DOKUZ EYLÜL UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

E-BOOK ANALYSIS AND REPRESENTATION

Assignment Report

by

Ali Şiyar ARSLAN

January 2021

İZMİR

Contents

1 Introduction 1

2 Methodology 1

2.1 Structure of Your Project 1

2.2 Encountered Problems and Solutions 2

2.3 Improvements 2

3 Experimentation 2

Appendix A: Code 3

Appendix B: Screenshots of your use cases 10

1 INTRODUCTION

This project aims to read the words from the website and determine the number of times which words occur. If the user enters one book, only the number of words in that book, if they enter two books, they sort common and different words from upper to lower. In this project,

2 METHODOLOGY

2.1 Structure of Your Project

BeautifulSoup is a powerful and fast library built for processing HTML or XML files. With this module, we can parse HTML codes in a resource and write bots. Its name is taken from the story told by a turtle in a wonderland. It is the "requests" module, one of the most useful modules used for the Python 3 programming language to interact with the internet. In short, the purpose of the requests module is to use the http(s) protocol. I used BeautifulSoup and request modules to extract data from the website. I asked the user how many books the user wanted to read in the first place. I read the h1 tags whose class is firstHeading and reached the title of the book here. Then I read the div tags whose class is mw-body and I have accessed all the contents of the book from here. I opened a txt file called book and printed and saved all the words read here. I read this file and I shifted all the letters to the small size and put the words in a new list without getting the stopwords. First I separated all the words by space character and put them in a separate list. then I separated it by the characters '(' and '.' and substituted '/' this character because in the re module '(' did not recognize it as a character. Regular expressions (re) It is a definition that is embedded in all programming languages. We can make very complex or very simple searches and replacements in a text, thanks to a pattern consisting of a series of special characters defined by a regular expression. Here are some rules and special syntax. Without regular expressions, it would be necessary to write many if-else ones one after the other. This module can do something that you can do in a few hours for you in seconds. Using the rstrip and lstrip commands, I deleted unwanted characters to the left and right of the word. I created an empty dictionary object and recorded the words as keywords and how many times it was typed in value. Program Create if not found before this word. Add 1 if found this word before .The user is asked if there are any words you want to be sorted specifically. If the user does not enter a special number, this number is accepted as 20. If the user has entered only 1 book, the program lists the words in that book in order of frequency from higher to lower. If the user has entered 2 books, the program lists common and different words among those books.

2.2 Encountered Problems and Solutions

I had to learn the request and beautifulsoup modules to scrapping data from the website. Before I started coding, I could not install these modules correctly on my computer. Against repeated attempts I was constantly getting this error 'You are using pip version 19.2.3, however version 20.2.2 is available. You should consider upgrading via the ‘python -m pip install –upgrade pip 'command.' I learned how to update the pip module after doing some research and now the appropriate environments are ready to start coding. After writing code to a certain level, I noticed that word counts were constantly missing while retrieving data from the website. When I examined the web site carefully from the beginning, I realized that the program only read the paragraphs and the program did not read the headings and some sentences. To fix this, I should have read a more comprehensive tag instead of <p> and I have considered the body tag, but the problem was still not resolved. When I examined the words read, words such as 'print (.....)' and 'print _....' were perceived differently and therefore were missing. After thinking about this for a long time, I realized that I should separate the words not only according to the space character but also according to characters like '/', '_', ')'.

2.3 Improvements

In addition to the program, I made visual improvements by using the '*' character.

3 EXPERIMENTATION

In the first case the user only enters a book and does not indicate that he has entered a specific number of words. In this case, the program takes the site given by the user and reads the words in it and sorts the last 20 words at most in descending order.

In the second case the user simply enters a book and enters a specific number of words. In this case, the program takes the site given by the user, reads the words in it and sorts the words in order from the largest to the lowest in the number of words specified by the most passing user.

In the third case the user enters two books and enters a specific number of words. In this case, the program takes the site given by the user, reads the words in it, and sorts the words in descending order in the number of users who have passed the word in only one book, which is common in both books.

In the fourth case, the user enters two books and does not enter a specific number of words. In this case, the program takes the site given by the user, reads the words in it and sorts the words in only one book, which is common in both books, at the most 20 words, in order from the largest to the smallest.

4 CONCLUSION

While doing this project, I spent most of my time researching and reviewing sample codes. Thanks to this project, I gained Analytical thinking skills. It allowed me to see the relationships between events or situations. It helped him think creatively. It would increase the problem-solving ability. It made me think systematically. Finally, I think it would be better if the project subject was not about a subject we had never learned but aimed to reinforce the subjects we learned before. I also think there may be a mistake in the homework document. Many words such as document, modified, cover, include are also wrong. according to the document only BOOK 1: Non Programmer's Tutorial for Python 2.6, the words to find in this book are also in the other book. I put two sample photos in the screenshots section.

