NumPy & Pandas Data Analysis Projects

Name: Suheal Ahmad

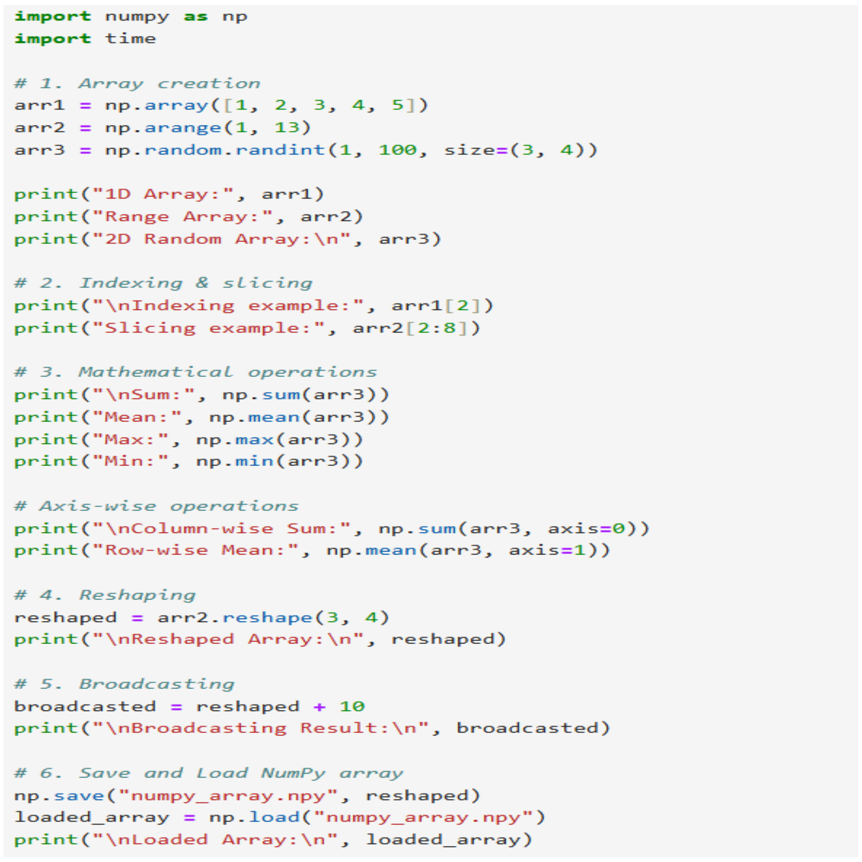
Role: Data Science Intern

Project – 1: NumPy Data Explorer

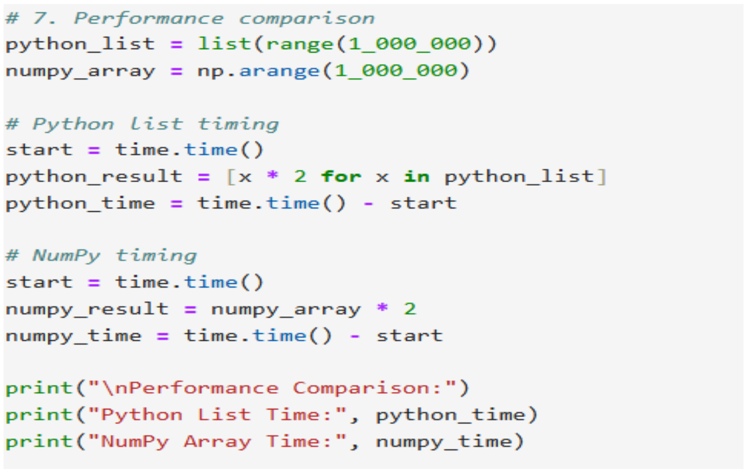
Objective:
To understand NumPy fundamentals and compare performance with Python lists.

Tools Used:
Python, NumPy

Codes:

