DOKUZ EYLÜL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

E-BOOK ANALYSIS AND REPRESENTATION

Assignment Report

by
Muzaffer Sevili

January 2021
İZMİR

Contents

1 INTRODUCTION

First of all, to put it simply, the project aims to write a web scraping application using python. Honestly, I didn't even know what web scraping means before I started the project. So I first researched what "web scraping" means. While researching "web scraping" I discovered libraries like requests and BeautifulSoup. Later, I added these libraries to the project. After that, thanks to the videos I watched and the articles I read, I learned to download the desired type of data from the desired website and to shred this data. Finally, with loops, I ranked the data by frequency.

2 METHODOLOGY

2.1 Structure of Your Project

Firstly, I used the "request" and "BeautifulSoup" libraries to get the url and content of the book and parse the book into words.

Secondly, I created dictionaries and variables to use after. After that, I used a list which named Stop_Words to determine stop words.

After that, I used functions to make my code more efficient.

First one of these functions is main_menu which make a menu for user. The user is asked what he wants to do and decision structures are reached according to the input. New inputs taken from the reached decision structures are sent to functions that need to be used. Also, I used global variables here because if the variable here changes it should have changed in the whole code.

Second one of these functions is clean_words which clean symbols and stop words from book. This function can clean words with built-in methods. These are replace(,) , split(), append() and sort(). "replace(,)" makes symbols change to "space", "split ()" allows words to be separated by the specified sign, "append()" allows adding new data to the existing list, "sort ()" allows the list to be sorted alphabetically.

Third one of these functions is read_file which read the file. After reading the file, it makes all the letters lowercase with lower (). I used this function because I realized reading operations more than one time.

Fourth one of these functions is count_words which count the words and sort words up to the entered limit. This function can sort words with “keys()” method. This method returns all the dictionaries keys as a sequence. So I used dictionaries in this function.

Fifth one of these functions is Compare which compares books. I used global variables here because if the variable here changes it should have changed in the whole code. After that I used loops for counting words. Afterward I used loops and built-in methods to add book words and book’s common words in lists. These built-in methods are append() which allows adding new data to the existing list, and sort() which allows the list to be sorted alphabetically. Thereafter, I used loops again with temporary variables to find and store distinct and common words. Also I used here “book” dictionaries which I created before.

Sixth one of these functions is one_book_info which shows one book’s frequency information. I used loop here to print book’s frequency information. Also I used conditional statements here and the len() method, which returns the length of the input to determine the length of the space.

Seventh one of these functions is comparison_books_info which shows two books’ comparison information. I used loop here to print books’ frequency information. Also I used conditional statements here and the len() method, which returns the length of the input to determine the length of the space.

The last one of these functions is download_book which downloads book(s) from the internet. In this function, firstly I splitted the book name with replace() method. Then, with the help of the requests library, I got the url in the desired formats. Then, with the help of the BeautifulSoup library, I obtained the fragmented version of the content (codes) in the url I got with the html.parser

command. After that, I made the data we want from the site as much as possible with a special html tag. Finally, I got only the text portion of the data I got with the html tag through a loop. In this function, I have performed the above operations 3 times. The reason for this is that there are different print versions on the website from which we collect the data. In the code I wrote in this function, I can get data from urls ending in print_version, Print_version and Printable_version. Also, if the url of the book does not end in these 3 ways and there is no such book, a .txt file is not created and the user gets an error message.

2.2 Encountered Problems and Solutions

Actually, when I first read the homework description, I thought I would have a hard time. However, as I mentioned in the bibliography, I watched different courses and read articles. Then I set a timesheet and every day tried to advance my code according to the homework description. When we reached the deadline for the assignment, I completed the assigned tasks.

I actually encountered many problems while doing my homework. The first of these was that when I first wrote my code, I could not download data by entering the title of the book. Then I solved this problem with the replace () built-in method. The second problem I encountered was that the end of the url address of some books was different, so I could not download some books. I solved this problem by changing my code to get data from different url addresses. My third problem is that I could not print an error message if there was no such book. While trying to solve this problem, I realized that the html tag I am using is not getting any data from the blank page. So if the content of the book is empty, I wrote the code that prevents creating a text file and stating that an incorrect name was entered. My fourth problem was that the output of the data did not look good enough. I solve this problem with conditional statements and len() method. The fifth problem I encountered was that the files were saved without extension. It was actually a minor problem and I solved it while trying to solve it. Another problem was sorting books with symbols encoded in utf-8. I solve this problem with “encoding = "utf-8"”. Additionally, I have a problem about common words and distinct words. Some words Although some words are found in only 1 book,

they appear to exist in both books. “documents” is one of them. The reason for this is that elements such as the information text and the front matter found in the first part of the website are also downloaded. The last problem is frequencies. According to the code I have written, the word frequencies may be slightly more or less than what is given in the example, or some words appear in my code when they are not in the example. The reason for this is that the running code in the background contains that word while the book does not. Even "name" is one of them. In addition, many words in the lists in the example were included in the list of stop words, but were not included in the article I used.

