

Lab Assignment-1

NAME: HARISH CHANDRA JYOSHI

CLASS ID: 12

TEAM ID: 04

EMAIL ID: hjddh@mail.umkc.edu

NAME: ATLURI VENKATA AKHILA KRISHNA

CLASS ID: 02

TEAM ID: 04

EMAIL ID: vagq2@mail.umkc.edu

Video link:

TASK 1:

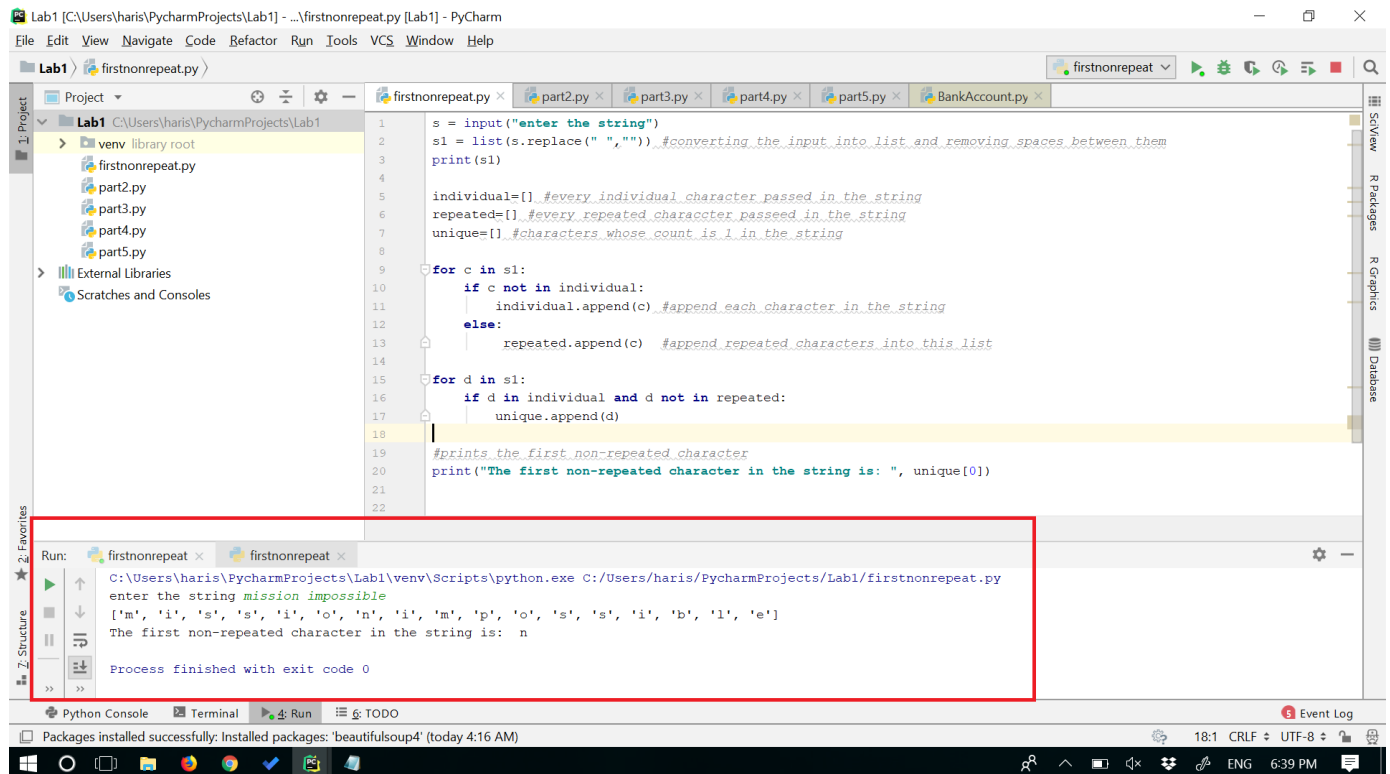
Objective

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

string 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**)

Entered an input as a string and converted the string into a list, later used for loop and if-else statements to find the first non-repeated character

Lab Assignment-1



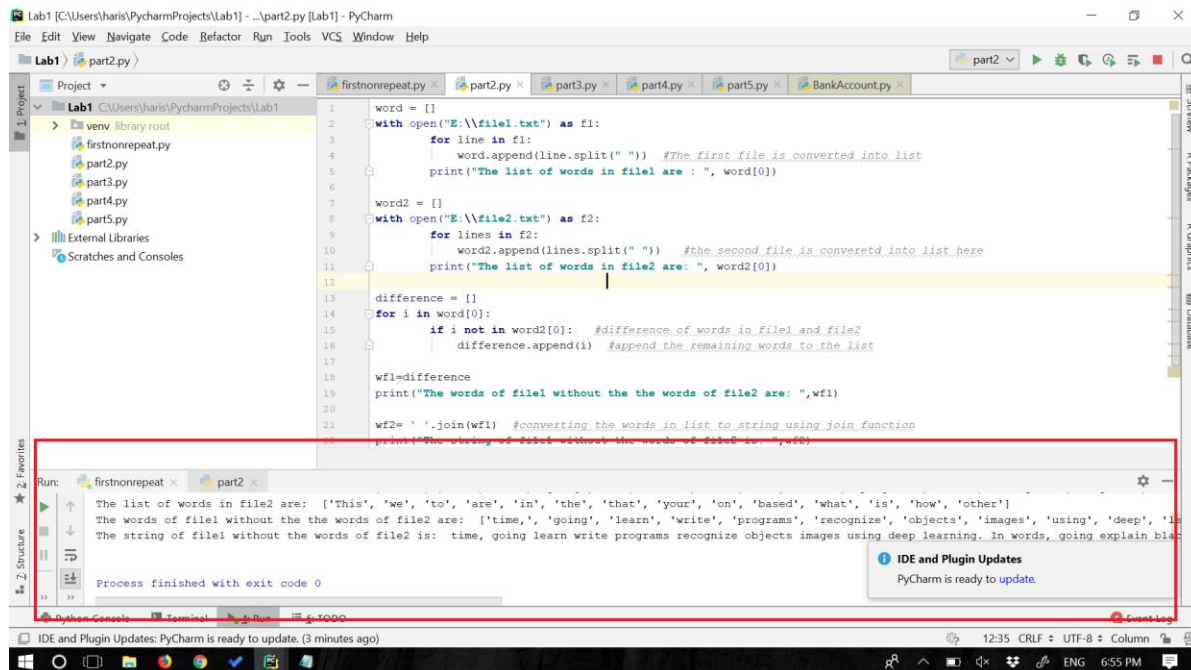
TASK 2:

Objective

2. Suppose you have two files. And what is inside the files are as follows: 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.

Opened both the files and converted the string in the file to a list of words and later using for loop removed the words in file1 which is inside file2.

Lab Assignment-1



The screenshot shows the PyCharm IDE with a project named 'Lab1'. The main editor displays a Python script named 'part2.py'. The script reads two text files, 'file1.txt' and 'file2.txt', into lists. It then calculates the difference between the two lists, finding words in 'file1.txt' that are not in 'file2.txt'. The output is printed to the console.

```
1 word = []
2 with open("E:\\file1.txt") as f1:
3     for line in f1:
4         word.append(line.split(" ")) #The first file is converted into list
5     print("The list of words in file1 are : ", word[0])
6
7 word2 = []
8 with open("E:\\file2.txt") as f2:
9     for lines in f2:
10        word2.append(lines.split(" ")) #the second file is converted into list here
11    print("The list of words in file2 are : ", word2[0])
12
13 difference = []
14 for i in word[0]:
15     if i not in word2[0]: #difference of words in file1 and file2
16         difference.append(i) #append the remaining words to the list
17
18 wfl=difference
19 print("The words of file1 without the the words of file2 are : ",wfl)
20
21 wf2= ' '.join(wfl) #converting the words in list to string using join function
22 print("The string of file1 without the words of file2 is : ",wf2)
```

The Run console shows the following output:

```
The list of words in file2 are: ['This', 'we', 'to', 'are', 'in', 'the', 'that', 'your', 'on', 'based', 'what', 'is', 'how', 'other']
The words of file1 without the the words of file2 are: ['time', 'going', 'learn', 'write', 'programs', 'recognize', 'objects', 'images', 'using', 'deep', '1']
The string of file1 without the words of file2 is: time, going learn write programs recognize objects images using deep learning. In words, going explain bla
```

Process finished with exit code 0

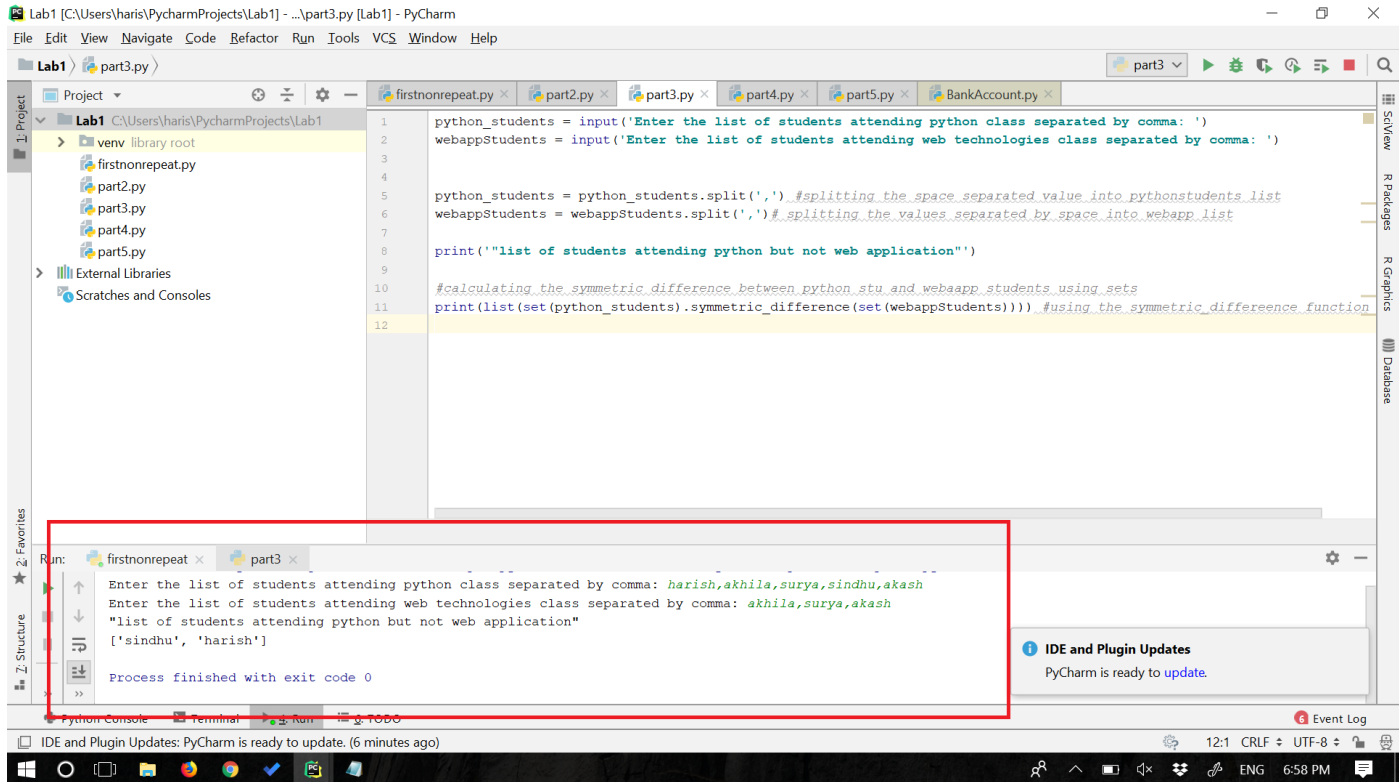
TASK 3:

