

Team: 14  
Professor: Yugyung Lee

Name: Sneha Mishra  
Class ID: 19  
Email: [smccr@mail.umkc.edu](mailto:smccr@mail.umkc.edu)  
[MyGitHub](#)

Technical Partner:  
Name: Raju Nekadi  
Class ID: 20  
Email: [rn8mh@mail.umkc.edu](mailto:rn8mh@mail.umkc.edu)  
[GitHub](#)

Source code for this lab work can be found [here](#)

## Objective

---

This Lab work is divided into 5 parts, to get familiar with python and understand the basic topics. Few of the topics covered in this lab are:

- Loop structures, Conditional statements, Functions, Tuples
- Complex datatypes, Dictionary, Sets, Python Functions, Web Scraping
- Classes, Object Oriented Concepts, Inheritance, instances, Scientific packages
- Data types, Operators, Conditional Statements

## Features

### Part 1:

---

#### Problem Statement:

Search in a string and find the first non-repeated characters in that string.

Example-

Input: Deep data structure.

Output: p

(hint: if there is space in the string you need to consider the whole as one string. In the above example Deepdatastructure).

## Solution:

1. Ask user to input a string of choice.
2. Convert the string to lower case (can go for upper case if you want) to make it case in-sensitive.
3. Form a dictionary of each character in the string with its count of occurrences.
4. Loop through this list to get the first character which does not repeat.

Code snippet is attached below:

```
1 # Ask user Input String
2 str = input('Enter String here - ')
3
4 # Removing spaces from input string and
5 # converting string to lower case as 0 not equal to d
6 # which will give incorrect output
7 new_str = str.replace(" ", "").lower()
8 length1 = len(new_str)
9
10 # defining a dictionary to hold for character from string
11 freq_dict1 = {}
12
13 # Building the dictionary with count from User input string
14 for i in range(0, length1):
15     c = new_str[i]
16     val = freq_dict1.get(c)
17     if val is not None:
18         freq_dict1[c] = val + 1
19     else:
20         freq_dict1[c] = 1
21
22 # looping through dictionary and printing first non repeating element.
23 for c in new_str:
24     if freq_dict1[c] == 1:
25         print('The first non repeating character in User str is: ', c)
26         break
```

The output after running the code is below:

```
Run: Part1 x
/Users/snehamishra/PycharmProjects/Test/venv/bin/python /Users/snehamishra/PycharmProjects/Test/venv/bin/python
Enter String here - data structure
The first non repeating character in User str is: e
Process finished with exit code 0
```

```
Run: Part1 x
/Users/snehamishra/PycharmProjects/Test/venv/bin/python /Users/snehamishra/PycharmProjects/Test/venv/bin/python
Enter String here - Deep data structure
The first non repeating character in User str is: p
Process finished with exit code 0
```

## Part 2:

---

### Problem Statement:

Suppose you have two files:

File1: "This time, we are going to learn how to write programs that recognize objects in images using deep learning. In other words, we are going to explain the black magic that allows Google Photos to search your photos based on what is in the picture"

File2: "this we to are in the that your on based what is how other"

Program a code such that you remove everything in the File1 which is inside File2.

The output of File1 will be: "time, going learn write programs recognize objects images using deep learning. words, going explain black magic allows Google Photos search photos picture".

### Solution:

1. Open both the input files in read mode (since these files are never updated, just read).
2. Check if the opened files are empty or not, if empty then display appropriate user messages.
3. When file not empty, Open a result file in write mode to update the File 1 contents.
4. Create a list of words from second file, which needs to be discarded from file 1.
5. Check if the word in file 1 is not in the above mentioned list of words, then only write the words in result file.

