Batch: C-5-(3)	**Roll No.:** 16014224055

Experiment / assignment / tutorial No. 7
Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Title: NumPy library of Python

AIM: To explore the Numpy library of Python

Expected OUTCOME of Experiment:

CO5: Use Numpy Library functions

Resource Needed: Python IDE

Theory:

NumPy: A Python library used for working with arrays.

- It also has functions for working in the domain of linear algebra, Fourier transform, and matrices.
- NumPy stands for Numerical Python.
- The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.

For Installation of NumPy:

- pip install numpy

Example

```
import numpy

arr=numpy.array([1, 2, 3, 4, 5])

print(arr)
```

output:?

For the creation of NumPy ndarray Object:

- NumPy is used to work with arrays. The array object in NumPy is called ndarray.
- We can create a NumPy ndarray object by using the array() function.
  Example:

```
import numpy as np
arr=np.array([1, 2, 3, 4, 5])
print(arr)
print(type(arr))
```

Creating ndarrays:

```
array = np.array([[0,1,2],[2,3,4]])
output:
[[0 1 2]
 [2 3 4]]

array = np.zeros((2,3))
[[0. 0. 0.]
 [0. 0. 0.]]

array = np.ones((2,3))
[[1. 1. 1.]
 [1. 1. 1.]]

array = np.eye(3)
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]

array = np.arange(0, 10, 2)
[0, 2, 4, 6, 8]

array = np.random.randint(0, 10, (3,3))
[[6 4 3]
 [1 5 6]
 [9 8 5]]
```

Slicing arrays

Slicing in Python means taking elements from one given index to another given index.
We pass slice instead of index like this: [start: end].
We can also define the step, like this: [start:end: step].

Arithmetic with NumPy Arrays:
Any arithmetic operations between equal-size arrays apply the operation element-wise

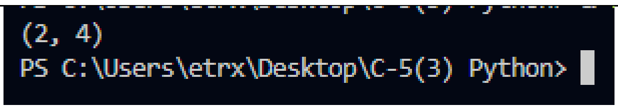
```
arr = np.array([[1., 2., 3.], [4., 5., 6.]])
print(arr)

[[1. 2. 3.]
 [4. 5. 6.]]
```

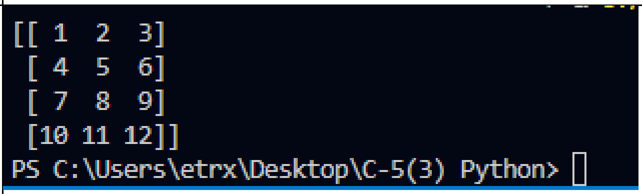
```
print(arr * arr)
[[ 1.  4.  9.]
 [16. 25. 36.]]

print(arr - arr)
[[0. 0. 0.]
 [0. 0. 0.]]
```

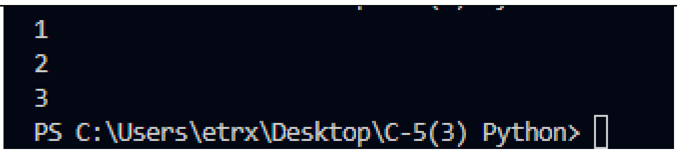
- **Shape of an Array**
  - The shape of an array is the number of elements in each dimension.
- **Reshaping arrays**
  - Reshaping means changing the shape of an array.
  - The shape of an array is the number of elements in each dimension.
  - By reshaping we can add or remove dimensions or change the number of elements
  - in each dimension.
- **Iterating Arrays**
  - Iterating means going through elements one by one.
  - As we deal with multi-dimensional arrays in numpy, we can do this using the basic
  - for loop of Python. If we iterate on a 1-D array it will go through each element
  - one by one.
- **Joining NumPy Arrays**
  - Joining means putting the contents of two or more arrays in a single array.
  - In SQL we join tables based on a key, whereas in NumPy we join arrays by axes.
  - We pass a sequence of arrays that we want to join to the concatenate() function, along with the axis. If the axis is not explicitly passed, it is taken as 0.
- **Splitting NumPy Arrays**
  - Splitting is the reverse operation of Joining.
  - Joining merges multiple arrays into one and Splitting breaks one array into multiple.
  - We use array_split() for splitting arrays, we pass it the array we want to split and the number of splits.
- **NumPy Searching Arrays**
  - You can search an array for a certain value, and return the indexes that get a match.
  - To search an array, use the where() method.
- **Sorting Arrays**
  - Sorting means putting elements in an ordered sequence.
  - An ordered sequence is any sequence that has an order corresponding to elements, like numeric or alphabetical, ascending or descending.
  - The NumPy ndarray object has a function called sort(), that will sort a specified array.
- **NumPy Filter Array**
  - Getting some elements out of an existing array and creating a new array out of them is called filtering. In NumPy, you filter an array using a boolean index list.

