In [11]:

```
from os import listdir

def fileids(path, ext='txt'):
    return [path+file for file in listdir(path) if file.split(".")[-1] == ext]


# def fileids(path, ext="txt"):
#     for file in listdir(path):
#         if file.split(".")[-1] == ext:
#             fileList.append(path+file)
#         return fileList


fileids("")
```

...

In [12]:

```
def filecontent(file):
    with open(file, encoding="utf-8") as fp:
        content = fp.read()
    return content
```

In [16]:

```
def ngram(term, n=2):
    return [term[i:i+n] for i in range(len(term) - n + 1)]
```

In [17]:

```
# 원시어절, 구두점, 불용어, 길이, 빈도, 정규식, 품사(형태소), ngram
from nltk.tokenize import word_tokenize
from konlpy.tag import Komoran
from string import punctuation
import re
from collections import defaultdict

ma = Komoran()

content = filecontent(fileids("")[-2])

indexTerm = defaultdict(int)
indexTerm1 = defaultdict(int)
indexTerm2 = defaultdict(int)
indexTerm3 = defaultdict(int)
indexTerm4 = defaultdict(int)
indexTerm5 = defaultdict(int)

for term in content.split():
    indexTerm1[term] += 1

for _ in indexTerm1:
    for t in ma.pos(_):
        indexTerm2[t] += 1
        indexTerm[t] += 1 #원시형태소 = 품사
        if len(t[0]) > 1: #음절 길이로 정규화
            indexTerm3[t[0]] += 1 # 원시형태소
            indexTerm[t[0]] += 1
        if t[1].startswith("N"):
            indexTerm4[t[0]] += 1 #명사
            indexTerm[t[0]] += 1
        for n in ngram(t[0]): #바이그램
            indexTerm5[n] += 1
            indexTerm[n] += 1

print(len(indexTerm1),len(indexTerm2), len(indexTerm3), W
len(indexTerm4), len(indexTerm5), len(indexTerm), W
# min(indexTerm5.values()), max(indexTerm5.values()),
min(indexTerm.values()), max(indexTerm.values()) )

indexTerm
#=> lexicon(controlled Vocabulary)
#=> 빈도(zipf) 정규화 빠져 있음 <- 색인어 도출
#=> indexing
```

Out[17]:

```
defaultdict(int,
            {('이인영', 'NNP'): 2,
             '이인영': 4,
             '이인': 2,
```

```
    '인영': 2,
    (',', 'SP'): 5,
    ('황교안', 'NNP'): 3,
    '황교안': 6,
    '황교': 3,
```

In [19]:

```
for file in fileids(""):
    with open(file, encoding="utf-8") as fp:
        contents = fp.read()
    tokens1 = contents.split() #원시어절
    tokens2 = word_tokenize(contents) #[word_tokenize(content) for token in tokens1] #구두점 분리
    tokens3 = [_ for token in tokens2 for _ in ma.pos(token)] #형태소-품사
#     tokens 1 + tokens2 + tokens3 => Lexicon(controlled vocab)
    tokens4 = [token[0] for token in tokens3]
    tokens5 = [token[0] for token in tokens3 if token[1].startswith("N")]
    tokens6 = [_ for token in tokens4 for _ in ngram(token)]
    #print(content, "Wn")
    #print(len(content.split()))
    print(len(tokens1))
    print(len(tokens1 + tokens2 + tokens3))
    print(len(tokens1 + tokens2 + tokens3 + tokens4 + tokens5 + tokens6))
    print(len(set(tokens1 + tokens2 + tokens3 + tokens4 + tokens5 + tokens6)))
    break
```

```
219
988
1999
648
```