Code snippet is attached below:

```
Part1.py Part2.py
1 # shutil module for its methods to copy files
2 import shutil
3
4 # Open the first file in read mode
5 file1 = open("file1.txt", "r")
6
7 # Get the 1st character of the file
8 file1FirstChar = file1.read(1)
9
10 # Getting data of the file
11 data1 = file1.read()
12
13 # If the 1st character is not available, then the file is empty
14 if not file1FirstChar:
15     print("The file 1 is empty!")
16 else:
17     # Open the second file in read mode
18     file2 = open("file2.txt", "r")
19
20     # Result file with deleted words
21     with open('result.txt', 'w') as result:
22         # Get the 1st character of the second file
23         file2FirstChar = file2.read(1)
24
25         # Getting data of the file 1
26         data2 = file2.read()
27
28         # If the 1st character is not available, then the file is empty
29         if not file2FirstChar:
30             print("The file 2 is empty! Thus output will be same as File 1!")
31             shutil.copyfile('file1.txt', 'result.txt')
32         else:
33             print("Comparing File 1 & 2 in else block 1!")
34             # Creating a list of the words in file 2 for comparing it with File 1 words later
35             list0fWord2 = []
36
37             # Getting all the comma separated words from the File 2
38             words2 = data2.split(" ")
```

```
24
25 # Getting data of the file 1
26 data1 = file1.read()
27
28 # If the 1st character is not available, then the file is empty
29 if not file2FirstChar:
30     print('The file 2 is empty! Thus output will be same as File 1!')
31     shutil.copyfile('file1.txt', 'result.txt')
32 else:
33     print('Comparing File 1 & 2 in else block 1')
34     # Creating a list of the words in file 2 for comparing it with File 1 words later
35     listOfWord2 = []
36
37     # Getting all the comma separated words from the File 2
38     words2 = data2.split(", ")
39
40     # Looping through all the words of File 2
41     for word2 in words2:
42         listOfWord2.append(word2)
43
44     # Getting all the comma separated words from the File 1
45     words1 = data1.split(", ")
46
47     # Looping through all the words in File 1
48     for word1 in words1:
49         if not word1 in listOfWord2:
50             print('writing to file 1 - ' + word1)
51
52             # Add the word in the result file only if the word does not match with the File 2 words
53             result.write(word1 + ", ")
54
55 # Closing all the opened files
56 file2.close()
57 file1.close()
58 result.close()
```

Snippet of the Input file to the code:

```
1 This time, we are going to learn how to write programs that recognize objects in images using deep learning.
2 In other words, we are going to explain the black magic that allows Google Photos to search your photos based on what is in the picture
```


```
1 |this we to are in the that your on based what is now other
```

The output after running the code is below:

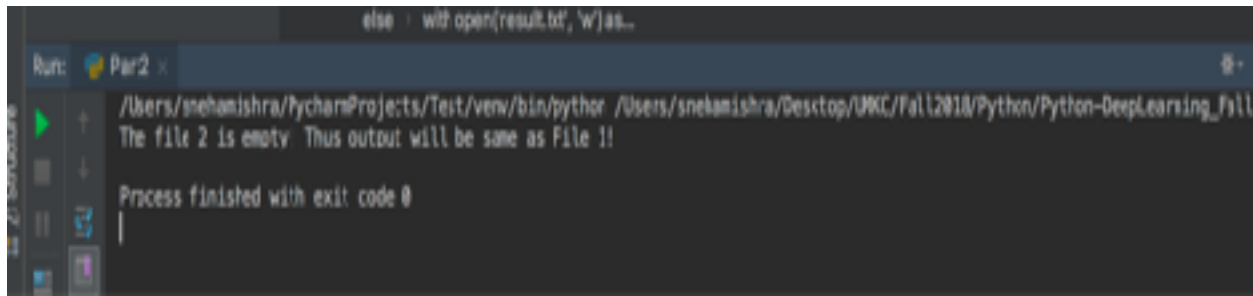
[illegible]

time, going learn write programs recognize objects images using deep learning. In words, going explain black magic allows Google Photos search photos pictures

When input file is empty, output will be:



The screenshot shows a code editor interface. On the left, a file explorer displays a project structure with folders 'Part1' through 'Part5' and a file 'file2.txt'. The 'file2.txt' file is selected, and its content is displayed in the main editor area. The content of 'file2.txt' is empty.

A screenshot of a Python IDE's console window. The window title is 'Run: Par2'. The console output shows a file operation error: 'FileNotFoundError: [Errno 2] No such file or directory: \'result.txt\''. The error message is partially visible at the top: 'else: with open(result.txt, \'w\') as...'. Below the error, the console shows the command prompt path: '/Users/snehamishra/PycharmProjects/Test/venv/bin/python /Users/snehamishra/Desktop/UMKC/Fall2018/Python/Python-DeepLearning\_Fall...'. The output text reads: 'The file 2 is empty. Thus output will be same as File 1!'. At the bottom, it says 'Process finished with exit code 0'.

```
else: with open(result.txt, 'w') as...
FileNotFoundError: [Errno 2] No such file or directory: 'result.txt'
/Users/snehamishra/PycharmProjects/Test/venv/bin/python /Users/snehamishra/Desktop/UMKC/Fall2018/Python/Python-DeepLearning_Fall...
The file 2 is empty. Thus output will be same as File 1!
Process finished with exit code 0
```

## Part 3:

### Problem Statement:

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 "python" classes but not "Web Application".