APPENDIX A: CODE

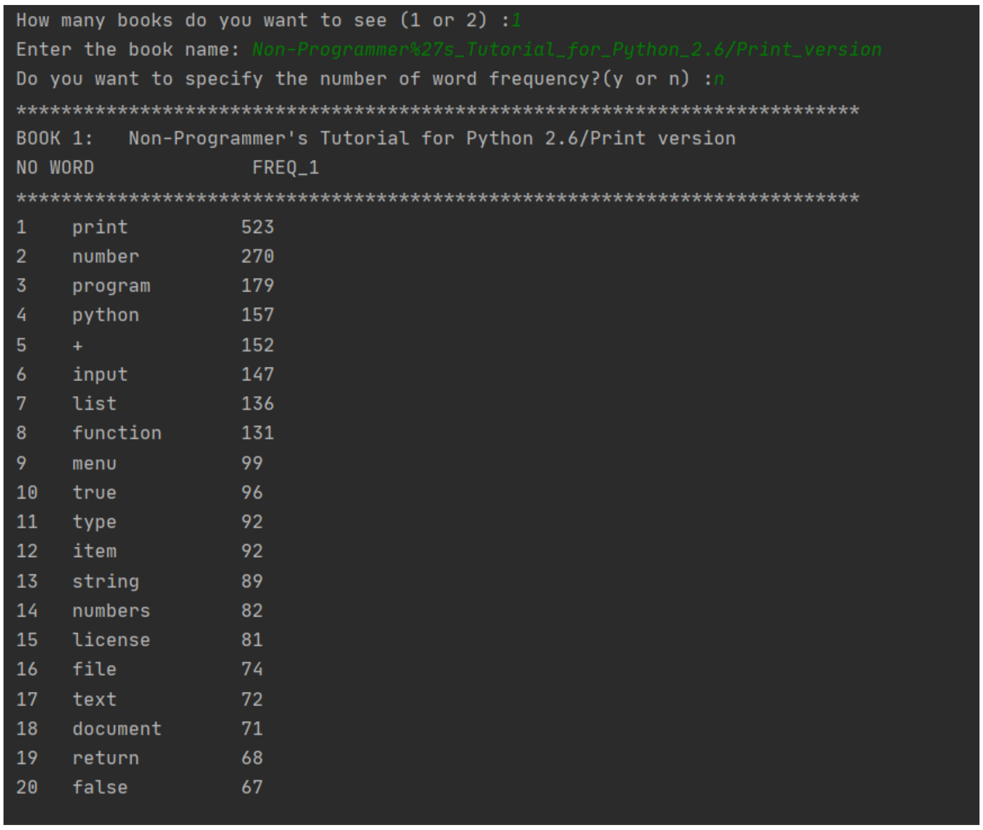
```
import re
import requests
from bs4 import BeautifulSoup
```

```
a=""
b=""
c=""
x=""
y=""
z=""
stop_words=["i","me","my","myself","we","our","ours","ourselves","you","your","yours","yourself","yorselves","←","use
","13","-3",
"he","him","his","himself","she","her","hers","herself","it","its","itself","they",":","it's","","//","$","b","(",")","'"","run","43",
"-23",
"them","their","theirs","themselves","what","which","who","whom","this","that","==","*","i'm","haven't","althouh","@#&
","]","name","[name]","k",
"these","those","am","is","are","was","were","be","been","being","have","has","had","<",">","<=","shouldn't","won't","--
",">=","[x]","line","value","next","first",
"having","do","does","did","doing","a","an","the","and","but","if","or","in","=","also","-
","!=","what's","0","1","2","3","4","5","6","7","8","9","  ","233",
"because","as","until","while","of","at","by","for","with","about","against","between","into","can't","couldn't","#",">>>","/
","'"","%",",","33",
"though","during","before","after","above","below","to","from","up","down","out","on","off","over","under","you,ve","the
re's","##","**",".","that's","243","b23",
"again","further","then","once","here","there","when","where","why","how","all","any","both","each","doesn't","don't","yo
u're","we're","here's","whether","next",
"few","more","most","other","some","such","no","nor","not","only","own","same","so","than","too","very","s","t","can","
will","just","don","should","now"]#stop words list
nofiltered_sentences=[]
filtered_sentences=[]
only_word=[]

count_book=int(input("How many books do you want to see (1 or 2) :"))

if  count_book==1:
   book_name=input("Enter the book name: ")
elif count_book==2:
   book_name = input("Enter the first book name: ")
url =  f"https://en.wikibooks.org/wiki/{book_name}" # Site link

response =  requests.get(url) #  pull  web page.

html_icerigi = response.content  #  get the content of  website.

soup =  BeautifulSoup(html_icerigi,"html.parser") # throw it into the BeautifulSoup object to break up  website..

for i in soup.find_all("h1",{"class":"firstHeading"}):#get the title of the first book from the site
   book_n1=i.text

def main():#printing web page articles to txt
   outfile=open("book.txt","w",encoding="utf-8")# If there is a file named book, open it, otherwise create it
   for i in soup.find_all("div",{"class":"mw-body"}):#get div tags whose class is mw-body
      outfile.write(i.text)
```

