HOMEWORK 4

SUBMITTED BY:-
PRAJNA BHANDARY

To run the program:
Input:
```
$ python3 homework4 cat dog mouse
```
Or you can give the weights as
```
$ python3 homework4 Wt 0.3 dog 0.4 cat 0.3 bird
```

Output:
The top 10 documents that have the word and their corresponding scores displayed

Approach

The posting.txt and index.txt files as generated in the previous homework 3 were used for this homework.

Following are the steps that were followed for the logic of this document retrieval process

1) If the term weights are given then we will map that to be the word query score for that word to be the weight provided.
2) The words must not contain unwanted escape characters.
3) Read the output files of homework 3 output. First consider the postings.txt file and stop when the word is encountered. Read the document weights.
4) Read the index.txt file and stop when the word is encountered and make a note of the term's frequency and start position of the file.
5) Consider all the items of the words given and update all relevant documents with the calculated weight taking query term factor into account. Add all the relavant documents.
6) Repeat the steps 3 to 5 for all the words in the query.
7) Filter and sort files based on their score. Exclude the documents that have the term score 0.
8) The output is displayed as the top 10 relevant documents.

Algorithm

The algorithm used in my program is a binary search for the index file in the disk approach. It searches every line for the query term until it finds the word that is queried. I believe the word we are looking for could be reached early in a document so instead of storing the whole document in memory and then searching for the word is not very effective.

Data Structure

Instead of using arrays to tabulate the document-query similarity scores, i have used hash table to keep a track of the weights for the terms. Also, if the document does not have the word then it has the value 0. If the document has the words in the query then everytime the document is encountered the score is updated.

The weights of the terms are considered so that the one with the least weight gets the least priority while searching for the terms and fetching the relavant documents. In order for the program to work for the query term weights the input should be given in the format. It should be followed by “Wt”.

```
$python3 homewor4.py Wt 0.3 dog 0.4 cat 0.3 bird
```

Complexity

N = number of documents to scan
Q = number of query terms
L = list of the documents the term encountered

O(N+Q+L)

Output:

```
$python3 homework4.py diet

o/p:
018.html 0.01914214327070349
263.html 0.0035424879247857913
009.html 0.003293748417721357
252.html 0.0026344566649044716
050.html 0.001918515932299721
152.html 0.0008378209910435484
328.html 0.0007835015031900953

$python3 homework4.py international affairs

o/p:
219.html 0.022957879186599604
133.html 0.017822348990989573
161.html 0.015292753518867361
138.html 0.009981344159126517
```

```
205.html 0.009713339998104979
247.html 0.00855741827610827
108.html 0.007745938956822141
125.html 0.007745938956822141
197.html 0.0076170052738020536
229.html 0.007270558530242678
```

```
$ python3 homework4.py Zimbabwe

o/p:
No results
```

```
$ python3 homework4.py computer network

o/p:
060.html 0.02901139287532376
140.html 0.023816716155180995
128.html 0.015480865500867648
156.html 0.012636382376873994
502.html 0.011322628253848928
223.html 0.010960466839325025
164.html 0.010002476316480324
501.html 0.009713484235838524
380.html 0.009668970783775857
038.html 0.008153314701981508
```

```
$ python3 homework4.py hydrotherapy
o/p:
223.html 0.00153593010985382
```

```
$ python3 homework4.py identity theft

o/p:
379.html 0.011160997157592216
380.html 0.0055308062808152245
245.html 0.003647451359624023
301.html 0.0035518629791649113
397.html 0.0012176568280189903
298.html 0.0011589111880713594
328.html 0.0009223252926072686
391.html 0.0008118224022366206
```

```
235.html 0.0007881674399775222
383.html 0.0007123770775408211
```

If you apply the weights to the query terms than the results change slightly because certain words get preference than the others. Consider the following examples we can observe that identoty theft without weights fetch us different results based on the score of the words and after giving weights it fetches us different relavant documents based on score.

```
$ python3 homework4.py Wt 0.7 identity 0.1 theft
```

```
245.html 0.0025532159517368163
301.html 0.0024863040854154377
379.html 0.0011160997157592216
397.html 0.0008523597796132932
298.html 0.0008112378316499515
328.html 0.000645627704825088
391.html 0.0005682756815656344
380.html 0.0005530806280815225
235.html 0.0005517172079842655
383.html 0.0004986639542785748
```

We can also observe in the below queries how different results are fetched when there are term weights given and not given.

```
$ python3 homework4.py Wt 0.7 quite 0.4 life 0.1 function
```

```
300.html 0.0037120711939182503
066.html 0.0033125331984867365
239.html 0.00279047884013637
452.html 0.002321953997841869
305.html 0.00228127286310879
100.html 0.0021535680208991377
237.html 0.002045564307464317
074.html 0.0019659749877197702
306.html 0.0017099273481586616
277.html 0.0016473359642538655
```

```
$ python3 homework4.py quite life function
```

```
300.html 0.00928017798479562б
224.html 0.00843097774534663
042.html 0.008406790419735417
066.html 0.00828133299621684
306.html 0.00709510998024665
239.html 0.006976197100340925
452.html 0.005804884994604672
305.html 0.005703182157771975
100.html 0.005383920052247844
068.html 0.005302148226517993
```