2.3 Improvements

I wrote a menu for actions to be taken. Thus, the user can do each action in the order they want.

3 EXPERIMENTATION

The first case is downloading book(s). If the user correctly enters the title of the book, the book will be downloaded to the folder where the .py file is located. For example, if the user types "Non-Programmer's Tutorial for Python 2.6", the book will be downloaded because the url ends in Print_version. Also my code works for books whose url ends with print_version and Printable_version. Planet Earth and How To Assemble A Desktop PC are examples, respectively. If the user types "Exit", it returns to the main menu.

The second case is counting one book’s word(s). If the user downloaded the book, it selects this option and sets a limit for the words to be sorted. If no limit is set, 20 words are sorted by default.

The third case is the comparison of two books. If the user downloaded the books, it selects this option and sets a limit for the words to be sorted. If no limit is set, 20 words are sorted by default. If this option is selected, the common words of the two books, the different words of the 1st book and the different words of the 2nd book are listed in order.

The fourth case is the exit option. If the user selects this option, he/she will get a good day message and the application will close.

4 CONCLUSION

Actually, I learned a lot from this project. First of all, I learned how to manage my time better and how to do good research. Secondly, I realized that I was not getting enough progress because I did not practice enough before doing this project. So this project has been a great lesson for me. Most importantly, I learned how to scrape data from the internet. The project was actually a very good project, but if it was a few more days, anyone could learn about the project. Due to the time problem, many people just tried to finish the project, not learn it.