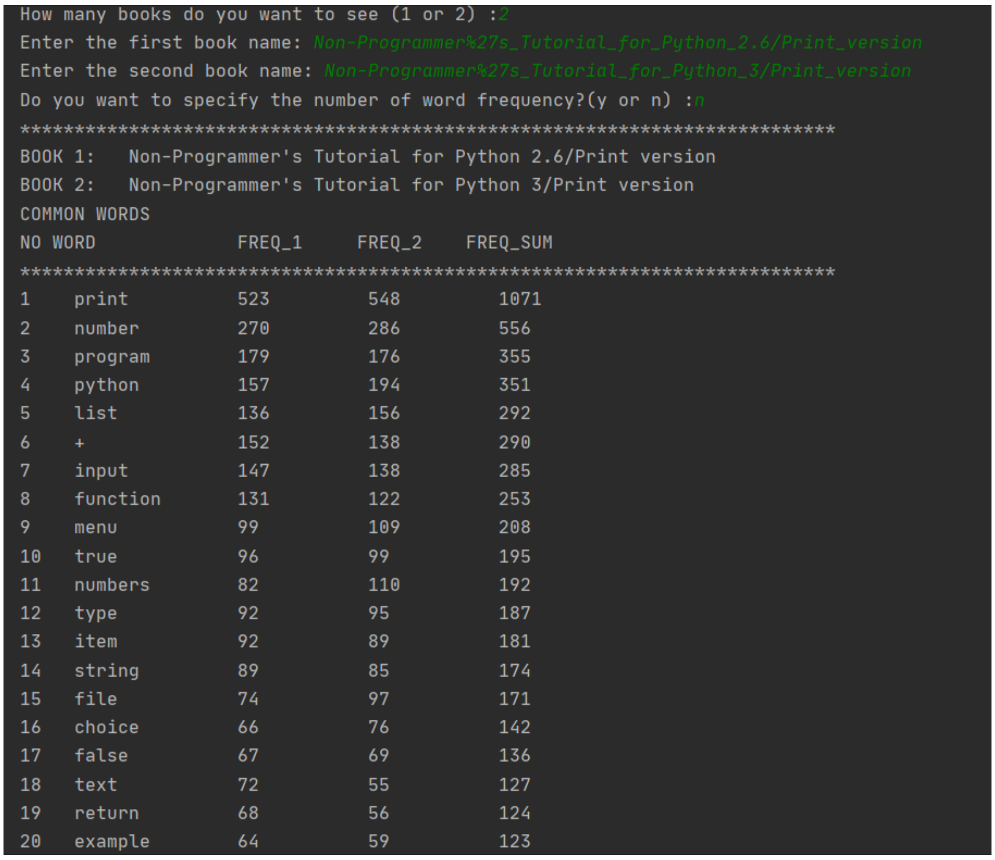
```
    outfile.close()
main()

def main():# reading from txt
    readfile=open("book.txt","r",encoding="utf-8")
    file_contents=readfile.read()
    readfile.close()
    file_contents=file_contents.lower()#make all letters lower size
    nofiltered_sentences.append(file_contents)
main()

for word in nofiltered_sentences:# Splitting text in txt by space character
    only_word=word.split()

for i in only_word:
    a=a+"/"+i

only_word2=a.split('(')# Splitting text in txt by '(' character
for i in only_word2:
    b=b+"/"+i

only_word2=b.split('.')# Splitting text in txt by '.' character
for i in only_word2:
    c=c+"/"+i

only_words=re.split("_| |/", c)# Splitting text in txt by '_',' 'and'/' characters at the same time

for word in only_words:#remove unwanted characters from the left and right of the word
    word=word.rstrip(",")
    word = word.rstrip("!")
    word = word.rstrip("!)")
    word=word.rstrip(".")
    word = word.rstrip(""")
    word = word.rstrip("0")
    word = word.rstrip("1")
    word = word.rstrip("2")
    word = word.rstrip("3")
    word = word.rstrip("4")
    word = word.rstrip("5")
    word = word.rstrip("6")
    word = word.rstrip("7")
    word = word.rstrip("8")
    word = word.rstrip("9")
    word = word.rstrip(">")
    word = word.lstrip("*")
    word=word.rstrip(":")
    word=word.rstrip(";")
    word = word.rstrip(")")
    word = word.rstrip("(")
    word = word.rstrip("?")
    word = word.lstrip("(")
    word = word.lstrip("##")
    word = word.lstrip("<")
    word = word.rstrip("##")
    word = word.lstrip('"')
    word = word.rstrip('"')
    word = word.rstrip(':"')

    if word not in stop_words:#remove stop words
        filtered_sentences.append(word)

word_counter={}

for w in filtered_sentences:#count which word has been found how many times
    if w  not in word_counter.keys():
        word_counter[w]=1#Create if not found before
    else:
        word_counter[w]+=1#add 1 if found before
#
if count_book==2:
    nofiltered_sentences2 = []
    filtered_sentences2 = []
    only_word22 = []



    book_name2 = input("Enter the second book name: ")
    url2 = f"https://en.wikibooks.org/wiki/{book_name2}"  # Site link

    response2 = requests.get(url2)  #  pull  web page.

    html_icerigi2 = response2.content  #  get the content of  website.

    soup2 = BeautifulSoup(html_icerigi2, "html.parser")  # throw it into the BeautifulSoup object to break up  website..

    for i in soup2.find_all("h1", {"class": "firstHeading"}):#get the title of the second book from the site
        book_n2 = i.text

    def main():  #printing web page articles to txt
        outfile2 = open("book2.txt", "w", encoding="utf-8")# If there is a file named book, open it, otherwise create it
        for i in soup2.find_all("div", {"class": "mw-body"}):#get div tags whose class is mw-body
            outfile2.write(i.text)
        outfile2.close()
    main()

    def main():  # reading from txt
        readfile2 = open("book2.txt", "r", encoding="utf-8")
        file_contents2 = readfile2.read()
        readfile2.close()
        file_contents2 = file_contents2.lower()#make all letters lower size
        nofiltered_sentences2.append(file_contents2)
    main()

    for word in nofiltered_sentences2:  # Splitting text in txt by space character
        only_word22 = word.split()

    for i in only_word22:
        x = x + "/" + i

    only_word2 = x.split('(')# Splitting text in txt by '(' character
    for i in only_word2:
        y = y + "/" + i

    only_word2 = y.split('.')# Splitting text in txt by '.' character
    for i in only_word2:
        z = z + "/" + i

    only_words2 = re.split("_| |/", z)# Splitting text in txt by '_',' 'and'/' characters at the same time

    for word in only_words2:  #remove unwanted characters from the left and right of the word
        word = word.rstrip(",")
        word = word.rstrip("!")
        word = word.rstrip("!)")
        word = word.rstrip(".")
        word = word.rstrip(""")
        word = word.rstrip("0")
        word = word.rstrip("1")
        word = word.rstrip("2")
        word = word.rstrip("4")
        word = word.rstrip("5")
        word = word.rstrip("6")
        word = word.rstrip("7")
        word = word.rstrip("8")
        word = word.rstrip("9")
        word = word.rstrip(">")
        word = word.lstrip("*")
        word = word.rstrip(":")
        word = word.rstrip(";")
        word = word.rstrip(")")
        word = word.rstrip("(")
        word = word.rstrip("?")
