# Beginner NumPy Practice Problems (20 Questions)

1. Create 1D and 2D arrays
- Create a 1D NumPy array from the Python list [2, 4, 6, 8].
- Create a 2D NumPy array with 2 rows and 3 columns using any numbers you want.

2. Check basic properties
Given:
X = np.array([[1, 2, 3],
[4, 5, 6]])
- Print X.shape
- Print X.ndim
- Print X.dtype

3. Zeros and ones
- Create a 3×3 array filled with zeros.
- Create a 2×4 array filled with ones.

4. arange vs linspace
- Use np.arange to create an array: [0, 2, 4, 6, 8].
- Use np.linspace to create 5 values from 0 to 1 (inclusive).

5. Indexing single elements
Given:
X = np.array([[800, 2, 15],
[950, 2, 10],
[1200, 3, 8],
[1500, 3, 5]])
- Get the first element (800)
- Get the value 3 (bedrooms of the 3rd row)
- Get the last element (5)

6. Row and column selection
Using the same X as above:
- Get the first row as an array.
- Get the second column (all bedroom values).

7. Slicing rows and columns
From the same X:
- Get the first 2 rows.
- Get rows 1 to 3 and columns 0 to 1.

8. Feature/target split
Consider:
data = np.array([
[800, 2, 15, 75000],
[950, 2, 10, 90000],

[1200, 3, 8, 120000],
[1500, 3, 5, 155000]
])
- Use slicing to create X (first 3 columns) and y (last column).

9. Mean and standard deviation
Using the same data array:
- Compute the overall mean of all numbers.
- Compute the standard deviation of all numbers.

10. Mean per column
For data above:
- Compute the mean of each column using axis=0.
- What does the mean of the last column represent?

11. Min and max per feature
For data above:
- Compute the minimum of each feature column (first 3 columns).
- Compute the maximum of each feature column (first 3 columns).

12. Reshape and flatten
- Create a = np.array([1, 2, 3, 4, 5, 6]).
- Reshape it into a 2×3 matrix.
- Then flatten it back to 1D.

13. Vertical and horizontal stacking
Given:
A = np.array([[1, 2],
[3, 4]])
B = np.array([[5, 6],
[7, 8]])
- Use np.vstack to stack them vertically.
- Use np.hstack to stack them horizontally.

14. Scalar operations
- Let X = np.array([10, 20, 30, 40]).
- Subtract 10 from all elements.
- Divide all elements by 10.
- Print the result after each step.

15. Elementwise operations
Given:
a = np.array([1, 2, 3])
b = np.array([10, 20, 30])
- Compute a + b
- Compute a * b
- Compute a / b
- Explain why these operations are called elementwise.

16. Vector dot product
Given:
w = np.array([1, 2, 3])
x = np.array([4, 5, 6])
- Compute np.dot(w, x).
- Compute w @ x.
- Verify both results are the same.

17. Matrix–vector multiplication (simple model)
Given:
X = np.array([
[1, 2, 3],
[4, 5, 6]
])
w = np.array([0.1, 0.2, 0.3])
b = 0.5
- Compute y_pred = X @ w + b.
- What does each element of y_pred represent?

18. Transpose and sums
Given:
A = np.array([[1, 2, 3],
[4, 5, 6]])
- Compute A.T.
- Compute A.sum(axis=0).
- Compute A.sum(axis=1).
- Explain the difference between sum over axis 0 and axis 1.

19. Boolean indexing (filtering)
Given:
prices = np.array([75000, 90000, 120000, 155000, 60000])
- Create a boolean mask for prices > 100000.
- Use it to get only the expensive houses.
- Count how many houses are above 100000.

20. Shuffling dataset (X and y together)
Using the data array from problem 8:
- Split into X (first 3 columns) and y (last column).
- Generate shuffled indices using np.random.permutation(len(X)).
- Create X_shuffled and y_shuffled so that each row and label still match after shuffling.