Objective:

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 "python" classes but not "Web Application"***

Entered the inputs of students attending python and web class separated by a comma, converted the string into a list and later converted the list into sets and calculated the difference between the sets and found out the students attending python class but not the web application.

Lab Assignment-1



The screenshot shows the PyCharm IDE interface. The main editor displays a Python script in `part3.py` with the following code:

```
1 python_students = input('Enter the list of students attending python class separated by comma: ')
2 webappStudents = input('Enter the list of students attending web technologies class separated by comma: ')
3
4
5 python_students = python_students.split(',') #splitting the space separated value into pythonstudents_list
6 webappStudents = webappStudents.split(',') #splitting the values separated by space into webapp_list
7
8 print("list of students attending python but not web application")
9
10 #calculating the symmetric difference between python_stu and websapp_students using sets
11 print(list(set(python_students).symmetric_difference(set(webappStudents)))) #using the symmetric difference function
12
```

The Run tool window at the bottom shows the execution output:

```
Enter the list of students attending python class separated by comma: harish,akhila,surya,sindhu,akash
Enter the list of students attending web technologies class separated by comma: akhila,surya,akash
"list of students attending python but not web application"
['sindhu', 'harish']
Process finished with exit code 0
```

A red rectangle highlights the Run tool window. An IDE update notification is visible on the right side of the Run window.

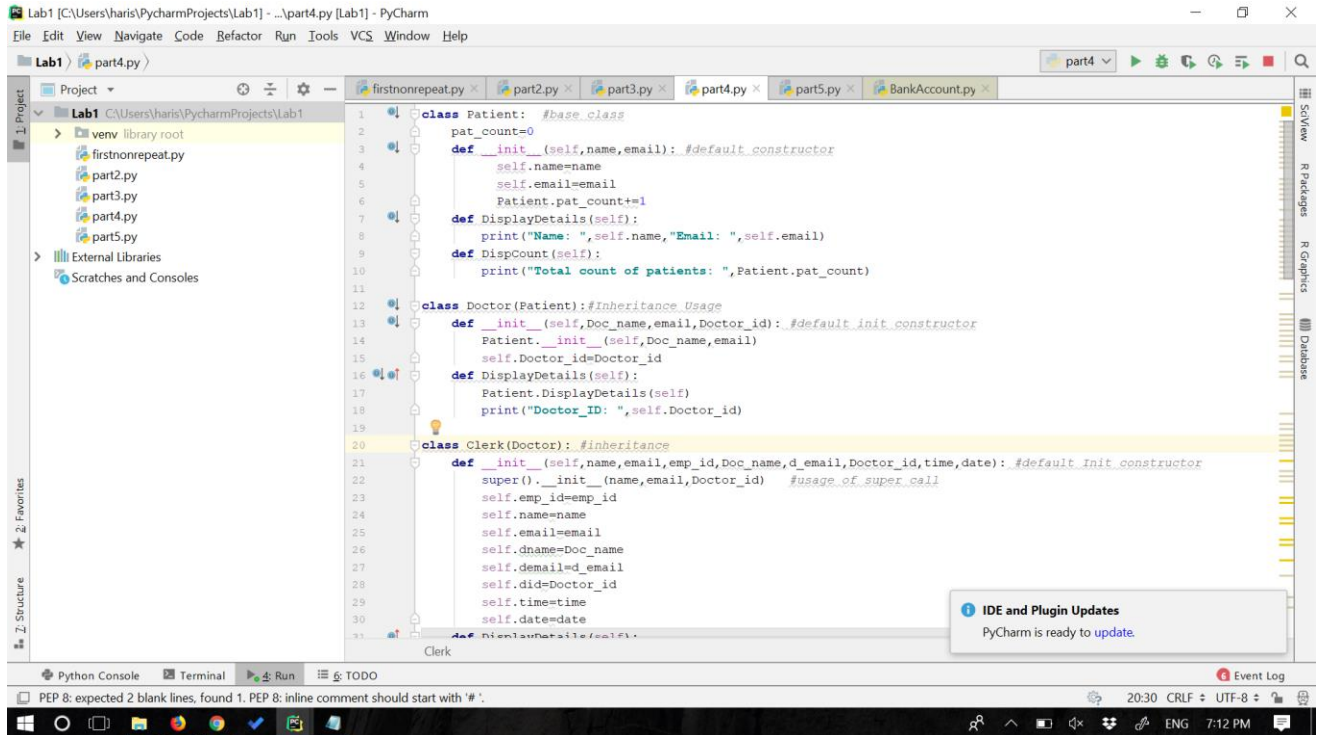
TASK 4:

Objective:

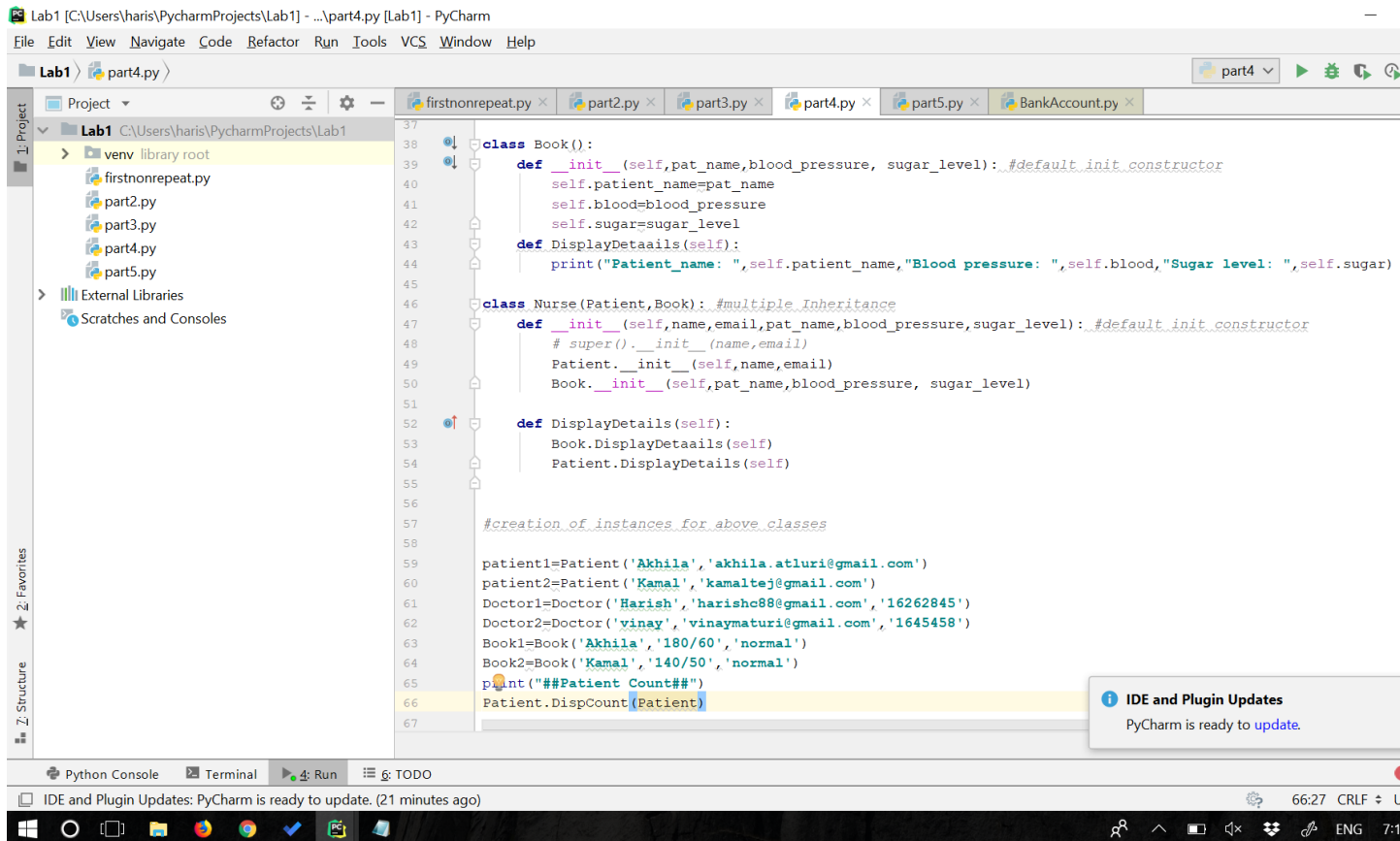
4. Write a python program to create the following management systems. a. Hospital admission System (e.g. classes Patient, Doctor, Medical Admission Clerk, Book, Nurse, etc.) Prerequisites: a. Your code should have at least five classes b. Your code should have *init* constructor in all the classes c. 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

Created 5 classes namely Patient, Doctor, clerk, book, nurse. Used inheritance between the classes and relationship between the classes is shown.

