twitter.csv")
```

Query engine → queryEngine.py

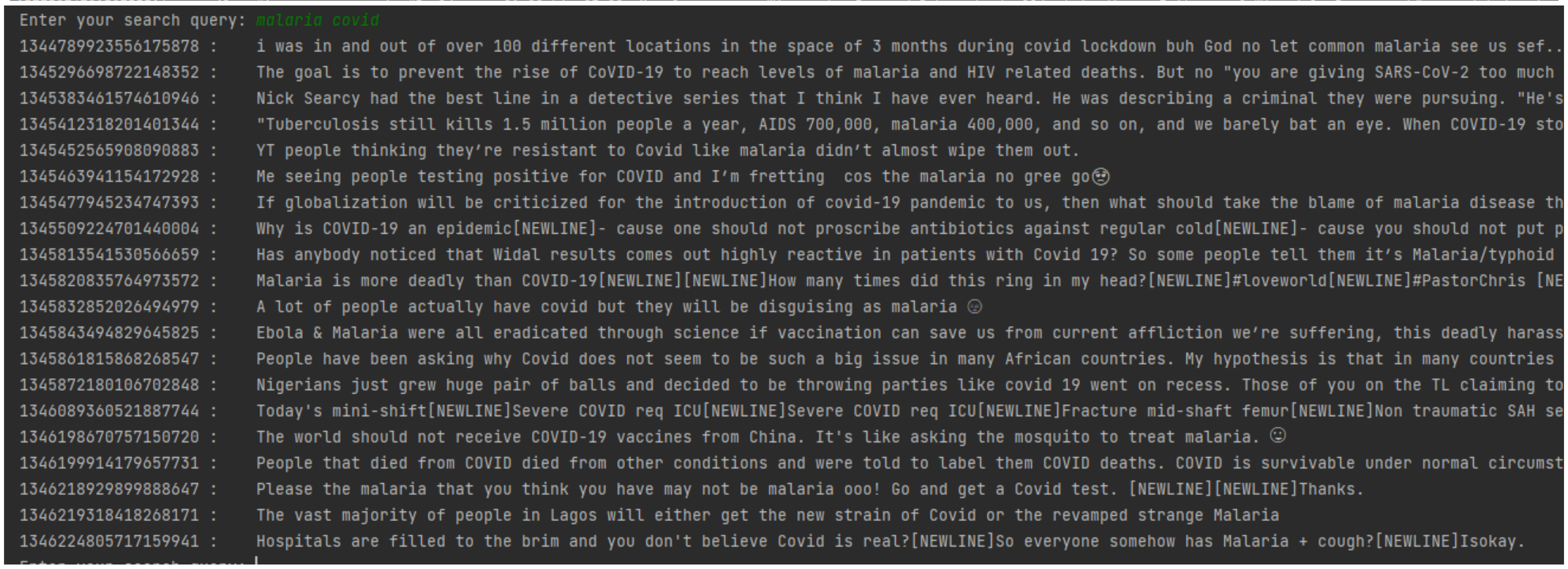
```python
import pickle
from pathlib import Path

import pandas as pd

# config
indexName = "20211108-14_03_27_index.pickle"
postings_dir = "E:/postings_lists/"
tweets_dir = "E:/tweets/"
showResults = 0
index = None

# functions
def query(term):
    if term in index:
        # read csv file from disk and return values as list
        postings = pd.read_csv(postings_dir + 'p' + str(index[term]['postings']) + '.csv')
        return postings.iloc[:, 0].values.tolist()
    else:
        # term not in index, return empty list
        return []

def queryAND(term1, term2):
    # get postings lists
    list1 = iter(query(term1))
    list2 = iter(query(term2))

    # intersection algorithm:

    # get start values smallValue, bigValue not sorted yet!
    smallV = next(list1, None)
    bigV = next(list2, None)

    # if one term has no results, AND query has no results
    if smallV == None or bigV == None:
        return []

    # init conditions → list1 with smallV tries to catchup to list2 to bigV
    if (smallV > bigV):
        temp = smallV
        temp2 = list1
        smallV = bigV
        list1 = list2
        bigV = temp
        list2 = temp2

    matches = []
    # intersect lists
    while (smallV <= bigV):
        # get value
        smallV = next(list1, None)

        # one list reached the end, break and return
        if (smallV == None):
            break

        # match found!
        if (smallV == bigV):
            matches.append(smallV)

        # smallValue passed bigValue => swap values and lists
        if (smallV > bigV):
            temp = smallV
            temp2 = list1
            smallV = bigV
            list1 = list2
            bigV = temp
            list2 = temp2

    return matches
```

```python
def displayResults(postingslists, nOfResults=showResults):
    results = []

    # if results > nOfResults show only nOfResults many results
    maxIndex = len(postingslists)
    if nOfResults != 0 and nOfResults < maxIndex:
        maxIndex = nOfResults

