CSEE5590/490: Python and Deep Learning Programming (2018 Fall)

LAB ASSGINEMENT 1

Team ID:- 13

Partner 1: Kamal Tej Veerapaneni Class id - 31

Partner 2: Vinay maturi vinay Class id - 17

YouTube Link: https://youtu.be/DrIvZlsu9jg

Introduction:

In this Lab Assignment we have worked on the following tasks.

1)Finding the first non-repeated character in a given string.

2)Removing the content from file_1 which present in file_2

3)List of students who attend “Python Class” but not “web Development”

4)Hospital Management System using various classes and inheritance.

5)Programming a code which downloads the html webpage containing a table using Request Library and then parsing the page using the beautifulsoup library.

Objectives:

Applying dictionaries, array, and iterations in strings for finding the first non-repeated characters in string.

Looping, splitting and respective methods are used for removing contents from file_1 which are present in also file_2

Creating 2 lists for each class and then using a loop we can remove the members from python class list who are in common with the web development class.

Using various classes, constructors inheritance and attribute keywords like **kw different sections of the task were completed.

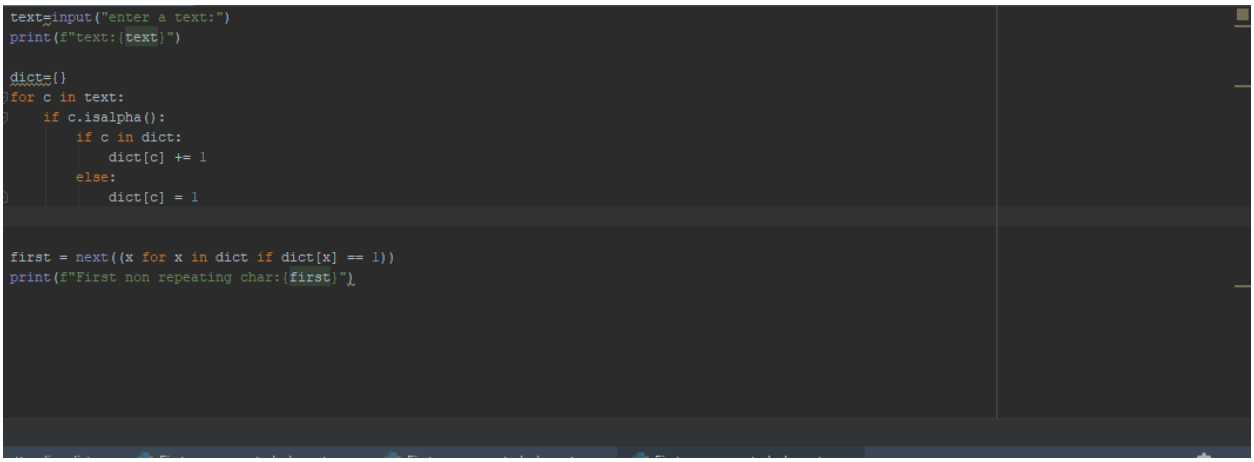
Downloaded the html webpages which contain a table and parsed it using beautifulsoup library.

Workflow:

Question 1:

We used dictionary to store the counts of each character in string input and then an array to store the orders of characters. We will iterate the array to check the first character whose count is 1 and returning the first character if it's count is 1. If there are no repeated characters, it will automatically return no repeated characters in the string.

Code Snippet

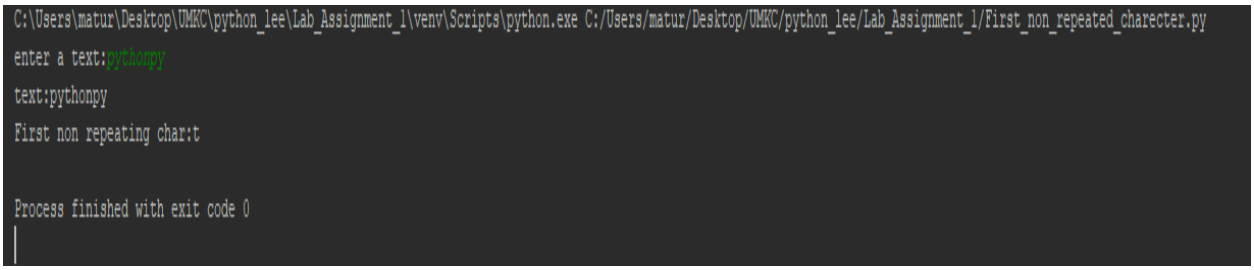


```
text=input("enter a text:")
print(f"text:{text}")

dict={}
for c in text:
    if c.isalpha():
        if c in dict:
            dict[c] += 1
        else:
            dict[c] = 1


first = next((x for x in dict if dict[x] == 1))
print(f"First non repeating char:{first}")
```