APPENDIX A: CODE

```
import requests     #TO GET URL
from bs4 import BeautifulSoup #TO PARSE AND ARRANGE DATA(S)

#Variables
choice = 0
limit = 0
book1 = {}
book2 = {}
distinct_words1 = {}
distinct_words2 = {}
commonWords = {}

Stop_words=["i", "me", "my", "myself", "we", "our", "ours", "ourselves"
, "you","your", "yours", "yourself", "yourselves", "he", "him", "his",
"himself","she", "her", "hers", "herself", "it", "its", "itself","they"
,"them","their", "theirs", "themselves", "what", "which", "who", "whom"
, "this","that", "these", "those", "am", "is", "are", "was", "were", "b
e", "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", "above", "bel
ow", "to","from", "up", "down", "in", "out", "on", "off", "over", "unde
r", "again","further", "then", "once", "here", "there", "when", "where"
,"why", "how","all", "any", "both", "each", "few", "more", "most", "oth
er", "some", "such","no", "nor", "not", "only", "own", "same", "so","th
```

```
an", "too", "very", "s","t", "can", "will", "just", "don", "should","no
w","has","int","name","line"] # I CREATE IT WITH SEARCH FOR STOPWORDS

#Functions
def main_menu(): #MENU

    print(" ")
    print(" >> 1- Download Books")
    print(" >> 2- Count one book's words")
    print(" >> 3- Compare two books words")
    print(" >> 4- Exit")

    is_choice = False
    global choice
    global limit
    global distinct_words1
    global distinct_words2

    while not is_choice:
        choice = int(input(" >> ENTER A CHOICE: "))
        if choice < 1 or choice > 4:
            is_choice = False
            print(" >> TRY AGAIN")
        else:
            is_choice = True

    if choice == 1: #DOWNLOAD A BOOK
        book_name= " "
        while (book_name!= "Exit"):
            book_name= input("Please enter your book name or enter 'Exit': ")
            if(book_name!= "Exit"):
                download_book(book_name)
        main_menu()

    elif choice == 2: # READ A BOOK
        limit_choice = (input("Enter the print limit: "))
        if(limit_choice == ""):
            limit = 20
        else:
            limit = int(limit_choice)

        book_name1 = input("Please enter your book's name (file): ")
        uncleaned_words = (read_file(book_name1+".txt"))
        book_words = count_words( clean_words(uncleaned_words) )
        one_book_info(book_words,book_name1)

    elif choice == 3: # COMPARE TWO BOOKS
```

```
        limit_choice = (input("Enter the print limit: "))
        if(limit_choice == ""):
            limit = 20
        else:
            limit = int(limit_choice)

        book_name1 = (input("Enter your 1. book's name: "))
        uncleaned_words1 = read_file(book_name1+".txt")
        book_name2 = (input("Enter your 2. book's name: "))
        uncleaned_words2 = read_file(book_name2+".txt")
        Compare(clean_words(uncleaned_words1),clean_words(uncleaned_words2))
        comparison_books_info (distinct_words1,distinct_words2,book_name1,book_name2)

    elif choice == 4: # EXIT
        print("Good Day")

def clean_words(Words): #CLEANING WORDS FROM SYMBOLS AND STOPWORDS
    cleaned_words= []

    Words = Words.replace(".", " ").replace("%"," ").replace("="," ").repl
ace(","," ").replace(";"," ").replace("_"," ").replace(")"," ").replace("
("," ").replace("!"," ").replace("?"," ").replace("<"," ").replace(">"," "
).replace("/"," ").replace("#"," ").replace("'"," ").replace('"'," ").repl
ace("*"," ").replace(":"," ").replace("&"," ").replace("-"," ")

    for word in (Words.split()):
        if " " in word:
            word = Words.replace(" ",'')
        if word not in Stop_words and (len(word) > 1):
              cleaned_words.append(word)
    cleaned_words.sort()

    return cleaned_words

def read_file(name): #READING FILE
    FILE = open(name,"r",encoding = "utf-8")
    script = FILE.read()
    script = script.lower()
    FILE.close()
    return script

def count_words(words): #COUNTING WORDS
    temp_count = 0
    temp_word = " "
    our_words={}
    word_count = {}
```

```
    for word in words: #COUNTING WORDS
        if word not in word_count:
            word_count[word] = 1
        else:
            word_count[word] =  word_count[word] + 1

    for a in range (limit): #SORTING WORDS UP TO THE ENTERED LIMIT
        for key in word_count.keys():
            if word_count[key] > temp_count and key not in our_words:
                temp_count = word_count[key]
                temp_word = key
        our_words[temp_word] = temp_count
        temp_count = 0
        temp_word = " "

    return our_words

def Compare(book_words1,book_words2):  #COMPARING TWO BOOKS
    global book1
    global book2
    global distinct_words1
    global distinct_words2
    global commonWords

    book1_words = []
    book2_words = []
    commonwords = []
    temp_count = 0
    temp_word = " "

    for word1 in book_words1: #BOOK1 WORDS
        if word1 not in book1:
            book1[word1] = 1
        else:
            book1[word1] +=  1

    for word2 in book_words2: #BOOK2 WORDS
        if word2 not in book2:
            book2[word2] = 1
        else:
            book2[word2] += 1

    for WORD in book1: #ADDING BOOK1 WORDS TO "BOOK1_WORDS"
        if WORD not in book2 and WORD not in book1_words:
            book1_words.append(WORD)
    book1_words.sort()
```

```
    for WORD in book2:  #ADDING BOOK2 WORDS TO "BOOK2_WORDS"
        if WORD not in book1 and WORD not in book2_words:
            book2_words.append(WORD)
    book2_words.sort()

    for WORD in book1: #ADDING COMMON WORDS TO "COMMONWORDS"
        if WORD in book2 and WORD not in commonwords:
            commonwords.append(WORD)
    commonwords.sort()

    for a in range (limit):

        for WORDS in book1_words: #DISTINCT WORDS OF BOOK1
            if (book1[WORDS]) > temp_count and WORDS not in distinct_words1:
                temp_count = (book1[WORDS])
                temp_word = WORDS
        distinct_words1[temp_word] = temp_count
        temp_count = 0
        temp_word = " "

        for WORDS in book2_words: #DISTINCT WORDS OF BOOK2
            if (book2[WORDS]) > temp_count and WORDS not in distinct_words2:
                temp_count = (book2[WORDS])
                temp_word = WORDS
        distinct_words2[temp_word] = temp_count
        temp_count = 0
        temp_word = " "

        for WORDS in commonwords: #COMMON WORDS
            if (book1[WORDS] + book2[WORDS]) > temp_count and WORDS not in commonWords:
                temp_count = (book1[WORDS]+book2[WORDS])
                temp_word = WORDS
        commonWords[temp_word] = temp_count
        temp_count = 0
        temp_word = " "

def one_book_info(Words,book_name1): #SHOWING ONE BOOK INFO

    no = 1
    print(" ")
    print(" ")
    print(">>> Book1: ",book_name1)
    print(" ")
    print("NO             WORD         FREQUENCE")
    print(" ")
```

