5590 PYTHON AND DEEP LEARNING PROGRAMS

LAB ASSIGNMENT-1

KARTHIK GUNDALAPALLI

SANTOSH GANDHI DIDDI

MOURYA PRAHARSHA BOBBILI

INTRODUCTION

OBJECTIVE

THE OBJECTIVE OF THIS LAB ASSIGNMENT IS TO PRACTICE TUPLES,LISTS, STRINGS, CLASSES, CLASS-BEHAVIOR, BEAUTIFUL-SOUP LIBRARIES

METHODS

Following are the problems given in lab assignment, screenshots of code and output are shown below

1) Write a program that computes the net amount of a bank account based a transaction log from console input.

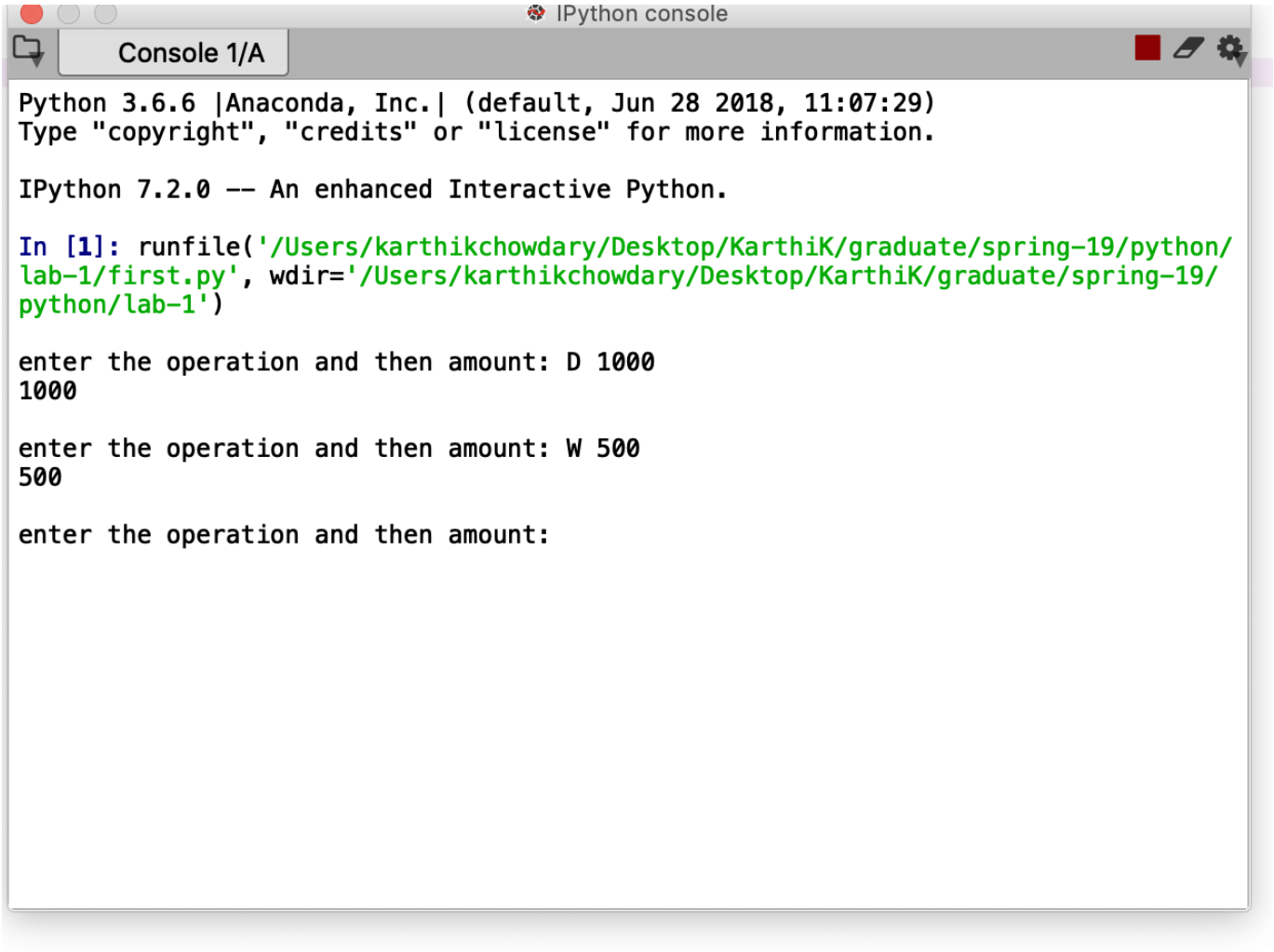
Work-Flow: In this we need to input the console with the required type of transaction and the amount and the program needs to output the balance after the transaction. In this program the user can input W for withdrawal and D for deposit and then the amount. In this we used the split operator to split the input into type of transaction and then the amount. The amount is then converted into int type and then the transaction is processed based on the type of transaction inputted.

