DOKUZ EYLÜL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

E-BOOK ANALYSIS AND REPRESENTATION

Assignment Report

by
Cemil DALAR
2019510025

January 2021
İZMİR

Contents

1 INTRODUCTION

I researched what the tasks requested from us in the project were, and investigated which algorithms I could solve these tasks.

1) How to data mining a website with Python
2) What is HTML programming
3) File operations in Python
4) string operations in Python
5) List operations

2 METHODOLOGY

2.1 Structure of Your Project

My project first asks the user how many books they want to look at, the name of the books they want and how many words in the book they want to see. Then my project first downloads and saves the desired books via the wikibooks site using the requests and beautifulsoup libraries according to the input from the user. My project then opens the downloaded book using file operations and finds the word in the book and how many of those words are, thanks to the loop and if-else statements.Finally,my project finds words that are different between two books . My project finds words that are same between two books.

2.2 Encountered Problems and Solutions

The first problem I encountered ,was punctuation marks in the downloaded book.And my project read these punctuation marks, it was reading the punctuation marks as words.So I deleted the punctuation marks in the downloaded file using the string library. So my project didn't find the punctuation marks as word again.

The second problem I encountered is that my project could not find some books. I used the try expect command and printed print_version at the end of the desired url. In this way, my problem was solved.

The third problem I encountered was that I could not print the words I found while finding different words. I put the words I found in a list, and then I printed the words in that list using a while loop starting from the most found word.

The fourth problem I encountered is that I was having trouble opening and using the books I downloaded. I was able to open the file by typing the "encoding = utf-8" command at the end of the code I used to open the file. And I was able to do operations on that file.

2.3 Improvements

Instead of using loops to sort the numbers of the words found, I used the sorted command so my code got shorter.

3 EXPERIMENTATION

. In the assignment given to us, the user was asked to enter a maximum of 2 books. Therefore, if the user tries to enter more than 2 books, I give an error message. And I want the user to enter the correct number of books.

4 CONCLUSION

I learned how to effectively solve the problems I encountered in this project by benefiting from many different sources. Thanks to this project assignment, I learned to write code in python more effectively and accurately. I have learned many things such as code writing strategies, algorithms, and the use of functions, and will continue to learn.

