Keerthana Aluru - 700778365

- ICP-2 (01/21/2025)

```
# Importing Numpy Library
import numpy as np
```

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.

```
# Employee Class creation
class Employee():
    # create data members to track count of employee and list of 
salaries
    emp_count = 0
    # constructor for initializing the variables
    def __init__(self,name,family,salary,department):
        self.name = name,
        self.family = family,
        self.salary = salary,
        self.department = department,
        Employee.emp_count += 1

    # funtion to return average of salary
    def avg_salary(employees:list):
        salaries_list = [emp.salary[0] for emp in employees] # method 
1 : generic
        # print(emp.salary[0] for emp in employees)
        avg_sal = sum(salaries_list)/Employee.emp_count      # method 
2 : using numpy
        # avg_sal = np.mean(salaries_list)
        return avg_sal

# create FullTimeEmployee class and inherit the properties from 
Employee class
class FullTimeEmployee(Employee):
    def __init__(self,name,family,salary,department):
        # calling parent class constructor
        super().__init__(name,family,salary,department)

employees = []
employees.append(Employee("Kumar","Yarva",1000000,"Data Science"))
employees.append(Employee("Ganesh","Aluru",2000000,"Data Analytics"))
employees.append(FullTimeEmployee("Renuka
Reddy","Pullalarevu",500000,"Artificial Intelligence"))
employees.append(FullTimeEmployee("Manoj
Reddy","Pullalarevu",600000,"Marketing"))


print("Output using employee class : ",
Employee.avg_salary(employees))
print("Output using fulltime employee class :
",FullTimeEmployee.avg_salary(employees))
```

```
Output using employee class :  1025000.0
Output using fulltime employee class :  1025000.0
```

Using NumPy create 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)

```
arr = np.random.uniform(1,20,20) # Random Vector creation of size 20
in the range of 1 - 20.
arr = arr.reshape(4,5) # reshaping the vector to array of (4,5) shape

# Task : replace the max in each row by 0 (axis = 1)

# Array before replacement
print("Array before replacement :\n\n",arr,end="\n\n\n")

# getting max elements of each row and reshaping it
each_row_max = np.max(arr,axis = 1).reshape(-1,1)

arr[arr == each_row_max] = 0

# print array after replacement
print("Array after replacing the max element of each row with 0: \n\
n", arr)
```

```
Array before replacement :

 [[ 1.         12.72881934 10.39936908 11.56366464  1.1618919 ]
 [ 6.35529243 14.9952966   1.          7.71322515  8.17244383]
 [18.04365682  1.          2.48707798 16.198048    8.55174478]
 [15.23909556  3.91705705  4.11098671 10.20258401  1.        ]]


Array after replacing the max element of each row with 0:

 [[ 1.          0.         10.39936908 11.56366464  1.1618919 ]
 [ 6.35529243  0.          1.          7.71322515  8.17244383]
 [ 0.          1.          2.48707798 16.198048    8.55174478]
 [ 0.          3.91705705  4.11098671 10.20258401  1.        ]]
```