Code:

first.py | second.py | third.py | four.py | Five_Classes.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Feb 13 17:46:25 2019

@author: karthikchowdary
"""
netAmount = 0
while True:
    user_s = input("enter the operation and then amount: ")
    if not user_s:
        break
    values = user_s.split()
    operation = values[0]
    amount = int(values[1])
    if operation == "D":
        netAmount += amount
    elif operation == "W":
        netAmount -= amount
    else:
        break
    print(netAmount)

```

Output:


IPython console
Console 1/A
Python 3.6.6 |Anaconda, Inc.| (default, Jun 28 2018, 11:07:29)
Type "copyright", "credits" or "license" for more information.
IPython 7.2.0 -- An enhanced Interactive Python.
In [1]: runfile('/Users/karthikchowdary/Desktop/KarthiK/graduate/spring-19/python/lab-1/first.py', wdir='/Users/karthikchowdary/Desktop/KarthiK/graduate/spring-19/python/lab-1')
enter the operation and then amount: D 1000
1000
enter the operation and then amount: W 500
500
enter the operation and then amount:

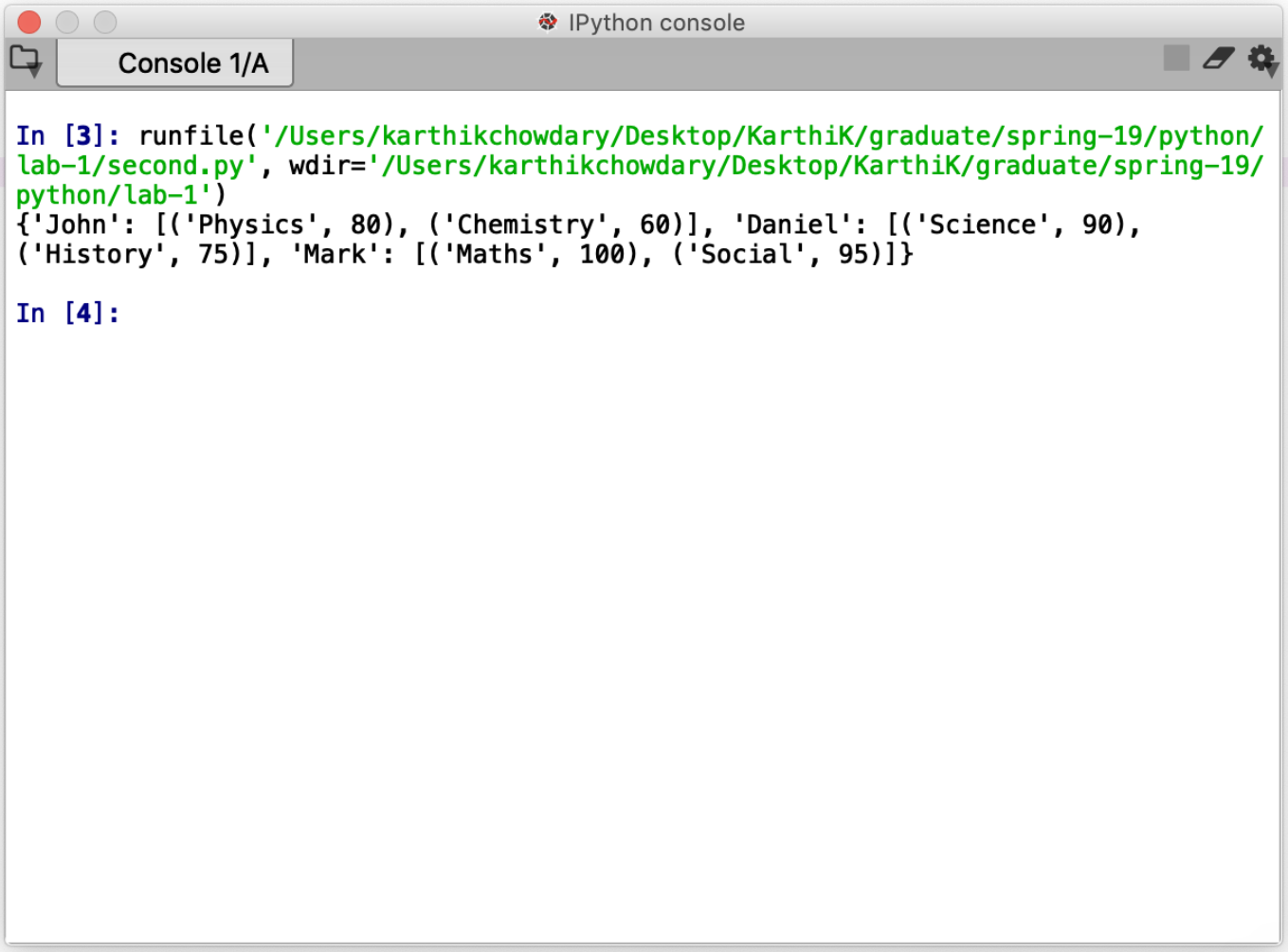
2) Create a dictionary with keys as names and values as list of (subjects, marks) in sorted order.

Work-Flow: In this we have tuples with students names and their marks in each subject. We need to create a dictionary which has the students name and marks of all subjects with his name. So, in this we use setdefault() function for the dictionary. This function returns the value of the key if it is available or creates a new key. So, all marks with same key are assigned to the same student or creates a new key if a new key is inputted.

Code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Feb 13 18:09:41 2019

@author: karthikchowdary
"""
def Con(tuple, dictonary):
    for a, b in tuple:
        dictonary.setdefault(a, []).append(b)
    return dictonary


tuple1 = ('John', ('Physics', 80))
tuple2 = ('Daniel', ('Science', 90))
tuple3 = ('John', ('Chemistry', 60))
tuple4 = ('Mark',('Maths',100))
tuple5 = ('Daniel',('History',75))
tuple6 = ('Mark',('Social', 95))


lt1 = [tuple1,tuple2,tuple3,tuple4,tuple5,tuple6]


dict = {}
dict1 = Con(lt1,dict)


print(dict1)
```

