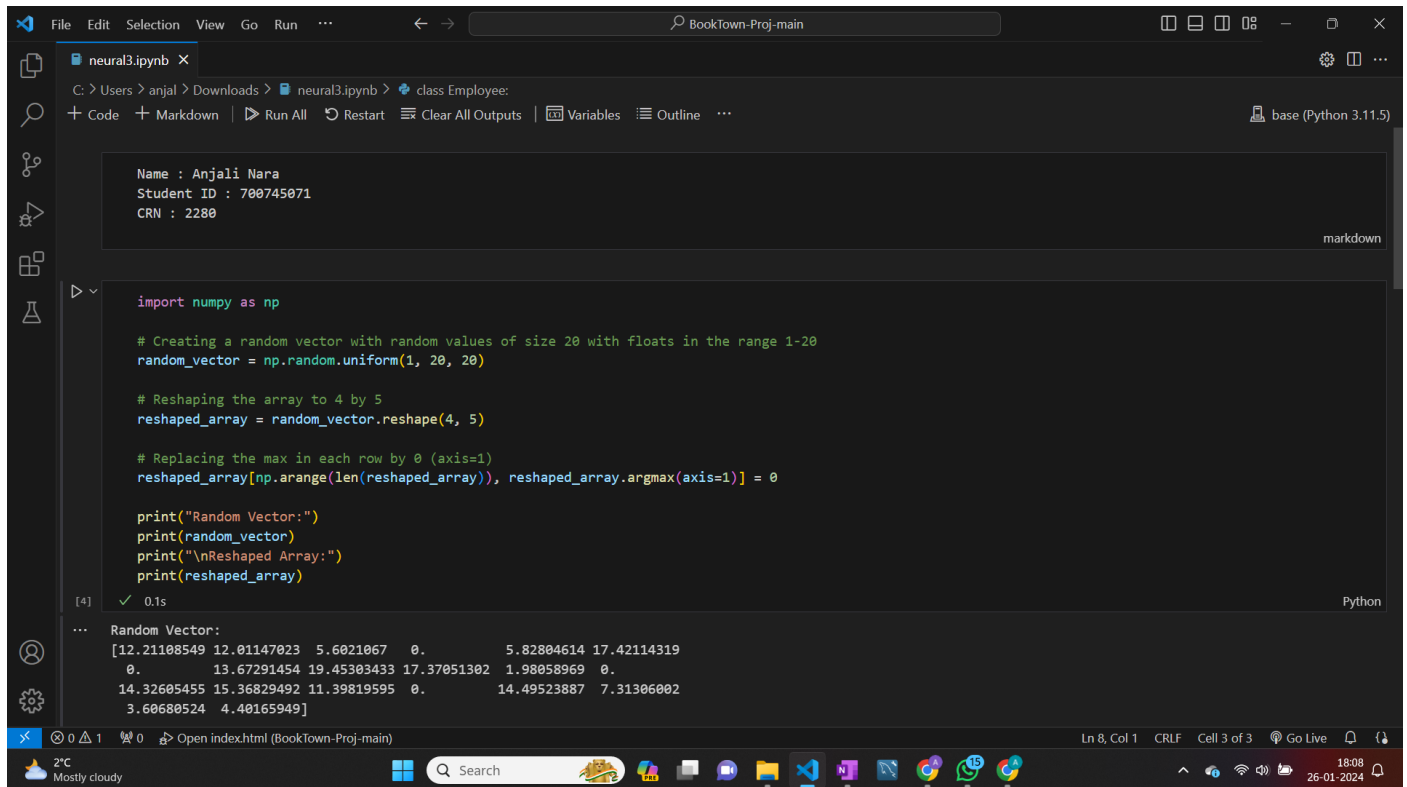


Github link: <https://github.com/naraanjali/Assignment3>

Video link: https://drive.google.com/file/d/1jaYC_9b0zVewWfQ96D8XldHpuHbBfVKb/view?usp=sharing



