NumPy Practice Questions

BASIC LEVEL (15 Questions)

1. Create a 1D array from 0 to 9 using np.arange(). Create a 2x3 array filled with zeros. Create a 3x3 identity matrix.

2. Given an array, print its shape, dimensions, data type, and memory size:

```
arr = np.array([[1, 2, 3], [4, 5, 6]], dtype=np.float32)
```

3. Convert array [0, 1, 2, 3, 4, 5] to shape (2, 3).

4. Create a 4x4 array and extract first row and last column. Extract elements at positions [0,1], [1,2], [2,0].

5. Add two arrays: [1, 2, 3] and [4, 5, 6]. Multiply arrays element-wise: [2, 3, 4] * [1, 2, 3].

6. Reverse array [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]. Flatten 2D array [[1, 2], [3, 4], [5, 6]] to 1D.

7. Stack [1, 2, 3] and [4, 5, 6] horizontally and vertically.

8. Replace all odd numbers in array with -1. Replace all values > 5 with 100 in array [1, 6, 3, 8, 2, 9].

9. Calculate mean, median, standard deviation of [10, 20, 30, 40, 50]. Find min and max values and their indices.

10. Normalize array [10, 20, 30, 40, 50] to range [0, 1].

11. Generate 10 random integers between 1 and 100. Create 3x3 random array and sort it row-wise.

12. Transpose matrix [[1, 2], [3, 4]]. Calculate dot product of [1, 2, 3] and [4, 5, 6].

13. Find common elements between [1, 2, 3, 4, 5] and [4, 5, 6, 7, 8]. Get positions where two arrays match.

14. Check if array contains NaN values: [1, 2, np.nan, 4]. Count non-zero elements in array.

15. Extract numbers between 5 and 10 from array [1, 3, 6, 8, 12, 7, 9]. Filter array elements based on condition arr > mean(arr).

INTERMEDIATE LEVEL (15 Questions)

16. Add scalar 10 to 2D array [[1, 2, 3], [4, 5, 6]]. Multiply 3x3 matrix by 1D array [1, 2, 3] using broadcasting.

17. Extract odd rows and even columns from 5x4 array. Use fancy indexing to select specific rows: [0, 2, 4].

18. Swap columns 1 and 2 in 2D array. Delete second column and insert new column [10, 10, 10].

19. Sort 2D array by values in second column. Find k smallest and k largest values in array.

20. Split array [0, 1, 2, 3, 4, 5, 6, 7, 8] into 3 equal parts. Join multiple arrays with different concatenation methods.

21. Calculate percentiles (25th, 50th, 75th) of array. Compute correlation coefficient between two arrays.

22. Calculate sum along different axes of 3D array. Find mean of each row and column in 2D array.

23. Count occurrences of each unique element in array. Find unique values and their frequencies simultaneously.

24. Use np.where() to replace values conditionally. Apply different operations based on conditions using np.select().

25. Find intersection, union, and difference of two arrays. Check element-wise equality with tolerance for floating-point arrays.

26. Create structured array with fields: name (string), age (int), salary (float). Filter structured array where age > 30 and salary > 50000.

27. Compare memory usage of different array dtypes. Optimize array operations using vectorization vs loops.

28. Implement matrix-vector operations using broadcasting. Handle shape mismatches in complex broadcasting scenarios.

29. Create universal function (ufunc) for custom mathematical operation. Apply function element-wise vs using loops (performance comparison).

30. Handle division by zero in arrays gracefully. Deal with memory limitations when working with large arrays.

ADVANCED LEVEL (12 Questions)

31. Calculate eigenvalues and eigenvectors of matrix [[4, 2], [1, 3]]. Verify eigendecomposition: $Av = \lambda v$. Handle complex eigenvalues and eigenvectors.

32. Solve system of linear equations: $2x + 3y = 7$, $x - y = 1$. Calculate matrix inverse and check for singularity. Compute QR decomposition of matrix.

33. Perform SVD on matrix and reconstruct original matrix. Use SVD for dimensionality reduction and noise filtering.

34. Calculate different matrix norms (Frobenius, spectral, nuclear). Compute condition number and assess numerical stability.

35. Implement sliding window operations on arrays. Create efficient algorithms for array rotations and circular shifts.

36. Implement complex filters using convolution. Process arrays with missing data and interpolation.

37. Optimize nested loops using vectorization. Compare performance of different NumPy approaches for same problem.

38. Work with 4D+ arrays (e.g., image batches, time series). Implement tensor operations and broadcasting in high dimensions.

39. Implement numerical integration using NumPy. Create finite difference methods for derivatives.

40. Create complex nested structured arrays. Implement efficient database-like operations on structured arrays.

BONUS CHALLENGE QUESTIONS

41. Implement Cholesky decomposition for positive definite matrices. Create custom matrix factorization algorithms.

42. Use NumPy to implement gradient descent. Solve nonlinear equations using Newton's method.