Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

GITHUB

Aim: Practical based on NumPy ndarray

IDE:

NumPy is a Python package created in 2005 that performs numerical calculations. It is generally used for working with arrays. NumPy also includes a wide range of mathematical functions, such as linear algebra, Fourier transforms, and random number generation, which can be applied to arrays.

Import NumPy in Python

We can import NumPy in Python using the import statement.

```
import numpy as np
```

The code above imports the numpy package in our program as an alias np. After this import statement, we can use NumPy functions and objects by calling them with np.

NumPy Array Creation

An array allows us to store a collection of multiple values in a single data structure. The NumPy array is similar to a list, but with added benefits such as being faster and more memory efficient. There are multiple techniques to generate arrays in NumPy.

Create Array Using Python List

We can create a NumPy array using a Python List. For example,

Example

```
import numpy as np
list1 = [2, 4, 6, 8]
array1 = np.array(list1)
print(array1)
```

Output:

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
1   import numpy as np
2   list1 = [2, 4, 6, 8]
3   array1 = np.array(list1)
4   print(array1)
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
[2 4 6 8]
PS E:\SEM 3\PWP>
```

Example

```
import numpy as np
array1 = np.array([2, 4, 6, 8])
print(array1)
```

Output

```
6   import numpy as np
7   array1 = np.array([2, 4, 6, 8])
8   print(array1)
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
[2 4 6 8]
PS E:\SEM 3\PWP>
```

Create an Array Using np.zeros()

The np.zeros() function allows us to create an array filled with all zeros. For example,

Example

```
import numpy as np
array1 = np.zeros(4)
print(array1)
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

Output

```
10  import numpy as np
11  array1 = np.zeros(4)
12  print(array1)
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
[0. 0. 0. 0.]
PS E:\SEM 3\PWP>
```

Create an Array With np.arange()

The np.arange() function returns an array with values within a specified interval. For example,

Example

```
import numpy as np
# create an array with values from 0 to 4
array1 = np.arange(5)
print("Using np.arange(5):", array1)
# create an array with values from 1 to 8 with a step of 2
array2 = np.arange(1, 9, 2)
print("Using np.arange(1, 9, 2):",array2)
```

Output

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
14   import numpy as np
15   # create an array with values from 0 to 4
16   array1 = np.arange(5)
17   print("Using np.arange(5):", array1)
18   # create an array with values from 1 to 8 with a step of 2
19   array2 = np.arange(1, 9, 2)
20   print("Using np.arange(1, 9, 2):",array2)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
Using np.arange(5): [0 1 2 3 4]
Using np.arange(1, 9, 2): [1 3 5 7]
PS E:\SEM 3\PWP>
```

Create an Array With np.random.rand()

The np.random.rand() function is used to create an array of random numbers. Let's see an example to create an array of **5** random numbers,

Example

```
import numpy as np
# generate an array of 5 random numbers
array1 = np.random.rand(5)
print(array1)
```

Output

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
22  import numpy as np
23  # generate an array of 5 random numbers
24  array1 = np.random.rand(5)
25  print(array1)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
[0.87767759 0.77972972 0.40657561 0.40114354 0.24053522]
PS E:\SEM 3\PWP>
```

Taks:

import numpy as np

Example 1: Creation of 1D array

arr1=np.array([10,20,30])

print("My 1D array:\n",arr1)

Output

```
27  import numpy as np
28  # Example 1: Creation of 1D array
29  arr1=np.array([10,20,30])
30  print("My 1D array:\n",arr1)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
My 1D array:
 [10 20 30]
PS E:\SEM 3\PWP>
```

Example 2: Create a 2D numpy array

arr2 = np.array([[10,20,30],[40,50,60]])

print("My 2D numpy array:\n", arr2)

Output

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
32  import numpy as np
33  arr2 = np.array([[10,20,30],[40,50,60]])
34  print("My 2D numpy array:\n", arr2)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
My 2D numpy array:
 [[10 20 30]
 [40 50 60]]
PS E:\SEM 3\PWP>
```

# Example 3: Create a sequence of integers
# from 0 to 20 with steps of 3
arr= np.arange(0, 20, 3)
print ("A sequential array with steps of 3:\n", arr)
Output

```
32  import numpy as np
33  arr= np.arange(0, 20, 3)
34  print ("A sequential array with steps of 3:\n", arr)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
A sequential array with steps of 3:
 [ 0  3  6  9 12 15 18]
PS E:\SEM 3\PWP>
```

# Example 4: Create a sequence of 5 values in range 0 to 3
arr= np.linspace(0, 3, 5)
print ("A sequential array with 5 values between 0 and 5:\n", arr)
Output

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
37   import numpy as np
38   arr= np.linspace(0, 3, 5)
39   print ("A sequential array with 5 values between 0 and 5:\n", arr)
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
A sequential array with 5 values between 0 and 5:
 [0.   0.75 1.5  2.25 3.  ]
PS E:\SEM 3\PWP>
```