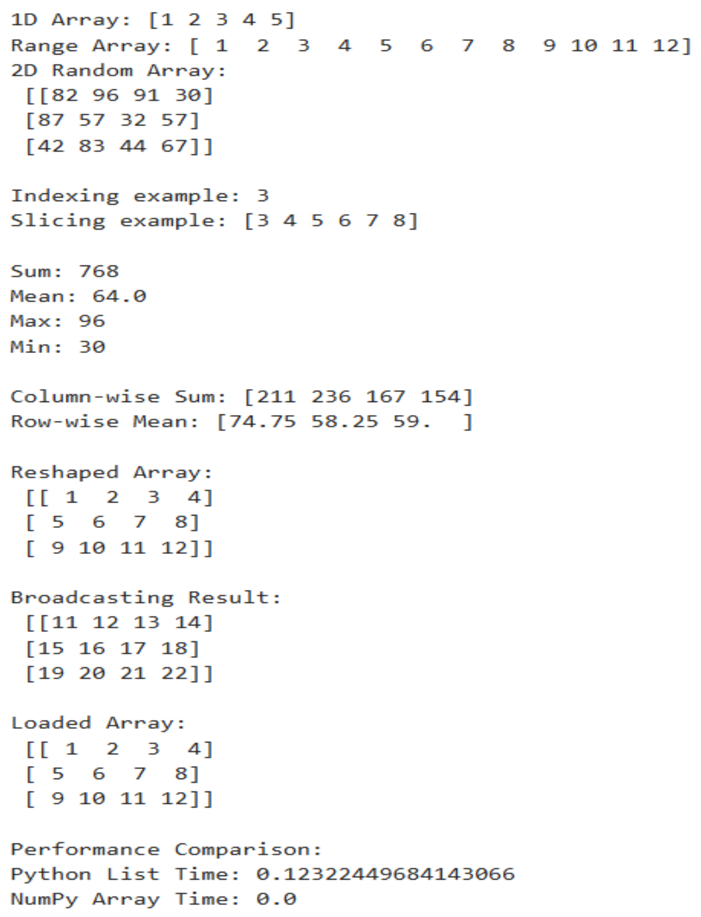


```python
import numpy as np
import time

# 1. Array creation
arr1 = np.array([1, 2, 3, 4, 5])
arr2 = np.arange(1, 13)
arr3 = np.random.randint(1, 100, size=(3, 4))

print("1D Array:", arr1)
print("Range Array:", arr2)
print("2D Random Array:\n", arr3)

# 2. Indexing & slicing
print("\nIndexing example:", arr1[2])
print("Slicing example:", arr2[2:8])

# 3. Mathematical operations
print("\nSum:", np.sum(arr3))
print("Mean:", np.mean(arr3))
print("Max:", np.max(arr3))
print("Min:", np.min(arr3))

# Axis-wise operations
print("\nColumn-wise Sum:", np.sum(arr3, axis=0))
print("Row-wise Mean:", np.mean(arr3, axis=1))

# 4. Reshaping
reshaped = arr2.reshape(3, 4)
print("\nReshaped Array:\n", reshaped)

# 5. Broadcasting
broadcasted = reshaped + 10
print("\nBroadcasting Result:\n", broadcasted)

# 6. Save and Load NumPy array
np.save("numpy_array.npy", reshaped)
loaded_array = np.load("numpy_array.npy")
print("\nLoaded Array:\n", loaded_array)
```



```python
# 7. Performance comparison
python_list = list(range(1_000_000))
numpy_array = np.arange(1_000_000)

# Python list timing
start = time.time()
python_result = [x * 2 for x in python_list]
python_time = time.time() - start

# NumPy timing
start = time.time()
numpy_result = numpy_array * 2
numpy_time = time.time() - start

print("\nPerformance Comparison:")
print("Python List Time:", python_time)
print("NumPy Array Time:", numpy_time)
```

Output:



```
1D Array: [1 2 3 4 5]
Range Array: [ 1  2  3  4  5  6  7  8  9 10 11 12]
2D Random Array:
 [[82 96 91 30]
 [87 57 32 57]
 [42 83 44 67]]

Indexing example: 3
Slicing example: [3 4 5 6 7 8]

Sum: 768
Mean: 64.0
Max: 96
Min: 30

Column-wise Sum: [211 236 167 154]
Row-wise Mean: [74.75 58.25 59.   ]

Reshaped Array:
 [[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]

Broadcasting Result:
 [[11 12 13 14]
 [15 16 17 18]
 [19 20 21 22]]

Loaded Array:
 [[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]

Performance Comparison:
Python List Time: 0.12322449684143066
NumPy Array Time: 0.0
```

Output Summary:

- *Successfully created 1D and 2D NumPy arrays using different methods such as array (), arrange (), and random.*
- *Performed indexing and slicing operations to extract required elements from arrays.*
- *Applied mathematical and statistical functions like sum, mean, minimum, maximum, and axis-wise operations.*
- *Reshaped arrays into different dimensions and used broadcasting to perform efficient computations without loops.*
- *Saved NumPy arrays to disk and loaded them back using. npy format.*
- *Compared execution time between Python lists and NumPy arrays and observed that NumPy performs significantly faster for large datasets.*

Result: *NumPy provides high-performance numerical computation and is more efficient than standard Python lists.*