### Solution:

1. Create two lists of students for the two courses mentioned in the above problem.
2. Create a third list which contains the list of students enrolled in first course but not in the 2nd course.
3. Run the loop for the students in list one, and put the names of only those students who are not in the second list.
4. Print the third list.

Code snippet is attached below:

```
Part1.py x Part2.py x Part3.py x
1 # Defining List for Python Class
2 Python = ['Sneha', 'Suzash', 'Gutlu', 'Bubba', 'Badiu', 'Plan', 'Aditya']
3
4 # Defining List for Web Application
5 WebApp = ['Gutlu', 'Swati', 'Aditya', 'Plan']
6
7 # List to hold number of students who are attending Python but not Web Application
8 pythonNotWebApp = []
9
10 # Logic to fill new list
11 for p in Python:
12     if p not in WebApp:
13         pythonNotWebApp.append(p)
14
15 # Driver Program
16 if __name__ == "__main__":
17     print('The student who are attending Python and not WebApplication Class are: ', pythonNotWebApp)
```

The output after running the code is below:

```
log in Python
Part3.py
C:\Users\ankandhara\PycharmProjects\Test\venv\Scripts\python /Users/ankandhara/Desktop/DMC/Fall2012/Python/Python-DeepLearning_Fall2012/Mod1_Lab1/Source/mod1_lab1/Part3.py
The student who are attending Python and not WebApplication Class are: ['Sneha', 'Suzash', 'Bubba', 'Badiu']
PYTHON PROGRAM EXECUTED SUCCESSFULLY
```

## Part 4:

### Problem Statement:

Write a python program to create a Hospital admission System (e.g. classes Patient, Doctor, Medical Admission Clerk, Book, Nurse, etc.)

Prerequisites:

- Your code should have at least five classes
- Your code should have *init* constructor in all the classes
- Your code should show inheritance at least once
- Your code should have one super call
- Use of self is required
- 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
- Comment your code appropriately to point out where all these things are present

## Solution:

1. Create Classes called Hospital, Procedure, Patient, Staff, Nurse and Doctor.
2. Define the attributes and getter/setter for each class as required.
3. For multiple inheritance, class Patient inherits properties from two classes Hospital and Procedure.
4. Class Staff inherits from class Hospital, while class Doctor and Nurse inherit from class Staff.
5. Create objects of the classes and print them out.

Code snippet is attached below:

```
Part1.py x Part2.py x Part3.py x Part4.py x
1 # Hospital Class with name and address public data attribute
2 class Hospital:
3     def __init__(self, n, a):
4         self.hname = n
5         self.haddress = a
6
7     # Dental Procedure class with procedure name , procedure code , procedure fee details
8     class Procedure:
9         def __init__(self, pcode, pname, pfee):
10             self.procedure_name = pname
11             self.procedure_code = pcode
12             self.procedure_fee = pfee
13
14     # Patient class with name, address, gender and dental procedure details extended from class procedure and hospital
15     class Patient(Hospital, Procedure): # Multiple Inheritance
16         total_patient = 0 # class attribute for counting number of in hospital
17
18         def __init__(self, pid, pname, page, psex, paddress, pcode, pname, pfee):
19             super(Patient, self).__init__(pname, paddress) # Super class Hospital call for Patient Class
20             Procedure.__init__(self, pcode, pname, pfee) # Call for __init__ Procedure
21             self.__patient_id = pid # Defining patient ID as private
22             self.patient_name = pname
23             self.patient_age = page
24             self.__class__.total_patient += 1 # Incrementing Patient Class by 1
25
26         def patient_display(self):
27             print('Patient Name:', self.patient_name, 'Hospital Procedure Undergone:',
28                   self.procedure_name, 'Fee paid of $', self.procedure_fee)
29
30         def getpatient_id(self): # Function to return Private Patient ID
31             return self.__patient_id
32
```

```

32
33 # Hospital Staff Class with Staff ID and Staff Type
34 class StaffHospitalis:
35     def __init__(self, scode, stype, hname, haddress):
36         super(Staff, self).__init__(hname, haddress)
37         self.staff_code = scode
38         self.staff_type = stype
39
40 # Doctor Class
41 class Doctor(Staff): # Multilevel Inheritance logic implemented here
42     total_doctor = 0 # Class attribute for counting number of doctors
43
44     def __init__(self, did, name, qual, city, spec, scode, stype, hname, haddress):
45         super(Doctor, self).__init__(scode, stype, hname, haddress) # Call to base class Staff using super method
46         self.__doc_id = did # Defining Doctor ID as Private data member
47         self.doc_name = name
48         self.doc_qual = qual
49         self.doc_city = city
50         self.doc_specaility = spec
51         self.__class__.total_doctor += 1 # Incrementing Doctor Count by 1
52
53     def doctor_display(self):
54         print('Doctor Name :', self.doc_name, 'Qualification:', self.doc_qual,
55               'Speciality:', self.doc_specaility, 'Hospital', self.hname)
56
57     def getdoctor_id(self): # Function to return private Doctor ID
58         return self.__doc_id
59
60 # Nurse Class
61 class Nurse(Staff):
62     total_nurse = 0 # Class attribute for counting number of Nurses
63
64     def __init__(self, nid, name, age, qual, city, scode, stype, hname, haddress):

