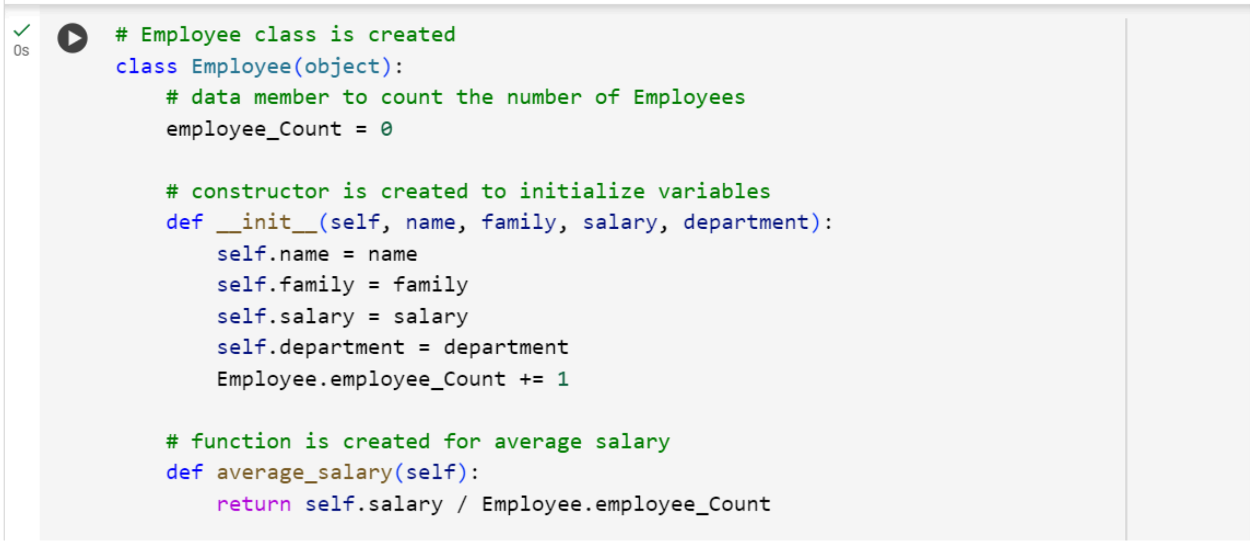
ICP 2

Kavya Reddy Nagulapally
700759486

GitHub: https://github.com/Kavyareddy03/neural_networks/tree/main/Assignment-2

Question 1

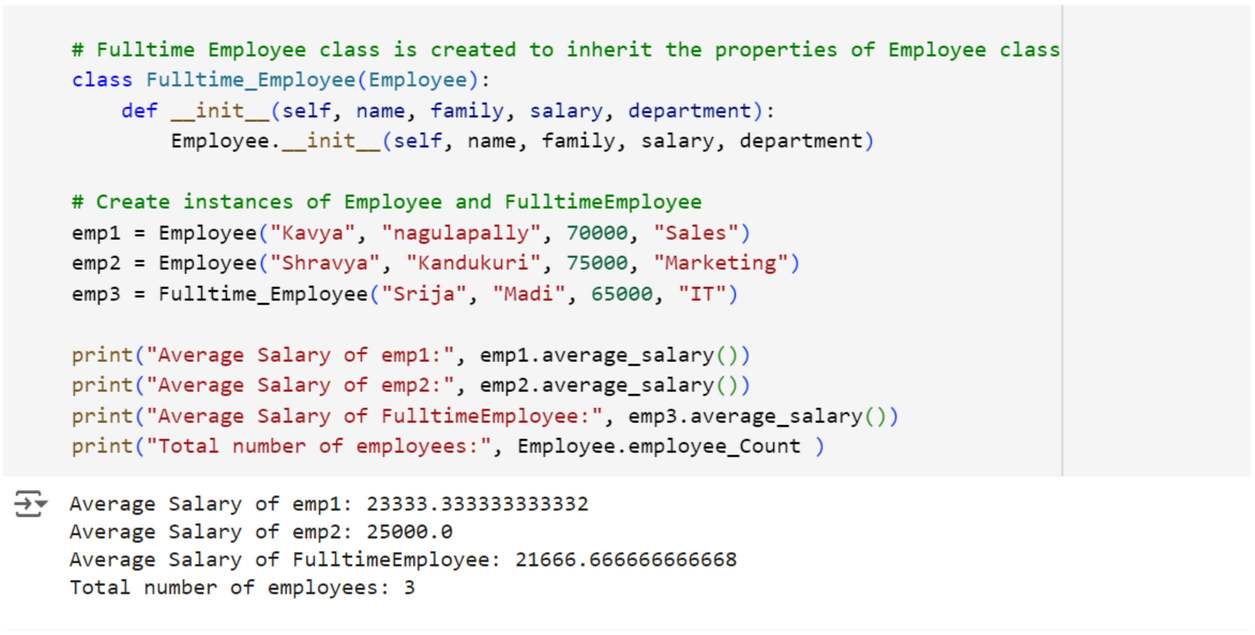
Code and output:



```
# Employee class is created
class Employee(object):
    # data member to count the number of Employees
    employee_Count = 0

    # constructor is created to initialize variables
    def __init__(self, name, family, salary, department):
        self.name = name
        self.family = family
        self.salary = salary
        self.department = department
        Employee.employee_Count += 1

    # function is created for average salary
    def average_salary(self):
        return self.salary / Employee.employee_Count
```



```
# Fulltime Employee class is created to inherit the properties of Employee class
class Fulltime_Employee(Employee):
    def __init__(self, name, family, salary, department):
        Employee.__init__(self, name, family, salary, department)

# Create instances of Employee and FulltimeEmployee
emp1 = Employee("Kavya", "nagulapally", 70000, "Sales")
emp2 = Employee("Shravya", "Kandukuri", 75000, "Marketing")
emp3 = Fulltime_Employee("Srija", "Madi", 65000, "IT")

print("Average Salary of emp1:", emp1.average_salary())
print("Average Salary of emp2:", emp2.average_salary())
print("Average Salary of FulltimeEmployee:", emp3.average_salary())
print("Total number of employees:", Employee.employee_Count )
```

```
Average Salary of emp1: 23333.333333333332
Average Salary of emp2: 25000.0
Average Salary of FulltimeEmployee: 21666.666666666668
Total number of employees: 3
```

Explanation

Employee Class :

- The class Variables ‘employee_count’ Counts the instances of `Employee’.
- And the class ‘total_salary’ Compiles the entire pay for each instance of `Employee’.
- ‘__init__’ constructor sets up the attributes of an employee ({name}, {family}, {salary}, {department{) initially.
- When a new instance is generated, it updates the class variables `employee_count` and `total_salary`.

Average_salary :

- Gives the average salary of each employee as calculated and returned.
- If there are no employees, return {0} in order to avoid division by zero.

Fulltime_Employee class:

- Takes inheritance from Employee.
- Initializes its attributes using the constructor of the parent class.

Instance of Use:

