

**CSEE 5590 0001- Python and Deep
Learning
Fall 2018**

Python Lab Assignment 1

Submitted On 9/18/2018

Name: Alexandria Piatt and Alexander
Larios

Class IDs (respectively): 23 15

TABLE OF CONTENTS

1. Author
2. Objective
3. Features
4. Configuration
5. Input/Output Screenshots
6. Implementation & Code Snippet
7. Limitations
8. References

AUTHORS

This is the lab report for the first Lab Assignment in CS 5590 0001 Special Topics. The authors of this document are ALEXANDRIA PIATT (ID: 23) and ALEXANDER LARIOS (ID: 15), both Seniors pursuing a Bachelors in Computer Science at University of Missouri Kansas City. The course is taught by Saria Goudarzvand.

OBJECTIVE

This lab's objective was to solidify the in class work to learn basic python structures and concepts. The specific skills practiced in this lab are listed below.

- Sets
- Web Scraping
- Searching through strings and list manipulation
- Classes and inheritance

This is accomplished by completing a series of six tasks. A brief description of these tasks are:

- Searching a string to find the first non-repeating characters
- After being given a list of students in two separate classes, return students that are in one class but not the other.
- Using BeautifulSoup, return a table on a given website.

FEATURES

Task 1:

Find the first non repeated character in a string

The program can take a user inputted string. It will search through the string for unique characters and will return the first one it finds.

Task 2:

Read in two text files and output the words from file1 that are not present in file2

The program reads in both text files and adds all the elements separated by spaces into two arrays. The words in the array from the second file are then checked against the words from the first file and if a word is present it is removed from the array of words from the first file. Once all the words have been checked the new array of words from the first file that were not present in the second file are output.

Task 3:

Find students in the python class but not in the web application class

Given a list of students for each class, the code will return which students are enrolled in only the python class.

Task 4:

Create the classes for a hospital admission system

The program creates a object of every class and calls their methods and then calls their print details() method to output their attribute data.

Task 6:

Using requests library output the results of a table from a specified website to a file

Given a website (listed in the assignment), use the requests library (and BeautifulSoup) to scrape the table from the website and out

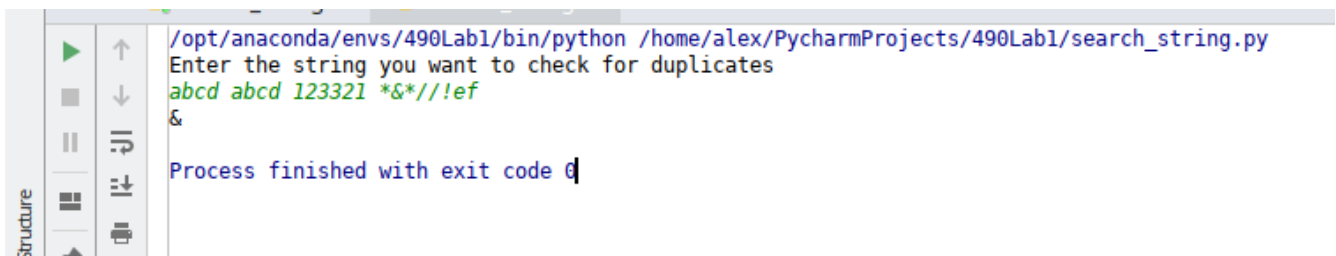
CONFIGURATION

All of the code in this lab was written and built using PYCHARM IDE, in an Anaconda3 environment and using Python 3.6.