Lab Assignment-1

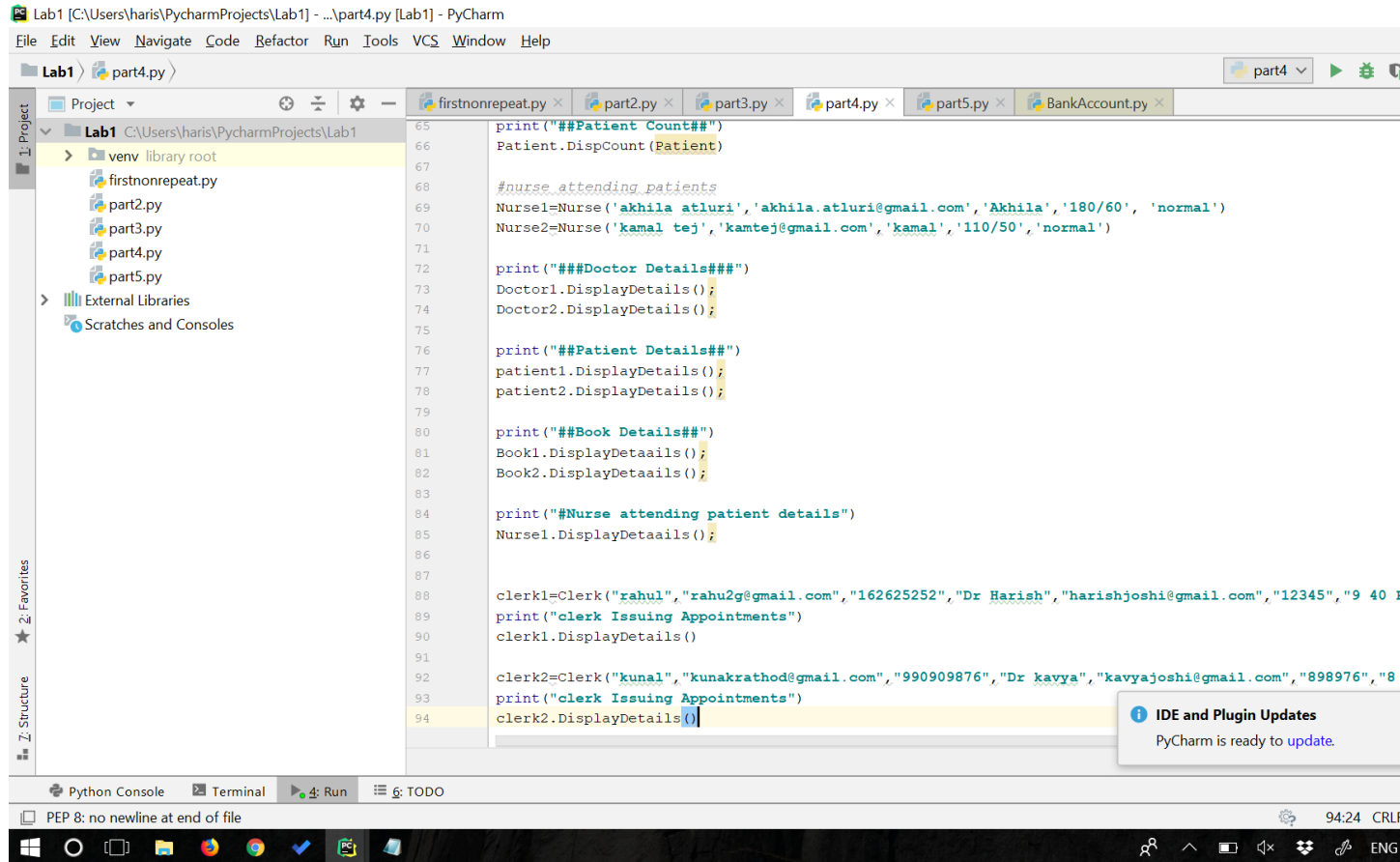


```
1 class Patient: #base class
2     pat_count=0
3     def __init__(self,name,email): #default constructor
4         self.name=name
5         self.email=email
6         Patient.pat_count+=1
7     def DisplayDetails(self):
8         print("Name: ",self.name,"Email: ",self.email)
9     def DispCount(self):
10        print("Total count of patients: ",Patient.pat_count)
11
12 class Doctor(Patient):#Inheritance Usage
13     def __init__(self,Doc_name,email,Doctor_id): #default init constructor
14         Patient.__init__(self,Doc_name,email)
15         self.Doctor_id=Doctor_id
16     def DisplayDetails(self):
17         Patient.DisplayDetails(self)
18         print("Doctor_ID: ",self.Doctor_id)
19
20 class Clerk(Doctor): #inheritance
21     def __init__(self,name,email,emp_id,Doc_name,d_email,Doctor_id,time,date): #default init constructor
22         super().__init__(name,email,Doctor_id) #usage of super call
23         self.emp_id=emp_id
24         self.name=name
25         self.email=email
26         self.dname=Doc_name
27         self.demail=d_email
28         self.did=Doctor_id
29         self.time=time
30         self.date=date
31     def DisplayDetails(self):
```