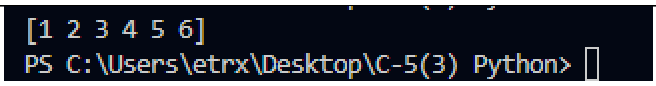
1. Problem statement:

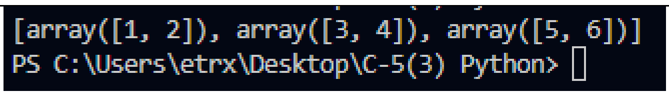
Python Code	Output

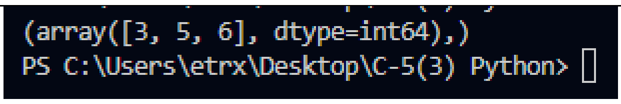
import numpy as np
arr = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print(arr.shape)
	
	`(2, 4)`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

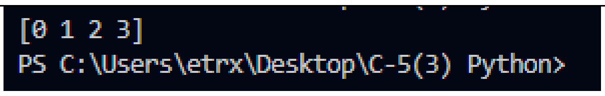
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
newarr = arr.reshape(4, 3)
print(newarr)
	
	`[[ 1  2  3]`
	` [ 4  5  6]`
	` [ 7  8  9]`
	` [10 11 12]]`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

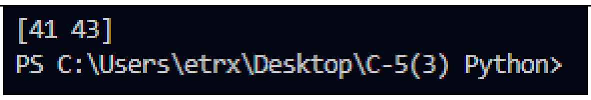
import numpy as np
arr = np.array([1, 2, 3])
for x in arr:
    print(x)
	
	`1`
	`2`
	`3`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.concatenate((arr1, arr2))
print(arr)
	
	`[1 2 3 4 5 6]`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6])
newarr = np.array_split(arr, 3)
print(newarr)
	
	`[array([1, 2]), array([3, 4]), array([5, 6])]`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

import numpy as np
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x = np.where(arr == 4)
print(x)
	
	`(array([3, 5, 6], dtype=int64),)`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

import numpy as np
arr = np.array([3, 2, 0, 1])
print(np.sort(arr))
	
	`[0 1 2 3]`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

import numpy as np
arr = np.array([41, 42, 43, 44])
x = [True, False, True, False]
newarr = arr[x]
print(newarr)
	
	`[41 43]`
	`PS C:\Users\etrx\Desktop\C-5(3) Python>`

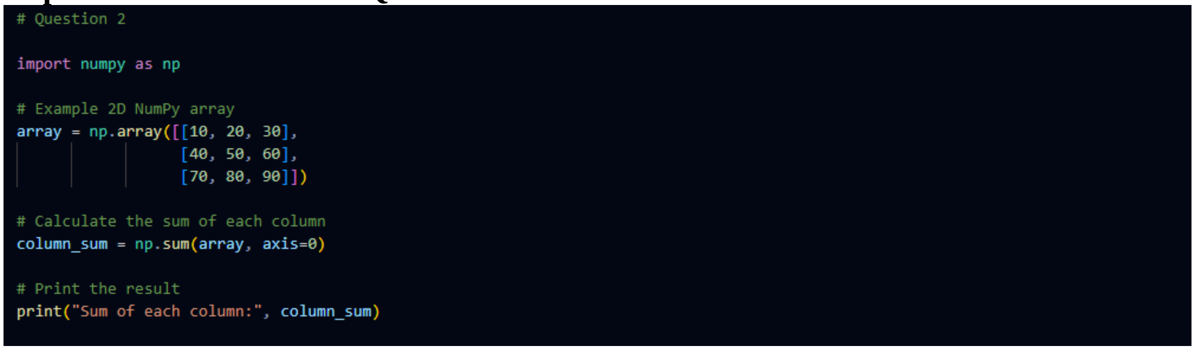
2. Write a Python program to calculate the sum of all columns in a 2D NumPy array.

3. Create two NumPy arrays representing monthly high and low temperatures for a year. Calculate the monthly average temperatures, and the overall average high and low temperatures, and identify the months with the highest and lowest average temperatures.

