

PHASE 1: SPECIFICATION:

- A. Creating a word processing helper cleanToken()
- B. Processing webpages in readDocs()
- C. Creating the inverted index with buildInvertedIndex()
- D. Searching webpages using findQueryMatches()
- E. Combine everything using mySearchEngine()

The programme asks the user to input a search term, then searches the index and displays the number of URLs and URLs where the matching phrase was located. Program exits when user press return/enter as a search query.

Stand by while building index...

Indexed 5 web pages containing 1882 unique terms.

Enter query sentence (RETURN/ENTER to quit):make

Found 5 matching pages

```
{'https://wmich.edu/you.html', 'https://cs.wmich.edu/elise/courses/cs531/assignments-SI19.html', 'https://cs.wmich.edu/~alfuqaha/Spring06/cs5550/projects.html', 'https://www.cs.wmich.edu/~gupta/teaching/cs603/wsnSp04/ClassPolicies.html', 'https://www.cs.wmich.edu/gupta/teaching/cs5950/5950F23PGSweb/TopicsCovered%20Pro gGradStu.html'}
```

Enter query sentence (RETURN/ENTER to quit):make -students

Found 1 matching pages

```
{'https://cs.wmich.edu/elise/courses/cs531/assignments-SI19.html'}
```

Enter query sentence (RETURN/ENTER to quit):non-token

Found 0 matching pages

```
{}
```

Enter query sentence (RETURN/ENTER to quit):

All Done!

Testing the search engine using the following steps:

- A. Simple query, single query with uppercase, single query with punctuations
- B. Compound query with '+' , '-' prefixes for the search term.
- C. Program should list the no of websites and the urls of the
- D. Program repeatedly asks user for input, should terminate when user press enter.

PHASE 2 DESIGN:

2.1 Functions and its purpose

This program consists of 5 functions

- A. Word-processing helper - ***string cleanToken(string token)***
This function accepts <string token> as input and returns clean token(string), removes the leading and trailing punctuations marks but not from inside a token. This function is reused for the purpose of cleaning tokens.
- B. Processing webpages – ***dict<string, Set<string>> readDocs(string dbfile)***
This function accepts the file name as parameter and returns dictionary of forward index(tokens as key & set of urls as value). This function reads the content of the file , process the each line contents into individual token which gets cleaned when passed to cleanToken() and prepares a forward index.
- C. Creating inverted index –
dict<string, Set<string>> buildInveretdIndex(dict<string, Set<string>> docs)
This function takes in forward index prepared by previous function and creates inverted index (dictionary) which contains mapping from words to set of urls where the word can be found.
- D. Searching query in webpages –
Set<string> findQueryMatches(dict<string, Set <string>> index, string query)
This function takes in query string as parameter and returns the set of url strings were the match was found. Query can either be a single term or a compound sequence of multiple terms. Besides the first term the other terms may or may not be preceded by + or – modifier.
- E. One function to encompass all the above methods –
void mySearchEngine(string dbfile)
This function takes file name as input, this function invokes all the above mentioned methods in orderly fashion to create a working search engine. This function provides the capability for users to enter the search query repeatedly until user provides the return to end the program.

2.2 Pseudocode for functions:

1. ***string cleanToken(string token)***
 match ← string should atleast match a single letter
 if no match and length of token is 1
 then discard token
 return "
 convert token to lower case
 clean the token to remove leading and trailing punctuation
 return token
2. ***dict<string, Set<string>> readDocs(string dbfile)***
 file handler ← open the file in read mode
 set next line as true
 initialise forward index empty dictionary
 while next line is true

```

line ← fetch the next line
if line starts with 'https'
    then set the url as key
else if line starts with 'page body'
    then create empty set of urls
else if the line starts with 'endPageBody'
    reset the key and value
else
    spilt the words in line
    for each word
        call cleanToken
        add the cleanToken to set of values
    end for
end if
end while
return forward index
3. dict<string, Set<string>> buildInveretdIndex(dict<string, Set<string>> docs)
   initialise inverted index to an empty dictionary
   for : fetch key,values from forward index
       for each individual tokens inside values:
           present ← try to search in inverted index
           if present
               then add it to the existing key
           else
               add the value under new key
           end if
       end for
   return inverted index
4. Set<string> findQueryMatches(dict<string, Set <string>> index, string query)
   Initialise empty final url set
   for each sub query in query
       If the prefix of sub query is '-'
           Then result set ← try to fetch the query from inverted index
           Final url set ← perform difference b/w result set and final url set
       else if the prefix of sub query is '+'
           Then result set ← try to fetch the query from inverted index
           Final url set ← perform intersection b/w result set and final url set
       else If there is no prefix in sub query
           Then result set ← try to fetch the query from inverted index
           Final url set ← perform union b/w result set and final url set
       end if
   end for
   return final url set
5. void mySearchEngine(string dbfile)
   forward index ← invoke the function readDocs
   inverted index ← invoke the function buildInvertedIndex
   while true
       query ← user input

```

```

        if query is empty
            break from while
        end if
        invoke findQueryMatches & pass inverted index and query
    end while

```

PHASE 3: RISK ANALYSIS

No risk.