INPUT/OUTPUT SCREENSHOTS

Task 1:

The below screenshot shows the user inputted string and returns the first unique character. The string is not just restricted to letters and numbers, but also works with characters such as !@#\$%^&*() as well.



```
/opt/anaconda/envs/490Lab1/bin/python /home/alex/PycharmProjects/490Lab1/search_string.py
Enter the string you want to check for duplicates
abcd abcd 123321 *&*///!ef
a
Process finished with exit code 0
```

Task 2:

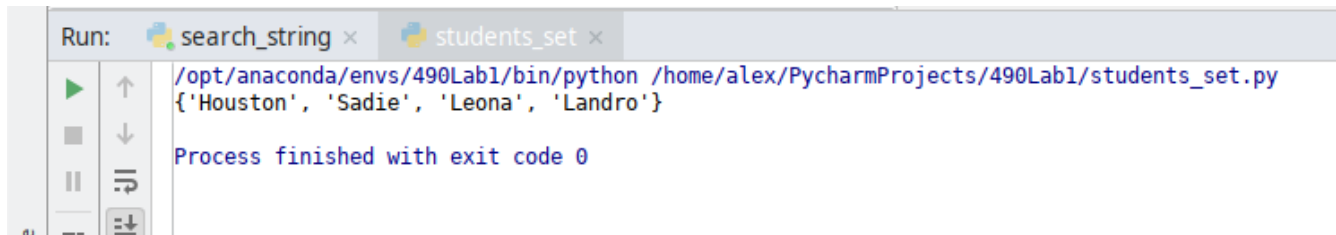
The below screenshot shows the output of FileDifference.py, Three arrays are output, one from the first text file, one from the second text file, and the words from the first text file that were not present in the second text file.



```
C:\Users\Snow\Anaconda3\envs\Lab1\python.exe "C:/Users/Snow/Dropbox/Fall 18/490/Lab1/fileDifference.py"
['time', 'going', 'learn', 'to', 'write', 'programs', 'recognize', 'objects', 'images', 'using', 'deep', 'learning.', 'In', 'words', 'weare', 'going', 'to', 'explain', 'black', 'magic', 'that', 'allows', 'Google', 'Photos', 'to', 'search', 'photos', 'in', 'the', 'picture']
['This', 'we', 'to', 'are', 'in', 'the', 'that', 'your', 'on', 'based', 'what', 'is', 'how', 'other']
['time', 'going', 'learn', 'to', 'write', 'programs', 'recognize', 'objects', 'images', 'using', 'deep', 'learning.', 'In', 'words', 'weare', 'going', 'to', 'explain', 'black', 'magic', 'that', 'allows', 'Google', 'Photos', 'to', 'search', 'photos', 'in', 'the', 'picture']
Process finished with exit code 0
```

Task 3:

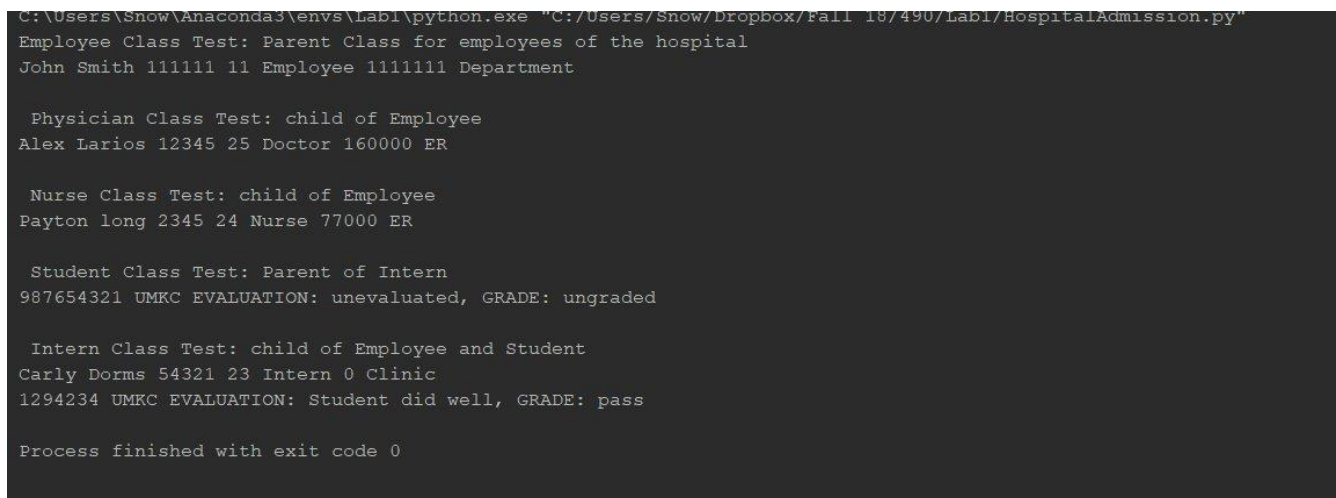
The below screenshot shows the output from the small script that determines the students that are only enrolled in Python Class but not Web Applications class.



```
Run: search_string x students_set x
/opt/anaconda/envs/490Lab1/bin/python /home/alex/PycharmProjects/490Lab1/students_set.py
{'Houston', 'Sadie', 'Leona', 'Landro'}
Process finished with exit code 0
```