Output:



```
C:\Users\matur\Desktop\UMKC\python_lee\Lab_Assignment_1\venv\Scripts\python.exe C:/Users/matur/Desktop/UMKC/python_lee/Lab_Assignment_1/First_non_repeated_charecter.py
enter a text:pythonpy
text:pythonpy
First non repeating char:t

Process finished with exit code 0
```

Question 2:

Each file is split into words and taken into a single string separately for both the files.

Those strings are split into separate lists where words are stored in lists separately for each string generated from each file.

Once the two lists are generated from the two files, each word from the list_1 is checked with the word list of list_2 and then removed if matched with the list_2 word list.

--File_1 is split into words and stored in

```
        for line in f:
  str1.append(line.strip().lower())
```

--File_2 is split into words and stored in

```
for line in f:
  str2.append(line.strip())
```

--List of words are created using the following code.

```
s1=list(str1[0].split(" "))
```

```
s2=list(str2[0].split(" "))
```

Using 2 for loops the common words of list_1 will be removed by matching with the list of words of List_2.

CODE:

```
#consider first file as from a.txt
#consider second file as from b.txt
a = "a.txt"
b = "b.txt"

# first file list
aWordsList = []
# second file list
bWordsList = []

# Open the File and then split it and read each word
#first file words list in an array
with open(a) as f:
    aWordsList = f.read().split()

#second file words list in an array in lowercase
with open(b) as f:

    bWordsList = f.read().lower().split()

# Write access opening for first file
aOut = open(a, 'w')

# Here we loop the first file list and compare whether words in first file word  are there in second file
for aWord in aWordsList:
    if aWord.lower() not in bWordsList:
        aOut.write(aWord+" ")#Write to the file if first file word cannot be seen in second file
```

Output:

```
time, going learn write programs recognize objects images using deep learning. words, weare going explain black magic allows Google Photos search photo
```

Question 3:

In this section we are taking the list of students of Python and Web_Devolopment classes separately into two lists using “while” loop. Next using a line of code we found the list of people who attend python class but not web development.

CODE:

```
x = input("enter the students list for python.Enter none to halt entering")
i = 0
ll = []
while(x!="none"):
    ll.append(x)
    i = i+1
    x = input("enter next student details in python class")
print(ll)
y = input("enter the students list for web development .Enter none to halt entering")
j = 0
l2 = []
while(y!="none"):
    l2.append(y)
    j = j+1

    y = input("enter next student details in python class")
print(l2)
print("Hello_2")
l3 = [x for x in ll if x not in l2]
print(l3)
print("hello3")
```

Output:

```
C:\Users\matur\Desktop\UMKC\python_lee\Lab_Assignment_1\venv\Scripts\python.exe C:/Users/m
enter the students list for python.Enter none to halt enteringvinay
enter next student details in python classkamal
enter next student details in python classakhila
enter next student details in python classharish
enter next student details in python classarchana
enter next student details in python classnone
['vinay', 'kamal', 'akhila', 'harish', 'archana']
enter the students list for web development .Enter none to halt enteringvinay
enter next student details in python classashish
enter next student details in python classpranoop
enter next student details in python classkoushik
enter next student details in python classnone
['vinay', 'ashish', 'pranoop', 'koushik']
Hello_2
['kamal', 'akhila', 'harish', 'archana']
hello3

Process finished with exit code 0
```

Question 4:

In this we have used 5 classes named as Patient, Doctor, Clerk, Nurse, Emergency.

In this we have showed inheritance by taking Patient class as base class and Multiple inheritance is applied Using “Nurse” class with “clerk” and “doctor” class. For passing the arguments in multiple inheritance we have used **super** key word and ****kw** for passing the arguments as key value pairs.

Along with this we have used private member in base class “Patient” to find out the number of patients received by the doctor.