In [21]:

```
ngram("테스트 31231")
```

Out[21]:

```
['테스', '스트', '트 ', ' 3', '31', '12', '23', '31']
```

In [22]:

```
corpus = filecontent(fileids(""))[-2])
#print(corpus)
print(len(corpus))

pattern = dict()

#구두점
pattern1 = re.compile(r"[{0}]".format(re.escape(punctuation)))
corpus = pattern1.sub(" ", corpus)
pattern["punc"] = pattern1
# print(len(corpus))
# corpus

#불용어
pattern2 = re.compile(r"[A-Za-z0-9]{7,}")
corpus = pattern2.sub(" ", corpus)
pattern["stop"] = pattern2
#print(len(corpus))

#이메일
pattern3 = re.compile(r"Ww{2,}@Ww{3,}.(.Ww{2,})+")
pattern3 = re.compile(r"Ww{2,}@(.?Ww{2,})+")
corpus = pattern3.sub("", corpus)
pattern["email"] = pattern3

#도메인
pattern4 = re.compile(r"(.?Ww{2,}){2,}")
corpus = pattern4.sub("", corpus)
pattern["domain"] = pattern4

#한글 이외
pattern5 = re.compile(r"[^가-힣0-9]+")
corpus = pattern5.sub(" ", corpus)
pattern["nonkorean"] = pattern5

#반복되는 공백문자
pattern6 = re.compile(r"Ws{2,}")
corpus = pattern6.sub(" ", corpus)
pattern["whitespace"] = pattern6

'''
id@domain
.com | net => 1차
.kr => 2차

URL 한글

문서가 짧을수록 부가정보가 더 많이 필요

'''
```

```
1032
```

Out[22]:

```
'Wnid@domainWn.com | net => 1차Wn.kr => 2차WnWnURL 한글WnWn문서가 짧을수록 부가정보
가 더 많이 필요WnWn'
```

In [23]:

```
content = filecontent(fileids(""))[-2])

for _ in ["email", "punc", "stop", "whitespace"]:
    content = pattern[_].sub(" ", content)

content
```

Out[23]:

```
'이인영 황교안에 5·18 망언 징계 요구 손잡고 광주 가자 더불어민주당 이인영 원내대표가 16일 오전 국회에서 열린 정책조정회의에서 발언하고 있다 강창광 기자 5·18 민주화운동 기념일을 이틀 앞둔 16일 정치권은 일제히 황교안 자유한국당 대표에게 ‘망언자 징계’를 요구했다 이인영 더불어민주당 원내대표는 이날 국회에서 열린 원내대책회의에 참석해 “황교안 대표는 세 가지를 하고 광주에 가야한다”며 “ 김순례·김진태·이종명 의원에 대한 당 내부 징계절차를 마무리 해야한다”고 말했다 이어 “국회에서의 징계 절차도 마무리 돼야한다 아직 한 걸음도 나가지 못한 채로 어제 국회 윤리특별위원회 간사회의가 끝내 불발됐다”고 덧붙였다 이 원내대표는 마지막 과제로 “재발방지를 위한 법제도를 국회에서 마련해야한다”며 “황교안 대표의 결단을 촉구하며 오늘이라도 매듭짓고 떳떳하게 함께 손 잡고 광주를 찾을 수 있어야 한다”고 주문했다 박지원 민주평화당 의원도 이날 자신의 페이스북에 “망언 의원 징계·퇴출 5·18 진상조사위원으로 적합한 인사를 추천하고 와야 한다”며 “오실 때는 반드시 숙제를 하고오라”고 적었다 국회 윤리특위 소속 민주당 의원들은 기자회견을 열고 ‘5 18 망언 의원’에 대한 국회 차원의 징계를 서두르자고 요구했다 민주당 윤리특위 간사인 권미혁 의원은 “ 자유한국당이 5 18 망언 의원을 징계할 의사가 있는 건지 아니면 계속 미루려고 꼼수를 부리는 건지 매우 걱정된다”고 말했다 국회법은 윤리특위가 의원 징계를 심사하기 전에 윤리심사자문위원회를 구성해 의견을 듣고 이를 존중하라고 규정하고 있다 윤리심사자문위는 자유한국당·바른미래당 추천 위원들이 자문위원장을 맡은 민주당 추천 위원의 자격 문제 등을 들어 심의를 거부하면서 파행을 빚고 있다 민주당은 ‘자문위 의견 없음’으로 간주해 윤리특위의 징계 절차를 진행하자는 입장이고 자유한국당과 바른미래당은 국회법 절차상 자문위의 의견이 있어야 한다고 맞서고 있다 김원철 기자 '
```

In [24]:

```
temp = "news.naver.com"
re.sub(r"(.?Ww{2,}){2,}","", temp)
re.sub(r"(https?Ww{2,}){2,}","", temp)
```

Out[24]:

```
'news.naver.com'
```

Kobill..