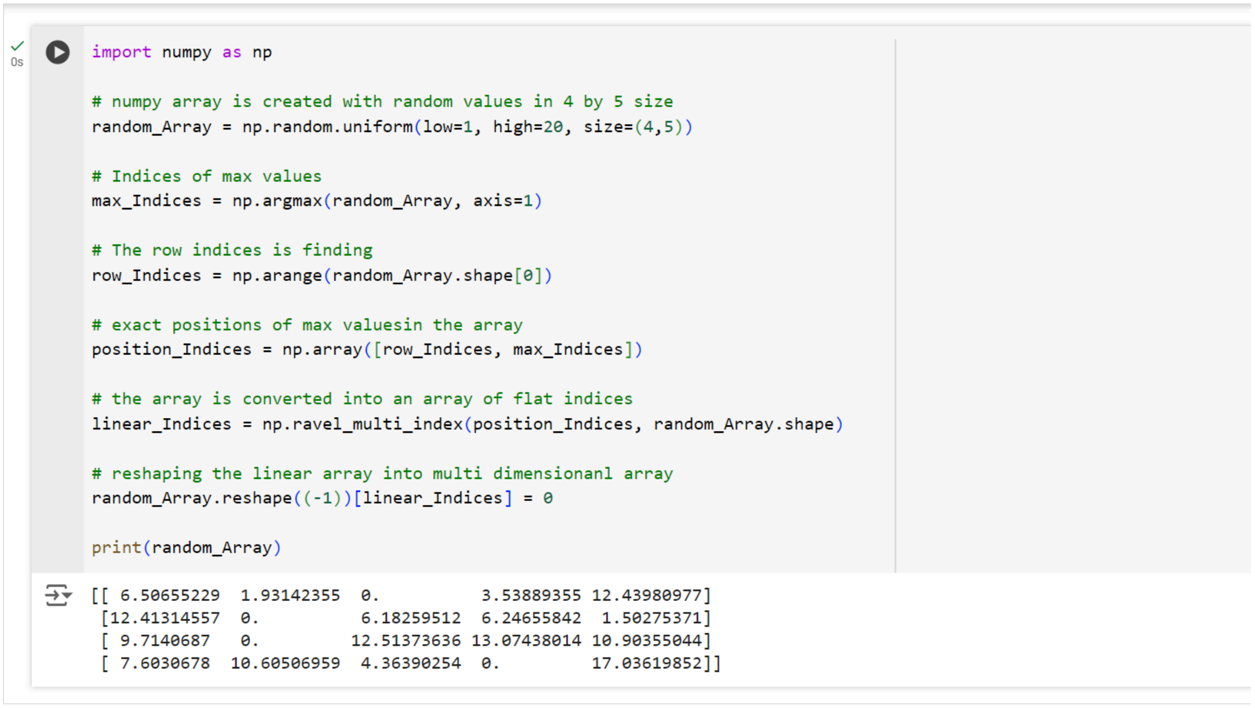
- The instances of Employee are {emp1} and `emp2}.
- ‘emp3’ is an instance of a `Fulltime_Employee’ that also modifies `total_salary` and `employee_count’.

Results:

- Prints each employee's average salary.
- Prints the entire staff count.

Question 2

Code and Output:



```
import numpy as np

# numpy array is created with random values in 4 by 5 size
random_Array = np.random.uniform(low=1, high=20, size=(4,5))

# Indices of max values
max_Indices = np.argmax(random_Array, axis=1)

# The row indices is finding
row_Indices = np.arange(random_Array.shape[0])

# exact positions of max valuesin the array
position_Indices = np.array([row_Indices, max_Indices])

# the array is converted into an array of flat indices
linear_Indices = np.ravel_multi_index(position_Indices, random_Array.shape)

# reshaping the linear array into multi dimensionanl array
random_Array.reshape((-1))[linear_Indices] = 0

print(random_Array)
```

```
[[ 6.50655229  1.93142355  0.          3.53889355 12.43980977]
 [12.41314557  0.          6.18259512  6.24655842  1.50275371]
 [ 9.7140687   0.         12.51373636 13.07438014 10.90355044]
 [ 7.6030678  10.60506959  4.36390254  0.         17.03619852]]
```

Explanation:

1. Import NumPy: The import numpy as np statement in Python is used to import the NumPy library, which is a fundamental package for numerical computing in Python. The code starts by importing the NumPy library, which is used for numerical operations.

2. Create a Random Array: The function `np.random.uniform(low=1, high=20, size=(4,5))` generates an array of random floating-point numbers uniformly distributed between a specified range.This function from the NumPy library is used to generate random numbers from a uniform distribution, which means that all values within the specified range are equally likely to occur.

3. Find Max Indices: For each row, it identifies the column index of the maximum value.**‘`np.argmax`’** function returns the indices of the maximum values along a specified axis of an array.

4. Generate Row Indices: It creates an array of row indices corresponding to each row in the 4x5 array.**`np.arange`** is a function which will create an array of evenly spaced values within a

specified range.`random_Array.shape[0]` returns a tuple representing the dimensions of the array. **Index** `[0]` will accesses the first element of the tuple

5. Determine Exact Positions: It combines the row and column indices to get the positions of the maximum values in the array. `np.array` is used to create a NumPy array from existing data, such as lists or tuples. These arrays are the primary data structure in NumPy and are used for storing and manipulating numerical data efficiently.

6. Convert to Flat Indices: It transforms these (row, column) positions into single flat indices for easier manipulation in a 1D view of the array. The function is used to convert a set of multi-dimensional indices into flat (1D) indices for use in a 1D view of the array.

7. Set Max Values to Zero: Using the flat indices, it sets the maximum values in the original array to zero.

8. Print the Modified Array: Finally, it prints the updated array where the maximum values have been replaced with zeros.