

## Practical-2: Implement Python Program for NumPy Arrays

```
import numpy as np

# Creating array object
arr = np.array([[1, 2, 3],
                [4, 2, 5]])

# Printing type of arr object
print("Array is of type: ", type(arr))

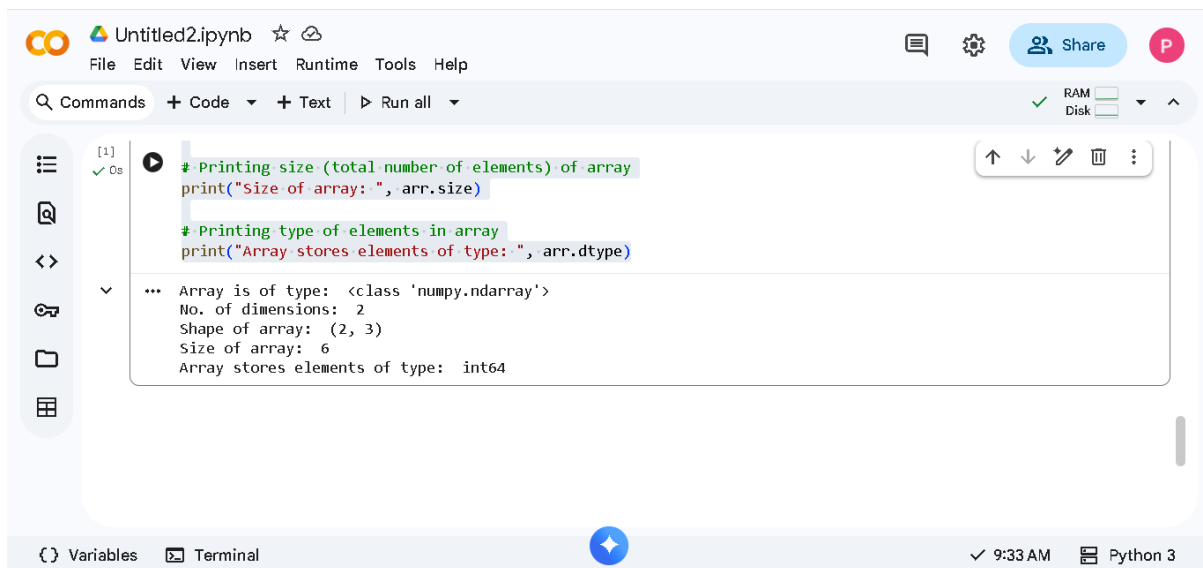
# Printing array dimensions (axes)
print("No. of dimensions: ", arr.ndim)

# Printing shape of array
print("Shape of array: ", arr.shape)

# Printing size (total number of elements) of array
print("Size of array: ", arr.size)

# Printing type of elements in array
print("Array stores elements of type: ", arr.dtype)
```

Output:



The screenshot shows a Jupyter Notebook window titled "Untitled2.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". Below the menu is a "Commands" search bar and tabs for "+ Code", "+ Text", and "Run all". On the right, there are icons for "Share" and a profile icon, along with RAM and Disk usage indicators. The main area displays a code cell with the following content:

```
[1] ✓ Os
# Printing size (total number of elements) of array
print("Size of array: ", arr.size)

# Printing type of elements in array
print("Array stores elements of type: ", arr.dtype)
```

Below the code, the output is shown in a collapsed state, indicated by a downward arrow. The output text is:

```
... Array is of type: <class 'numpy.ndarray'>
No. of dimensions: 2
Shape of array: (2, 3)
Size of array: 6
Array stores elements of type: int64
```

At the bottom of the window, there are tabs for "Variables" and "Terminal", a blue circular icon with a white star, and a status bar showing "9:33 AM" and "Python 3".

