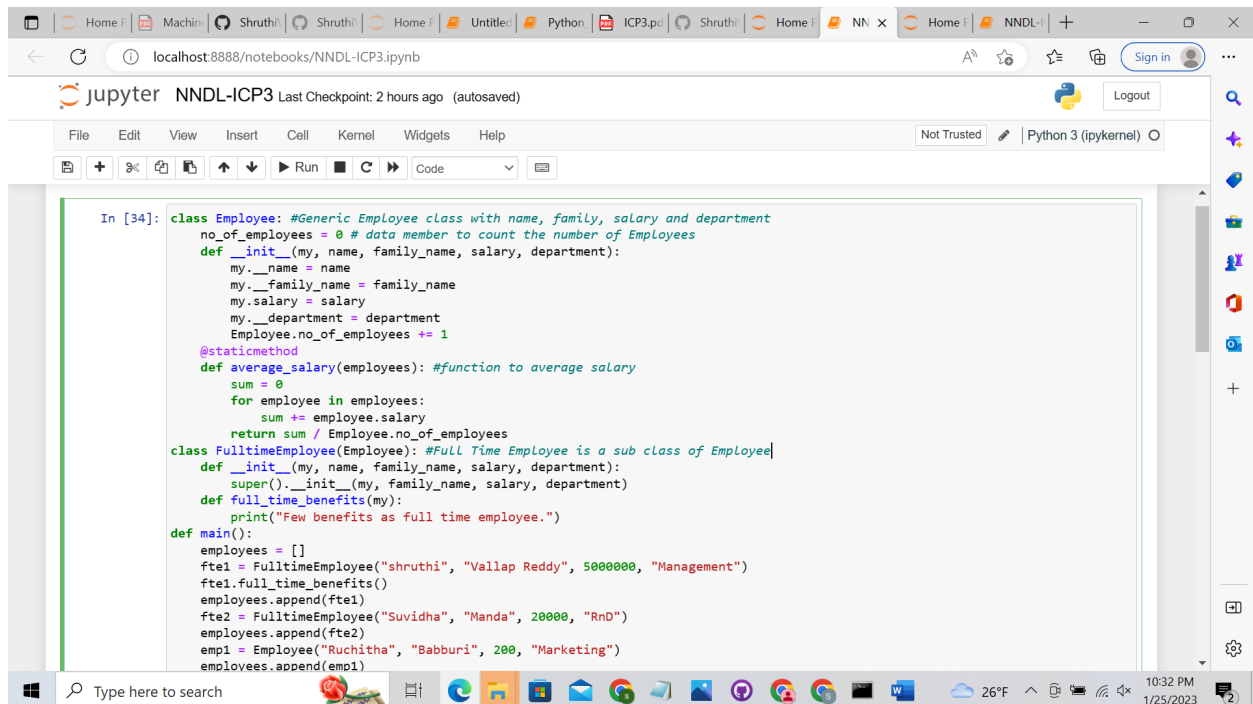


Lesson3: ICP3 In class programming:

1) Create a class Employee and then do the following

- Create a data member to count the number of Employees
- Create a constructor to initialize name, family, salary, department
- Create a function to average salary
- Create a Fulltime Employee class and it should inherit the properties of Employee class
- Create the instances of Fulltime Employee class and Employee class and call their member functions

Ans



```
In [34]: class Employee: #Generic Employee class with name, family, salary and department
no_of_employees = 0 # data member to count the number of Employees
def __init__(my, name, family_name, salary, department):
    my._name = name
    my._family_name = family_name
    my.salary = salary
    my._department = department
    Employee.no_of_employees += 1
    @staticmethod
    def average_salary(employees): #function to average salary
        sum = 0
        for employee in employees:
            sum += employee.salary
        return sum / Employee.no_of_employees
class FulltimeEmployee(Employee): #Full Time Employee is a sub class of Employee
    def __init__(my, name, family_name, salary, department):
        super().__init__(my, family_name, salary, department)
    def full_time_benefits(my):
        print("Few benefits as full time employee.")
def main():
    employees = []
    fte1 = FulltimeEmployee("shruthi", "Vallap Reddy", 5000000, "Management")
    fte1.full_time_benefits()
    employees.append(fte1)
    fte2 = FulltimeEmployee("Suvidha", "Manda", 20000, "RnD")
    employees.append(fte2)
    emp1 = Employee("Ruchiha", "Babburi", 200, "Marketing")
    employees.append(emp1)
```