APPENDIX A: CODE

```
import requests  # Adding requests library
from bs4 import BeautifulSoup  # Adding beautifulsoup library
import string #Adding string library
import operator  #Adding operator library
book_name =""
```

```
number_of_books=int(input("please enter number of books(1 or 2)"))
frequiences=int(input("how many word frequencies you wish to see?"))
stop_words= {'i', 'me', '•','my','·','like','he', 'myself', 'we',
'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll",
"you'd",
             'your', 'yours', 'yourself', 'yourselves', 'he', 'him',
'his', 'himself', 'she', "she's", 'her', 'hers',
             'herself', 'it', "it's", 'its', 'itself', 'they',
'them', 'their', 'theirs', 'themselves', 'what', 'which',
             'who', 'whom', 'this', 'that', "that'll", 'these',
'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been',
             'being', 'have', 'has', 'had', 'having', 'do', 'does',
'did', 'doing', 'a', 'an', 'the', 'and', 'but',
             'if', 'or', 'because', 'as', 'until', 'while', 'of',
'at', 'by', 'for', 'with', 'about', 'against',
             'between', 'into', 'through', 'during', 'before',
'after','I', 'above', 'below', 'to', 'from', 'up', 'down',
             'in', 'out', 'on', 'off', 'over', 'under', 'again',
'further', 'then', 'once', 'here', 'there', 'when',
             'where', 'why', 'how', 'all', 'any', 'both', 'each',
'few', 'more', 'most', 'other', 'some', 'such', 'no',
             'nor', 'not', 'only', 'own', 'same', 'so', 'than',
'too', 'The','very', 's', 't', 'can', 'will', 'just', 'don',
             "don't", 'should', "should've",'de' 'now', 'd', 'll',
'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't",
             'couldn', "couldn't", 'didn', "didn't", 'doesn',
"doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven',
             "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't",
'mustn', "mustn't", 'needn', "needn't", 'shan',
             "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't",
'weren', "weren't", 'won', "won't", 'wouldn',

"wouldn't","1","2","3","4","5","6","7","8","9","0","a","b","c","d","
e","f","g","h","ı","i","j","k","l","m","n","o","p","r","s","t","u","
v","y","z","x"}
def book_scraping(book_name):
    try:#find the book name
        url =
"https://en.wikibooks.org/wiki/"+book_name+"/Print_version"
    except AttributeError as error:
        url =
"https://en.wikibooks.org/wiki/"+book_name+"/Print_Version"
    req = requests.get(url)
    soup = BeautifulSoup(req.content, 'html.parser')
    a = soup.find("div", class_="mw-parser-output")
    b=a.find_all(["p","li","h2"])
    c=[]
    for i in b:
       c.append(i.text)
    with open(book_name+".txt","w",encoding="utf-8") as file:#save
the book
        file.writelines(c)
def word_frequencies_one_book():
   with open(book_name+".txt","r",encoding="utf-8") as file:#opening
the first book
       line=file.read()
       line1=line.lower()
       line2=""
       for i in line1:
```

```
            if i not in string.punctuation:
                line2 += i
        splitwords = line2.split(" ")
        all_words = {}
    print("NO  WORD  FREQ_1")
    for word in splitwords:# for word in splitwords:find words and
numbers of words in the book
        if word not in all_words:
            if word not in stop_words:#separating words from
stop_words
                all_words[word] = 1
        else:
            if word not in stop_words:
                all_words[word] += 1
    all_words_sorted = sorted(all_words.items(), key=lambda x: x[1],
reverse=True)#sorting by the number of words we find
    i=0
    while  i<frequiences:#count the words we find
      print(i+1,str( all_words_sorted[i][0]),str( all_words_sorted[i]
[1]))
      i=i+1

def word_frequencies_two_book():
    with open(book_name + ".txt", "r", encoding="utf-8") as
file:#opening the first book
        line = file.read()
        line1 = line.lower()
        line2 = ""
        for i in line1:
            if i not in string.punctuation:
                line2 += i
        splitwords = line2.split(" ")
        all_words = {}
    print("NO  WORD  FREQ_1")
    for word in splitwords:# for word in splitwords:find words and
numbers of words in the book
        if word not in all_words:
            if word not in stop_words:#separating words from
stop_words
                all_words[word] = 1
        else:
            if word not in stop_words:
                all_words[word] += 1
    all_words_sorted = sorted(all_words.items(), key=lambda x: x[1],
reverse=True)#sorting by the number of words we find
    i = 0
    while i < frequiences:#count the words we find
        print(i + 1, str(all_words_sorted[i][0]),
str(all_words_sorted[i][1]))
        i = i + 1
    with open(book_name2 + ".txt", "r", encoding="utf-8") as
file:#opening the second book
        line = file.read()
        line1 = line.lower()
        line2 = ""
        for i in line1:
            if i not in string.punctuation:
                line2 += i
        splitwords = line2.split(" ")
```

```
        all_words = {}
    print("NO  WORD  FREQ_2")
    for word in splitwords:# for word in splitwords:find words and
numbers of words in the book
        if word not in all_words:
            if word not in stop_words:#separating words from
stop_words
                all_words[word] = 1
        else:
            if word not in stop_words:
                all_words[word] += 1
    all_words_sorted = sorted(all_words.items(), key=lambda x: x[1],
reverse=True)#sorting by the number of words we find
    i = 0
    while i < frequiences:#count the words we find
        print(i + 1, str(all_words_sorted[i][0]),
str(all_words_sorted[i][1]))
        i = i + 1
def distinct():
    print("DISTINCT WORDS")
    print("NO WORD FREQ_1")
    with open(book_name + ".txt", "r", encoding="utf-8") as
file:#opening the first book
        line = file.read()
        line1 = line.lower()
        line2 = ""
        for i in line1:
            if i not in string.punctuation:
                line2 += i
        splitwords = line2.split(" ")
    with open(book_name2 + ".txt", "r", encoding="utf-8") as
file:#opening the second book
        line = file.read()
        line1 = line.lower()
        line2 = ""
        for i in line1:
            if i not in string.punctuation:
                line2 += i
        splitwords2 = line2.split(" ")

    distinct_words={}
    for i in splitwords:#find words in the first book that are
different from the second book
      if i not in splitwords2:
        if i not in stop_words:
            if i not in distinct_words:
                distinct_words[i]=1
            else:
                distinct_words[i]+=1

sorting=sorted(distinct_words.items(),key=operator.itemgetter(1),rev
erse=True)
    j=0
    while j<frequiences:#write words in the first book that are
different from the second book
        print(j + 1, str(sorting[j][0]), str(sorting[j][1]))
        j=j+1
    print("DISTINCT WORDS")
    print("NO WORD FREQ_2")
```

```
    distinct2_words = {}
    for i in splitwords2:#find words in the second book that are
different from the first book
      if i not in splitwords:
        if i not in stop_words:
            if i not in distinct2_words:
                distinct2_words[i] = 1
            else:
                distinct2_words[i] += 1
    sorting2= sorted(distinct2_words.items(),
key=operator.itemgetter(1), reverse=True)
    j = 0
    while j < frequiences:#write words in the second book that are
different from the first book
        print(j + 1, str(sorting2[j][0]), str(sorting2[j][1]))
        j = j + 1
    common1={}
    common2={}
    for i in splitwords:#find common words found in two books
      if i in splitwords2:
       if i not in stop_words:
            if i not in common1:
                common1[i]=1
            else:
                common1[i]+=1
    sorting3 = sorted(common1.items(), key=operator.itemgetter(1),
reverse=True)
    for j in splitwords2:#find common words found in two books
        if j in splitwords:
          if j not in stop_words:
            if j not in common2:
                common2[j] = 1
            else:
                common2[j] += 1
    sorting4 = sorted(common2.items(), key=operator.itemgetter(1),
reverse=True)
    p=0
    print("COMMON WORDS")
    print("NO WORD FREQ_1 FREQ_2 FREQ_SUM")
    while  p<frequiences:#write common words found in two books
        print(str(p+1),str(sorting3[p][0]),str(sorting3[p]
[1]),str(sorting4[p][1]),str(sorting3[p][1]+sorting4[p][1]))
        p=p+1
if number_of_books==1:
    book_name = input("please enter the book name")
    book_name = book_name.replace(" ","_")
    book_name = book_name.replace("'","%27")
    print(book_name)
    book_scraping(book_name)
    word_frequencies_one_book()
elif number_of_books==2:
    book_name = input("please enter the first book name")
    book_name = book_name.replace(" ", "_")
    book_name = book_name.replace("'", "%27")
    book_name2 = input("please enter the second book name")
    book_name2 = book_name2.replace(" ", "_")
    book_name2= book_name2.replace("'", "%27")
    book_scraping(book_name)
    book_scraping(book_name2)
```

```
    word_frequencies_two_book()
    distinct()
elif number_of_books>2:
  print("Error!!.Only You can enter maximum two books.Please enter
the number of books correctly.")
```