PHASE 4: VERIFICATION

In order to successfully run the program the name of the file has to exactly match with the input file name and the input file has to be present in the specified directory. The program has been tested several times in order to check build of inverted index and it successfully returns the url list for the provide input query and program exits when user press enter.

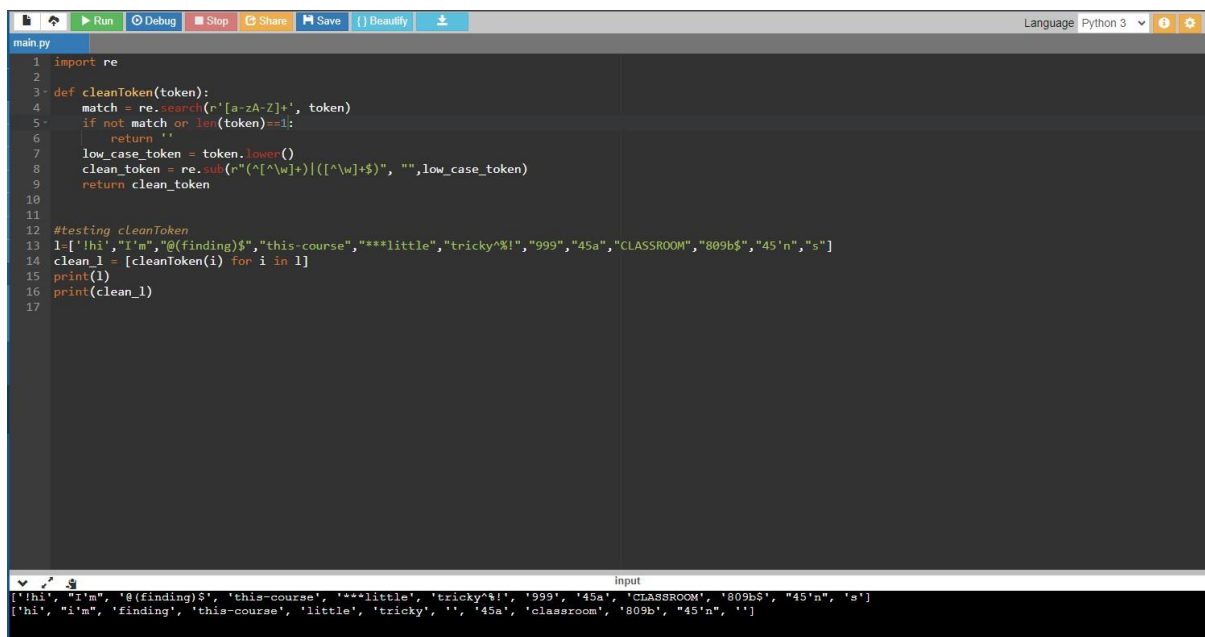
PHASE 5: CODING

The code has been pushed to the git hub , and the link has been provided below.

https://github.com/akgWMU/SearchEngine_pa2-Sathya-Ramesh

PHASE 6: TESTING

Testing of cleanToken()



```

main.py
1 import re
2
3 def cleanToken(token):
4     match = re.search(r'[a-zA-Z]+', token)
5     if not match or len(token)==1:
6         return ''
7     low_case_token = token.lower()
8     clean_token = re.sub(r'(^[\w]+)|([\w]+$)', '', low_case_token)
9     return clean_token
10
11
12 #testing cleanToken
13 l=['!hi', 'I'm', '@(finding)$', 'this-course', '***little', 'tricky^%', '999', '45a', 'CLASSROOM', '809b$', '45'n', 's']
14 clean_l = [cleanToken(i) for i in l]
15 print(l)
16 print(clean_l)
17

```

Input

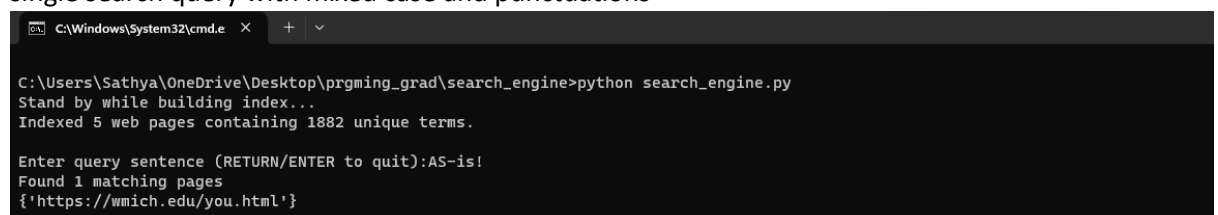
```

['!hi', 'I'm', '@(finding)$', 'this-course', '***little', 'tricky^%', '999', '45a', 'CLASSROOM', '809b$', '45'n', 's']
['!hi', 'I'm', '@(finding)', 'this-course', 'little', 'tricky', '', '45a', 'classroom', '809b', '45'n', '']