Task 4:

The below screenshot shows output from the program that constructs objects of multiple hospital classes and outputs their attribute details.



```
C:\Users\Snow\Anaconda3\envs\Lab1\python.exe "C:/Users/Snow/Dropbox/Fall 18/490/Lab1/HospitalAdmission.py"
Employee Class Test: Parent Class for employees of the hospital
John Smith 111111 11 Employee 1111111 Department

Physician Class Test: child of Employee
Alex Larios 12345 25 Doctor 160000 ER

Nurse Class Test: child of Employee
Payton long 2345 24 Nurse 77000 ER

Student Class Test: Parent of Intern
987654321 UMKC EVALUATION: unevaluated, GRADE: ungraded

Intern Class Test: child of Employee and Student
Carly Dorms 54321 23 Intern 0 Clinic
1294234 UMKC EVALUATION: Student did well, GRADE: pass

Process finished with exit code 0
```

Task 6:

The code takes a URL and parses the HTML code. It finds a table and then outputs the table on the page to a text document.

The below screenshot shows the tables outputted to the file. There are multiple tables on the page, so there are multiple outputs.

cellpadding="0" cellspacing="0" class="table table-bordered table-striped table-hover" id="data">	
<thead>	
<tr>	
<th style="width:60px;">Rank</th>	
<th class="player-label">Player</th><th style="width:70px;">Team</th><th style="width:60px;">Points</th>	
<th style="width:60px;">Game</th><th style="width:60px;">Avg</th>	
</tr>	
</thead>	
<tbody>	
<tr><td class="center">16</td><td class="player-label">Cam Newton</td><td class="center">CAR</td><td class="center">389.1</td><td class="center">16</td><td class="center">16</td><td class="center">24.3</td></tr><tr><td class="center">27</td><td class="player-label">Tony Brady</td><td class="center">NE</td><td class="center">343.7</td><td class="center">27</td><td class="center">27</td><td class="center">25.7</td></tr><tr><td class="center">3</td><td class="player-label">Russell Wilson</td><td class="center">SEA</td><td class="center">336.4</td><td class="center">3</td><td class="center">3</td><td class="center">26.4</td></tr><tr><td class="center">16</td><td class="center">21.0</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Blake Bortles</td><td class="center">JAC</td><td class="center">316.1</td><td class="center">16</td><td class="center">16</td><td class="center">19.8</td></tr><tr><td class="center">5</td><td class="center">5</td><td class="center">5</td><td class="center">5</td><td class="center">5</td><td class="center">5</td><td class="player-label">Carson Palmer</td><td class="center">FA</td><td class="center">309.2</td><td class="center">5</td><td class="center">5</td><td class="center">19.8</td></tr><tr><td class="center">6</td><td class="center">6</td><td class="center">6</td><td class="center">6</td><td class="center">6</td><td class="center">6</td><td class="player-label">Drew Brees</td><td class="center">NO</td><td class="center">306.5</td><td class="center">6</td><td class="center">6</td><td class="center">26.1</td></tr><tr><td class="center">7</td><td class="center">7</td><td class="center">7</td><td class="center">7</td><td class="center">7</td><td class="center">7</td><td class="player-label">Aaron Rodgers</td><td class="center">GB</td><td class="center">301.3</td><td class="center">7</td><td class="center">7</td><td class="center">23.8</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Kirk Cousins</td><td class="center">MIN</td><td class="center">293.5</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Matthew Stafford</td><td class="center">DET</td><td class="center">289.7</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Eli Manning</td><td class="center">NYG</td><td class="center">287.6</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Ryan Fitzpatrick</td><td class="center">TB</td><td class="center">285.1</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Philip Rivers</td><td class="center">LAC</td><td class="center">284.3</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Jameis Winston</td><td class="center">TB</td><td class="center">275.2</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Derek Carer</td><td class="center">OAK</td><td class="center">273.3</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Alex Smith</td><td class="center">WAS</td><td class="center">271.6</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="center">16</td><td class="player-label">Tyrod Taylor</td><td class="center">CLE</td><td class="center">278.6</td><td class="center">16</td><td class="center">16</td><td class="center">18.3</td></tr><tr><td class="center">16</td><td class="center">16</td><td class="center">	

