CS 2302
Data Structure Fall 2020

Lab Report #4

Due:11/09/2020
Professor:Olac, Fuentes
TA: Nath, Anindita
Student: Manuel Gutierrez

Introduction

For this lab, we were asked to work with Hash Tables, Dictionaries, time complexity, and code methods that use this data structure to Store-specific elements. For this lab, it is key to remember the basics of how Hash Tables work and how you can use them to Store data inside them. In this case, we are recreating a Dictionary of some sort where every word is stored in a bucket and we have to check for repeating words also, It is important to note how to translate implementation from every data structure. The main objective of this lab is to get a deeper understanding of how Hash Tables work.

Proposed Solution Design and Implementation

Operation #1

First, ask the user to choose between three books this operation is O(1) because it is just storing an input into a variable to be used later on. I tried to use an edge case statement to help with cases but it turns out that the input variable in python is a string type by default so integer checks were not possible.

Operation #2

After the user picks a book all the sentences will be stored in a hashtable using the method def createHashTable(word_lists): we are creating a table based on every single sentence in the book by setting the length of the Hashtable to the total length of the hashtable. The reason we do this is to fill each bucket with a different word. After that, the user answers questions about what word they would like to search in theory we basically replicate a search engine for a book or the control + f function on windows. After the hashtable is created in O(n) it will then ask the user to look for a word.

Operation #3

After the user searches for the word it will print out how many times the word is repeated in the book and up to 10 sentences where the word appears.

Experimental Results.

I used 3 different results, in this case, the word Buck and jhdjak

Buck
Output:
Reloaded modules: graph_AL, graph_AM, graph_EL
Choose among the following books:
1) The Call of the Wild
2) Dracula
3) The Adventure of Sherlok Holmes

selection: 1

Enter a word to Search: buck
The word buck appears in 296 sentences in The Call of the Wild
The word buck the word appeard in the Folloing sentence:
Buck did not read the newspapers, or he would have known that trouble was brewing, not alone for himself, but for every tide-water dog, strong of muscle and with warm, long hair, from Puget Sound to San Diego.

Buck lived at a big house in the sun-kissed Santa Clara Valley.

And over this great demesne Buck ruled.

But Buck was neither house-dog nor kennel-dog.

Bernard, had been the Judge's inseparable companion, and Buck bid fair to follow in the way of his father.

And this was the manner of dog Buck was in the fall of 1897, when the Klondike strike dragged men from all the world into the frozen North.

But Buck did not read the newspapers, and he did not know that Manuel, one of the gardener's helpers, was an undesirable acquaintance.

No one saw him and Buck go off through the orchard on what Buck imagined was merely a stroll.

Buck had accepted the rope with quiet dignity.

Then the rope tightened mercilessly, while Buck struggled in a fury, his tongue lolling out of his mouth and his great chest panting futilely.

Do you wish to continue y/n?

Jhdjak

output:
Choose among the following books:
1) The Call of the Wild
2) Dracula
3) The Adventure of Sherlok Holmes

selection: 1

Enter a word to Search: jhdjak
Traceback (most recent call last):

File "/Users/manuelgutierrez/Desktop/CS3/lab4/Gutierrez_Manuel_lab4.py", line 88, in <module>
continue_process(H)

File "/Users/manuelgutierrez/Desktop/CS3/lab4/Gutierrez_Manuel_lab4.py", line 55, in continue_process
print('The word '+word+' appears in '+ str(len(bucket))+' sentences in '+bn[int(selection)-1])

TypeError: object of type 'NoneType' has no len()

Conclusion

This lab was made to help us understand more about hashtables in order to fully understand them we must break them down into dictionaries it was really fun to recreate a search machine but I feel like I did not cover as many edge cases as would have liked to. Overall I did learn a more in-depth knowledge on Has Tables thanks to this lab.

'''
Name: Manuel Gutierrez

```
Class: CS2403
Proffessor: Dr.Fuentes
TA: Nath, Anindita
Last modofied November 6 2020
'''

import bs4 as bs
import urllib.request
import hash_table_chain as htc

# You will need to install urllib and BeautifulSoup4
lowercase = ''.join(chr(i) for i in range(97,123)) + ' '

def createHashTable(word_lists):
    H=htc.HashTableChain(len(word_lists))
    for i in range(len(word_lists)):
        for j in word_lists[i]:
            lis = H.retrieve(j)
            if lis == None:
                H.insert(j,[i])
            else:
                #checks for duplicates in a single sentence
                if i != (H.retrieve(j))[len(H.retrieve(j))-1]:
                    lis += [i]
                    H.update(j,lis)
    return H

def find_index(H,word):
    bucket = H.retrieve(word)
    return bucket

def getSentence(bucket,sentece_list):
    for i in range(10):
        #checks just in case if the word does not appear it will break out of the loop
        if(i > len(bucket)):
            break
        print(sentence_list[bucket[i]])
        print()

def get_words(st):
    st = st.lower()
    st = st.replace('\r\n', ' ')
    st = ''.join( c for c in st if  c in lowercase)
    words = st.split()
```

```
    return words

def continue_process(H):
    res = 'y'
    while res == 'y':
        word = input('Enter a word to Search: ')
        bucket = find_index(H,word)

        print('The word '+word+' appears in '+ str(len(bucket))+' sentences in '+bn[int(selection)-1])
        print('The word '+word+' the word appeard in the Folloing sentence:')

        getSentence(bucket,sentence_list)
        res =input('Do you wish to continue y/n?')

if __name__ == "__main__":
    res = 'y'
    bn = ['The Call of the Wild','Dracula','The Adventure of Sherlok Holmes']
    url_list = ['http://www.gutenberg.org/files/215/215-h/215-h.htm',
'http://www.gutenberg.org/files/345/345-h/345-h.htm',
'http://www.gutenberg.org/files/1661/1661-h/1661-h.htm']
    print("Choose among the following books:")
    print('1) The Call of the Wild')
    print('2) Dracula')
    print('3) The Adventure of Sherlok Holmes')
    selection = input("selection: ")

    #funtes code
    url = url_list[int(selection)-1]
    word_lists = []
    sentence_list = []
    data = urllib.request.urlopen(url).read()
    soup = bs.BeautifulSoup(data,'lxml')
    count = 0
    for paragraph in soup.find_all('p'):
        par  = paragraph.string
        if par:
            par = par.replace('\r\n', ' ')
            sent = par.split('.')
            for s in sent:
                sentence_list.append(s+'.')
                words = get_words(s)
                word_lists.append(words)
    H = createHashTable(word_lists)
    continue_process(H)
```

```
    '''
    # print('This is the first sentence:')
    # print(sentence_list[0])
    # print('This is the first word list:')
    # print(word_lists[0])
    '''
```

I Manuel Gutierrez certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class