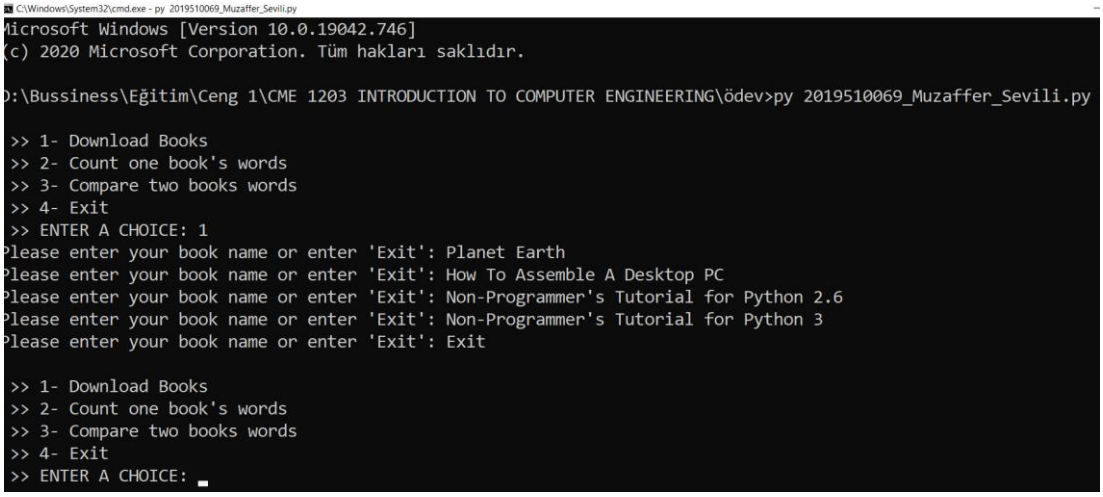
```
    for key in Words.keys(): #MAKING INFORMATION APPEAR PROPERLY
        if(no<10):
             first_space=" "*11
        elif(no>=10 and no<100):
             first_space=" "*10
        elif(no>=100):
             first_space=" "*9

        space=15-len(key)
        space2=space*" "
        print(no,first_space,key,space2,Words[key])
        no = no + 1


def comparison_books_info (Words1,Words2,book_name1,book_name2): # SHOW
 COMPARISON OF TWO BOOKS INFO

    no = 1
    print(" ")
    print(" ")
    print(">>> Book1: ",book_name1)
    print(">>> Book2: ",book_name2)
    print(" ")
    print("COMMON WORDS")
    print(" ")
    print("NO          WORD        FREQUENCE.1    FREQUENCE.2        TOTAL")
    print(" ")

    for key in commonWords.keys(): #MAKING INFORMATION APPEAR PROPERLY

        if(no<10):

            first_space=" "*10

            if(book1[key]>=100):
                if(book2[key]>=100):
                    second_space=" "*10
                    third_space=" "*11
                elif(book2[key]<100):
                    second_space=" "*11
                    third_space=" "*12

            elif(book1[key]<100):
                if(book2[key]>=100):
                    second_space=" "*11
                    third_space=" "*11
                elif(book2[key]<100):
```

```
                    second_space=" "*10
                    third_space=" "*12

        elif(no>=10 and no<100):

            first_space=" "*9

            if(book1[key]>=100):
                if(book2[key]>=100):
                    second_space=" "*9
                    third_space=" "*11
                elif(book2[key]<100):
                    second_space=" "*10
                    third_space=" "*12

            elif(book1[key]<100):
                if(book2[key]>=100):
                    second_space=" "*11
                    third_space=" "*11
                elif(book2[key]<100):
                    second_space=" "*11
                    third_space=" "*12

        elif(no>=100):
            first_space=" "*10
            second_space=" "*10
            third_space=" "*11

        space=15-len(key)
        space2=space*" "
        print(no,first_space,key,space2,book1[key],second_space,book2[k
ey],third_space,commonWords[key])
        no = no + 1

    #SHOWING BOOK1 DISTINCT WORDS
    no = 1
    print(" ")
    print(" ")
    print(">>> Book1: ",book_name1)
    print(" ")
    print("DISTINCT WORDS")
    print(" ")
    print("NO           WORD         FREQUENCE")
    print(" ")
    for key in Words1.keys(): #MAKING INFORMATION APPEAR PROPERLY
        if(no<10):
             first_space=" "*10
```