[illegible]

IMPLEMENTATION & CODE SNIPPET

Task 1:

Python has flexible variable typing. By breaking the string into a list of characters, it was simple to iterate through the list and count each character's occurrences in the list. Because of the flexibility of the string variable type, there are not any restrictions on what the user can type in.

```
# get the user input
sString = input("Enter the string you want to check for duplicates\n")
# remove spaces
sString = sString.replace(" ", "")
# convert the string to a list of characters
listString = list(sString)
returnList = []
# iterate through the characters
for i in listString:
    # in python the count function counts the number of occurrences of a specific character in a list
    result = listString.count(i)
    # if the returned integer is 1 then add the character to the unique character list
    if result == 1:
        returnList.append(i)
# print out the first character returned
print(returnList[0])
```

Task 2:

There are two functions. One for reading in the initial text files and adding the words to two arrays. The second function takes both arrays and checks if any of the words in the second array are present in the first, if so then it removes that word from the first array

```
#This program takes the words in file1.txt and file2.txt and returns the words from file1 that were not in file 2.
file1Words = []
file2Words = []
# read_in: reads the words in the text file into an array
# @param fname: string file name
def read_in(fname):
    fileWords = []
    with open(fname, 'r') as f:
        for line in f:
            for word in line.split(' '):
                fileWords.append(word)
    return fileWords
# remove words: loops through the words in the second array and checks if they are present in the first array, if the word
# exists in both arrays then it removed from the first array of words from file1
# @param firstWords: array containing string elements
# @param secondWords: array containing string elements
def remove_words(firstWords, secondWords):
    for word in secondWords:
        if word in firstWords:
            firstWords.remove(word)
    return firstWords
file1Words = read_in('file1.txt')
file2Words = read_in('file2.txt')
result = remove_words(file1Words, file2Words)
# print first file words and second file words then the result
print(file1Words)
print(file2Words)
print(result)

#NOTE: I noticed in the example that the words included the punctuation
# so I used a space as a delimiter when reading the words into the arrays
```

Task 3:

Finding the differences between two unique lists of data (sets) in python is very

```
ren_string.py  students_set.py  table_scraping.py
#students enrolled in web application class
web_application = {"Susan", "Alex", "Courtney", "John", "Sam", "Louisa"}
#students enrolled in python class
python_class = {"Alex", "Landro", "Leona", "Susan", "Houston", "Sadie", "John"}
#print the difference between the two sets
print(python_class - web_application)
```

simple in python. With the use of sets, it will only take a few lines of code.