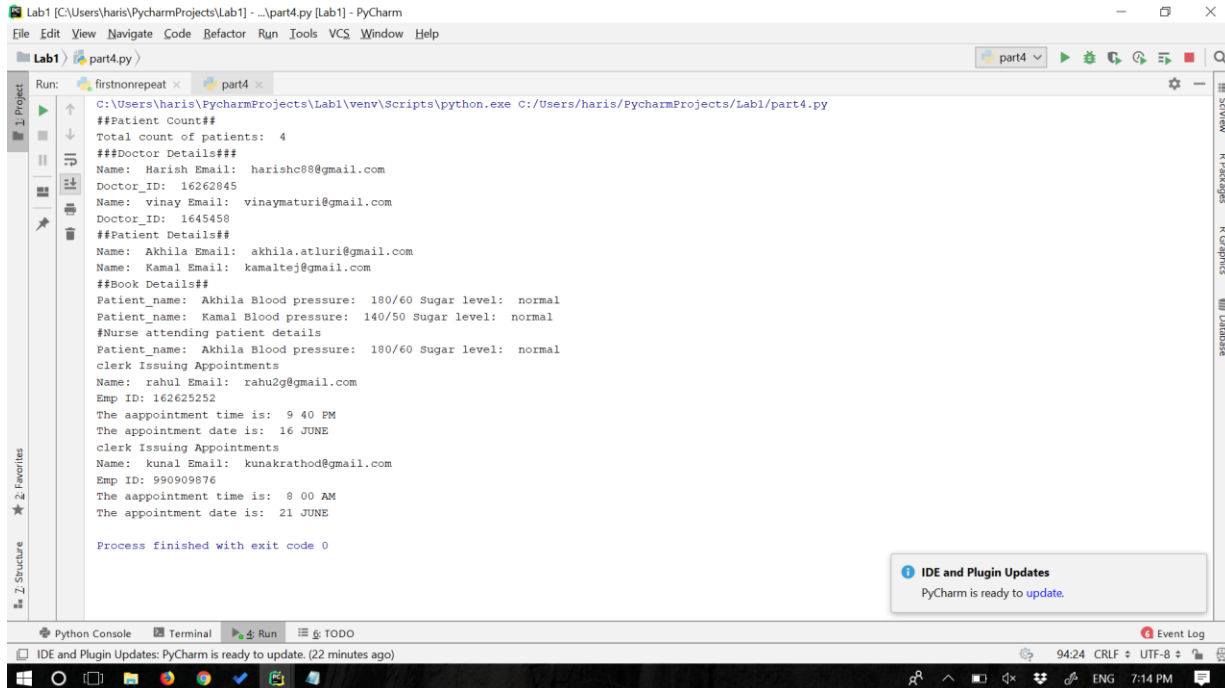
```
37
38 class Book():
39     def __init__(self,pat_name,blood_pressure, sugar_level): #default init constructor
40         self.patient_name=pat_name
41         self.blood=blood_pressure
42         self.sugar=sugar_level
43     def DisplayDetails(self):
44         print("Patient name: ",self.patient_name,"Blood pressure: ",self.blood,"Sugar level: ",self.sugar)
45
46 class Nurse(Patient,Book): #multiple inheritance
47     def __init__(self,name,email,pat_name,blood_pressure,sugar_level): #default init constructor
48         # super().__init__(name,email)
49         Patient.__init__(self,name,email)
50         Book.__init__(self,pat_name,blood_pressure, sugar_level)
51
52     def DisplayDetails(self):
53         Book.DisplayDetails(self)
54         Patient.DisplayDetails(self)
55
56
57 #creation of instances for above classes
58
59 patient1=Patient('Akhila','akhila.atluri@gmail.com')
60 patient2=Patient('Kamal','kamaltej@gmail.com')
61 Doctor1=Doctor('Harish','harishc88@gmail.com','16262845')
62 Doctor2=Doctor('vinay','vinaymaturi@gmail.com','1645458')
63 Book1=Book('Akhila','180/60','normal')
64 Book2=Book('Kamal','140/50','normal')
65 print("##Patient Count##")
66 Patient.DispCount(Patient)
67
```

Lab Assignment-1



Output:

Lab Assignment-1



```
Run: firstnonrepeat x part4 x
C:\Users\haris\PycharmProjects\Lab1\venv\Scripts\python.exe C:/Users/haris/PycharmProjects/Lab1/part4.py
##Patient Count##
Total count of patients: 4
###Doctor Details###
Name: Harish Email: harishc88@gmail.com
Doctor_ID: 16262845
Name: vinay Email: vinaymaturi@gmail.com
Doctor_ID: 1645458
##Patient Details##
Name: Akhila Email: akhila.atluri@gmail.com
Name: Kamal Email: kamaltej@gmail.com
##Book Details##
Patient_name: Akhila Blood pressure: 180/60 Sugar level: normal
Patient_name: Kamal Blood pressure: 140/50 Sugar level: normal
#Nurse attending patient details
Patient_name: Akhila Blood pressure: 180/60 Sugar level: normal
clerk Issuing Appointments
Name: rahul Email: rahul2g@gmail.com
Emp ID: 162625252
The aappointment time is: 9 40 PM
The appointment date is: 16 JUNE
clerk Issuing Appointments
Name: kunal Email: kunakrathod@gmail.com
Emp ID: 990909876
The aappointment time is: 8 00 AM
The appointment date is: 21 JUNE

Process finished with exit code 0
```

IDE and Plugin Updates: PyCharm is ready to update. (22 minutes ago)

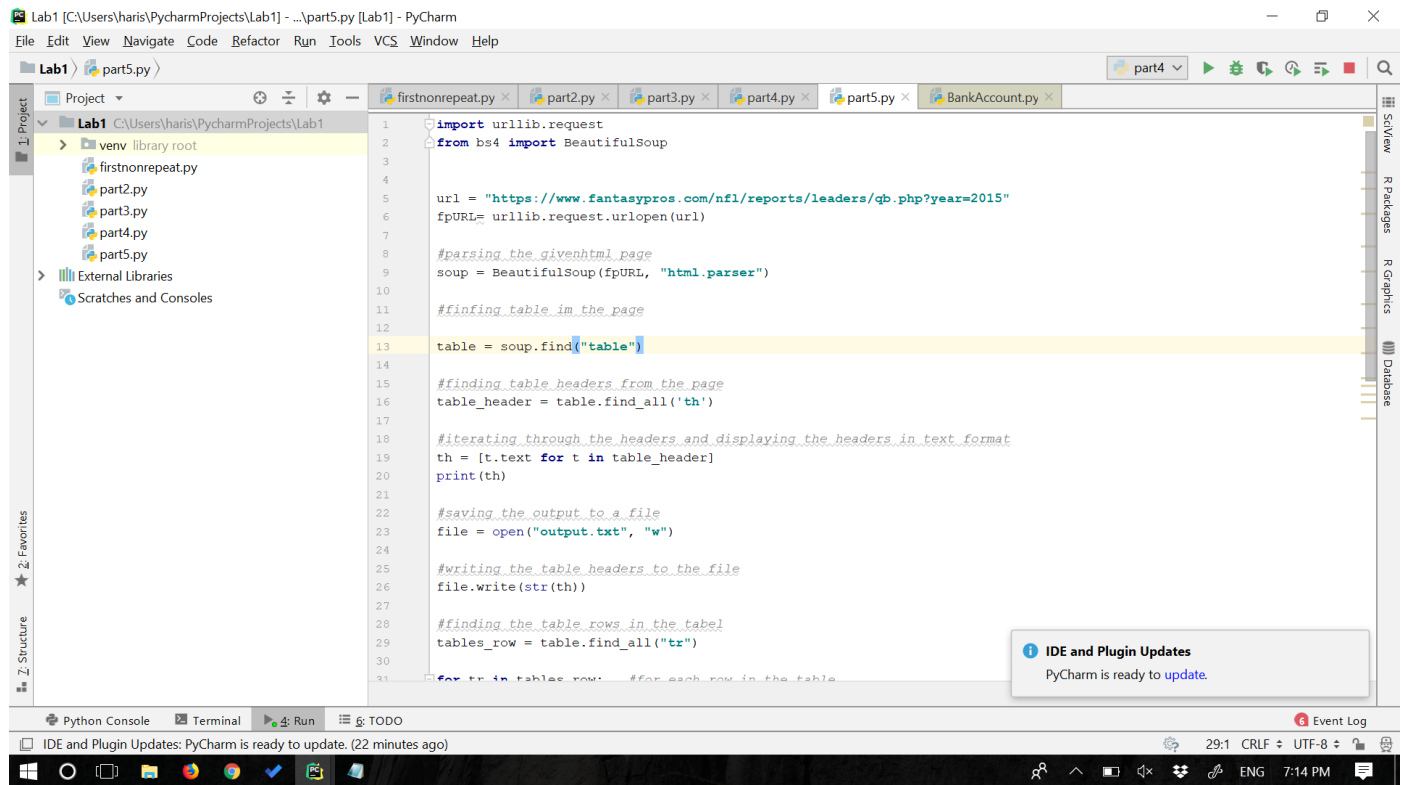
TASK 5:

