Name of student: Abhay Omprakash Prajapati		
Roll no: 41		**Tutorial No: 9**
Title of LAB Assignment: **To implement NumPy library in Python**		
DOP: 30-10-2023		**DOS: 04-11-2023**
CO Mapped:	**PO Mapped:**	**Signature:**

1. Aim:
To understand and implement basic operations in NumPy, including creating ndarray objects, performing matrix multiplication, indexing and slicing NumPy arrays, and working with data types.

2. Theory:
In this practical, we will cover the following topics:

Creating ndarray objects using array() in NumPy.
Implementing 2D arrays to perform matrix multiplication using the dot() function.
Indexing and slicing in NumPy arrays to access and manipulate elements.

Exploring NumPy data types for efficient memory usage and performance.
3. Code:

Here's the Python code for each part of your practical:

1. **Creating ndarray objects using array() in NumPy:**

```
import numpy as np

# Create a 1D array
arr1 = np.array([1, 2, 3, 4, 5])

# Create a 2D array
arr2 = np.array([[1, 2, 3], [4, 5, 6]])

print("1D Array:")
print(arr1)
print("2D Array:")
print(arr2)
```

2. **Program for Indexing and Slicing in NumPy arrays:**

```
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

element = arr[1, 2]
row_slice = arr[0:2, 1:]
print("Element at (1, 2):", element)
print("Sliced Array:")
print(row_slice)
```

3. **Implementing NumPy - Data Types:**

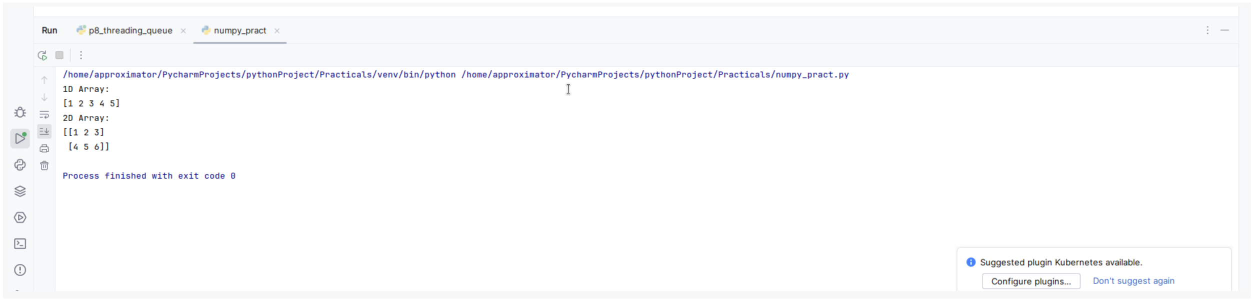
```
import numpy as np

# Create arrays with specific data types
int_array = np.array([1, 2, 3], dtype=np.int32)
float_array = np.array([1.2, 2.3, 3.4], dtype=np.float64)
complex_array = np.array([1 + 2j, 2 + 3j], dtype=np.complex128)

print("Int Array:", int_array)
print("Float Array:", float_array)
print("Complex Array:", complex_array)
```

Conclusion:

In this practical, we learned how to create ndarray objects in NumPy using the array() function, perform matrix multiplication with 2D arrays, and use indexing and slicing to access specific elements in arrays. Additionally, we explored the use of NumPy data types for efficient memory and performance management.

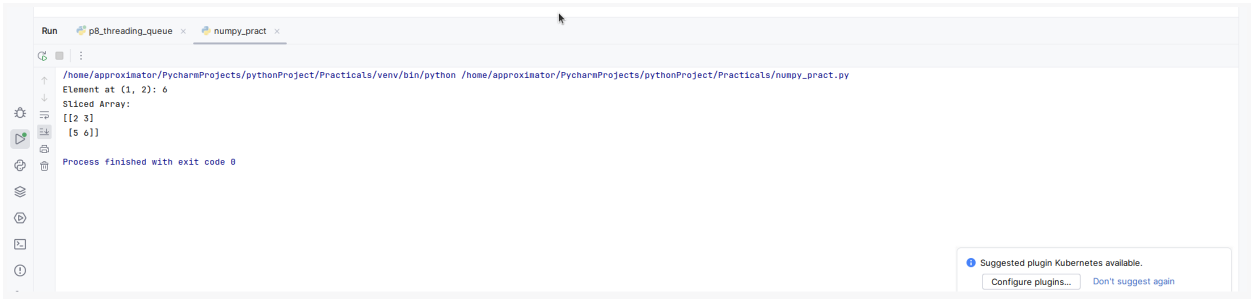
5. Output:

1.



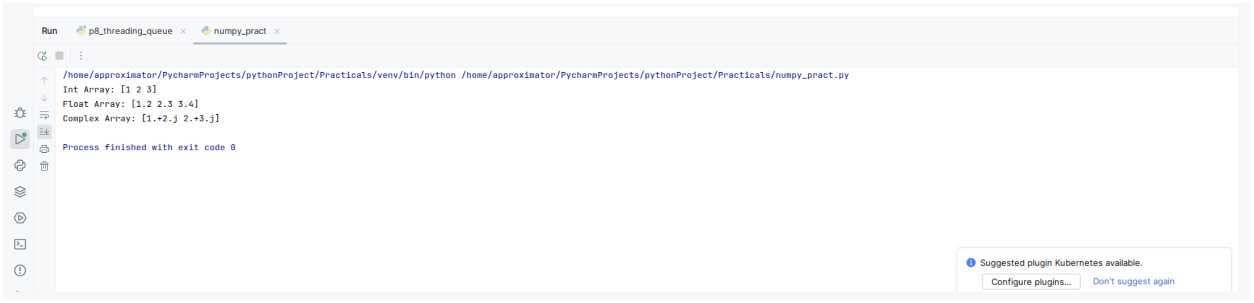
```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/numpy_pract.py
1D Array:
[1 2 3 4 5]
2D Array:
[[1 2 3]
 [4 5 6]]

Process finished with exit code 0
```

2.



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/numpy_pract.py
Element at (1, 2): 6
Sliced Array:
[[2 3]
 [5 6]]

Process finished with exit code 0
```

3.



```
/home/approximator/PycharmProjects/pythonProject/Practicals/venv/bin/python /home/approximator/PycharmProjects/pythonProject/Practicals/numpy_pract.py
Int Array: [1 2 3]
Float Array: [1.2 2.3 3.4]
Complex Array: [1.+2.j 2.+3.j]

Process finished with exit code 0
```