The code and output are as follows

Code Snippet:

```
class patient:
                                                    #BASE CLASS FOR THE NEXT CLASSES
    i=0
    def __init__(self,name,**kwargs):               #  INIT  function used
        self.name = name
        patient.i=patient.i+1
    def book_appointment(self):
        print("The appoitment is booked for tommorowm Mr.", self.name)
    def basic_info(self):
        print("the patients info is ")

class doctor(patient):
                                                    #INHERITENCE USING THE BASIC CLASS
                                                    #class inherited from patient
    j = 0
    def __init__(self,name, age, D_O_B,**kwargs):           #  INIT  function used
        self.name=name
        self.age = age
        self.D_O_B = D_O_B
        super(doctor,self).__init__(name=name, age=age, D_O_B=D_O_B)
    def prescription(self):
        print("Go to medical shop")
    def fees_payment(self):
        print("pay to clerk... baraaluu levuuu")
class clerk(doctor):
    k = 0
    def __init__(self, disease,**kwargs):       #  INIT  function used
        self.disease = disease
        super(clerk,self).__init__(**kwargs)
    def fees_payment(self):
        print("payment of", self.disease, "100$")               #SUPER Key WORD USED

class Nurse(clerk,doctor):
    def __init__(self, name, age, disease, D_O_B, weight):
        self.weight = weight
```

```
        print("payment of", self.disease, "100$")               #SUPER Key WORD USED

class Nurse(clerk,doctor):
    def __init__(self, name, age, disease, D_O_B, weight):
        self.weight = weight
        super(Nurse, self).__init__(name=name, age=age, disease=disease, D_O_B=D_O_B)
    def suggestions(self):
        print("Better chekc you weight", self.weight)
class emergency():
    def __init__(self, is_it,gender):
        self.is_it=is_it
        self.gender=gender
    def join_emergency(self):
        print("the situation is worst?",self.is_it)
        print("the admit is ",self.gender)

print("Patient is Base class where object creation is shown")
obj_1 = patient("vinay")
obj_1.book_appointment()
print(".")
print(".")
print(".")

print("Here super key word is used to pass arguments")
obj_2 = doctor("vinay", 23, "08/22/1996")
obj_2.basic_info()
print(".")
print(".")
print(".")

                                                        #print("THIS IS OBJECT 3 INFO")
                                                        #obj_3 =clerk("vinay")
                                                           #obj_3.fees_payment()
print("Here, Multiple Inheritence is used ")
```