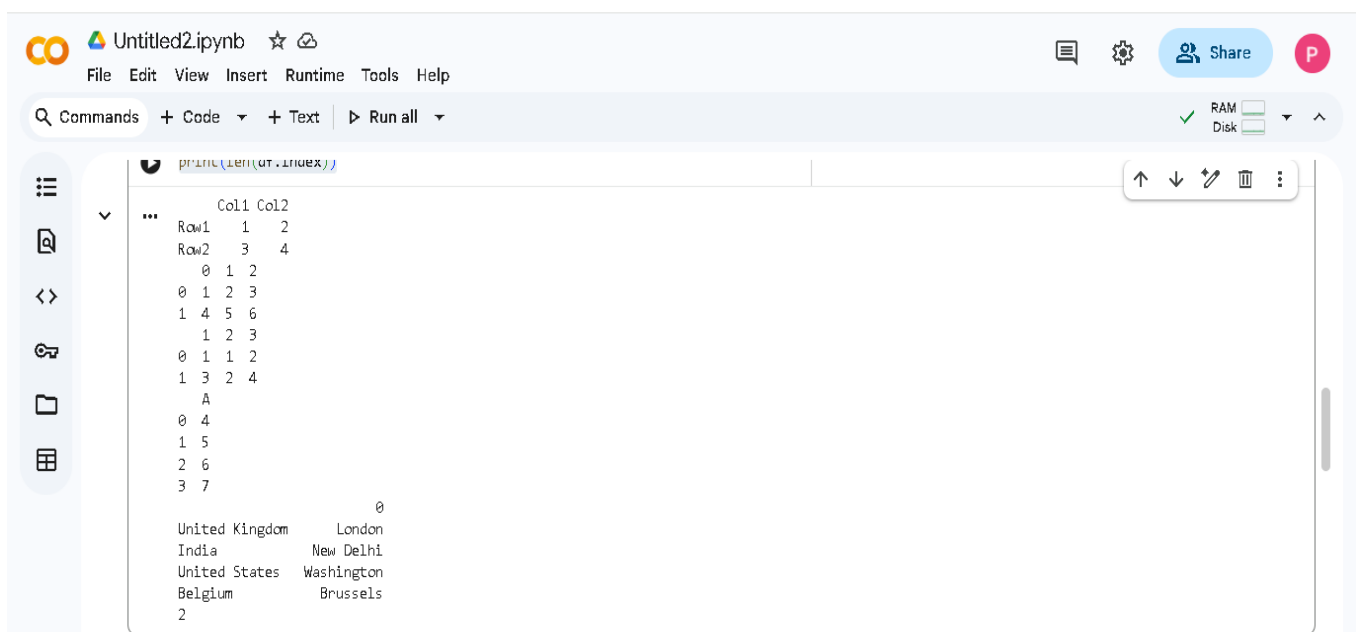
## Extra:

### Practical-2.1: More about NumPy Arrays and data frames

```
import numpy as np
import pandas as pd
data = np.array([[ ' ', 'Col1', 'Col2'], [ 'Row1', 1, 2],
[ 'Row2', 3, 4]])
print(pd.DataFrame(data=data[1:,1:],
index = data[1:,0], columns=data[0,1:]))
# Take a 2D array as input to your DataFrame
my_2darray = np.array([[1, 2, 3], [4, 5, 6]])
print(pd.DataFrame(my_2darray))
# Take a dictionary as input to your DataFrame
my_dict = {1: ['1', '3'], 2: ['1', '2'], 3: ['2', '4']}
print(pd.DataFrame(my_dict))
# Take a DataFrame as input to your DataFrame
my_df = pd.DataFrame(data=[4,5,6,7], index=range(0,4), columns=['A'])
print(pd.DataFrame(my_df))
# Take a Series as input to your DataFrame
my_series = pd.Series({"United Kingdom":"London", "India":"New Delhi",
"United States":"Washington", "Belgium":"Brussels"})
print(pd.DataFrame(my_series))
df = pd.DataFrame(np.array([[1, 2, 3], [4, 5, 6]]))
# Use the `shape` property print(df.shape)

# Or use the `len()` function with the `index` property
print(len(df.index))
```

## Output:



The screenshot shows a Jupyter Notebook interface with the following components:

- Toolbar:** Includes icons for file operations, search, and a 'Share' button.
- Search Bar:** Labeled 'Commands', with options for '+ Code', '+ Text', and 'Run all'.
- Code Cell:** Contains the same Python code as the previous block, including imports, array creation, DataFrame construction from various inputs, and printing of DataFrame shape and index length.
- Output Cell:** Displays the results of the code execution. It shows the DataFrame created from the NumPy array, the DataFrame from the 2D array, the DataFrame from the dictionary, the DataFrame from the Series, and the DataFrame from the NumPy array. The output also shows the shape of the DataFrame and the length of its index.

The output of the code execution is as follows:

```
Col1 Col2
Row1    1    2
Row2    3    4
0      1    2
0      1    2    3
1      4    5    6
1      1    2    3
0      1    1    2
1      3    2    4
A
0      4
1      5
2      6
3      7

0
United Kingdom    London
India             New Delhi
United States     Washington
Belgium           Brussels
2
```