Objective:

**5. program a code which downloads a webpage contains a table using Request library, then parse the page using Beautiful soup library. You should save all the information in the table in a file. **

Code is programmed which downloads a table in a webpage using request library and the webpage is parsed using beautiful Soup library extract the contents of a web page and all the contents of the table are stored in a file.

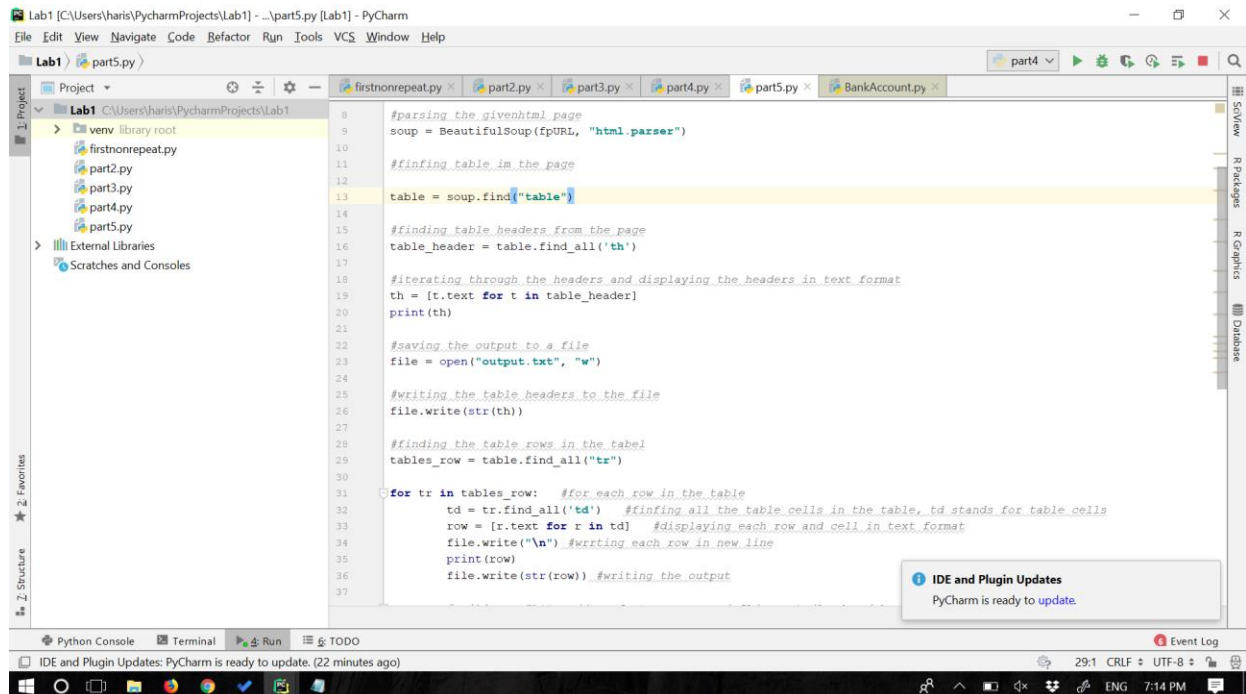
Lab Assignment-1



This screenshot shows the PyCharm IDE with the file `part5.py` open. The code is a Python script that uses `urllib.request` and `BeautifulSoup` to fetch and parse a webpage. The script finds a table, extracts its headers, and saves the output to a file named `output.txt`. The code is as follows:

```
1 import urllib.request
2 from bs4 import BeautifulSoup
3
4
5 url = "https://www.fantasypros.com/nfl/reports/leaders/qb.php?year=2015"
6 fpURL= urllib.request.urlopen(url)
7
8 #parsing the givenhtml page
9 soup = BeautifulSoup(fpURL, "html.parser")
10
11 #finfing table in the page
12
13 table = soup.find("table")
14
15 #finding table headers from the page
16 table_header = table.find_all('th')
17
18 #iterating through the headers and displaying the headers in text format
19 th = [t.text for t in table_header]
20 print(th)
21
22 #saving the output to a file
23 file = open("output.txt", "w")
24
25 #writing the table headers to the file
26 file.write(str(th))
27
28 #finding the table rows in the tabel
29 tables_row = table.find_all("tr")
30
31 for tr in tables_row: #for each row in the table
```

The IDE interface includes a project view on the left, a toolbar at the top, and a status bar at the bottom. A notification for IDE and Plugin Updates is visible in the bottom right corner.