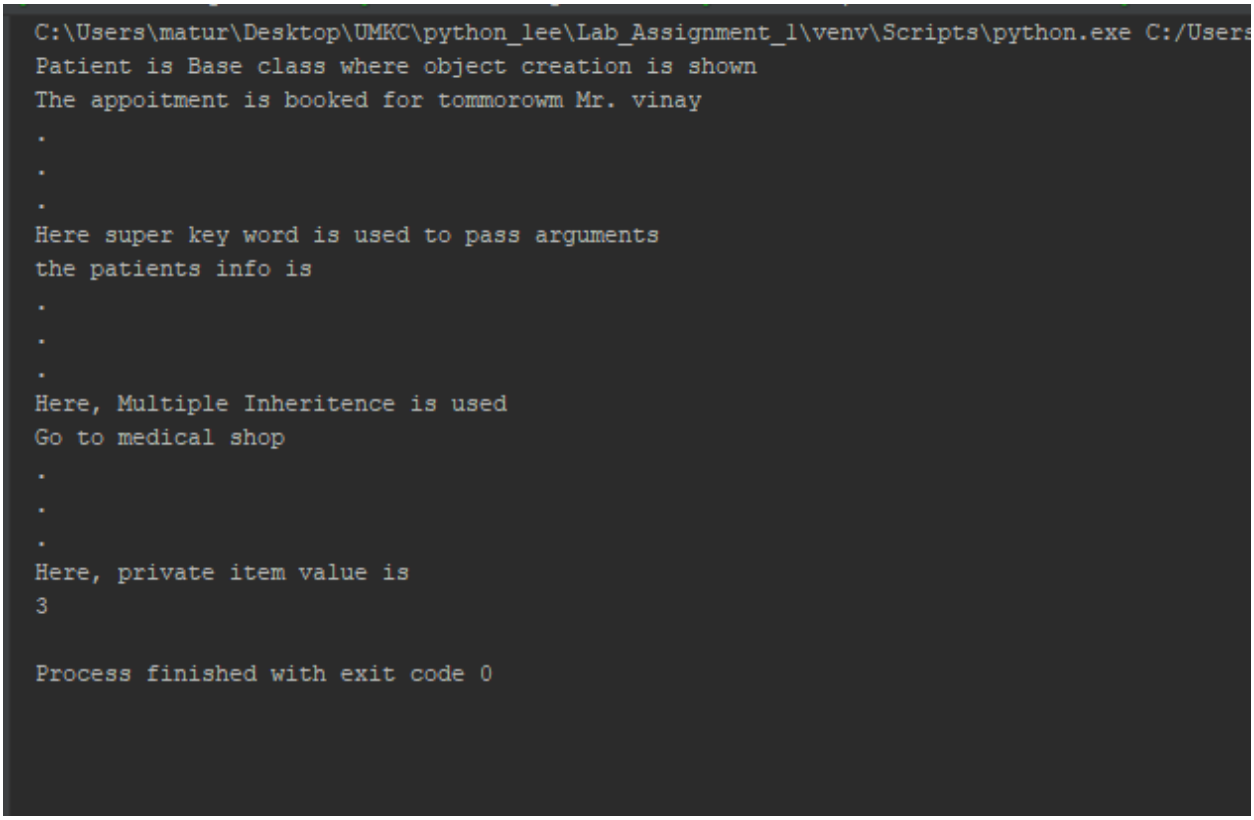
```
print("Patient is Base class where object creation is shown")
obj_1 = patient("vinay")
obj_1.book_appointment()
print(".")
print(".")
print(".")

print("Here super key word is used to pass arguments")
obj_2 = doctor("vinay", 23, "08/22/1996")
obj_2.basic_info()
print(".")
print(".")
print(".")

                                                        #print("THIS IS OBJECT 3 INFO")
                                                        #obj_3 =clerk("vinay")
                                                           #obj_3.fees_payment()

print("Here, Multiple Inheritence is used ")
obj_4 = Nurse("vinay", 23, "Fever", "08/22/1996",72)              #Multiple Inheritence is uaing this object.
obj_4.prescription()
print(".")
print(".")
print(".")
print("Here, private item value is")
print(patient.i)
```

Output:



Question 5:

In this, we import the url through request library and assign link to a variable and then open the link and converts it in to html using soup = BeautifulSoup(getLink, “html.parser”) and prints the header for the wiki page

Code Snippet

```
import requests
from bs4 import BeautifulSoup

my_list=[] #used to store the links

my_link="https://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015"
# our link which we want to print
link=requests.get(my_link)
# requesting the link

obj=BeautifulSoup(link.content,"html.parser")
 #print the title of the webpage
print(obj.title)
my_list.append(obj.find_all('a'))
# collecting data/links on to list
#goes through each  'a' to get the reference
for i in obj.find_all('a'):
    print(i.get('href'))

#c=finds the table and prints the table data.
table = obj.find("table", {"class": "wikitable sortable plainrowheaders"})
for row in table.findAll("tr"):
    cells = row.findAll("td")
    heading = row.findAll("th")
    # we are checking all contents in tables
    print(cells,heading)
```

Output:

```
/affiliate/
/affiliate/
https://secure.fantasypros.com/accounts/profile/
/advertise/
/about/
http://sports.yahoo.com/
http://sportradar.us/
http://www.myfantasyleague.com/
http://www.usatoday.com/sports/
https://www.teamrankings.com/?trk=af_fp_footer
https://www.facebook.com/FantasyPros
https://secure.fantasypros.com/accounts/register/?next=http://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015
https://secure.fantasypros.com/accounts/login/?next=http://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015
https://secure.fantasypros.com/accounts/profile/
https://secure.fantasypros.com/accounts/logout/?next=http://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015
https://twitter.com/FantasyPros
https://twitter.com/FantasyProsMLB
https://twitter.com/FantasyProsNBA
https://www.facebook.com/FantasyPros
https://www.facebook.com/FantasyPros
https://twitter.com/FantasyPros
https://www.instagram.com/fantasypros/
https://www.youtube.com/FantasyPros
/about/
```