Output:


IPython console
Console 1/A
In [3]: runfile('/Users/karthikchowdary/Desktop/KarthiK/graduate/spring-19/python/lab-1/second.py', wdir='/Users/karthikchowdary/Desktop/KarthiK/graduate/spring-19/python/lab-1')
{'John': [('Physics', 80), ('Chemistry', 60)], 'Daniel': [('Science', 90), ('History', 75)], 'Mark': [('Maths', 100), ('Social', 95)]}
In [4]:

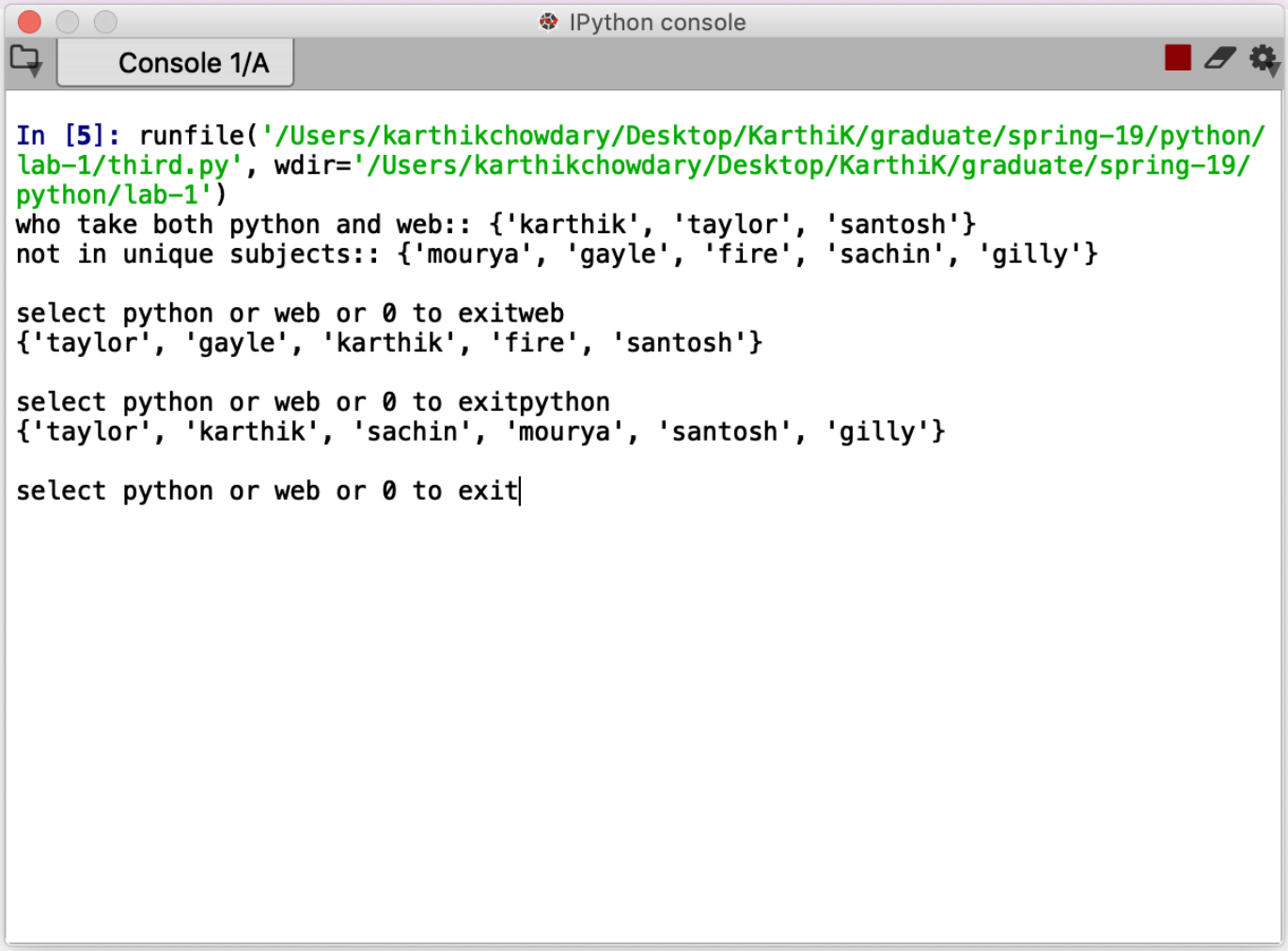
3) Consider the following scenario. You have a list of students who are attending class "Python" and another list of students who are attending class "Web Application".•Find the list of students who are attending both the classes.•Also find the list of students who are not common in both the classes. Print the both lists. Consider accepting the input from the console for list of students that belong to class “Python” and class “Web Application”.

Work-Flow: In this we use and operator "&" between the lists to to get the list of students who go to both the subjects and use union function on students who go only for python and only for web lists to the list of students who don't have both the subjects in common. And at last we run a while loop to get the required list of students going to python or web.

Code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Feb 13 17:30:49 2019

@author: karthikchowdary
"""

Py = {"karthik", "santosh", "mourya","sachin", "taylor", "gilly"}

# students list who took web
web = {"karthik", "fire", "gayle", "taylor", "santosh"}



print("who take both python and web::",Py & web)

onlypython = Py-web


onlyweb= web-Py

print("not in unique subjects::",onlypython.union(onlyweb))
i=1
while(i):
    i=input("select python or web or 0 to exit")
    if(i=="python"):
        print(Py)
    elif(i=="web"):
        print(web)
    else:
        break



"""print("are in python but not in web::",onlypython)