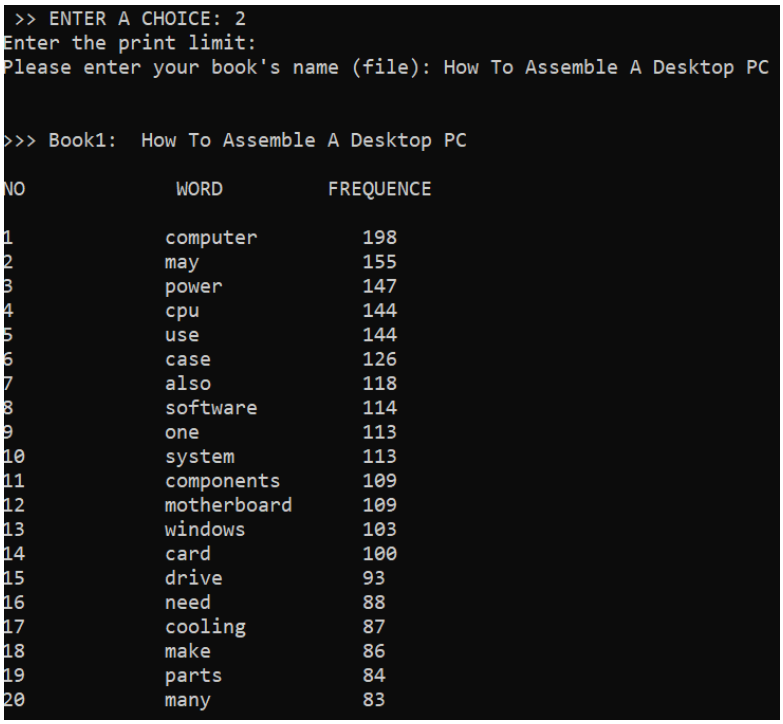
```
        elif(no>=10 and no<100):
             first_space=" "*9

        elif(no>=100):
             first_space=" "*8

        space=15-len(key)
        space2=space*" "

        print(no,first_space,key,space2,distinct_words1[key])
        no = no + 1

    #SHOWING BOOK2 DISTINCT WORDS
    no = 1
    print(" ")
    print(" ")
    print(">>> Book2: ",book_name2)
    print(" ")
    print("DISTINCT WORDS")
    print(" ")
    print("NO           WORD         FREQUENCE")
    print(" ")
    for key in Words2.keys(): #MAKING INFORMATION APPEAR PROPERLY
        if(no<10):
             first_space=" "*10

        elif(no>=10 and no<100):
             first_space=" "*9

        elif(no>=100):
             first_space=" "*8

        space=15-len(key)
        space2=space*" "

        print(no,first_space,key,space2,distinct_words2[key])
        no = no + 1

def download_book(book_name): #DOWNLOADING A BOOK

    book = " "
    splitted_book_name = book_name.replace(" ","_")

    wikiurl = requests.get('https://en.wikibooks.org/wiki/' + splitted_
book_name + "/Print_version")
    wikiurl_2 = BeautifulSoup(wikiurl.content,"html.parser")
```

```
    all_site = wikiurl_2.find_all("div",{"class":"mw-parser-output"})

    for htmlseperator in all_site: #TAKING ONLY TEXTS
        book += (" " + htmlseperator.text)

    if(book==" "): #IF WE CANNOT DOWNLOAD WITH "Print_version"
        wikiurl = requests.get('https://en.wikibooks.org/wiki/' + split
ted_book_name + "/print_version")
        wikiurl_2 = BeautifulSoup(wikiurl.content,"html.parser")

        all_site = wikiurl_2.find_all("div",{"class":"mw-parser-output"})

        for htmlseperator in all_site: #TAKING ONLY TEXTS
            book += (" " + htmlseperator.text)
        if(book==" "):#IF WE CANNOT DOWNLOAD WITH "print_version"

            wikiurl = requests.get('https://en.wikibooks.org/wiki/' + s
plitted_book_name + "Printable_version")
            wikiurl_2 = BeautifulSoup(wikiurl.content,"html.parser")

            all_site = wikiurl_2.find_all("div",{"class":"mw-parser-output"})

            for htmlseperator in all_site: #TAKING ONLY TEXTS
                book += (" " + htmlseperator.text)
            if(book==" "):
                print(" ")
                print("***You had entered a wrong book name!***")
                print(" ")
            else:
                file = open(book_name+".txt","w",encoding = "utf-8")
                file.write(book)
                file.close()

        else:         #IF WE CAN DOWNLOAD WITH "print_version"
            file = open(book_name+".txt","w",encoding = "utf-8")
            file.write(book)
            file.close()

    else: #IF WE CAN DOWNLOAD WITH "Print_version"
        file = open(book_name+".txt","w",encoding = "utf-8")
        file.write(book)
        file.close()

main_menu()
```

