IRTM
Wi 20/21
Assigment 1

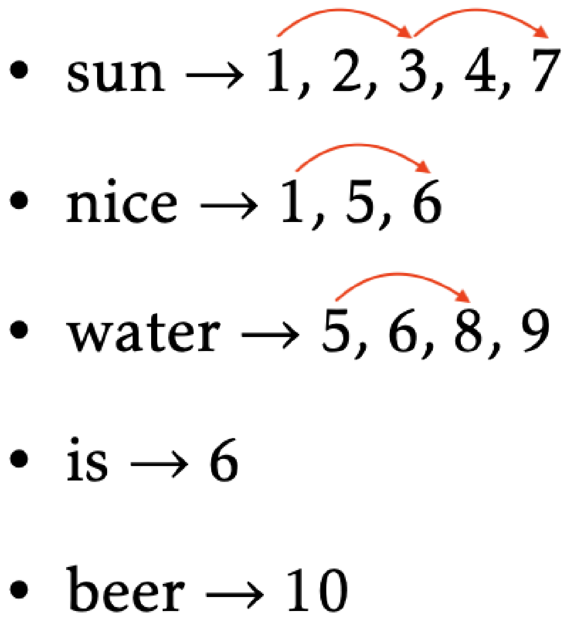
Alberto Saponaro - saponaroalberto97@gmail.com
Walter Väth - walter.vaeth@gmail.com
Chong Shen - st143575@stud.uni-stuttgart.de
Xin Pang - Email

Pen and Paper Task 1

Subtask A

- sun → 1, 2, 3, 4, 7
- nice → 1, 5, 6
- water → 5, 6, 8, 9
- is → 6
- beer → 10

Subtask B


- sun → 1, 2, 3, 4, 7
- nice → 1, 5, 6
- water → 5, 6, 8, 9
- is → 6
- beer → 10

Example query: *nice AND is*
1. Comparisons without skip pointers: 1 & 6, 5 & 6, 6 & 6 ⇒ 3 comparisons
2. Comparisons with skip pointers: 1 & 6, 6 & 6 ⇒ 2 comparisons

Without skip pointers we must compare the terms step by step, although 5 in **nice** is still smaller than 6 in **is**. With skip pointers we can skip the 5 in **nice** and directly go to the 6 in **nice**.

IRTM
Wi 20/21
Assigment 1

Alberto Saponaro - saponaroalberto97@gmail.com
Walter Väth - walter.vaeth@gmail.com
Chong Shen - st143575@stud.uni-stuttgart.de
Xin Pang - Email

Pen and Paper Task 2

```
tokenize(text: string):
    token = ''
    list_of_tokens = []

    for char in text:
        if char == ' ':  #whitespace
            list_of_tokens.add(token)
            token = '' #empty string
        if char is a symbol:
            list_of_tokens.add(token)
            list_of_token.add(char)
            token = '' #empty string
        else:
            token += char

    return list_of_tokens
```

Alberto Saponaro - saponaroalberto97@gmail.com
Walter Väth - walter.vaeth@gmail.com
Chong Shen - st143575@stud.uni-stuttgart.de
Xin Pang - Email

Pen and Paper Task 3

Query: *Gates /2 Microsoft*

(Gates, 4): [{1:[3], 2:[6], 3:[2, 17], 4:[1]}]
(Microsoft, 4): [{1:[1], 2:[1, 21], 3:[3], 5:[16, 22, 51]}]

cross product = { 1:[(3,1)], 2:[(6,1), (6,21)], 3:[(2,3), (17,3)], 4:[(1,16), (1,22), (1,51)] }

From all tuples in the cross product, the tuples (3,1) and (2,3) fulfill the query's condition abs(tuple[1]-tuple[0]) $\leq$ 2. So the answer is: document 1, document 3.

Alberto Saponaro - saponaroalberto97@gmail.com
Walter Väth - walter.vaeth@gmail.com
Chong Shen - st143575@stud.uni-stuttgart.de
Xin Pang - Email

Programming Task 1

```
import csv, re, nltk

def index(filename: str ='code/postillon.csv'):
    index = {}
    dictionary = {}
    postings_lists = []

    tokenizer = nltk.RegexpTokenizer(r"\w+")

    with open(filename, 'r') as file:
        reader = csv.reader(file, delimiter = '\t')
        postings = []

        #iterate through each row of the table
        for row in reader:
            (doc_id, url, pub_date, title, news_text) = row

            #tokenize and normalize news text
            #this procedure will remove symbols like !?() etc.
            #the set data structure will remove all duplicates
            news_text_norm = set(tokenizer.tokenize(news_text.lower()))

            #generate postings
            #iterate through each term
            for term in news_text_norm:
                postings.append((term, doc_id))

        #sort postings
        postings = sorted(postings[1:], key = lambda tup: tup[0])


        post_id = 0
        post_size = 0
        #iterate through postings
        for posting in postings:
            term, doc_id = posting

            if term not in dictionary:
                #upate the dictionary with the new term
                #initialize the postings size
                #save the postings id,
                #witch is the position of the postings list
                #into the postings lists
                dictionary.update({term: [post_size+1, post_id]})

                #initialize a new postings list
                postings_lists.append([doc_id])

                #update postings id
                post_id +=1
            else:
                #update size of posting
                dictionary[term][0] += 1
```

Alberto Saponaro - saponaroalberto97@gmail.com
Walter Väth - walter.vaeth@gmail.com
Chong Shen - st143575@stud.uni-stuttgart.de
Xin Pang - Email

```

                #update postings list
                postings_lists[-1].append(doc_id)

    return dictionary, postings_lists


def query(term_1: str, term_2: str = ''):
    if term_2 == '':
        #only term_1
        pass
    else:
        #term_1 AND term_2
        pass
    pass


if __name__ == "__main__":
    index()
```

code/script.py