    # if no results
    if maxIndex == 0:
        print("No results...")
        return results

    # iterate over ressults, open tweet.txt file from disk and print tweet in a line
    for i in range(0, maxIndex):
        file = tweets_dir + str(postingslists[i]) + ".txt"
        with open(file, 'r', encoding='utf-8') as f:
            results.append(str(postingslists[i])+" :    "+f.read())
    for i in range(0, len(results)):
        print(results[i])

    return results

# run the query loop:

# check if index file exists
results = Path(indexName);
if results.is_file():

    # unpickle index file
    print("file found, loading...")
    pickleFile = open(indexName, 'rb')
    index = pickle.load(pickleFile)

    # UI loop, let user enter queries
    while True:
        queryString = input("Enter your search query: ")

        # exit program with q
        if (queryString == "q"):
            print("Quit called. Goodbye, see you later alligator...")
            break

        # process query
        terms = queryString.lower().split()
        if (len(terms) == 1):
            # one term, use simple query
            results = query(terms[0])
        else:
            # two terms, use AND query
            results = queryAND(terms[0], terms[1])

        # display results
        resultDocuments = displayResults(results)
```

Results & Example query



```
Enter your search query: minecraft monday
1346239551342166016 :    Starting with some Minecraft Monday for my week. Catch me on #Twitch streaming some Stoneblock2. Hope you are having a great Monday 💖
1363261466334306305 :    #Minecraft #livestream Monday 10AM MT, come hang out on #twitch link in bio and will post at the start of my stream!
1371236823121399809 :    Schlatt said he and Connor ran a business in another server, Quackity had canon PTSD of Techno from Minecraft Monday. So what we can conclude is that Schlatt and Connor
1398773118928408577 :    people mentioning minecraft monday is bringing me back to watching vik and getting so pissed when techno used /top to beat him because I had no clue who techno was and
1398775542124486656 :    People who got really really mad on mcc today probably wouldn't survive watching the previous mcc or minecraft monday.
1399408095055343617 :    #MINECRAFT #MONDAY TIME!!
1412233747697598464 :    Playing some MineCraft with my nephew. How's everyone's Monday been so far? #monday
1427345246912974870 :    Running a little late, but Minecraft Monday will be going live imminently! Do tune in! It's not to be missed #Minecraft #monday #gameshow
1429221700978184195 :    Thanks everyone who came out to my #Minecraft stream tonight; I had the absolute BEST time (despite the constant fear of unseen mobs and getting HORRENDOUSLY lost) and
1429593841015967745 :    Hey all! So as most know I recently got tested positive for Covid and delayed my streams. But now I'm feeling a little better. So we will be back at it tomorrow with ou
1429924186101895175 :    What is up everyone I am no longer #DownWithTheSickness. So its time to get down with this weeks schedules….[NEWLINE][NEWLINE]Monday- CANCELED [NEWLINE][NEWLINE] Wednes
Enter your search query: stuttgart silvester
1344936702369017859 :    Wir leben in einem Deutschland, in dem an Silvester #Covidioten in #Stuttgart bis ins neue Jahr hinein demonstrieren dürfen. [NEWLINE][NEWLINE]Wir leben in einem Deutsc
1345509982398251008 :    Das Maß ist nach Silvester endgültig voll - Demokratie hat ihre Grenzen! Zeit jeden Corona-Leugner und #Querdenker0711 vor die Tore vor #Stuttgart zu treiben! Wir lasse
Enter your search query: berlin bahn
1345800756021108738 :    Heute in der Bahn wieder einen netten Mitmenschen getroffen. Jedoch war er mir nicht ganz geheuer, als er mir ein Gram Schore angeboten hat. Ich musste dankend ablehnen
1346476380565897219 :    Ich fahre gerade leider (aus einem triftigen Grund) mit einer S-Bahn aus Berlin raus. Sagen wir so: Mehr Homeoffice wäre möglich und nötig.
1347972850863529987 :    Deutschland Wirtschaft Läuft gut Unternehmer führen gut trotzdem wollen sich Politiker immer mehr einmischen jetzt Grün mit HomeOffice Pflicht stellt Grün Geräte Sicher
1349000173322706946 :    Unterschied einer S-Bahn-Station: vor den 6 km: Inzidenz Berlin-Köpenick: 220. Schulen und Kitas zu. Hinter den 6 km: Inzidenz Dahme-Spreewald: 304. Abschlussklassen in
1350792695132921856 :    Funktioniert bei euch die #CoronaApp noch? Hatte seit Weihnachten keine Risikobegnung mehr. Meine Freunde auch nicht.[NEWLINE]Und ich fahre täglich mit der Bahn quer du
1352934211641077760 :    Schon intrressant, wie viele Menschen aufgrund eines triftigen Anlasses an einem Samstag Mittag mit der S-Bahn unterwegs sind. #CoronaVirusDE #lockdown2021 #Berlin
1361944955103698944 :    Crazy, in der Berliner U-Bahn ist die #Covid Pandemie schon vorbei. Unglaublich wie voll die ist und wie viele  Menschen wieder zur Arbeit pendeln... #berlin
1363851241873428480 :    Es ist Montag, 15 Uhr, die Berliner U-Bahn ist VOLL. So voll wie vor Corona. Man kann keinen Abstand halten.[NEWLINE][NEWLINE]Wie kann das „harter“ Lockdown sein? Wie s
1372453498470723584 :    Covid19 Varianten verbreiten sich. Die S Bahn Berlin hilft fleissig mit und hat Fahrkarten kontrollen wieder aufgenommen, damits schneller geht #CoronaVirusDE
1374750711301484547 :    Ich habe die Arbeit und die Kollegen während der Quarantäne vermisst.[NEWLINE]Andere Dinge wieder gar nicht...:[NEWLINE]Penisnasen oder Maskenlosen in Bahn und  Bahnhof
1375159963094552583 :    Pandemie[NEWLINE]Ein Reisender kommt aus Spanien, sein Flug nach Berlin wurde Teil Storniert, dieser endet in Dortmund. Von da aus geht´s in die U-Bahn, Fernzug mit Ums
1376566683855376388 :    An April sollen in Berlin geflüchtete geimpft werden! #Impfdesaster #NieMehrCDU #Coronakrise [NEWLINE]Macht total Sinn! Die die das Geld verdienen und mit Bus und Bahn
1377326647146393607 :    Könnt ihr euch noch an Berlin zu Ostern erinnern? So mit tausenden Touristen? Die ihren Rucksack auf dem Bauch tragen, weil sie Angst haben, in der U-Bahn ausgeraubt zu
```