Task 4:

5 classes are initialized. 3 of the classes are child classes that inherent from Employee or both Employee and Student. Objects are then constructed at the end and a call is made to their print_details() method to output their attributes.

```

1  #Patient Class
2  class Patient:
3      def __init__(self, first, phone, last, ID, age, primary):
4          self.first = first
5          self.last = last
6          self.phone = phone
7          self.ID = ID
8          self.age = age
9          self.salary = primary
10
11     def print_patient_details(self):
12         print(self.first, self.last, self.phone, self.ID, self.age, self.salary)
13
14     #Student Class
15     class Student:
16         def __init__(self, sID, school):
17             self.sID = sID
18             self.school = school
19             self.grade = "ungraded"
20             self.evaluation = "unevaluated"
21
22         #private member
23         def __give_grade(self, grade, eval):
24             self.grade = grade
25             self.evaluation = eval
26
27         def evaluate(self, evaluation, grade):
28             self.__give_grade(grade, evaluation)
29
30         def print_student_details(self):
31             print(self.sID, self.school, "EVALUATION: " + self.evaluation + ", ", "GRADE: " + self.grade)
32
33     #Parent class for Employees of the hospital
34     class Employee:
35         def __init__(self, first, last, ID, age, title, salary, department):
36             self.first = first
37             self.last = last
38             self.ID = ID
39             self.age = age
40             self.title = title
41             self.salary = salary
42             self.department = department
43
44         def print_details(self):
45             print(self.first, self.last, self.ID, self.age, self.title, self.salary, self.department)
46
47     #Intern Class that is a child of both Employee and student
48     class Intern(Employee, Student):
49         def __init__(self, first, last, ID, age, title, salary, department, sID, school):
50             Employee.__init__(self, first, last, ID, age, title, salary, department)
51             Student.__init__(self, sID, school)
52
53     #Physician class that is child of Employee and uses super()
54     class Physician(Employee):
55         def __init__(self, first, last, ID, age, title, salary, department):
56             super().__init__(first, last, ID, age, title, salary, department)
57
58     #Nurse class that is child of Employee
59     class Nurse(Employee):
60         def __init__(self, first, last, ID, age, title, salary, department, nurseManager):
61             Employee.__init__(self, first, last, ID, age, title, salary, department)
62             self.nurseManager = nurseManager
63
64
65     #Tests
66     print("Employee Class Test: Parent Class for employees of the hospital")
67     testEmp = Employee("John", "Smith", 11111, 11, "Employee", 111111, "Department")
68     testEmp.print_details()
69
70     print("\n Physician Class Test: child of Employee")
71     testDR = Physician("Alex", "Larios", 12345, 25, "Doctor", 160000, "ER")
72     testDR.print_details()
73
74     print("\n Nurse Class Test: child of Employee")
75     testNurse = Nurse("Payton", "Long", 2345, 24, "Nurse", 77000, "ER", "Manager: cindy")
76     testNurse.print_details()

```

Task 6:

The first step of working with websites in python is importing the appropriate libraries. In this case, I used BeautifulSoup and requests. The first step is to process the url and store the data from the website in a variable. This is done using the request library. Next is to convert the raw data from the website into a text format. Afterwards, use BeautifulSoup to parse the text into an object that uses the power of the BeautifulSoup library. The find_all function in the BeautifulSoup library searches through the parsed html data and returns all of the tables on the site. The last thing to do is output it to a text file.

```
from bs4 import BeautifulSoup
import requests

# cache link
link = "https://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015"
source_material = requests.get(link)

# convert saved website data as a text file
source_text = source_material.text

# parse with beautiful soup library
soup_response = BeautifulSoup(source_text, "html.parser")

# find the table
table = soup_response.find_all('table')

# output table to
file_out = open('table_out.txt', 'w')
file_out.write(str(table))
```

LIMITATIONS

REFERENCES

<https://docs.python.org/3/tutorial/inputoutput.html>

<https://www.analyticsvidhya.com/blog/2015/10/beginner-guide-web-scraping-beautiful-soup-python/>