- 1) Created Employee class
- 2) Created the data member to count the number of employees
- 3) Created the constructor for employee using `__init__` method.
- 4) Created the function `average_salary()` to calculate the average salary of all employees
- 5) Created the full time employee class which inherited the employee class
- 6) Created the instances with some data in `main()` and calculated the average salary.

Home | Machin | Shruthi | Shruthi | Home | Untitled | Python | ICP3.py | Shruthi | Home | NN x | Home | NNDL- | + | - | x

localhost:8888/notebooks/NNDL-ICP3.ipynb

jupyter NNDL-ICP3 Last Checkpoint: 2 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

```
def average_salary(employees): #function to average salary
    sum = 0
    for employee in employees:
        sum += employee.salary
    return sum / Employee.no_of_employees
class FulltimeEmployee(Employee): #Full Time Employee is a sub class of Employee
    def __init__(my, name, family_name, salary, department):
        super().__init__(my, family_name, salary, department)
    def full_time_benefits(my):
        print("Few benefits as full time employee.")
def main():
    employees = []
    fte1 = FulltimeEmployee("shruthi", "Vallap Reddy", 5000000, "Management")
    fte1.full_time_benefits()
    employees.append(fte1)
    fte2 = FulltimeEmployee("Suvidha", "Manda", 20000, "RnD")
    employees.append(fte2)
    emp1 = Employee("Ruchitha", "Babburi", 200, "Marketing")
    employees.append(emp1)
    emp2 = Employee("Sindhu", "Maryadha", 10000, "HR")
    employees.append(emp2)
    print("Average salary of all employees :", FulltimeEmployee.average_salary(employees))
    print("No of Employees", Employee.no_of_employees)
if __name__ == "__main__":
    main()

Few benefits as full time employee.
Average salary of all employees : 1257550.0
No of Employees 4
```