print("only in web",onlyweb)"""

```

Output:


IPython console
Console 1/A
In [5]: runfile('/Users/karthikchowdary/Desktop/KarthiK/graduate/spring-19/python/lab-1/third.py', wdir='/Users/karthikchowdary/Desktop/KarthiK/graduate/spring-19/python/lab-1')
who take both python and web:: {'karthik', 'taylor', 'santosh'}
not in unique subjects:: {'mourya', 'gayle', 'fire', 'sachin', 'gilly'}
select python or web or 0 to exitweb
{'taylor', 'gayle', 'karthik', 'fire', 'santosh'}
select python or web or 0 to exitpython
{'taylor', 'karthik', 'sachin', 'mourya', 'santosh', 'gilly'}
select python or web or 0 to exit

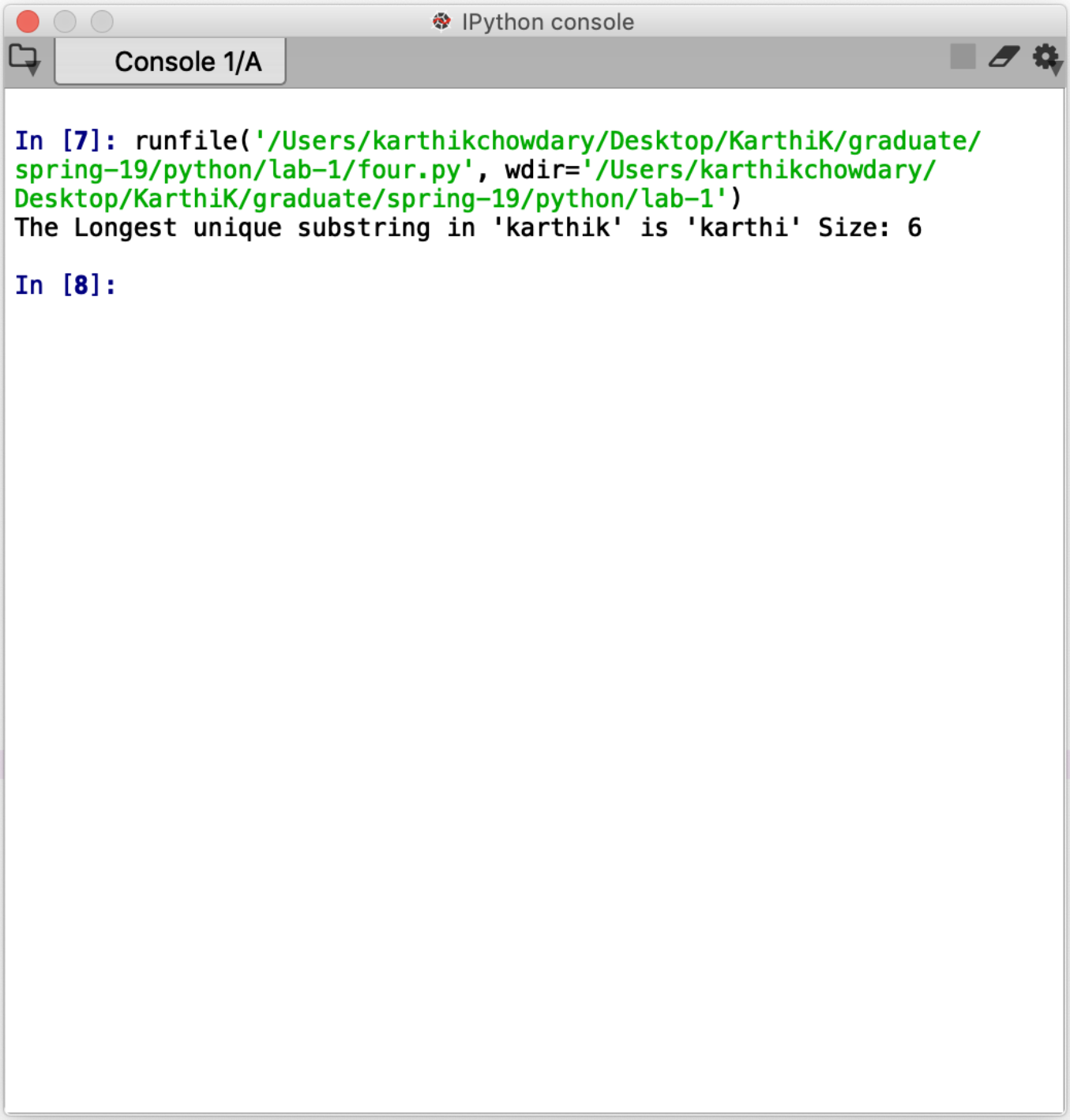
4) Given a string, find the longest sub-string without repeating characters along with the length.

Work-Flow: In this we need to find the longest sub-string present in a given string without repeating characters along with its length. This can be done as shown below:

Code:

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Feb 13 18:09:49 2019

@author: karthikchowdary
"""

def uniquesubstring(input_string):

  last_occurrence = {}
  longest_length = 0
  longest_position = 0
  starting_position = 0
  current_length = 0


  for a, b in enumerate(input_string):
    l = last_occurrence.get(b, -1)

    if l < starting_position:
        current_length += 1
    else:

        if current_length > longest_length:
            longest_position = starting_position
            longest_length = current_length

        current_length -= l - starting_position
        starting_position = l + 1

    last_occurrence[b] = a

  if current_length > longest_length:
    longest_position = starting_position
    longest_length = current_length

  return input_string[longest_position:longest_position + longest_length]




input = 'karthik'

print(f"The Longest unique substring in '{input}' is '{uniquesubstring(input)}' Size: {len(uniquesubstring(inpu
```