# Example 8: Use ones() create array
arr = np.ones((2,3))
print("numpy array:\n", arr)
print("Type:", type(arr))
Output

```
44   import numpy as np
45   arr = np.ones((2,3))
46   print("numpy array:\n", arr)
47   print("Type:", type(arr))
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
numpy array:
 [[1. 1. 1.]
 [1. 1. 1.]]
Type: <class 'numpy.ndarray'>
PS E:\SEM 3\PWP>
```

NumPy Data Types
A data type is a way to specify the type of data that will be stored in an array. For example,
array1 = np.array([2, 4, 6])

NumPy Data Types

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

NumPy offers a wider range of numerical data types than what is available in Python. Here's the list of most commonly used numeric data types in NumPy:

1. int8, int16, int32, int64 - signed integer types with different bit sizes
2. uint8, uint16, uint32, uint64 - unsigned integer types with different bit sizes
3. float32, float64 - floating-point types with different precision levels
4. complex64, complex128 - complex number types with different precision levels

Check Data Type of a NumPy Array

```
import numpy as np
# create an array of  integers
int_array = np.array([-3, -1, 0, 1])
# create an array of floating-point numbers
float_array = np.array([0.1, 0.2, 0.3])
# create an array of complex numbers
complex_array = np.array([1+2j, 2+3j, 3+4j])
# check the data type of int_array
print(int_array.dtype)  # prints int64
# check the data type of float_array
print(float_array.dtype)  # prints float64
# check the data type of complex_array
print(complex_array.dtype)  # prints complex128
```

Output

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```python
import numpy as np
# create an array of  integers
int_array = np.array([-3, -1, 0, 1])
# create an array of floating-point numbers
float_array = np.array([0.1, 0.2, 0.3])
# create an array of complex numbers
complex_array = np.array([1+2j, 2+3j, 3+4j])
# check the data type of int_array
print(int_array.dtype)  # prints int64
# check the data type of float_array
print(float_array.dtype)  # prints float64
# check the data type of complex_array
print(complex_array.dtype)  # prints complex128
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
int32
float64
complex128
PS E:\SEM 3\PWP>
```

Creating NumPy Arrays With a Defined Data Type

In NumPy, we can create an array with a defined data type by passing the dtype parameter while calling the np.array() function. For example,

```
import numpy as np
# create an array of 8-bit integers
array1 = np.array([1, 3, 7], dtype='int8')
# create an array of unsigned 16-bit integers
array2 = np.array([2, 4, 6], dtype='uint16')
# create an array of 32-bit floating-point numbers
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

array3 = np.array([1.2, 2.3, 3.4], dtype='float32')
# create an array of 64-bit complex numbers
array4 = np.array([1+2j, 2+3j, 3+4j], dtype='complex64')
# print the arrays and their data types
print(array1, array1.dtype)
print(array2, array2.dtype)
print(array3, array3.dtype)
print(array4, array4.dtype)
Output

```
64  import numpy as np
65  # create an array of 8-bit integers
66  array1 = np.array([1, 3, 7], dtype='int8')
67  # create an array of unsigned 16-bit integers
68  array2 = np.array([2, 4, 6], dtype='uint16')
69  # create an array of 32-bit floating-point numbers
70  array3 = np.array([1.2, 2.3, 3.4], dtype='float32')
71  # create an array of 64-bit complex numbers
72  array4 = np.array([1+2j, 2+3j, 3+4j], dtype='complex64')
73  # print the arrays and their data types
74  print(array1, array1.dtype)
75  print(array2, array2.dtype)
76  print(array3, array3.dtype)
77  print(array4, array4.dtype)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
[1 3 7] int8
[2 4 6] uint16
[1.2 2.3 3.4] float32
[1.+2.j 2.+3.j 3.+4.j] complex64
PS E:\SEM 3\PWP>
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

NumPy Type Conversion

In NumPy, we can convert the data type of an array using the astype() method. For example,

```
import numpy as np
# create an array of integers
int_array = np.array([1, 3, 5, 7])
# convert data type of int_array to float
float_array = int_array.astype('float')
# print the arrays and their data types
print(int_array, int_array.dtype)
print(float_array, float_array.dtype)
```

Output