        word = word.lstrip("(")
        word = word.lstrip("##")
        word = word.lstrip("<")
        word = word.rstrip("##")
        word = word.lstrip("")
        word = word.lstrip(',')
```

```
        word = word.rstrip("")
        word = word.rstrip(':"')
        word = word.rstrip('=')

        if word not in stop_words:#remove stop words
            filtered_sentences2.append(word)

    word_counter2 = {}

    for w in filtered_sentences2:#count which word has been found how many times
        if w not in word_counter2.keys():
            word_counter2[w] = 1#Create if not found before
        else:
            word_counter2[w] += 1#add 1 if found before
#
common_words={}
only_first={}
only_second={}
hm_counter=0
hm=input("Do you want to specify the number of word frequency?(y or n) :")
if hm == "y":
    hmc = int(input("how many word frequency do you want to see :"))
elif hm == "n":
    hmc = 20
if count_book==1:

    s= sorted(word_counter.items(),key=lambda x:x[1],reverse=True)#sort by values in the dictionary in ascending order

    for i in s:#print book1
        if hm_counter<hmc:
            if hm_counter==0:
                print("*****************************************************************************")
                print("BOOK 1:  ",book_n1)
                print('{:<5}{:^33} '.format('NO WORD', ' FREQ_1'))
                print("*****************************************************************************")
            print("{:<5}{:<15}{:<15}".format(hm_counter+1,i[0],i[1]))
            hm_counter+=1
        elif hm_counter==hmc:
            break

elif count_book==2:

    for i,k in word_counter.items():#common words
        for j,l in word_counter2.items():
            if i==j:
                common_words[i]= k+l

    for i,j in word_counter.items():#only words in the first book
        if i not in word_counter2.keys():
            only_first[i]=j

    for i,j in word_counter2.items():#only words in the second book
        if i not in word_counter.keys():
            only_second[i]=j

    s2 = sorted(common_words.items(), key=lambda x: x[1], reverse=True)# common words#sort by values in the dictionary
in ascending order

    for i in s2:#print common words
        if hm_counter<hmc:
            if hm_counter==0:
                print("***************************************************************************")
                print("BOOK 1:  ", book_n1)
                print("BOOK 2:  ", book_n2)
                print("COMMON WORDS")
                print('NO WORD             FREQ_1     FREQ_2    FREQ_SUM')
                print("***************************************************************************")
            print("{:<5}{:<15}{:<12}{:<12}{:<12}".format(hm_counter+1,i[0],word_counter[i[0]],word_counter2[i[0]],i[1]))
            hm_counter+=1
        elif hm_counter==hmc:
            break
    print()
    print()
    s3 = sorted(only_first.items(), key=lambda x: x[1], reverse=True)  #only words in the first book#sort by values in the
dictionary in ascending order
    hm_counter = 0
    for i in s3:#print only words in the first book
        if hm_counter < hmc:
            if hm_counter==0:
                print("***************************************************************************")
                print("BOOK 1:  ", book_n1)
                print("DISTINCT WORDS")
                print('{:<5}{:^33} '.format('NO WORD', ' FREQ_1'))
                print("***************************************************************************")
            print("{:<5}{:<15}{:<15}".format(hm_counter + 1, i[0],i[1]))


            hm_counter += 1
        elif hm_counter == hmc:
            break
    print()
    print()
    s4 = sorted(only_second.items(), key=lambda x: x[1], reverse=True)  #only words in the second book#sort by values in
the dictionary in ascending order
    hm_counter = 0
    for i in s4:#print only words in the second book
        if hm_counter < hmc:
            if hm_counter == 0:
                print("***************************************************************************")
                print("BOOK 2:  ", book_n2)
                print("DISTINCT WORDS")
                print('{:<5}{:^33} '.format('NO WORD', ' FREQ_2'))
                print("****************************************************************************")
            print("{:<5}{:<15}{:<15}".format(hm_counter + 1, i[0], i[1]))

            hm_counter += 1
        elif hm_counter == hmc:
            break
```

