Programmming Task 1:

```
#!/usr/bin/env python3

from collections import defaultdict

name = ""
inv_index = defaultdict(list)
postings = defaultdict(set)
stop_words = {'a', 'an', 'and', 'are', 'as', 'at', 'be', 'by', 'for', 'from',
              'has', 'he', 'in', 'is', 'it', 'its', 'of', 'on', 'that', 'the',
              'to', 'was', 'were', 'will', 'with'}


def normalize(term):
    """
    normalize word and remove useless stuff
    """
    return term.lower().\
        replace(":", "").\
        replace(";", "").\
        replace(".", "").\
        replace(",", "").\
        replace("/", "").\
        replace("#", "")


def index(filename):
    """
    indexes a given file and saves terms to a posting and non-positional
inverted index
    """
    global name
    name = filename
    try:
        # open file
        with open(filename, "r") as file:
            docID = 0
            # iterate over each line in file
            for line in file:
                # split them to list of terms
                tweet = line.split()

                for term in tweet:
                    # remove clutter
                    term = normalize(term)
                    # check if term is in stop words to save some memory
                    if term in stop_words:
                        continue
                    # add docID
                    postings[term].add(docID)
                    # update inv_index
                    # we use the term as pointer, because python does not
                    # support pointers, and storing int by indexes will have
an
                    # massive overhead
                    inv_index[term] = (len(postings[term]), term)
                # increase line number counter
                docID += 1
                # this is for displaying a progress while indexing
                if docID % 10000 == 0:
                    print(str(int(docID / 10000)) + " %")
    except FileNotFoundError as e:
        raise SystemExit("Could not open file: " + str(e))
    return


def getLines(lines):
    result = ""
    try:
        # open file
        with open(name, "r") as file:
            # iterate over the lines and print the lines, which match the
terms
            for i, line in enumerate(file):
                if i in lines:
                    result += str(i) + "\t" + line
    except FileNotFoundError as e:
        raise SystemExit("Could not open file: " + str(e))
    return result


def query(term1, term2=""):
    """
    you can query your search terms. If only one term given it only searches
    for one, otherwise they both have to exist in the tweet
    """
    lines = []
    # remove clutter
    term1 = normalize(term1)
    term2 = normalize(term2)
    # if only one term given look for it
    if term1 in inv_index and not term2:
        (postings_len, postings_pointer) = inv_index[term1]
        # the sorted document_id list out of the postings_list
        lines = sorted(list(postings[postings_pointer]))
    # if two terms are given look for both
    elif term2 and term2 in inv_index:
        (postings_len, postings_pointer) = inv_index[term1]
        # the sorted document_id list out of the postings_list
        lines1 = sorted(list(postings[postings_pointer]))
        (postings_len, postings_pointer) = inv_index[term2]
        # the sorted document_id list out of the postings_list
        lines2 = sorted(list(postings[postings_pointer]))
        # init of iterators
        listiter1 = iter(lines1)
        listiter2 = iter(lines2)
        tmp1 = -1
        tmp2 = -1
        # and intersect the lists to see which lines match both terms
        # if the have the same lines, it will be added to 'lines'
        while True:
            try:
                # like discused in the lesson
                if tmp1 <= tmp2:
                    tmp1 = next(listiter1)
                else:
                    tmp2 = next(listiter2)
                if tmp1 == tmp2:
                    lines.append(tmp1)
            except StopIteration:
                break
    else:
        print("nothing found")
    # we could end here, but we want to get the lines from the file
    return getLines(lines)


if __name__ == '__main__':
    index("tweets")
    print("finished indexing")
    # print(query("geldern"))
    print(query("stuttgart", "bahn"))
```

Ausgabe für das Programm:

```
12310  2016-05-06 01:39:52 +0200  728368645106221056  @Jobs_Stuttgart_
Jobbörse Stuttgart  🔍 Deutsche Bahn AG sucht Quereinsteiger zum Wagenmeister
im Güterverkehr (w/m) 🔎[NEWLINE] #Jobs
#Stuttgart[NEWLINE]https://t.co/9O01C5rWJj
81065  2016-06-04 00:41:04 +0200  738863098908528640  @chrispillennews  Chris
Pillen  Die Bahn hat Kosten und Risiken des Projekts Stuttgart 21 nach rund
drei Jahren nochmals genau überprüft
508713  2016-06-04 01:12:15 +0200  738870945083838467  @Der_Finanzfuchs
Inffox.com  Umstrittenes Bahnprojekt: Stuttgart 21 wird noch teurer und kommt
später: Die Bahn hat Kosten und Risiken des... https://t.co/EC4jdr0oU4
633778  2016-06-03 13:20:32 +0200  738691836131299328  @airjibeer  airjibeer
die #Bahn behauptet ja, daß sie in Stuttgart 60 km Tunnel inkl 3 Bahnhöfe für
6,8 Mrd baut.. der 57 km lange #Gotthard-Tunnel kostete 11 Mrd
750847  2016-06-02 09:52:30 +0200  738277096368295936  @GLiAdM  PinSel  Unter
#SPD -Finanzminister nicht möglich:  #S21: Land will Schadenersatz von #Bahn
prüfen - Stuttgart 21 - https://t.co/rbw6lQX6Jg
820750  2016-06-03 18:15:25 +0200  738766043871612928  @SvenBoell  Sven Böll
Achtung Überraschung: #Stuttgart 21 wird 500 Millionen Euro teurer - und
später fertig https://t.co/G3xuRMreLs #Bahn
```