Submitted By

- Ruthvic Punyamurtula (Problems - 1,2,4)
- Sai Charan Kothapalli (problems - 3,5,6)

Resources

- youtube demo
- Source Code

Introduction

This is lab assignment 1 of cs5590 - python/Deep Learning class. This assignment gives a head start for python programming required for ML and DL. This lab is based on the tasks done in ICE 1, ICE 2 & ICE 3 which can be found here

Objective

This lab assignment focuses on Implementing the below concepts

- Classes & inheritance
- Handling tuples and dictionaries
- Practice Lists & Sets
- String operation
- Solving a use case based problem
- Sorting on values

Approaches/Methods

We have used python3 to create classes and their objects based on the efficient algorithm time complexities. Most of the work here is based on user's standard input but not the static ones.

Workflow

1. Write a program to find the net final amount of a bank account

Code Snippets

```
net_amount = 0
while 1:
    trans_detail = input("Enter transaction: ")
    # we split on space to separate transaction type & amount
    trans_detail = trans_detail.split(" ")
    trans_type = trans_detail [0] # type
    trans_amount = int (trans_detail [1]) #  amount
    if trans_type=="Deposit" or trans_type=="deposit":
        net_amount += trans_amount
    elif trans_type=="Withdraw" or trans_type=="withdraw":
        net_amount -= trans_amount
    else:
        print("Please enter either Deposit or Withdraw amount only")
    #user choice to continue or not
    choice = input("Enter Y/y to Continue or any other character to exit: ")
    if not (choice =="Y" or choice =="y") :
        break

# print the net amount
print("Net amount: ", net_amount)
```

Output

```
C:\Users\ruthv\Anaconda3\python.exe C:/Users/ruthv/Documents/GitHub/CS5590
Enter transaction: Deposit 1000
Enter Y/y to Continue or any other character to exit: y
Enter transaction: withdraw 300
Enter Y/y to Continue or any other character to exit: y
Enter transaction: Deposit 500
Enter Y/y to Continue or any other character to exit: y
Enter transaction: Withdraw 200
Enter Y/y to Continue or any other character to exit: n
Net amount:  1000
```

2. Create a dictionary from list of tuples & sort them

Code Snippets

```python
def tup_to_dict(tup, dict):
    for a, b in tup:
        dict.setdefault(a, []).append(b)
    return dict

# function to sort dictionary whose values are list of tuples
def sort_dict(dict):
    for idx,list_of_tups in dict.items():
        # key - idx, value = list_of_tups
        dict[idx] = sorted(list_of_tups,key=lambda x: x[1]) # sorts on 1st value of list
    return dict
```

Output

```
C:\Users\ruthv\Anaconda3\python.exe C:/Users/ruthv/Documents/GitHub/CS5590_PyDL/Module1/Lab_Assignment/Lab1/Source/Tuples.py
Output before sorting is :
{'John': [('Physics', 80), ('Science', 95)], 'Daniel': [('Science', 90), ('History', 75)], 'Mark': [('Maths', 100), ('Social', 95)]}
Output after sorting is :
{'John': [('Physics', 80), ('Science', 95)], 'Daniel': [('History', 75), ('Science', 90)], 'Mark': [('Social', 95), ('Maths', 100)]}

Process finished with exit code 0
```

3. Finding i) A intersection B ii) (A union B) - (A intersection B)

Code Snippets

```python
def findStudents(Python, BigData):
    intersectionList = []
    unionList = []
    for Student in Python:
        if Student in BigData:
            intersectionList.append(Student)
            BigData.remove(Student)
        else:
            unionList.append(Student)

    print("Common list of students in both the subjects are: ",intersectionList)
    print("Un Common list of students in both the subjects are: ", unionList + BigData)

if __name__ == '__main__':
    Python = ['Charan','Ram','Kottapalli','Sri']
    WebApplications = ['Charan', 'Kottapalli','Shankar']
    findStudents(Python, WebApplications)
```

Output

```
findStudents() › for Student in Python › else
CommonStudents
/Users/charankottapalli/Desktop/Lab1/venv/bin/python /Users/charankottapalli/Desktop/Lab1/CommonStudents.py
('Common list of students in both the subjects are: ', ['Charan', 'Kottapalli'])
('Un Common list of students in both the subjects are: ', ['Ram', 'Sri', 'Shankar'])

Process finished with exit code 0
```