Output:




```
ıbstring(input))}")
```

5) Write a python program to create any one of the following management systems. 1. Airline Booking Reservation System (e.g. classes Flight, Person, Employee, Passenger etc.) 2. Library Management System(eg: Student, Book, Faculty, Department etc.) Prerequisites: a. Your code should have at least five classes b. Your code should have *init* constructor in all the classes c. Your code should show inheritance at least once d. Your code should have one super call e. Use of self is required f. Use at least one private data member in your code g. Use multiple Inheritance at least once h. Create instances of all classes and show the relationship between them

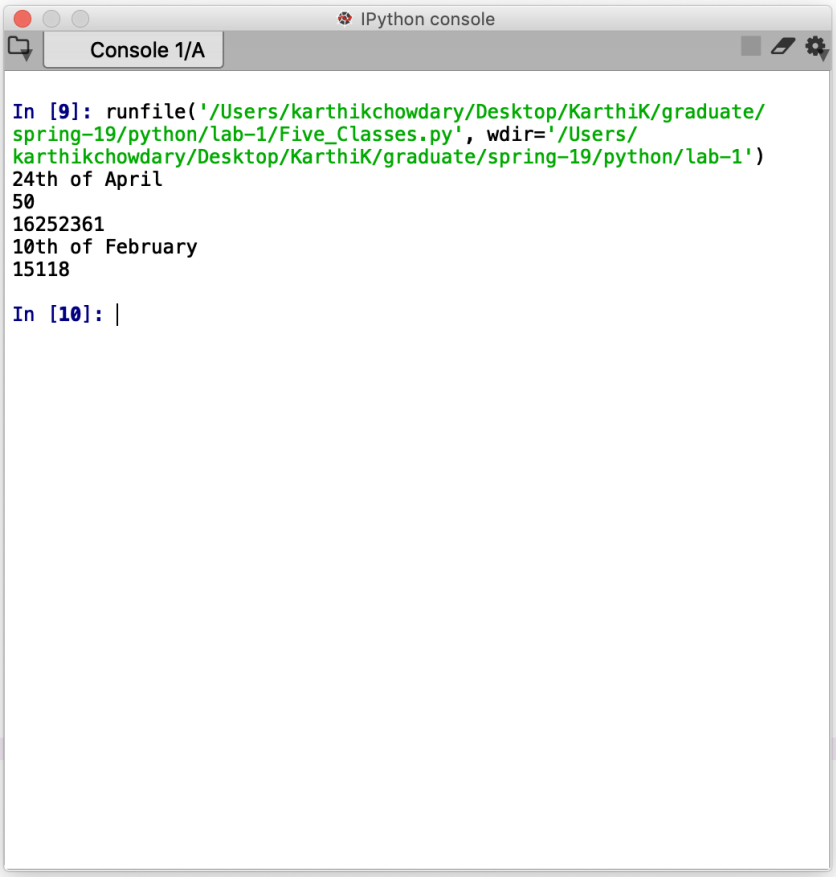
Work-Flow: All the requirements can be seen in the program code below:

Code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Feb 13 16:52:30 2019

@author: karthikchowdary
"""

class Person:

    def __init__(self,n,a,add):
        self.name=n
        self.age=a
        self.address=add


""" Person Class is created it can be inherited by Passenger and also Employee """
class Passenger(Person):

    def __init__(self,n,a,add,luggweight):
         Person.__init__(self,n,a,add)
         self.luggage_weight=luggweight

    def gettraveldate(self):
        print("24th of April")
    def getluggage(self):
        print(self.luggage_weight)
""" Passenger is a class extending Person class """

class Employee(Person):

    def __init__(self,n,a,add,idnumber):
         Person.__init__(self,n,a,add)
         self.id=idnumber

    def getjoindate(self):
        print("10th of February")

    def getid(self):
        print(self.id)