This screenshot shows the PyCharm IDE with the file `part5.py` open. The code is a Python script that uses `urllib.request` and `BeautifulSoup` to fetch and parse a webpage. The script finds a table, extracts its headers, and saves the output to a file named `output.txt`. The code is as follows:

```
8 #parsing the givenhtml page
9 soup = BeautifulSoup(fpURL, "html.parser")
10
11 #finfing table in the page
12
13 table = soup.find("table")
14
15 #finding table headers from the page
16 table_header = table.find_all('th')
17
18 #iterating through the headers and displaying the headers in text format
19 th = [t.text for t in table_header]
20 print(th)
21
22 #saving the output to a file
23 file = open("output.txt", "w")
24
25 #writing the table headers to the file
26 file.write(str(th))
27
28 #finding the table rows in the tabel
29 tables_row = table.find_all("tr")
30
31 for tr in tables_row: #for each row in the table
32     td = tr.find_all('td') #finfing all the table cells in the table, td stands for table cells
33     row = [r.text for r in td] #displaying each row and cell in text format
34     file.write("\n") #wrting each row in new line
35     print(row)
36     file.write(str(row)) #writing the output
```

The IDE interface includes a project view on the left, a toolbar at the top, and a status bar at the bottom. A notification for IDE and Plugin Updates is visible in the bottom right corner.

Output:

Lab Assignment-1

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The project structure on the left shows a project named 'Lab1' with a 'venv' directory and several Python files: 'firstnonrepeat.py', 'output.txt', 'part2.py', 'part3.py', 'part4.py', and 'part5.py'. The main editor window displays the content of 'firstnonrepeat.py', which is a list of NFL player statistics. The list is structured as a list of lists, where each inner list represents a player's data: Rank, Player, Team, Points, Games, and Avg. The data is sorted by points in descending order. A notification in the bottom right corner states 'IDE and Plugin Updates: PyCharm is ready to update.' The bottom status bar shows '1:1 CRLF + UT'.

```
1 ['Rank', 'Player', 'Team', 'Points', 'Games', 'Avg']
2 []
3 ['1', 'Cam Newton', 'CAR', '389.1', '16', '24.3']
4 ['2', 'Tom Brady', 'NE', '343.7', '16', '21.5']
5 ['3', 'Russell Wilson', 'SEA', '336.4', '16', '21.0']
6 ['4', 'Blake Bortles', 'JAC', '316.1', '16', '19.8']
7 ['5', 'Carson Palmer', 'FA', '309.2', '16', '19.3']
8 ['6', 'Drew Brees', 'NO', '306.5', '15', '20.4']
9 ['7', 'Aaron Rodgers', 'GB', '301.3', '16', '18.8']
10 ['8', 'Kirk Cousins', 'MIN', '293.5', '16', '18.3']
11 ['9', 'Matthew Stafford', 'DET', '289.7', '16', '18.1']
12 ['10', 'Eli Manning', 'NYG', '287.6', '16', '18.0']
13 ['11', 'Ryan Fitzpatrick', 'TB', '285.1', '16', '17.8']
14 ['12', 'Philip Rivers', 'LAC', '284.3', '16', '17.8']
15 ['13', 'Jameis Winston', 'TB', '275.2', '16', '17.2']
16 ['14', 'Derek Carr', 'OAK', '273.3', '16', '17.1']
17 ['15', 'Alex Smith', 'WAS', '271.0', '16', '16.9']
18 ['16', 'Tyrod Taylor', 'CLE', '270.6', '14', '19.3']
19 ['17', 'Ryan Tannehill', 'MIA', '257.3', '16', '16.1']
20 ['18', 'Andy Dalton', 'CIN', '244.1', '13', '18.8']
21 ['19', 'Matt Ryan', 'ATL', '233.9', '16', '14.6']
22 ['20', 'Ben Roethlisberger', 'PIT', '227.6', '12', '19.0']
23 ['21', 'Jay Cutler', 'FA', '226.3', '15', '15.1']
24 ['22', 'Marcus Mariota', 'TEN', '210.1', '12', '17.5']
25 ['23', 'Teddy Bridgewater', 'NO', '200.4', '16', '12.5']
26 ['24', 'Sam Bradford', 'ARI', '194.8', '14', '13.9']
27 ['25', 'Brian Hoyer', 'NE', '166.6', '11', '15.1']
28 ['26', 'Joe Flacco', 'BAL', '162.1', '10', '16.2']
29 ['27', 'Josh McCown', 'NYJ', '132.1', '8', '16.5']
30 ['28', 'Andrew Luck', 'IND', '130.8', '7', '18.7']
31 ['29', 'Blaine Gabbert', 'TEN', '129.8', '8', '16.2']
32 ['30', 'Brock Osweiler', 'MIA', '116.7', '8', '14.6']
```

References:

1. <https://www.digitalocean.com/community/tutorials/how-to-scrape-web-pages-with-beautiful-soup-and-python-3>
2. <https://stackoverflow.com/questions/576169/understanding-python-super-with-init-methods>