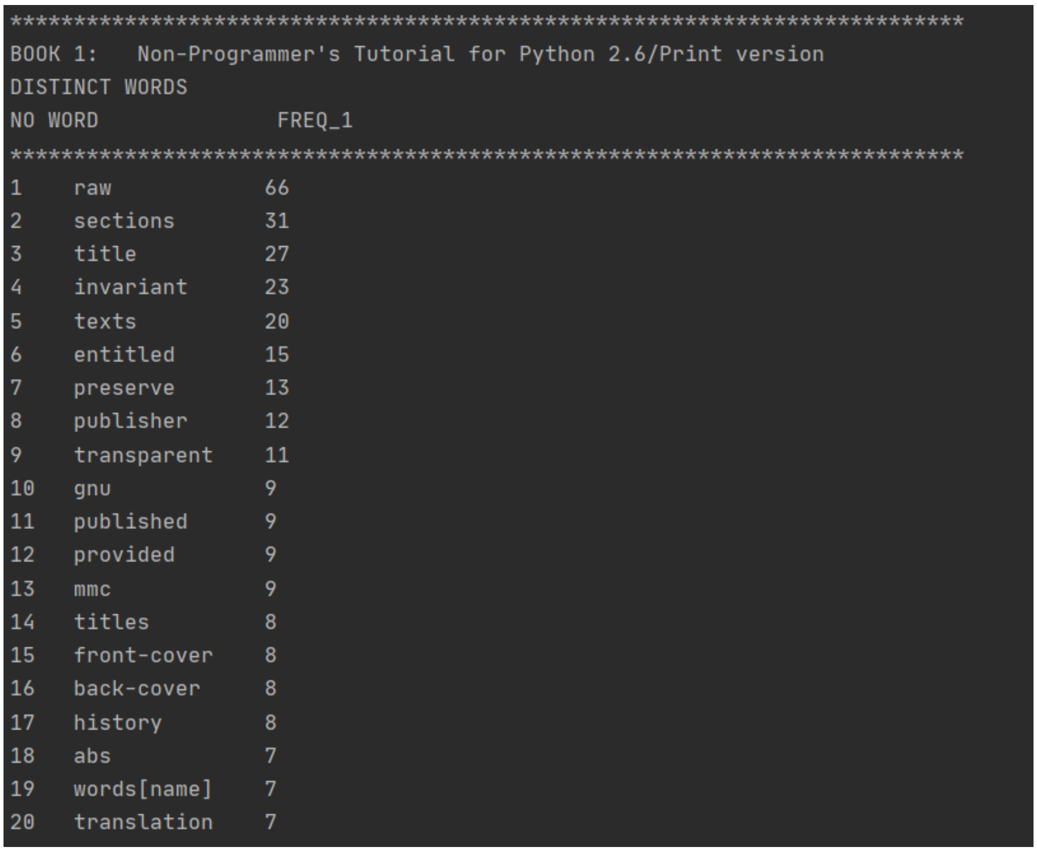
APPENDIX B: SCREENSHOTS OF YOUR USE CASES