The screenshot shows a Jupyter Notebook titled 'neural3.ipynb' in VS Code. The code in the cell is as follows:

```
import numpy as np

# Creating a random vector with random values of size 20 with floats in the range 1-20
random_vector = np.random.uniform(1, 20, 20)

# Reshaping the array to 4 by 5
reshaped_array = random_vector.reshape(4, 5)

# Replacing the max in each row by 0 (axis=1)
reshaped_array[np.arange(len(reshaped_array)), reshaped_array.argmax(axis=1)] = 0

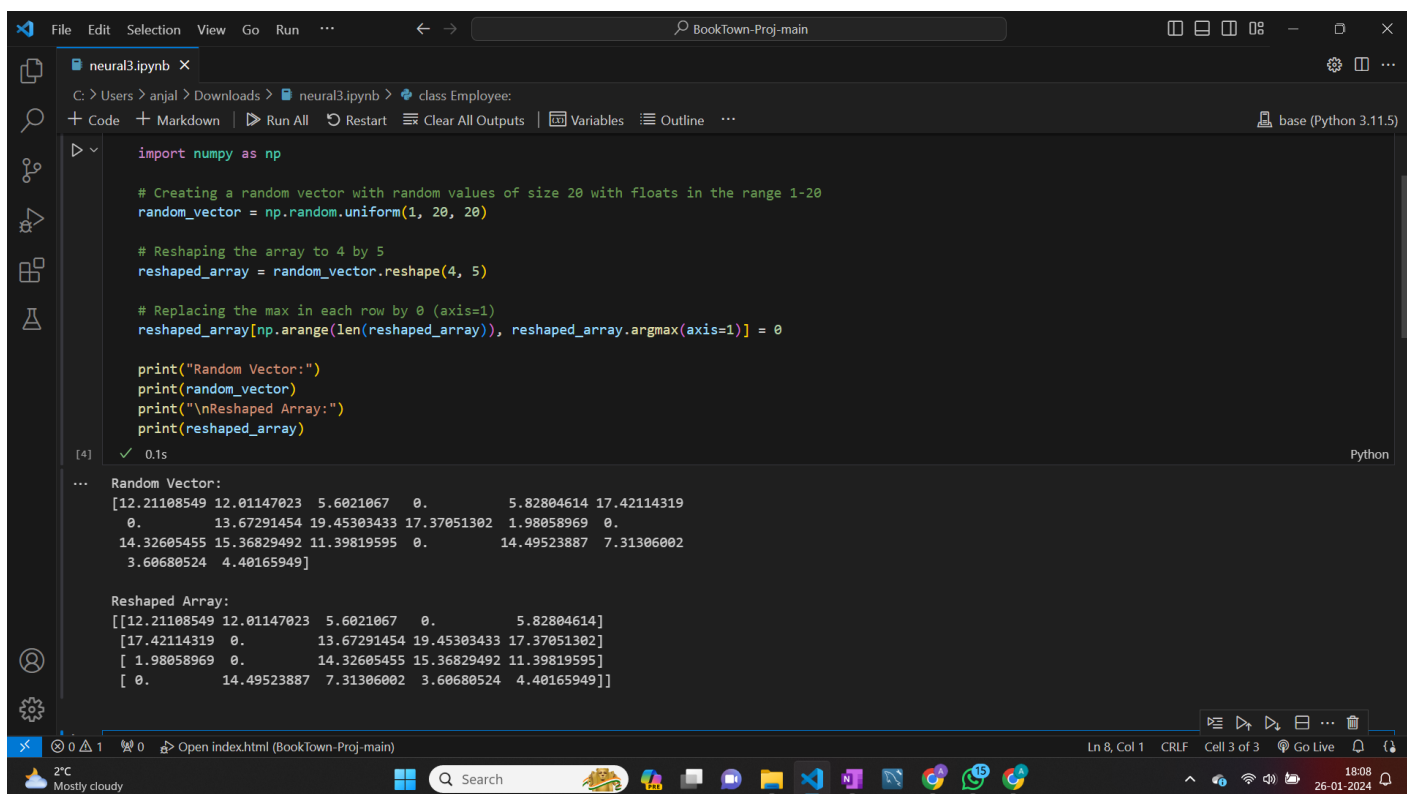
print("Random Vector:")
print(random_vector)
print("\nReshaped Array:")
print(reshaped_array)
```

The output of the code is displayed below the cell:

```
[4] ✓ 0.1s

Random Vector:
[12.21108549 12.01147023  5.6021067  0.          5.82804614 17.42114319
  0.          13.67291454 19.45303433 17.37051302  1.98058969  0.
 14.32605455 15.36829492 11.39819595  0.          14.49523887  7.31306002
 3.60680524  4.40165949]
```

The bottom status bar shows 'Ln 8, Col 1', 'CRLF', 'Cell 3 of 3', 'Go Live', and the system clock '18:08 26-01-2024'.