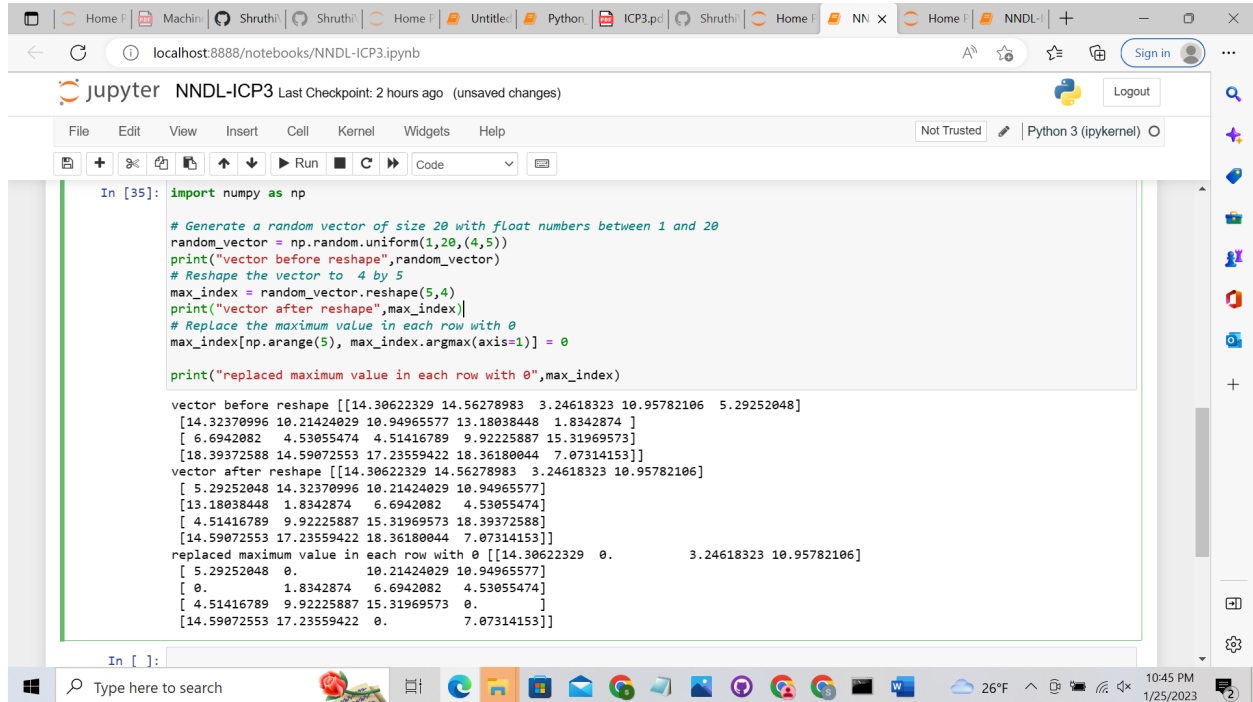
Type here to search

26°F 10:32 PM 1/25/2023

2. Numpy

- Using NumPy creates a random vector of size 20 having only float in the range 1-20.
- Then reshape the array to 4 by 5 Then replace the max in each row by 0 (axis=1) (you can NOT implement it via for loop)

Ans



The screenshot shows a Jupyter Notebook interface with a code cell containing the following Python code:

```
In [35]: import numpy as np

# Generate a random vector of size 20 with float numbers between 1 and 20
random_vector = np.random.uniform(1,20,(4,5))
print("vector before reshape",random_vector)
# Reshape the vector to 4 by 5
max_index = random_vector.reshape(5,4)
print("vector after reshape",max_index)
# Replace the maximum value in each row with 0
max_index[np.arange(5), max_index.argmax(axis=1)] = 0

print("replaced maximum value in each row with 0",max_index)
```

The output of the code is displayed below the code cell:

```
vector before reshape [[14.30622329 14.56278983  3.24618323 10.95782106  5.29252048]
 [14.32370996 10.21424029 10.94965577 13.18038448  1.8342874 ]
 [ 6.6942082  4.53055474  4.51416789  9.92225887 15.31969573]
 [18.39372588 14.59072553 17.23559422 18.36180044  7.07314153]]
vector after reshape [[14.30622329 14.56278983  3.24618323 10.95782106]
 [ 5.29252048 14.32370996 10.21424029 10.94965577]
 [13.18038448 1.8342874  6.6942082  4.53055474]
 [ 4.51416789  9.92225887 15.31969573 18.39372588]
 [14.59072553 17.23559422 18.36180044  7.07314153]]
replaced maximum value in each row with 0 [[14.30622329  0.          3.24618323 10.95782106]
 [ 5.29252048  0.          10.21424029 10.94965577]
 [ 0.          1.8342874  6.6942082  4.53055474]
 [ 4.51416789  9.92225887 15.31969573  0.          ]
 [14.59072553 17.23559422  0.          7.07314153]]
```

- 1) Using numpy created the random vector to get the float numbers between 1 to 20 numbers as 4 by 5 matrix and printed the output
- 2) Reshaped the matrix using reshape() method
- 3) Replaced the maximum value with using code `max_index[np.arange(5), max_index.argmax(axis=1)] = 0`

Git Repo link = <https://github.com/ShruthiVallapReddy/NNDL---ICP3.git>