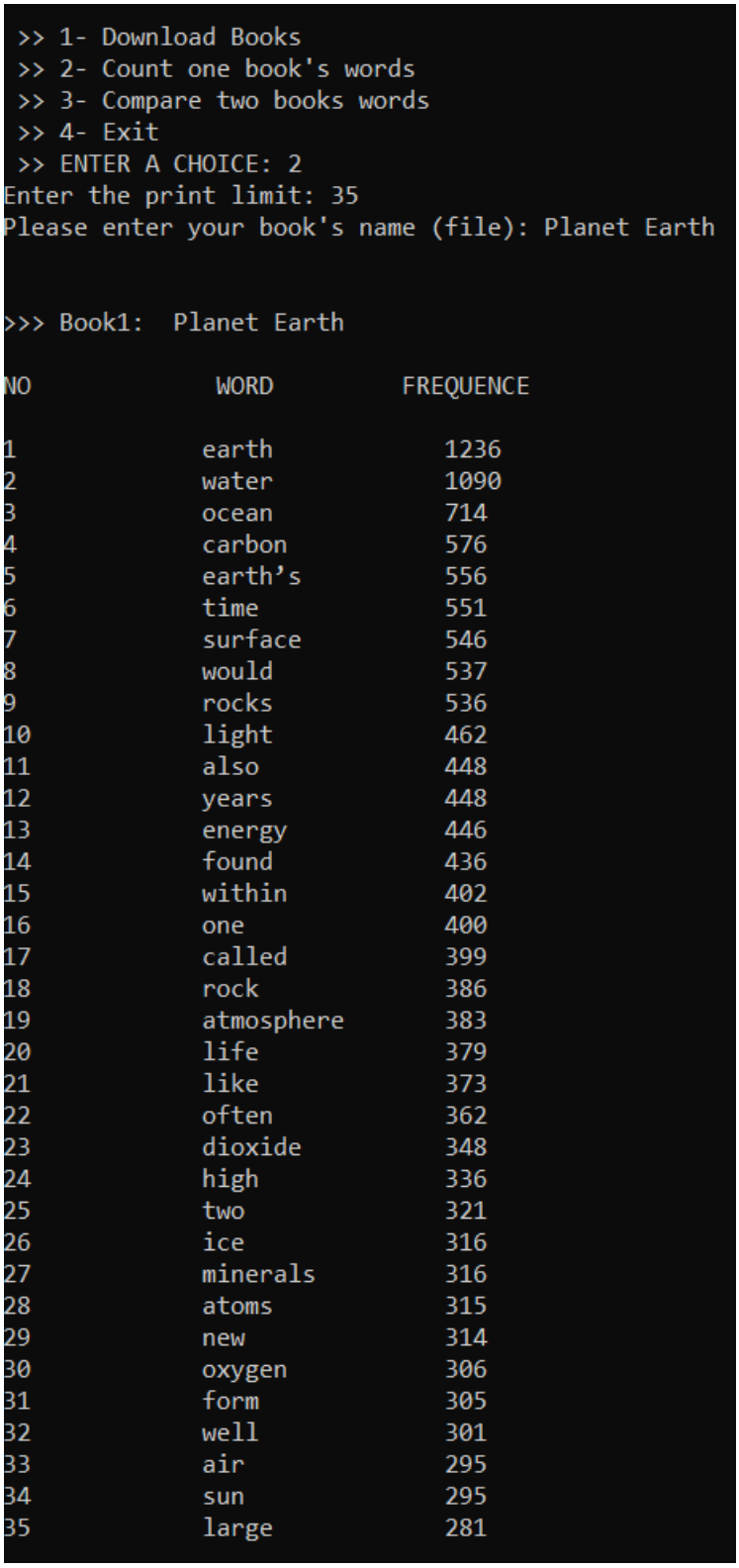
APPENDIX B: SCREENSHOTS OF YOUR USE CASES



This screenshot shows that books ending with different url can be downloaded.



This screenshot shows counting the words of a book when the limit is default.



This screenshot shows counting the words of a book when the limit is specified.

```
 >> 1- Download Books
 >> 2- Count one book's words
 >> 3- Compare two books words
 >> 4- Exit
 >> ENTER A CHOICE: 3
Enter the print limit:
Enter your 1. book's name: Non-Programmer's Tutorial for Python 2.6
Enter your 2. book's name: Planet Earth


>>> Book1:  Non-Programmer's Tutorial for Python 2.6
>>> Book2:  Planet Earth

COMMON WORDS

NO          WORD         FREQUENCE.1    FREQUENCE.2         TOTAL

1           water            5             1090            1095
2           time             45             551            596
3           would            32             537            569
4           print            521             8             529
5           also             33             448            481
6           found            34             436            470
7           one              58             400            458
8           years            2             448            450
9           called           25             399            424
10          number           268            156            424
11          within           13             402            415
12          life             25             379            404
13          like             31             373            404
14          rock             1              386            387
15          often            7              362            369
16          new              49             314            363
17          two              32             321            353
18          first            109            241            350
19          high             11             336            347
20          well             17             301            318
```

This screenshot shows common words of two books when limit is default.