```
80  import numpy as np
81  # create an array of integers
82  int_array = np.array([1, 3, 5, 7])
83  # convert data type of int_array to float
84  float_array = int_array.astype('float')
85  # print the arrays and their data types
86  print(int_array, int_array.dtype)
87  print(float_array, float_array.dtype)
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
[1 3 5 7] int32
[1. 3. 5. 7.] float64
PS E:\SEM 3\PWP>
```

NumPy Array Attributes

In NumPy, attributes are properties of NumPy arrays that provide information about the array's shape, size, data type, dimension, and so on.

Common NumPy Attributes

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

Here are some of the commonly used NumPy attributes:

Attributes	Description
ndim	returns number of dimension of the array
size	returns number of elements in the array
dtype	returns data type of elements in the array
shape	returns the size of the array in each dimension.
itemsize	returns the size (in bytes) of each elements in the array
data	returns the buffer containing actual elements of the array in memory

The ndim attribute returns the number of dimensions in the numpy array. For example,

```
import numpy as np
# create a 2-D array
array1 = np.array([[2, 4, 6],
                   [1, 3, 5]])
# check the dimension of array1
print(array1.ndim)
```

Output

```
89  import numpy as np
90  # create a 2-D array
91  array1 = np.array([[2, 4, 6],
92                     [1, 3, 5]])
93  # check the dimension of array1
94  print(array1.ndim)

PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
2
PS E:\SEM 3\PWP>
```

NumPy Array size Attribute

The size attribute returns the total number of elements in the given array.

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
import numpy as np
array1 = np.array([[1, 2, 3],
                   [6, 7, 8]])
# return total number of elements in array1
print(array1.size)
```

Output

```
 96  import numpy as np
 97  array1 = np.array([[1, 2, 3],
 98                     [6, 7, 8]])
 99  # return total number of elements in array1
100  print(array1.size)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
6
PS E:\SEM 3\PWP>
```

NumPy Array shape Attribute

In NumPy, the shape attribute returns a tuple of integers that gives the size of the array in each dimension. For example,

```
import numpy as np
array1 = np.array([[1, 2, 3],
                   [6, 7, 8]])
# return a tuple that gives size of array in each dimension
print(array1.shape)
```

Output

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
102   import numpy as np
103   array1 = np.array([[1, 2, 3],
104                      [6, 7, 8]])
105   # return a tuple that gives size of array in each dimension
106   print(array1.shape)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
(2, 3)
PS E:\SEM 3\PWP>
```

NumPy Array dtype Attribute

We can use the dtype attribute to check the datatype of a NumPy array. For example,

```
import numpy as np
# create an array of integers
array1 = np.array([6, 7, 8])
# check the data type of array1
print(array1.dtype)
```

Output

```
108   import numpy as np
109   # create an array of integers
110   array1 = np.array([6, 7, 8])
111   # check the data type of array1
112   print(array1.dtype)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
int32
PS E:\SEM 3\PWP>
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

NumPy Array itemsize Attribute

In NumPy, the itemsize attribute determines size (in bytes) of each element in the array. For example,

```
import numpy as np
# create a default 1-D array of integers
array1 = np.array([6, 7, 8, 10, 13])
# create a 1-D array of 32-bit integers
array2 = np.array([6, 7, 8, 10, 13], dtype=np.int32)
# use of itemsize to determine size of each array element of array1 and array2
print(array1.itemsize)  # prints 8
print(array2.itemsize)  # prints 4
```

Output

```
114  import numpy as np
115  # create a default 1-D array of integers
116  array1 = np.array([6, 7, 8, 10, 13])
117  # create a 1-D array of 32-bit integers
118  array2 = np.array([6, 7, 8, 10, 13], dtype=np.int32)
119  # use of itemsize to determine size of each array element of array1 and array2
120  print(array1.itemsize)  # prints 8
121  print(array2.itemsize)  # prints 4
```

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  Code
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
4
4
PS E:\SEM 3\PWP>
```

NumPy Array data Attribute

In NumPy, we can get a buffer containing actual elements of the array in memory using the data attribute. In simpler terms, the data attribute is like a pointer to the memory location where the array's data is stored in the computer's memory.

