CS5590/490 - Python and Deep Learning Programming

In Class Programming Report - 3 Class ID 24 - Anurag Thantharate

Date Submitted: 02/09/2020

Video Link: https://www.loom.com/share/88d66a59cab442898ab1ff616ef72c95

- 1. Create a class Employee and then do the following
 - a. Create a data member to count the number of Employees
 - b. Create a constructor to initialize name, family, salary, department
 - c. Create a function to average salary
 - d. Create a Fulltime Employee class and it should inherit the properties of Employee class
 - e. Create the instances of Fulltime Employee class and Employee class and call their member functions.

```
Spring 2020 > Python with DL > III ICP3 > 6 employee.py
         avg_Salary = (Employee.totalSalary / Employee.empCount)
return avg_Salary
       # Create instances for passing employee data
       emp1 = Employee('Andy', 'T', 3000, 'IT')
emp2 = Employee('Brett', 'W', 4000, 'Finance')
emp3 = Employee('Jondy', 'A', 5000, 'Development')
        # Defining sub-class for full time employee inheriting Employee class functions
26
27
       class FulltimeEmp(Employee):
            FTEcount = 0
28
29
30
            FTEtotalSalary = 0
          def __init__(self, name, family, salary, department):
31
32
33
               Employee.__init__(self, name, family, salary, department)
                self.__class__.FTEcount += 1
               self.__class__.FTEtotalSalary = FulltimeEmp.FTEtotalSalary + salary
34
35
36
37
38
39
            # Creating a member function associated with full time employees class
            def fulltimeCount(self):
               fulltimeCount = FulltimeEmp.FTEcount
               return fulltimeCount
           # Create function to call average salary
           def avgFTEsalarv(self):
              avgFTEsalary = (FulltimeEmp.FTEtotalSalary / FulltimeEmp.FTEcount)
                return avgFTEsalary
        # Create instances (passing arguments) of the full time employee and calling their member functions
        emp4 = FulltimeEmp('Kristin', 'Y', 6000, 'Product')
        emp5 = FulltimeEmp('Dale', 'W', 6000, 'PMO')
```

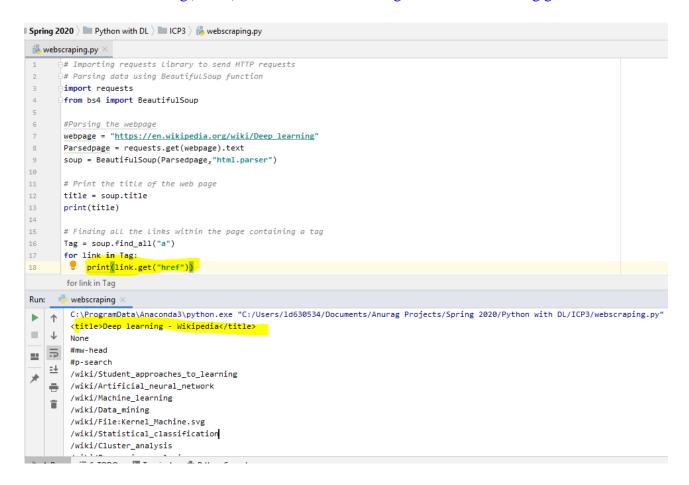
```
# Create instances (passing arguments) of the full time employee and calling their member functions
emp4 = FulltimeEmp('Kristin', 'Y', 6000, 'Product')
emp5 = FulltimeEmp('Dale', 'W', 6000, 'PMO')

print(emp1.name)
print(emp2.name)
print(emp3.name)
print(emp5.name)
print(emp5.name)
(emp3.avg_salary())
print("Average Salary of Consultant (Not Full-time)", emp3.avg_salary())
(emp4.avgFTEsalary())
print("Average Salary of Full time Employee", emp4.avgFTEsalary())

# Calling full time employee function to identify the number of employees
totalEmp = emp5.fulltimeCount() + emp1.empCount
print("Total no. of Employees in the Company are ", totalEmp)
```

2. Web Scraping - Write a simple program that parse a Wiki page mentioned below and follow the instructions: https://en.wikipedia.org/wiki/Deep_learning

- a. Print out the title of the page
- b. Find all the links in the page ('a' tag)
- c. Iterate over each tag(above) then return the link using attribute "href" using get



- 3. Using NumPy create random vector of size 15 having only Integers in the range 1-20.
 - a. Then reshape the array to 3 by 5.
 - b. Then replace the max in each row by 0. You can NOT implement it via for loop. You need to use np.where, reshape)