This screenshot is identical to the one above, showing the same Jupyter Notebook cell and output. The output of the code is displayed below the cell:

```
[4] ✓ 0.1s

Random Vector:
[12.21108549 12.01147023  5.6021067  0.          5.82804614 17.42114319
  0.          13.67291454 19.45303433 17.37051302  1.98058969  0.
 14.32605455 15.36829492 11.39819595  0.          14.49523887  7.31306002
 3.60680524  4.40165949]

Reshaped Array:
[[12.21108549 12.01147023  5.6021067  0.          5.82804614]
 [17.42114319  0.          13.67291454 19.45303433 17.37051302]
 [ 1.98058969  0.          14.32605455 15.36829492 11.39819595]
 [ 0.          14.49523887  7.31306002  3.60680524  4.40165949]]
```

The bottom status bar shows 'Ln 8, Col 1', 'CRLF', 'Cell 3 of 3', 'Go Live', and the system clock '18:08 26-01-2024'.

```
File Edit Selection View Go Run ... BookTown-Proj-main
neural3.ipynb X
C: > Users > anjal > Downloads > neural3.ipynb > class Employee:
+ Code + Markdown ▶ Run All ↺ Restart 🗑 Clear All Outputs 📄 Variables 📖 Outline ... base (Python 3.11.5)
class Employee:
    # Class variable to count the number of employees
    num_employees = 0

    def __init__(self, name, family, salary, department):
        # Instance variables
        self.name = name
        self.family = family
        self.salary = salary
        self.department = department

        # Increment the count of employees
        Employee.num_employees += 1

    def average_salary(self):
        # Function to calculate average salary
        # For simplicity, let's assume a fixed number of employees for this example
        total_salary = Employee.num_employees * self.salary
        return total_salary / Employee.num_employees

class FulltimeEmployee(Employee):
    def __init__(self, name, family, salary, department, hours_worked):
        # Call the constructor of the parent class (Employee)
        super().__init__(name, family, salary, department)

        # New property for FulltimeEmployee
        self.hours_worked = hours_worked
Ln 8, Col 1 CRLF Cell 3 of 3 Go Live 18:08 26-01-2024
2°C Mostly cloudy Search
```

```
File Edit Selection View Go Run ... BookTown-Proj-main
neural3.ipynb X
C: > Users > anjal > Downloads > neural3.ipynb > class Employee:
+ Code + Markdown ▶ Run All ↺ Restart 🗑 Clear All Outputs 📄 Variables 📖 Outline ... base (Python 3.11.5)
# Function to calculate average salary
# For simplicity, let's assume a fixed number of employees for this example
total_salary = Employee.num_employees * self.salary
return total_salary / Employee.num_employees

class FulltimeEmployee(Employee):
    def __init__(self, name, family, salary, department, hours_worked):
        # Call the constructor of the parent class (Employee)
        super().__init__(name, family, salary, department)

        # New property for FulltimeEmployee
        self.hours_worked = hours_worked

# Creating instances of Employee and FulltimeEmployee classes
employee1 = Employee("John Doe", "Doe Family", 50000, "HR")
employee2 = FulltimeEmployee("Alice Smith", "Smith Family", 60000, "IT", 40)

# Calling member function
average_salary = employee1.average_salary()
print(f"Average Salary: ${average_salary}")

# Accessing the inherited properties of FulltimeEmployee
print(f"{employee2.name} works in {employee2.department} department and works {employee2.hours_worked} hours per week.")
[5] ✓ 0.0s Python
Average Salary: $50000.0
Alice Smith works in IT department and works 40 hours per week.
Ln 8, Col 1 CRLF Cell 3 of 3 Go Live 18:08 26-01-2024
2°C Mostly cloudy Search
```