

**NumPy Exam Paper (Total 30 Questions - 2 Marks Each)**



**Section A: NumPy Creation (13 Questions)**

1. **From a Python List**   
 Write a code to create a NumPy array from the following list:

list1 = [1, 2, 3, 4, 5]

**Answer:**

import numpy as np

list = [1,2,3,4,5]

arr = np.array(list)

print(arr)

2. **From a Tuple**   
 Convert the tuple (10, 20, 30, 40) into a NumPy array.

**Answer:**

import numpy as np

tuple\_arr = (10, 20, 30, 40)

arr = np.array(tuple\_arr)

print(arr)

3. **Using arange**   
 Create a NumPy array from 0 to 30 with a step of 5 using arange .

**Answer:**

import numpy as np

arr = np.arange(0, 30, 5)

print(arr)

4. **Using linspace**   
 Create an array of 6 evenly spaced values between 0 and 50 using linspace .

**Answer:**

import numpy as np

arr = np.linspace(0, 50, 6)

print(arr)

5. **Using ones**   
 Create a 4x4 matrix of ones using NumPy.

**Answer:**

import numpy as np

arr = np.ones((4, 4))

print(arr)

6. **Using zeros**   
 How do you create a 3x3 matrix of zeros in NumPy?

**Answer:**

arr = np.zeros((3, 3))

print(arr)

7. **Using empty**   
 What is the purpose of empty in NumPy? Create a 2x2 uninitialized array.

**Answer:**

**Purpose of empty is to allocate memory when the initial values are not ready and can be assigned later**

arr = np.empty((2, 2))

print(arr)

8. **Using full**   
 Create a 5x5 array where all elements are equal to 9 using full .

**Answer:**

arr = np.full((5, 5), 9)

print(arr)

9. **Using eye**   
 Create a 3x3 identity matrix using eye .

**Answer:**

arr = np.eye(3)

print(arr)

10. **Using random**   
Generate a 2x2 matrix of random integers between 1 and 100 using NumPy’s random module.

**Answer:**

arr = np.random.randint(1, 101, size=(2, 2))

print(arr)

11. **Using astype**   
 Convert the array np.array([10, 20, 30]) to a float array using astype .

**Answer:**

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

float\_arr = arr.astype(float)

print(float\_arr)

12. **Using reshape**   
 Reshape the array np.arange(9) into a 3x3 matrix.

**Answer:**

arr = np.arange(9)

arr = arr.reshape(3, 3)

print(arr)

13. **Using diag**   
 Create a 4x4 matrix with the diagonal elements [10, 20, 30, 40] using diag .

**Answer:**

arr = np.diag([10, 20, 30, 40])

print(arr)

**Section B: Indexing, Slicing, and Fancy Indexing (12 Questions)**

14. **Accessing Elements in 1D Array**   
 Access the third element of the array np.array([5, 10, 15, 20, 25]) .

**Answer:**

arr = np.array([5, 10, 15, 20, 25])

print(arr[2])

15. **Accessing Elements in 2D Array**   
Retrieve the element at row 2, column 3 from the 2D array np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) .

**Answer:**

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

print(arr[1, 2])

16. **Negative Indexing in 1D Array**   
Use negative indexing to access the last element of the array np.array([12, 23, 34, 45]) .

**Answer:**

arr = np.array([12, 23, 34, 45])

print(arr[-1])

17. **Slicing a 1D Array**   
 Slice the array np.array([10, 20, 30, 40, 50, 60]) to get the first four elements.

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

print(arr[:4])

18. Slicing a 2D **Array**   
From the array np.array([[10, 20, 30], [40, 50, 60], [70, 80, 90]]) , slice out the first two rows and the first two columns.

**Answer:**

arr = np.array([[10, 20, 30], [40, 50, 60], [70, 80, 90]])

print(arr[:2, :2])

19. **Reverse a 1D Array Using Slicing**   
 Reverse the array np.array([1, 2, 3, 4, 5]) using slicing.

**Answer:**

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

print(arr[::-1])

20. **Fancy Indexing in 1D Array**   
Using fancy indexing, select the 1st, 3rd, and 4th elements from the array arr = np.array([10, 20, 30, 40, 50]) .

**Answer:**

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

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

subset = arr[indices]

print(subset)

21. **Fancy Indexing in 2D Array**   
 Use fancy indexing to retrieve elements at positions (0, 1), (1, 2), and (2, 0) from the array arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) .

**Answer:**

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

**indices = np.array([(0, 1), (1, 2), (2, 0)])**

**subset = arr[indices]**

**print(subset)**

22. **Slice every second element in 1D Array**   
 From the array np.array([1, 2, 3, 4, 5, 6, 7, 8]) , slice every second element.

**Answer:**

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

print(arr[::2])

23. **Slice every second column in a 2D Array**   
For the array np.array([[10, 20, 30, 40], [50, 60, 70, 80], [90, 100, 110, 120]]) , slice every second column.

**Answer:**

arr = np.array([[10, 20, 30, 40], [50, 60, 70, 80], [90, 100, 110, 120]])

print(arr[:, ::2])

24. **Access last row using negative indexing**   
Retrieve the last row from the array np.array([[1, 2], [3, 4], [5, 6], [7, 8]]) using negative indexing.

**Answer:**

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

print(arr[::-4])

25. **Reverse each row in a 2D Array**   
Reverse the order of elements in each row of the array np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) .

**Answer:**

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

reversed\_arr = np.flip(arr, axis=1)

print(reversed\_arr)

**Section C: NumPy Copying (5 Questions)**

26. **Shallow Copy Using view()**   
 What is a shallow copy in NumPy? Demonstrate with a code example using view() .

**Answer:**

**Shallow copy – New object contains the pointers to the same element as the original element. Changed made to the original or the copy would impact one another**

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

view\_arr = arr.view()

print(view\_arr)

27. **Shallow Copy Modification**   
In a shallow copy, how does modifying an element affect the original array? Provide a code example.

**Answer:**

It modifies the original array

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

arr[0] = 7

view\_arr = arr.view()

print(view\_arr)

[7 2 3 4 5]

28. **Deep Copy Using copy()**   
What is a deep copy in NumPy? Show how to create a deep copy using the copy() method.

**Answer:**

**A deep copy creates an independent copy of an object and all its nested objects. This means that changes made to the copy will affect the original and vice versa**

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

**view\_arr = arr.copy()**

**print(view\_arr)**

**[1 2 3 4 5]**

29. **Effect of Modifying Deep Copy**   
 Does modifying a deep copy affect the original array? Illustrate with an example.

**Answer:**

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

arr[0] = 7

view\_arr = arr.copy()

view\_arr[1] = 8

print(view\_arr)

print(arr)

[7 8 3 4 5]

[7 2 3 4 5]

30. **Difference Between Shallow and Deep Copy**   
 Briefly explain the difference between shallow and deep copy in NumPy with examples.

**Answer:**

**Shallow copy – New object contains the pointers to the same element as the original element. Changed made to the original or the copy would impact one another**

**A deep copy creates an independent copy of an object and all its nested objects. This means that changes made to the copy will affect the original and vice versa**