""" Employee is a class extending Person class """
class Flight():
    fno=0
    def __init__(self,fno):
        self.flight=fno

    def getflight(self):
        print(self.fno)

""" flight is a class"""

class Pilot(Person, Flight):
    def __init__(self,n,a,add,fno,id):
        Person.__init__(self,n,a,add)
        Flight.__init__(self,fno)
        self.id=id
```

```
54      def __init__(self,n,a,add,fno,id):
55          Person.__init__(self,n,a,add)
56          Flight.__init__(self,fno)
57          self.id=id
58
59      def getpilotid(self):
60          print(self.id)
61
62  """ Multiple Inheritance Pilot class extends Person and Flight """
63
64
65
66
67  pass1=Passenger("karthik",22,"india",50)
68  pass1.gettraveldate()
69  pass1.getluggage()
70
71
72  emp=Employee("mourya",22,"usa",16252361)
73  emp.getid()
74  emp.getjoindate()
75
76
77  pilot=Pilot("santy",22,"india",1665,15118)
78  pilot.getpilotid()
79
80
```

Output:



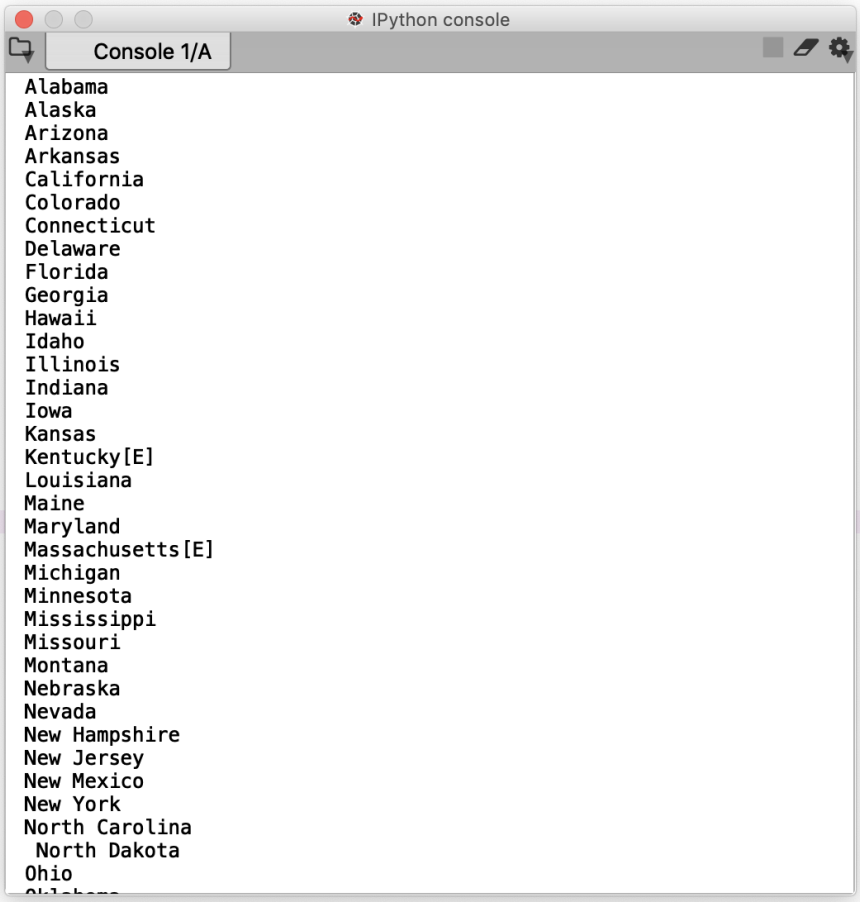
6) Program a code which download a web-page contains a table using Request library, then parse the page using Beautiful soup library. You should save the information about the states and their capitals in a file.

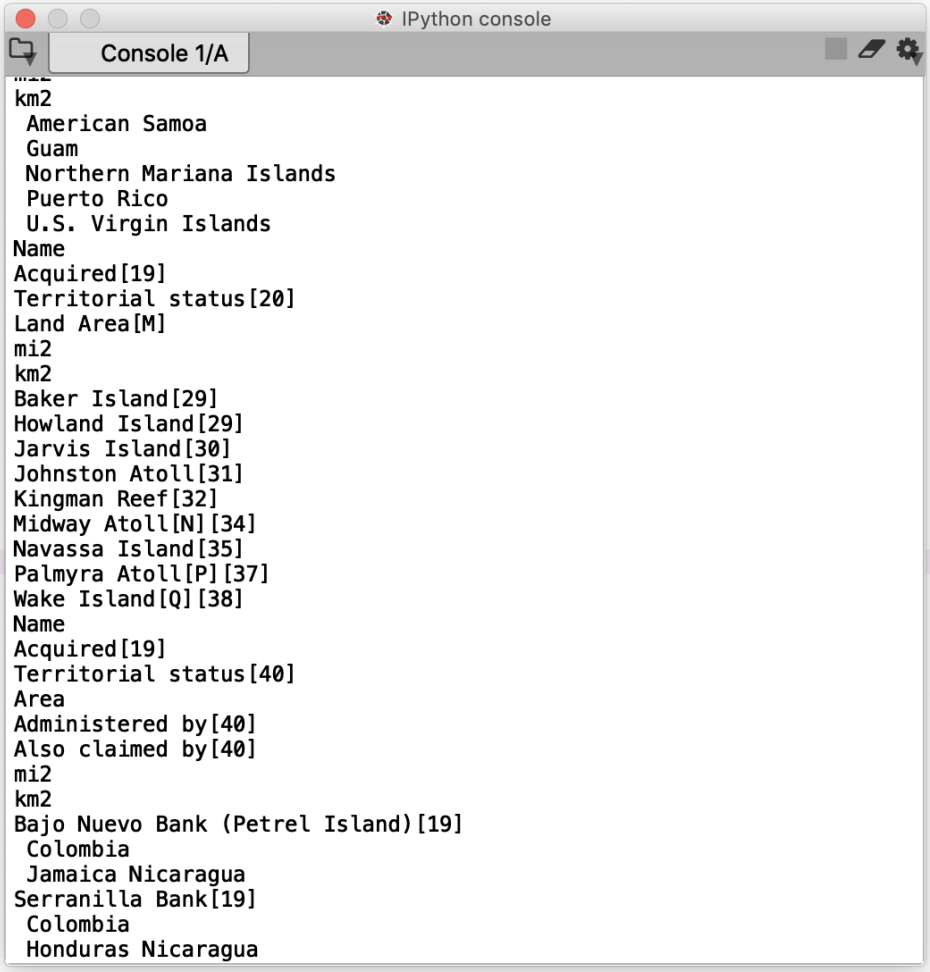
Work-Flow: In this we need to parse the given website to get the required names or data from the website. In this case we have a wikipage and the output needs to be the list of states in a table in the page. so when we inspect the site we can see that all the names of states are in th tags so we find them all and save them to a file to get the required output. this can be done using find_all function imported from beautiful soap 4.The following are the respective code and outputs:

Code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Feb 13 18:58:44 2019

@author: karthikchowdary
"""
import urllib.request
from bs4 import BeautifulSoup

file1 = open("table_txt", "w+")
wikiurl = "https://en.wikipedia.org/wiki/List_of_states_and_territories_of_the_United_States"


openURL = urllib.request.urlopen(wikiurl)


soup = BeautifulSoup(openURL, "html.parser")


for rows in soup.find_all('th'):

    file1.write(str(rows.text))

file1.seek(0,0)
string1 = file1.read()
print(string1)
file1.close()
```

Output:


IPython console
Console 1/A
Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky[E]
Louisiana
Maine
Maryland
Massachusetts[E]
Michigan
Minnesota
Mississippi
Missouri
Montana
Nebraska
Nevada
New Hampshire
New Jersey
New Mexico
New York
North Carolina
North Dakota
Ohio


IPython console
Console 1/A
km2
American Samoa
Guam
Northern Mariana Islands
Puerto Rico
U.S. Virgin Islands
Name
Acquired[19]
Territorial status[20]
Land Area[M]
mi2
km2
Baker Island[29]
Howland Island[29]
Jarvis Island[30]
Johnston Atoll[31]
Kingman Reef[32]
Midway Atoll[N][34]
Navassa Island[35]
Palmyra Atoll[P][37]
Wake Island[Q][38]
Name
Acquired[19]
Territorial status[40]
Area
Administered by[40]
Also claimed by[40]
mi2
km2
Bajo Nuevo Bank (Petrel Island)[19]
Colombia
Jamaica Nicaragua
Serranilla Bank[19]
Colombia
Honduras Nicaragua