Books/ Journals/ Websites referred:

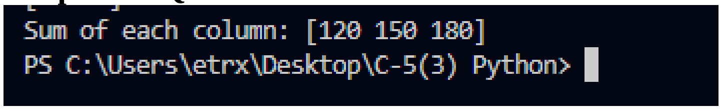
1. Reema Thareja, *Python Programming: Using Problem-Solving Approach*, Oxford University Press, First Edition 2017, India
2. Sheetal Taneja and Naveen Kumar, *Python Programming: A modular Approach*, Pearson India, Second Edition 2018, India

Implementation detail for Question 2:



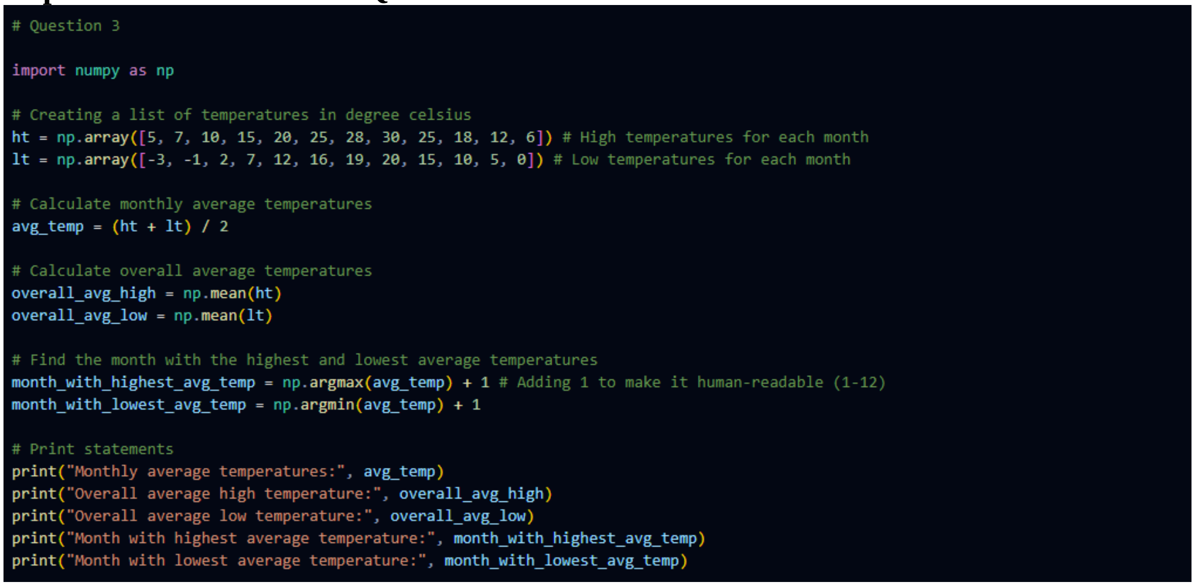
```python
# Question 2

import numpy as np

# Example 2D NumPy array
array = np.array([[10, 20, 30],
                  [40, 50, 60],
                  [70, 80, 90]])

# Calculate the sum of each column
column_sum = np.sum(array, axis=0)

# Print the result
print("Sum of each column:", column_sum)
```

Output for Question 2:



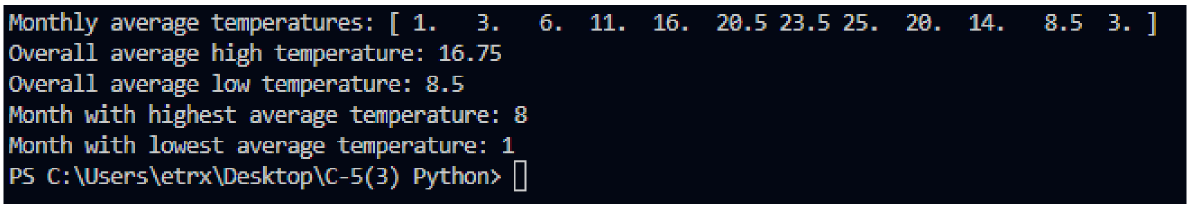
```
Sum of each column: [120 150 180]
PS C:\Users\etrx\Desktop\C-5(3) Python>
```

Implementation detail for Question 3:



```
# Question 3

import numpy as np

# Creating a list of temperatures in degree celsius
ht = np.array([5, 7, 10, 15, 20, 25, 28, 30, 25, 18, 12, 6]) # High temperatures for each month
lt = np.array([-3, -1, 2, 7, 12, 16, 19, 20, 15, 10, 5, 0]) # Low temperatures for each month

# Calculate monthly average temperatures
avg_temp = (ht + lt) / 2

# Calculate overall average temperatures
overall_avg_high = np.mean(ht)
overall_avg_low = np.mean(lt)

# Find the month with the highest and lowest average temperatures
month_with_highest_avg_temp = np.argmax(avg_temp) + 1 # Adding 1 to make it human-readable (1-12)
month_with_lowest_avg_temp = np.argmin(avg_temp) + 1

# Print statements
print("Monthly average temperatures:", avg_temp)
print("Overall average high temperature:", overall_avg_high)
print("Overall average low temperature:", overall_avg_low)
print("Month with highest average temperature:", month_with_highest_avg_temp)
print("Month with lowest average temperature:", month_with_lowest_avg_temp)
```