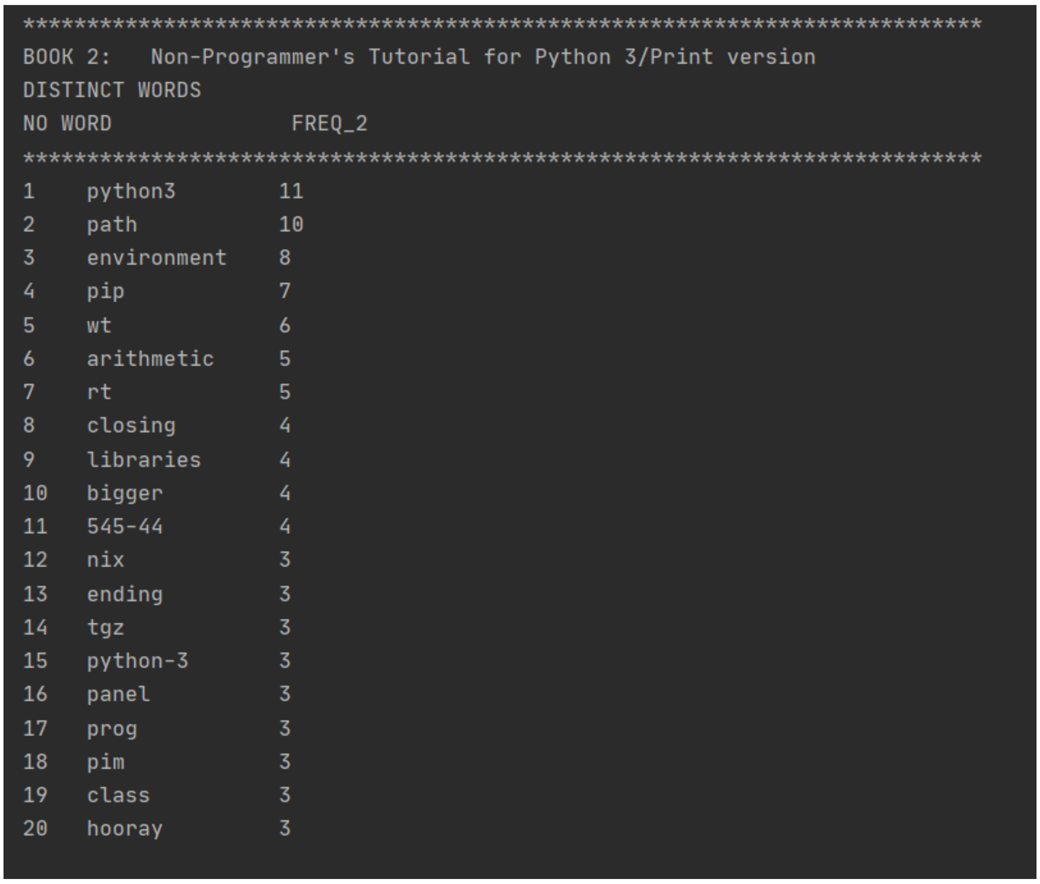
```
How many books do you want to see (1 or 2) :1
Enter the book name: Non-Programmer%27s_Tutorial_for_Python_2.6/Print_version
Do you want to specify the number of word frequency?(y or n) :n
****************************************************************************
BOOK 1:   Non-Programmer's Tutorial for Python 2.6/Print version
NO WORD                FREQ_1
****************************************************************************
1    print          523
2    number         270
3    program        179
4    python         157
5    +              152
6    input          147
7    list           136
8    function       131
9    menu           99
10   true           96
11   type           92
12   item           92
13   string         89
14   numbers        82
15   license        81
16   file           74
17   text           72
18   document       71
19   return         68
20   false          67
```