```
Enter your search query: malaria covid
1344789923556175878 :    i was in and out of over 100 different locations in the space of 3 months during covid lockdown buh God no let common malaria see us sef..
1345296698722148352 :    The goal is to prevent the rise of CoVID-19 to reach levels of malaria and HIV related deaths. But no "you are giving SARS-CoV-2 too much
1345383461574610946 :    Nick Searcy had the best line in a detective series that I think I have ever heard. He was describing a criminal they were pursuing. "He's
1345412318201401344 :    "Tuberculosis still kills 1.5 million people a year, AIDS 700,000, malaria 400,000, and so on, and we barely bat an eye. When COVID-19 sto
1345452565908090883 :    YT people thinking they're resistant to Covid like malaria didn't almost wipe them out.
1345463941154172928 :    Me seeing people testing positive for COVID and I'm fretting  cos the malaria no gree go🙄
1345477945234747393 :    If globalization will be criticized for the introduction of covid-19 pandemic to us, then what should take the blame of malaria disease th
1345509224701440004 :    Why is COVID-19 an epidemic[NEWLINE]- cause one should not proscribe antibiotics against regular cold[NEWLINE]- cause you should not put p
1345813541530566659 :    Has anybody noticed that Widal results comes out highly reactive in patients with Covid 19? So some people tell them it's Malaria/typhoid
1345820835764973572 :    Malaria is more deadly than COVID-19[NEWLINE][NEWLINE]How many times did this ring in my head?[NEWLINE]#loveworld[NEWLINE]#PastorChris [NE
1345832852026494979 :    A lot of people actually have covid but they will be disguising as malaria 😒
1345843494829645825 :    Ebola & Malaria were all eradicated through science if vaccination can save us from current affliction we're suffering, this deadly harass
1345861815868268547 :    People have been asking why Covid does not seem to be such a big issue in many African countries. My hypothesis is that in many countries
1345872180106702848 :    Nigerians just grew huge pair of balls and decided to be throwing parties like covid 19 went on recess. Those of you on the TL claiming to
1346089360521887744 :    Today's mini-shift[NEWLINE]Severe COVID req ICU[NEWLINE]Severe COVID req ICU[NEWLINE]Fracture mid-shaft femur[NEWLINE]Non traumatic SAH se
1346198670757150720 :    The world should not receive COVID-19 vaccines from China. It's like asking the mosquito to treat malaria. 🙂
1346199914179657731 :    People that died from COVID died from other conditions and were told to label them COVID deaths. COVID is survivable under normal circumst
1346218929899888647 :    Please the malaria that you think you have may not be malaria ooo! Go and get a Covid test. [NEWLINE][NEWLINE]Thanks.
1346219318418268171 :    The vast majority of people in Lagos will either get the new strain of Covid or the revamped strange Malaria
1346224805717159941 :    Hospitals are filled to the brim and you don't believe Covid is real?[NEWLINE]So everyone somehow has Malaria + cough?[NEWLINE]Isokay.
```

- The information need **„show me tweets of people who talk about the side effects of malaria and COVID vaccines“** is a phrase query
- We need **3 or 4 words** to describe the information need → e.g. **malaria covid vaccine effect**
- However our query engine is currently build to allow phrase queries only for 2 words, therefore the query **„malaria covid“** is the best we can do with this implementation.
- But it is easy to extend the query engine to allow phrase queries with more than 2 words, by consecutively intersecting the postings lists of the words.
  - E.g. r1 = query(term1, term2)
  - Intersect r1 with query(term3) etc.