Output for Question 3:

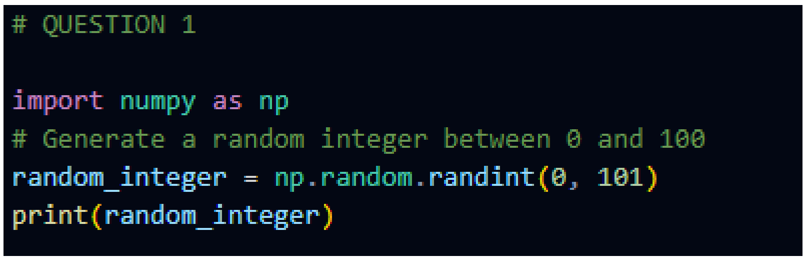


```
Monthly average temperatures: [ 1.   3.   6.  11.  16.  20.5 23.5 25.  20.  14.   8.5  3. ]
Overall average high temperature: 16.75
Overall average low temperature: 8.5
Month with highest average temperature: 8
Month with lowest average temperature: 1
PS C:\Users\etrx\Desktop\C-5(3) Python>
```

Conclusion:
In this module we learned about NumPy,
NumPy is a fundamental library for numerical and scientific computing in Python that can be used for a variety of applications
Also about how to install NumPy, its features and creation of NumPy and array Object:

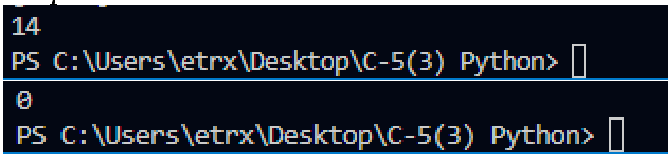
Post Lab Descriptive Questions

1. **Generate a random integer from 0 to 100 using the NumPy random function.**
   Code:



```python
# QUESTION 1

import numpy as np
# Generate a random integer between 0 and 100
random_integer = np.random.randint(0, 101)
print(random_integer)
```

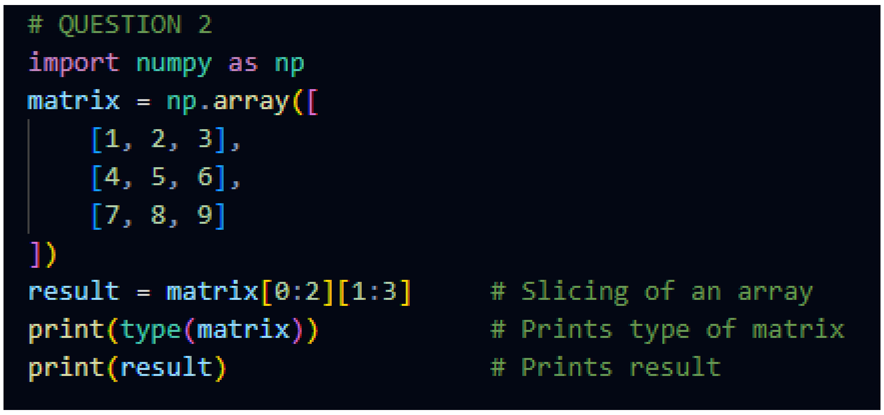
Output:



```
14
PS C:\Users\etrx\Desktop\C-5(3) Python> 
0
PS C:\Users\etrx\Desktop\C-5(3) Python> 
```

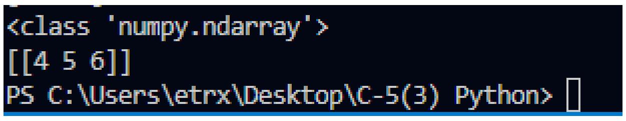
2. **Explain the slicing of 2-D Array**

Ans: Slicing a 2-D array in Python (using NumPy) means accessing or extracting a portion of the array by specifying a range of indices. A 2-D array has rows and columns, so slicing involves selecting a range of rows and/or columns.
Example:
Code:



```python
# QUESTION 2
import numpy as np
matrix = np.array([
        [1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]
])
result = matrix[0:2][1:3]      # Slicing of an array
print(type(matrix))            # Prints type of matrix
print(result)                  # Prints result
```

Output:



```
<class 'numpy.ndarray'>
[[4 5 6]]
PS C:\Users\etrx\Desktop\C-5(3) Python> 
```

Date: 08-11-2024 **Signature of faculty in-charge**