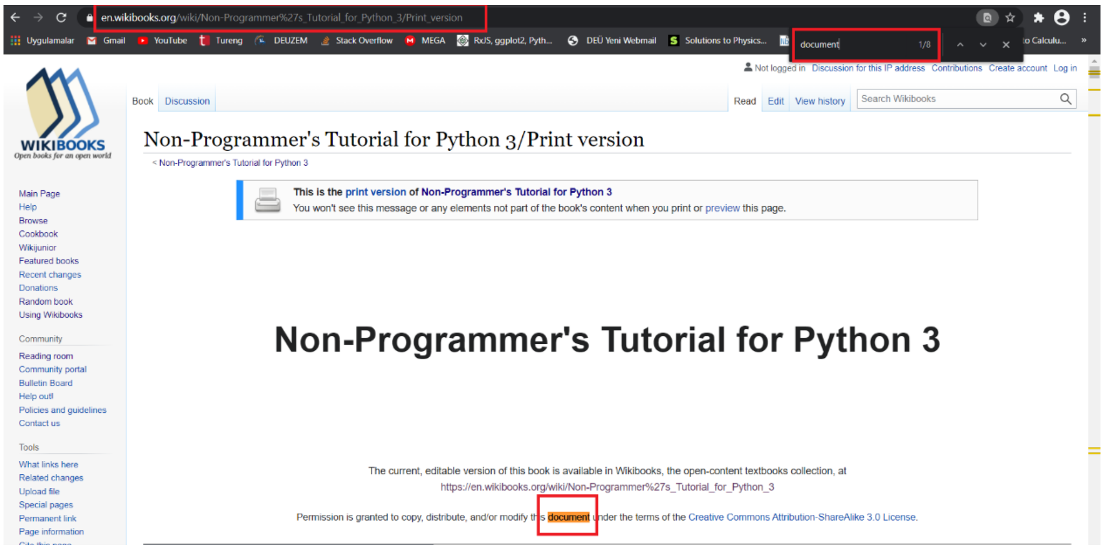
```
How many books do you want to see (1 or 2) :2
Enter the first book name: Non-Programmer%27s_Tutorial_for_Python_2.6/Print_version
Enter the second book name: Non-Programmer%27s_Tutorial_for_Python_3/Print_version
Do you want to specify the number of word frequency?(y or n) :n
***************************************************************************
BOOK 1:   Non-Programmer's Tutorial for Python 2.6/Print version
BOOK 2:   Non-Programmer's Tutorial for Python 3/Print version
COMMON WORDS
NO WORD             FREQ_1     FREQ_2    FREQ_SUM
***************************************************************************
1    print          523         548         1071
2    number         270         286         556
3    program        179         176         355
4    python         157         194         351
5    list           136         156         292
6    +              152         138         290
7    input          147         138         285
8    function       131         122         253
9    menu           99          109         208
10   true           96          99          195
11   numbers        82          110         192
12   type           92          95          187
13   item           92          89          181
14   string         89          85          174
15   file           74          97          171
16   choice         66          76          142
17   false          67          69          136
18   text           72          55          127
19   return         68          56          124
20   example        64          59          123
```



```
***************************************************************************
BOOK 1:   Non-Programmer's Tutorial for Python 2.6/Print version
DISTINCT WORDS
NO WORD                 FREQ_1
***************************************************************************
1    raw            66
2    sections       31
3    title          27
4    invariant      23
5    texts          20
6    entitled       15
7    preserve       13
8    publisher      12
9    transparent    11
10   gnu            9
11   published      9
12   provided       9
13   mmc            9
14   titles         8
15   front-cover    8
16   back-cover     8
17   history        8
18   abs            7
19   words[name]    7
20   translation    7
```



```
***************************************************************************
BOOK 2:   Non-Programmer's Tutorial for Python 3/Print version
DISTINCT WORDS
NO WORD              FREQ_2
***************************************************************************
1    python3        11
2    path           10
3    environment    8
4    pip            7
5    wt             6
6    arithmetic     5
7    rt             5
8    closing        4
9    libraries      4
10   bigger         4
11   545-44         4
12   nix            3
13   ending         3
14   tgz            3
15   python-3       3
16   panel          3
17   prog           3
18   pim            3
19   class          3
20   hooray         3
```