```

Program a code which download a webpage contains a table using Request library, then parse the page using BeautifulSoup library. You should save all

the information of the table in a file.

Sample input: <https://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015>

Sample output: Save the table in this link into a file

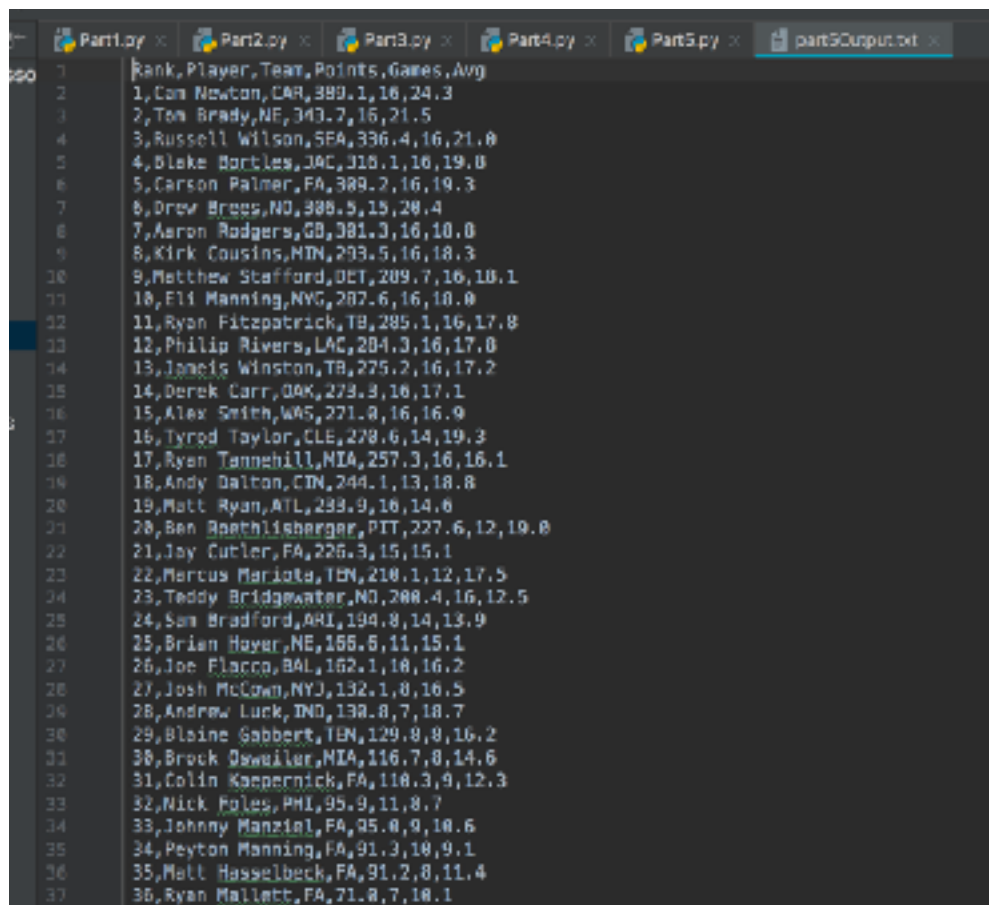
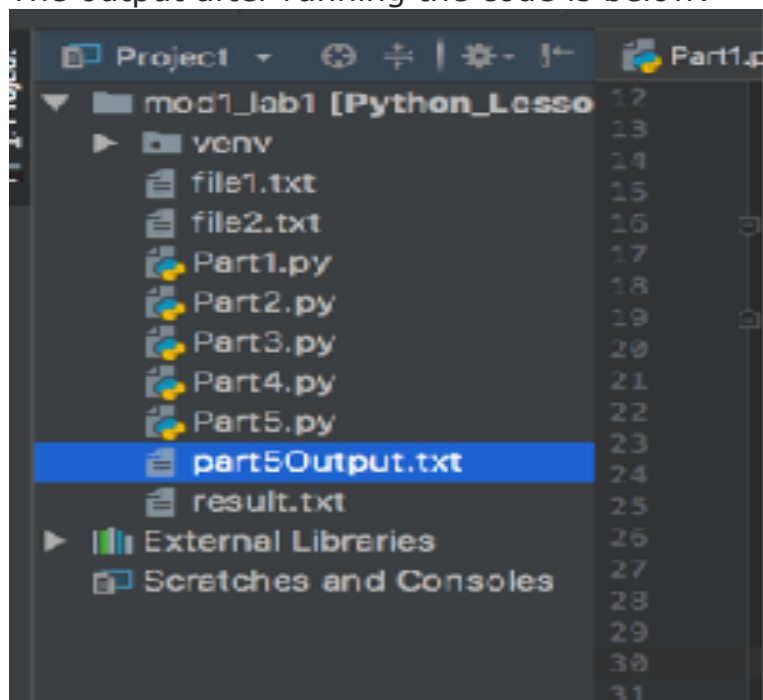
## Solution:

1. Get the url and fetch the contents of that web page.
2. The html is parsed using the BeautifulSoup libraries.
3. Open another text file as output to save the contents parsed from this web page.
4. Extract the data from the given url using the inbuilt functions and write them to the output text file.

Code snippet is attached below:

```
Part1.py x Part2.py x Part3.py x Part4.py x Part5.py x
1 import urllib.request
2 from bs4 import BeautifulSoup
3 import codecs
4
5 # Defining the url and doing parse operation from request response
6 url = "https://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015"
7 source = urllib.request.urlopen(url)
8 # plan_txt = source.text
9 soup = BeautifulSoup(source, "html.parser")
10
11 # logic to extract table header(Column Name) from <thead> tag
12 table_head = soup.table.thead
13 table_head_rows = table_head.findAll('tr')
14
15 header = [] # list to hold all the head column name using
16 for tr in table_head_rows:
17     th = tr.find_all('th')
18     head = [h.text for h in th]
19     header.extend(head)
20
21 # Logic to write header(Column Name) using utf8 coding encoding from Codec registry base classes
22 fl = codecs.open('part5Output.txt', 'wb', 'utf8')
23 head_line = ','.join(header)
24 fl.write(head_line + u'\r\n')
25 fl.close()
26
27 # logic to extract table data from within <tbody> tag
28 table_data = soup.table.tbody
29 table_data_rows = table_data.findAll('tr')
30 rows_data = [] # list to hold table data from <tbody>
31
32 for tr in table_data_rows:
33     td = tr.find_all('td')
34     data = [d.text for d in td]
35     rows_data.append(data)
36
37 # logic to append data record one by from list rows_data
38 for row in rows_data:
39     fl = codecs.open('part5Output.txt', 'ab', 'utf8')
40     data_line = ','.join(row)
41     fl.write(data_line + u'\r\n')
42     fl.close()
```

The output after running the code is below:



# References:

---

- <https://stackoverflow.com/questions/15137769/how-to-delete-same-words-from-different-text-file-using-python>
- <https://stackabuse.com/how-to-copy-a-file-in-python/>
- <https://docs.python.org/2/library/shutil.html>
- <https://www.sanfoundry.com/python-program-copy-contents-file-another/>
- <https://www.journaldev.com/14408/python-read-file-open-write-delete-copy>
- <https://unix.stackexchange.com/questions/145079/remove-all-lines-in-file-a-which-contain-the-strings-in-file-b>