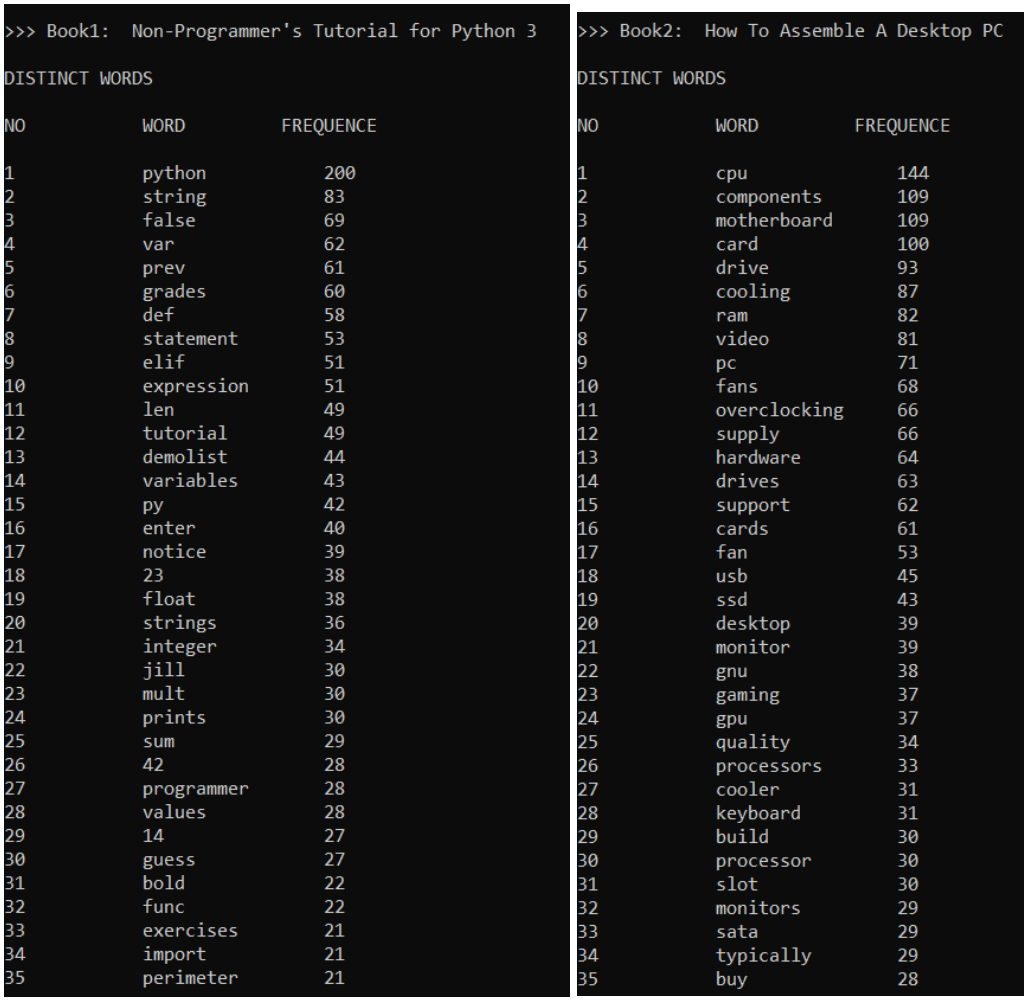
```
>>> Book1:  Non-Programmer's Tutorial for Python 2.6

DISTINCT WORDS

NO          WORD         FREQUENCE

1           python           154
2           menu             99
3           file             75
4           def              63
5           prev             61
6           len              53
7           demolist         49
8           mult             48
9           elif             46
10          password         44
11          py               39
12          hello            32
13          guess            31
14          programming      30
15          tutorial         28
16          jill             27
17          func             24
18          invariant        23
19          dictionary       22
20          filename         20
```

```
>>> Book2:  Planet Earth

DISTINCT WORDS

NO          WORD         FREQUENCE

1           earth            1236
2           ocean            714
3           carbon           576
4           earth's          556
5           surface          546
6           rocks            536
7           light            462
8           energy           446
9           atmosphere       383
10          dioxide          348
11          ice              316
12          minerals         316
13          atoms            315
14          oxygen           306
15          air              295
16          gas              273
17          pressure         265
18          molecules        235
19          north            229
20          species          225
```

This screenshots shows distinct words of two books when limit is default.

```
 >> 4- Exit
 >> ENTER A CHOICE: 3
Enter the print limit: 35
Enter your 1. book's name: Non-Programmer's Tutorial for Python 3
Enter your 2. book's name: How To Assemble A Desktop PC


>>> Book1:  Non-Programmer's Tutorial for Python 3
>>> Book2:  How To Assemble A Desktop PC

COMMON WORDS

NO          WORD         FREQUENCE.1    FREQUENCE.2         TOTAL

1           print            545             13              558
2           number           288             7               295
3           name             229             2               231
4           computer         30              198             228
5           use              70              144             214
6           program          177             8               185
7           may              21              155             176
8           line             153             20              173
9           list             157             8               165
10          one              52              113             165
11          case             22              126             148
12          power            1               147             148
13          also             26              118             144
14          input            132             5               137
15          function         123             12              135
16          first            110             21              131
17          10               64              66              130
18          value            129             1               130
19          system           9               113             122
20          make             34              86              120
21          windows          16              103             119
22          run              78              39              117
23          software         3               114             117
24          end              59              57              116
25          need             25              88              113
26          numbers          111             2               113
27          type             95              18              113
28          want             36              77              113
29          used             51              61              112
30          menu             109             2               111
31          file             97              11              108
32          time             43              63              106
33          get              49              56              105
34          next             94              11              105
35          true             99              3               102
```