```

Testing of search.py

- 1) single search query with mixed case and punctuations



```

C:\Windows\System32\cmd.exe
C:\Users\Sathya\OneDrive\Desktop\prgming_grad\search_engine>python search_engine.py
Stand by while building index...
Indexed 5 web pages containing 1882 unique terms.

Enter query sentence (RETURN/ENTER to quit):AS-is!
Found 1 matching pages
{'https://wmich.edu/you.html'}

```

2) single search queries with no punctuations

```
Enter query sentence (RETURN/ENTER to quit):goals
Found 2 matching pages
{'https://wmich.edu/you.html', 'https://www.cs.wmich.edu/gupta/teaching/cs5950/5950F23PGSweb/TopicsCovered%20ProgGradStu.html'}

Enter query sentence (RETURN/ENTER to quit):make
Found 5 matching pages
{'https://wmich.edu/you.html', 'https://cs.wmich.edu/elise/courses/cs531/assignments-SI19.html', 'https://cs.wmich.edu/~alfuqaha/Spring06/cs5550/projects.html', 'https://www.cs.wmich.edu/~gupta/teaching/cs603/wsn5p04/ClassPolicies.html', 'https://www.cs.wmich.edu/gupta/teaching/cs5950/5950F23PGSweb/TopicsCovered%20ProgGradStu.html'}
```

3) compound query with '-' prefix

```
Enter query sentence (RETURN/ENTER to quit):make -students
Found 1 matching pages
{'https://cs.wmich.edu/elise/courses/cs531/assignments-SI19.html'}
```

4) compound query with no prefix

```
Enter query sentence (RETURN/ENTER to quit):goals tables
Found 2 matching pages
{'https://wmich.edu/you.html', 'https://www.cs.wmich.edu/gupta/teaching/cs5950/5950F23PGSweb/TopicsCovered%20ProgGradStu.html'}
```

5) compound query with mix of prefixes

```
Enter query sentence (RETURN/ENTER to quit):make -tables goals +students
Found 4 matching pages
{'https://cs.wmich.edu/~alfuqaha/Spring06/cs5550/projects.html', 'https://www.cs.wmich.edu/~gupta/teaching/cs603/wsn5p04/ClassPolicies.html', 'https://wmich.edu/you.html', 'https://www.cs.wmich.edu/gupta/teaching/cs5950/5950F23PGSweb/TopicsCovered%20ProgGradStu.html'}
```

6) query which has no results

```
Enter query sentence (RETURN/ENTER to quit):non-token
Found 0 matching pages
{}
```

7) user press enter

```
Enter query sentence (RETURN/ENTER to quit):

All Done!
```

Additional test cases for search engine

```
main.py
1 def findQueryMatches(index,query):
2     #print('----',query)
3     process_query = query.split()
4
5     final_url_set = set()
6     if len(process_query)==1:
7         final_url_set = index.get(query.lower(),'')
8     else:
9         for sub_query in process_query:
10             if sub_query[0] == '-':
11                 srch_query = sub_query[1:].lower()
12                 urls = index.get(srch_query,'')
13                 final_url_set = final_url_set.difference(urls)
14             elif sub_query[0] == '+':
15                 srch_query = sub_query[1:].lower()
16                 urls = index.get(srch_query,'')
17                 final_url_set = final_url_set.intersection(urls)
18             else:
19                 urls = index.get(sub_query.lower(),'')
20                 final_url_set = final_url_set.union(urls)
21     if final_url_set:
22         print(final_url_set)
23     else:
24         print('{}')
25
26 index = {'tasty':{'url1','url2','url3','url4','url5','url6','url7'},
27         'mushrooms':{'url2','url4'},
28         'simple':{'url6','url8'},
29         'cheap':{'url3','url5','url9'}}
30 findQueryMatches(index,'tasty -mushrooms simple +cheap')

input
'url15', 'url13'
```

PHASE 7: REFINING THE PROGRAM

The program consists of a file name which is currently static in nature, any mis spelled file name or if the input file is not present at specified directory, it might throw file exception. The enhancement of program could be made to take the input file name from user also. The program should follow the good practice of closing the file handler all though it's opened in read mode.

PHASE 8: PRODUCTION

Folder containing search.py, sampleWebsiteData.txt & report has been compressed to a zip file and the same has been added to the dropbox submission.