In [25]:

```
from collections import defaultdict
from konlpy.corpus import kobill

collection = list()
for filename in kobill.fileids():
    collection.append([filename, kobill.open(filename).read()])

print(collection)

globalLexicon = dict() #단어:위치 <- 튜플 no
globalDocument = list()
globalPosting = list()
weightPosting = list()
documentLength = list(0 for _ in range(len(collection)))
globalMaxTF = dict()
globalTotalTF = dict()
```

```
[['1809890.txt', '지방공무원법 일부개정법률안WnWn(정의화의원 대표발의 )WnWn 의 안
Wn 번 호WnWn9890WnWn발의연월일 : 2010. 11. 12. WnWn발 의 자 : 정의화.이명수.
김을동 WnWn이사철.여상규.안규백WnWn황영철.박영아.김정훈WnWn김학송 의원(10인)WnWn제
안이유 및 주요내용WnWn 초등학교 저학년의 경우에도 부모의 따뜻한 사랑과 보살핌이
필요WnWn한 나이이나, 현재 공무원이 자녀를 양육하기 위하여 육아휴직을 할 WnWn수 있
는 자녀의 나이는 만 6세 이하로 되어 있어 초등학교 저학년인 WnWn자녀를 돌보기 위해
서는 해당 부모님은 일자리를 그만 두어야 하고 WnWn이는 곧 출산의욕을 저하시키는 문
제로 이어질 수 있을 것임.WnWn 따라서 육아휴직이 가능한 자녀의 연령을 만 8세 이하
로 개정하려WnWn는 것임(안 제63조제2항제4호).WnWn- 1 -WnWnWx0c법률 제 호Wn
Wn지방공무원법 일부개정법률안WnWn지방공무원법 일부를 다음과 같이 개정한다.WnWn제63
조제2항제4호 중 “만 6세 이하의 초등학교 취학 전 자녀를”을 “만 WnWn8세 이하(취학
중인 경우에는 초등학교 2학년 이하를 말한다)의 자녀를”WnWn로 한다.WnWn부 칙Wn
Wn이 법은 공포한 날부터 시행한다.WnWn- 3 -WnWnWx0c신 ·구조문대비표WnWn현 행
WnWn개 정 안WnWn제63조(휴직) ① (생 략)WnWn제63조(휴직) ① (현행과 같음)WnWn
② 공무원이 다음 각 호의 어WnWn ② ----------WnWn느 하나에 해당하
는 사유로 휴WnWn----------WnWn직을 원하면 임용권자는 휴직WnWn---
----------WnWn을 명할 수 있다. 다만, 제4호WnWn----------.--------
--------WnWn의 경우에는 대통령령으로 정WnWn----------WnWn하는 특
별한 사정이 없으면 휴WnWn----------WnWn직을 명하여야 한다.WnWn--
```

In [26]:

```
N = len(globalDocument)

for term, postingIdx in globalLexicon.items():
    old = postingIdx
    df = 0
    while True:
        if postingIdx == -1:
            break
        df += 1
        postingIdx = globalPosting[postingIdx][2]

    postingIdx = old
    idf1 = idf(df, N)

    while True:
        if postingIdx == -1:
            break

        data = globalPosting[postingIdx]
        postingIdx = data[2]
        tf = doubleTF(data[1], globalMaxTF[data[0]])
        print("{0}-{1}".format(term, globalDocument[data[0]]))
        print("=> {0} * {1} = {2}".format(tf, idf1, tf *idf1))
```

In [27]:

```
from nltk import word_tokenize
from konlpy.tag import Komoran

ma = Komoran()

for docName, docContent in collection:
    docIdx = len(globalDocument)
    globalDocument.append(docName)

    localPosting = defaultdict(int) #단어:위치
    for token in word_tokenize(docContent):
        for term in ma.pos(token):
            if len(term[0]) > 1 and term[1].startswith("N"):
                localPosting[term[0]] += 1
    globalMaxTF[docIdx] = max(localPosting.values())
    globalTotalTF[docIdx] = sum(localPosting.values())
### Local end ###
### skip sorting ###
    for term, freq in localPosting.items():
        if term in globalLexicon.keys():
            termIdx = list(globalLexicon.keys()).index(term)
            postingIdx = len(globalPosting)
            globalPosting.append([docIdx, freq, globalLexicon[term]])
            globalLexicon[term] = postingIdx
        #Lexicon term ? => 기록, 위치를 업데이트
        else:
            termIdx = len(globalLexicon.keys())
            postingIdx = len(globalPosting)
            globalLexicon[term] = postingIdx
            globalPosting.append([docIdx, freq, -1])
        #Posting 기록, 위치도 기록

print(globalDocument)
```

```
['1809890.txt', '1809891.txt', '1809892.txt', '1809893.txt', '1809894.txt', '180989
5.txt', '1809896.txt', '1809897.txt', '1809898.txt', '1809899.txt']
```