4. Longest substring with unique characters

Code Snippets

```python
def LongestSubStrWithLength(self, s):
    idx = 0
    max_l = 0
    longestSubStr = ''
    for position in range(1, len(s)):
        # print("position is ---> ", position)
        if(s[position] in s[idx:position]):
            # print("s is  ---> ", s[idx:position])
            max_l = len(s[idx:position]) if (len(s[idx:position]) > max_l) else max_l
            longestSubStr = s[idx:position]
            idx = s[idx:position].index(s[position]) + 1 + idx
            # print("idx is ---> ", idx)
        else:
            if(position == len(s) - 1):
                max_l = max([max_l, len(s[idx:])])
                # print("max is ---> ", max_l)
                # print("s is -----> ", s[idx:])
                if len(s[idx:]) > max_l :
                    longestSubStr = s[idx:]
    return longestSubStr, (max_l if(max_l != 0) else len(s))
```

Output

```
LongestSubString
C:\Users\ruthv\Anaconda3\python.exe C:/Users/ruthv/Documents/
Enter a string --> abccdefghhij
Longest substring is cdefgh and its length is 6

Process finished with exit code 0
```

5. Flight Reservation System

Code Snippets

```python
class Flight(object):
    flight_count = 0
    def __init__(self, Flight_Number, From, To, Date):
        self.Flight_Number = Flight_Number
        self.From_Loc = From
        self.To_Loc = To
        self.Date = Date
        Flight.flight_count += 1

    def getFlightDetails(self):
        #print("Details of flight are: ", self.Flight_Number, self.From_Loc
        return self.Flight_Number, self.From_Loc, self.To_Loc, self.Date
    def getFlightCount(self):
        print("Total number of flights are: " , self.flight_count)


class Person(object):
    person_count = 0
    def __init__(self, Name, Age, Sex):
        self.Name = Name
        self.Age = Age
        self.Sex = Sex
        Person.person_count += 1
```

Output

```
Passenger › __init__()
Flight
/usr/local/bin/python3.6 /Users/charankottapalli/Desktop/Lab1/Flight.py
Employee Details are ('Shankar', 'Male', 36) E898202
Employee Details are ('Rohit', 'Female', 73) E987442
Passenger Details are ('Kottapalli', 'Male', 18) Q789 and Flight details are (9893, 'Kansas-City', 'Chicago-IL', 'Feb-14-19')
Passenger Details are ('Sita', 'Female', 19) A8789 and Flight details are (1235, 'San-Diego', 'Dallas', 'Feb-15-19')
Passenger Details are ('Ram', 'Male', 40) U4232 and Flight details are (9893, 'Kansas-City', 'Chicago-IL', 'Feb-14-19')
Pilot Details are ('John', 'Male', 35) P842748 and assigned flight detils are (9893, 'Kansas-City', 'Chicago-IL', 'Feb-14-19')
Totlal number of pilots are:  1
Total number of persons are:  7
Total number of flights are:  2
Total number of persons are:  7
Total Number of employees are:  2

Process finished with exit code 0
```

6. Web Scraping to find states and Capitals

Code Snippets

```python
import requests
from bs4 import BeautifulSoup

def collect_href():
    url = "https://en.wikipedia.org/wiki/List_of_state_and_union_territory_capitals_in_India"

    source_code = requests.get(url)
    source_text = source_code.text
    soup_text = BeautifulSoup(source_text, 'html.parser')
    #print(soup_text)
    for link in soup_text.findAll('a'):
        href = link.get('href')

    table = soup_text.find("table",{"class":"wikitable sortable plainrowheaders"})

    for rows in table.findAll('td'):
        print(rows.text)



if __name__ == '__main__':
    collect_href()
```

Output

This output is also saved to file

(https://github.com/Ruthvicp/CS5590_PyDL/blob/master/Module1/Lab_Ass

```
StatesandCapitals
/usr/local/bin/python3.6 /Users/charankottapalli/Desktop/Lab1/venv/StatesandCapitals.py
Port Blair

 –

Kolkata

1955

Calcutta (1945–1955)

Hyderabad (de jure to 2024)Amaravati (de facto from 2017)[3] [4] [a]

Amaravati [3]

Amaravati

19562017

Kurnool (1953–1956)

Itanagar

Itanagar

Guwahati

1986
```

Conclusion

We have understood and implemented the concepts of tuples, dictionaries and used them in classes. Also used beautiful soup as a part of web scraping and developed a use case based approach