Non-Programmer's Tutorial for Python 3/Print version

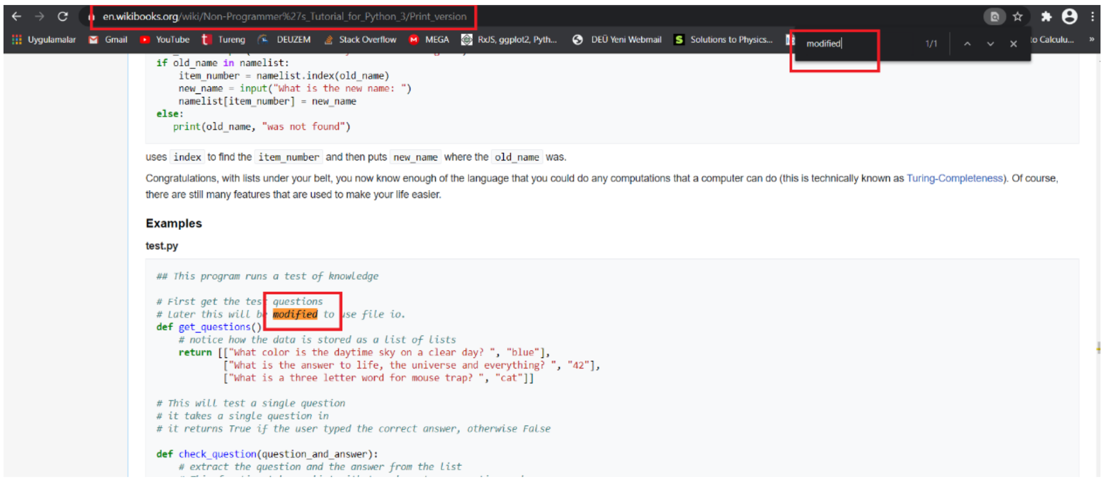
< Non-Programmer's Tutorial for Python 3

This is the print version of Non-Programmer's Tutorial for Python 3
You won't see this message or any elements not part of the book's content when you print or preview this page.

Non-Programmer's Tutorial for Python 3

The current, editable version of this book is available in Wikibooks, the open-content textbooks collection, at
https://en.wikibooks.org/wiki/Non-Programmer%27s_Tutorial_for_Python_3

Permission is granted to copy, distribute, and/or modify this document under the terms of the Creative Commons Attribution-ShareAlike 3.0 License.



```
if old_name in namelist:
    item_number = namelist.index(old_name)
    new_name = input("What is the new name: ")
    namelist[item_number] = new_name
else:
   print(old_name, "was not found")
```

uses `index` to find the `item_number` and then puts `new_name` where the `old_name` was.

Congratulations, with lists under your belt, you now know enough of the language that you could do any computations that a computer can do (this is technically known as Turing-Completeness). Of course, there are still many features that are used to make your life easier.

Examples

test.py

```
## This program runs a test of knowledge

# First get the tes questions
# Later this will b modified to se file io.
def get_questions()
    # notice how the data is stored as a list of lists
    return [["What color is the daytime sky on a clear day? ", "blue"],
            ["What is the answer to life, the universe and everything? ", "42"],
            ["What is a three letter word for mouse trap? ", "cat"]]

# This will test a single question
# it takes a single question in
# it returns True if the user typed the correct answer, otherwise False

def check_question(question_and_answer):
    # extract the question and the answer from the list
```

REFERENCES

ERDINC S.(2017) *python beautifulsoup modulü.* sinanerdin.com

https://www.sinanerdinc.com/python-beautifulsoup-modulu

ANNONYMOUS(2016) *Python error no python at c users dell appdata local programs python python37/32 Python exe.*EDUREKA

https://www.edureka.co/community/73328/python-python-appdata-local-programs-python-python37-python

(2020) *How to upgrade pip in windows* .datofish

https://datatofish.com/upgrade-pip/

(2018)*python sorted()*.programiz.com

https://www.programiz.com/python-programming/methods/built-in/sorted

YASAR(2015)*python list- bir çırpıda listeler*.ysar.net

https://ysar.net/python/list-dersleri-tutorial.html

(2016) *take control of your python print() statements*.scientificallysound

https://scientificallysound.org/2016/10/17/python-print3/

python string strip() method.tutorialspoint

https://www.tutorialspoint.com/python/string_strip.htm

temel dosya işlemleri.. python-istihza.yazbel

https://python-istihza.yazbel.com/temel_dosya_islemleri.html