APPENDIX B: SCREENSHOTS OF YOUR USE CASES

In this appendix section you Give screenshots of at least three use cases.

```
please enter number of books(1 or 2)2
how many word frequencies you wish to see?20
please enter the first book nameNon-Programmer's Tutorial for Python 2.6
please enter the second book nameNon-Programmer's Tutorial for Python 3
```

NO	WORD	FREQ_1
1		536
2	program	132
3	python	109
4	list	89
5	function	87
6	license	70
7	document	64
8	line	61
9	use	60
10	value	54
11	first	52
12	example	45
13	may	45
14	section	45
15	used	45
16	statement	45
17	next	44
18	version	43
19	string	43
20	name	42

NO	WORD	FREQ_2
1		555
2	python	137
3	program	136
4	function	90
5	list	89
6	use	61
7	file	58
8	line	58
9	first	57
10	name	52
11	used	46
12	next	46
13	run	44
14	string	42
15	statement	42
16	example	41
17	code	41
18	value	41
19	true	41
20	variable	39

DISTINCT WORDS

NO	WORD	FREQ_1
1	document	64
2	sections	28
3	title	26
4	invariant	21
5	modified	19
6	copyright	16
7	texts	15
8	entitled	14
9	distribute	12
10	transparent	11
11	publisher	10
12	provided	9
13	mmc	9
14	published	8
15	history	8
16	word	7
17	holder	7
18	preserve	7
19	exception	6
20	verbatim	6

```
DISTINCT WORDS
NO WORD FREQ_2
1 path 12
2 environment 10
3 arithmetic 6
4 advanced 5
5 pip 5
6 python3 5
7 subprocess 5
8 configuring 4
9 closing 4
10 27 4
11 users

2231 3
12 printing

321 3
13 expressions

331 3
14 ending 3
15 module

14211 3
16 libraries 3
17 panel 3
18 spam 3
19 imported 3
20 installing 2
```

```
COMMON WORDS
NO WORD FREQ_1 FREQ_2 FREQ_SUM
1  536 555 1091
2 program 132 137 269
3 python 109 136 245
4 list 89 90 179
5 function 87 89 176
6 license 70 61 131
7 line 61 58 119
8 use 60 58 118
9 value 54 57 111
10 first 52 52 104
11 example 45 46 91
12 may 45 46 91
13 section 45 44 89
14 used 45 42 87
15 statement 45 42 87
16 next 44 41 85
17 version 43 41 84
18 string 43 41 84
19 name 42 41 83
20 text 41 39 80
```

REFERENCES

They gave information about how to write the code to these sources. I used them.

https://www.ranks.nl/stopwords

https://requests.readthedocs.io/en/master/

https://sites.google.com/site/egitimbilgileri/home/a---python---twisted---qt/03---string-islemleri

https://stackoverflow.com/questions/30042334/attribute-error-list-object-has-no-attribute-split

https://stackoverflow.com/questions/9233027/unicodedecodeerror-charmap-codec-cant-decode-byte-x-in-position-y-character

https://stackoverflow.com/questions/17569679/python-attributeerror-io-textiowrapper-object-has-no-attribute-split

https://hackersandslackers.com/scraping-urls-with-beautifulsoup/

https://caylakyazilimci.com/post/python-dosya-islemleri

https://stackoverflow.com/questions/20837786/changing-the-referer-url-in-python-requests

https://realpython.com/python-requests/

https://www.crummy.com/software/BeautifulSoup/bs4/doc/