```
import numpy as np
array1 = np.array([6, 7, 8])
array2 = np.array([[1, 2, 3],
                   [6, 7, 8]])
# print memory address of array1's and array2's data
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

print("\nData of array1 is: ",array1.data)
print("Data of array2 is: ",array2.data)

```
123    import numpy as np
124    array1 = np.array([6, 7, 8])
125    array2 = np.array([[1, 2, 3],
126                       [6, 7, 8]])
127    # print memory address of array1's and array2's data
128    print("\nData of array1 is: ",array1.data)
129    print("Data of array2 is: ",array2.data)
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"

Data of array1 is:  <memory at 0x02278B68>
Data of array2 is:  <memory at 0x0240F338>
PS E:\SEM 3\PWP>
```

Task
Multiplication of two given matrixes

```
import numpy as np
p = [[1, 0], [0, 1]]
q = [[1, 2], [3, 4]]
print("Original matrices:")
print(p)
print(q)
# Perform matrix multiplication using np.dot
result1 = np.dot(p, q)
print("Result of the matrix multiplication:")
print(result1)
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

Output

```
131   import numpy as np
132   p = [[1, 0], [0, 1]]
133   q = [[1, 2], [3, 4]]
134   print("Original matrices:")
135   print(p)
136   print(q)
137   # Perform matrix multiplication using np.dot
138   result1 = np.dot(p, q)
139   print("Result of the matrix multiplication:")
140   print(result1)
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
Original matrices:
[[1, 0], [0, 1]]
[[1, 2], [3, 4]]
Result of the matrix multiplication:
[[1 2]
 [3 4]]
PS E:\SEM 3\PWP>
```

Compute the determinant of a given square array.

```
import numpy as np
from numpy import linalg as LA
a = np.array([[1, 0], [1, 2]])
# Display the original 2x2 array 'a'
print("Original 2-d array")
print(a)
print("Determinant of the said 2-D array:")
print(np.linalg.det(a))
```

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

Output

```
142    import numpy as np
143    from numpy import linalg as LA
144    a = np.array([[1, 0], [1, 2]])
145    # Display the original 2x2 array 'a'
146    print("Original 2-d array")
147    print(a)
148    print("Determinant of the said 2-D array:")
149    print(np.linalg.det(a))
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\lab_7.py"
Original 2-d array
[[1 0]
 [1 2]]
Determinant of the said 2-D array:
2.0
PS E:\SEM 3\PWP>
```

Post Lab Exercise:

a. Write a NumPy program to create a 3x3 matrix with values ranging from 2 to 10.

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```python
import numpy as np

#Question 1
matrix = np.arange(2, 11).reshape(3, 3)
print(matrix)
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\post_lab_7.py"
[[ 2  3  4]
 [ 5  6  7]
 [ 8  9 10]]
PS E:\SEM 3\PWP>
```

b. Write a NumPy program to reverse an array (the first element becomes the last).

```python
#Question 2
arr = np.array([1, 2, 3, 4, 5])
reversed_arr = np.flip(arr)
print(reversed_arr)
```

```
PROBLEMS 3  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\post_lab_7.py"
[5 4 3 2 1]
PS E:\SEM 3\PWP>
```

c. Write a NumPy program to find common values between two arrays.

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
12  #Question 3
13  a1 = np.array([11,12,13,14])
14  a2 = np.array([13,14,15,16])
15  common = np.intersect1d(a1, a2)
16  print(common)
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\post_lab_7.py"
[13 14]
PS E:\SEM 3\PWP>
```

d. Write a NumPy program to repeat array elements.

```
18  #Question 4
19  a = np.array([1, 2, 3])
20  repeated = np.repeat(a, 3)
21  print(repeated)
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\post_lab_7.py"
[1 1 1 2 2 2 3 3 3]
PS E:\SEM 3\PWP>
```

e. Write a NumPy program to find the memory size of a NumPy array.

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
23  #Question 5
24  a = np.array([1, 2, 3, 4, 5])
25  memory_size = a.nbytes
26  print("Memory size in bytes:", memory_size)
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\post_lab_7.py"
Memory size in bytes: 20
PS E:\SEM 3\PWP>
```

f. Write a NumPy program to create an array of ones and zeros.

```
28  #Question 6
29  zeros = np.zeros((3, 3))
30  ones = np.ones((2, 4))
31  print("Array of zeroes:\n", zeros)
32  print("Array of ones:\n", ones)
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\post_lab_7.py"
Array of zeroes:
 [[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
Array of ones:
 [[1. 1. 1. 1.]
 [1. 1. 1. 1.]]
PS E:\SEM 3\PWP>
```

g. Write a NumPy program to find the 4th element of a specified array.

Marwadi University Marwadi Chandarana Group NAAC A+	**Marwadi University** **Faculty of Engineering & Technology** **Department of Information and Communication Technology**	
Subject: Programming With Python (01CT1309)	**Aim:** Practical based on NumPy ndarray	
Experiment No: 07	**Date:**	**Enrollment No: 92400133055**

```
34  #Question 7
35  a = np.array([10, 20, 30, 40, 50])
36  fourth_element = a[3]
37  print("4th element:", fourth_element)
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\SEM 3\PWP> python -u "e:\SEM 3\PWP\Class Tutorials\post_lab_7.py"
4th element: 40
PS E:\SEM 3\PWP>
```