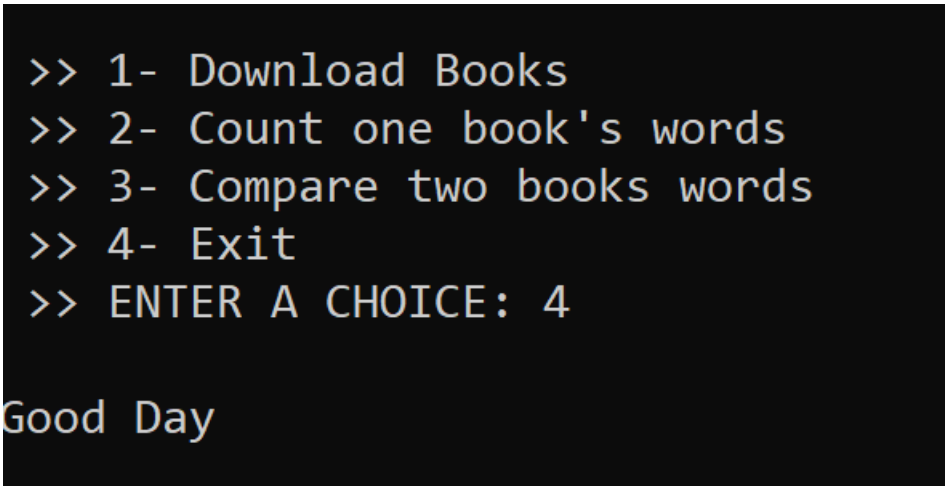
This screenshot shows common words of two books when limit is specified.



```
>>> Book1:  Non-Programmer's Tutorial for Python 3

DISTINCT WORDS

NO          WORD         FREQUENCE

1           python           200
2           string           83
3           false            69
4           var              62
5           prev             61
6           grades           60
7           def              58
8           statement        53
9           elif             51
10          expression       51
11          len              49
12          tutorial         49
13          demolist         44
14          variables        43
15          py               42
16          enter            40
17          notice           39
18          23               38
19          float            38
20          strings          36
21          integer          34
22          jill             30
23          mult             30
24          prints           30
25          sum              29
26          42               28
27          programmer       28
28          values           28
29          14               27
30          guess            27
31          bold             22
32          func             22
33          exercises        21
34          import           21
35          perimeter        21
```

```
>>> Book2:  How To Assemble A Desktop PC

DISTINCT WORDS

NO          WORD         FREQUENCE

1           cpu              144
2           components       109
3           motherboard      109
4           card             100
5           drive            93
6           cooling          87
7           ram              82
8           video            81
9           pc               71
10          fans             68
11          overclocking     66
12          supply           66
13          hardware         64
14          drives           63
15          support          62
16          cards            61
17          fan              53
18          usb              45
19          ssd              43
20          desktop          39
21          monitor          39
22          gnu              38
23          gaming           37
24          gpu              37
25          quality          34
26          processors       33
27          cooler           31
28          keyboard         31
29          build            30
30          processor        30
31          slot             30
32          monitors         29
33          sata             29
34          typically        29
35          buy              28
```

This screenshot shows distinct words of two books when limit is specified.



```
>> 1- Download Books
>> 2- Count one book's words
>> 3- Compare two books words
>> 4- Exit
>> ENTER A CHOICE: 4

Good Day
```

This screenshot shows that when the user selects the "Exit" option, the program gives a "Good Day" message.

REFERENCES

All resources you used during project have to be specified here. You must give citations in the text when you refer a resource. You can use IEEE reference style as a standard [1]. It will be easier if you can use References menu of your word processor.

[1] Goran Jevtic, "How To Install PIP To Manage Python Packages On Windows," 2019. [Online]. Available: https://phoenixnap.com/kb/install-pip-windows.

[2] Martin Breuss, "Beautiful Soup: Build a Web Scraper With Python," 2020. [Online]. Available: https://realpython.com/beautiful-soup-web-scraper-python/.

[3] Mehmet Tek, "Python Programlama Eğitimi" 2020. [Online]. Available: https://www.udemy.com/course/veri-bilimi-ve-makine-ogrenmesi-icin-python/.

[4] Arup Jyoti Dutta, "Removing stop words with NLTK in Python," 2017. [Online]. Available: https://www.geeksforgeeks.org/removing-stop-words-nltk-python/.

[5] David Amos, " A Practical Introduction to Web Scraping in Python," 2020. [Online]. Available: https://realpython.com/python-web-scraping-practical-introduction/.

[6] Yunus Emre Gündoğmuş, "Python ile Veri Kazıma(Web Scraping) Çalışması," 2019. [Online]. Available: https://medium.com/kaveai/web-scraping-453e96a86195.

[7] Onur Eroğlu, "Python Kelime Sayma Programı," 2018. [Online]. Available: https://www.youtube.com/watch?v=5X23dW7lLs8.

[8] W3Schools, "Python String replace() Method," 2018. [Unknown Date]. Available: https://www.w3schools.com/python/ref_string_replace.asp.

[9] Unat, "Python split() Metodu," 2019. [Online]. Available: https://medium.com/@robuno/python-split-metodu-6066ef795d0d#