In [28]:

```
from math import log10
from math import log

def idf(df, N):
    return log10(N / df)

def doubleTF(freq, maxFreq, alpha=0.5):
    return alpha+(1-alpha)*(freq / maxFreq)

N = len(globalDocument)

print(N)

for term, postingIdx in globalLexicon.items():
    old = postingIdx
    df = 0

    while True:
        if postingIdx == -1:
            break
        df += 1
        postingIdx = globalPosting[postingIdx][2]

    postingIdx = old
    idf1 = idf(df, N)

    while True:
        if postingIdx == -1:
            break

        data = globalPosting[postingIdx]
        postingIdx = data[2]
        tf = doubleTF(data[1], globalMaxTF[data[0]])
        data[1] = tf*idf1
        #0.5 + 0.5(data[1]/globalMaxTF[data[0]] * log(N/df))
        documentLength[data[0]] += (tf*idf1)**2

print(idf1)
```

```
10
1.0
```

In [29]:

```
print(data)
```

```
[9, 0.5121951219512195, -1]
```

In [30]:

```
documentLength, globalPosting
```

Out[30]:

```
([5.548214914470586,
  5.559732825289965,
  8.483731930057433,
  9.023741064859358,
  17.279175272435793,
  20.64748891939265,
  58.9681599039947,
  29.00664707822145,
  30.27732729969868,
  46.86567607365623],
 [[0, 0.13719593726274673, -1],
  [0, 0.23038632081012705, -1],
  [0, 0.053555533504452225, -1],
  [0, 0.058656060504876255, -1],
  [0, 0.057380928754770244, -1],
  [0, 0.2094421098273882, -1],
  [0, 0.05100527000424022, -1],
  [0, 0.04973013825413422, -1],
```

In [31]:

```
from math import log2

query = "국방의 의무와 보편적 의무에 대한 의무를 찾아주세요."

QWM = defaultdict(float)

for token in word_tokenize(query):
    for term in ma.pos(token):
        if len(term[0]) > 1 and term[1].startswith("N"):
            QWM[term[0]] += 1

maxfreq = max(QWM.values())
for term, freq in QWM.items():
    df = 1
    if term in globalLexicon:
        postingIdx = globalLexicon[term]
        while True:
            if postingIdx == -1:
                break
#             data = globalPosting[postingIdx]
            postingIdx = globalPosting[postingIdx][2]
            df += 1

        QWM[term] = (0.5 + (1-0.5)*(freq/maxfreq)*(log2(N/df)))
```

In [32]:

```
print(QWM)
```

```
defaultdict(<class 'float'>, {'국방': 0.7894942656943676, '의무': 1.368482797083103,
'보편': 1.0})
```

In [33]:

```
from math import sqrt

searchResult = defaultdict(float)
for term, postingIdx in globalLexicon.items():
    while True:
        if postingIdx == -1:
            break

        data = globalPosting[postingIdx]
        searchResult[globalDocument[data[0]]] += (data[1] - QWM[term]) ** 2
        postingIdx = data[2]

for doc, distance in searchResult.items():
    searchResult[doc] = sqrt(distance)
```

In [34]:

```
K = 10
for doc, dist in sorted(searchResult.items(), key=lambda x:x[1])[:K]:
    print(doc,dist, len(collection[globalDocument.index(doc)][1]))
```

```
1809890.txt 2.3554649041050446 4201
1809891.txt 2.3579085701718725 4188
1809892.txt 2.9126846602503047 4929
1809893.txt 3.003954238143344 4152
1809894.txt 4.156822737673065 1097
1809895.txt 4.5439508051246165 1857
1809897.txt 5.389788153999409 4135
1809898.txt 5.5073582257331 4288
1809899.txt 6.909068887318202 8549
1809896.txt 7.735158201615523 8919
```

In [35]:

```
searchResult = defaultdict(float)
for term, weight in QWM.items():
    if term in globalLexicon:
        postingIdx = globalLexicon[term]
        while True:
            if postingIdx == -1:
                break

            data = globalPosting[postingIdx]
            searchResult[globalDocument[data[0]]] += (data[1] - QWM[term])
            postingIdx = data[2]

for doc, distance in searchResult.items():
    searchResult[doc] /= documentLength[globalDocument.index(doc)]
```

In [36]:

```
K = 3
for doc, dist in sorted(searchResult.items(), key=lambda x:x[1], reverse=True)[:K]:
    print(doc,dist, len(collection[globalDocument.index(doc)][1]))
```

```
1809890.txt 4.931630475658667 4201
1809891.txt 4.924964325560612 4188
1809892.txt 3.885075872429039 4929
```