**Document explaining code for CA OOP Adlane Boulmelh (C19367031) 08/11/2020**

Firstly in this assignment I imported the string function by using   
“import string”. I needed this to be able to clear whitespace and punctuation when storing each word in the document as a key in the dictionary.

I used five functions for this assignment labelled def mainmenu(), def menu1(), def menu2(), def menu3() and def doc\_reader().

Def mainmenu():

This function was used to display the main menu and to keep the menu in a while loop. This menu also had error checking and would display an appropriate error message each time user entered something incorrect.

This function declared three main variables which were crucial in order to check for errors entered by user. These variables were options, runner and checker\_opt1.

A while loop was implemented to repeatedly display the menu. This was while runner == 0: . Menu text will be printed in the loop and user will be asked for input.

The variable options was used to check which option in the menu was selected by the user. This was done by using an If statement 🡪 if options == “1”:

If the user selected one it would run the code in the if statement which was menu1(). This called the menu1() function and will run the code in menu1() function (menu1 function explained below). In options1 achecker\_opt1 was incremented. This was to allow the user to then select option 2 after they completed a search. The user cannot run the program and select option 2 straight away and an appropriate error message will be displayed. This was the same for options == “2”. It executed the code in the elif options == “2”: statement. The elif would then check in an if statement whether checker\_opt1 is equal to 0. If this is the case the user will be moved to the mainmenu again and will be asked to select option 1 first before they can proceed. Next an else statement is there if the user has selected option 1 in the menu before option 2. In the else statement it will call function menu2() to execute the code inside it. Runner will increment in here as well so the menu will not re-appear for user. Another elif statement will be done for options == “3”. This will execute the code in the function menu3(). This is for the option to quit the program. Runner will increment in order for the while loop to not execute. Then an else statement will be implemented to catch any errors the user may enter. Code will be executed in the else if 1,2 or 3 is not entered in the menu. An error message saying “ERROR: Invalid Entry. Please try again”. Runner will be equal to zero in order for the menu to appear again and ask for user input.

Next function is def menu1():

This function is used to read file ap\_docs.txt and store each word in the text as a key in the dictionary. The value will be related to what document it is and will increment when the word tag is found when reading lines.

Variabes declared in this function are new\_doc\_check, word\_tag, dict\_store

doc\_find.

New\_doc\_check is used to keep track of where each word was found. i.e the word “stock” was found in 12,17,22 🡪 {“stock”:”12,17,22”}

Word\_tag is the tag used to figure out whether a new document has been found.

Dict\_store is the dictionary where all keys are stored and their values.

Doc\_find was used to ask user what they were looking for and then it would be searched in the dictionary.

In this function I used the with open(“ap\_docs.txt”,”r”)as file: I tried other ways we learned in class but I encountered issues whenever I used another way. This has worked best for me.

I then executed a for loop to read in each line in the text document.

For line in file.readlines(): was used.

Then I lowercased all the words in every line in the document.

Splitline\_lower = line.lower() This lowercased all characters in each line.

The variable splitline\_split was used to store the split in each line

splitline\_split = splitline\_lower.split()

Next I used another for loop to get each word on its own line in the text document. For word in splitline\_split:

I declared a new variable called new\_words in the for loop and this stored all the words and stripped them of any punctuation. I done this like this.

new\_words = word.strip(string.punctuation)

Next I used an if statement to check if a word tag has been found in the words to increment the document counter.

If word\_tag in new\_words:

Then new\_doc\_check would increment by one each time.

Then an else statement is next and I stored in the dictionary each word as the key and then the value would be added through the variable new\_doc\_check

Dict\_store[new\_words] = new\_doc\_check

I encountered a problem with storing each word as a key and then its value related to which document it is in. I have got the words in each key but when I go to increment the counter that is keeping track of the tag each times it messes up the count for some strange reason.

Then the variable doc\_find would be used to store input from user after a message is displayed saying to enter what word they would like to find in a document.

Then I used a try and except for error checking. The try is executed if the word is found in the dictionary and then it will display the document numbers that are related to the search using dict\_store[doc\_find]. If this doesn’t exist the except will execute and a message saying “search query has not been found”

I then returned dict\_store, doc\_find and doc\_store to use in function menu2()

Def menu2():

This was used to allow a user to read a document they select. I declared

set1 = set() This is used to store every value a specific key is found in. I then called the return variables. I done this by doing in order of returned variables in def menu1().

Dict = menu1()

Word\_search = menu1()

Doc\_nums = menu1()

This variable was returned from function doc\_reader. I will explain this function below

Each\_doc = doc\_reader()

In def menu2() I tried to store what documents that are related to each search query entered by user.

I used a for loop to get each key and value of the dictionary by using a for loop.

For key,value in dict.items()

Then following with an if statement, if key == word\_search.

I tried to add the value of the key found from the search into the set.

Since the set storing wouldn’t work and I encountered an issue I printed out where the searched word has been found e.g if user searches the word “wet” it will print “found in document 15” . Then the user is asked what document they want to open, this is stored in doc\_nums.

Then next I used a print(each\_doc[doc\_nums]). This works because it will go to the dictionary each\_doc then the key which the user stored in doc\_nums will be used to print the value which will be that whole document.

Def menu3()

Contains a print saying “Ending program…” to end the program for the user.

Then an exit is followed to end the program properly.

My final function is def doc\_reader():

This has the main variables of word\_tag which is used to store the tag “<NEW DOCUMENT>”

Then I declared dict\_docs = {} , this is an empty dictionary which will then store each document. Next I had a counter which was equal to 0.

I then opened the “ap\_docs.txt” as a read file.

With open(“ap\_docs.txt”,”r”)as file:

Then a for loop using readlines() to read each line in the file.

Then followed by an if statement to check for the tag in each line. If the tag is found it’ll then execute the code inside the if which is to increment the counter by 1. Then an else is followed by a try and except, code will try to be executed and dict\_docs[counter] += line. This will create a key of the counter then the value will add the next line then the one after until it reaches the next tag in the document to increase the counter and start storing each line in the next key. The except is there if the dictionary cannot += the line it will just assign each line in to it.

Then at the end of the line I started the function mainmenu() to run the program.

The main problem I encountered was storing values in the set.

I could store each word in a key in the first dictionary, but I could not sort out the increment counter to increase counter for each time the tag is found I don’t know why I tried lots of different methods but it was no use.

From making this program I realised a problem with the way I set up the functions. As I returned variables from menu1() and then called these variables in menu2() it creates a problem. When I run the program menu will be displayed then I select one and It asks for a search query. E.g I type in “wet” It will tell me its been found in document 15. Then the menu will re-appear and I then select 2 to read the document it will ask the same prompt from menu option 1 three times. After you enter the search query three times you will be asked to open a document from 1-224 different documents.

I tried to solve this issue and I now what is causing it but I did not have enough time to try and fix it.