**🔢 NumPy – Questions & Answers**

**Q1. What is NumPy and why is it useful?**  
**A:** NumPy is a Python library for numerical computations. It provides support for arrays, matrices, and high-level mathematical functions. It's faster and more memory efficient than Python lists.

**Q2. How do you create a NumPy array?**

import numpy as np

a = np.array([1, 2, 3])

**Q3. How do you generate a range of values?**

np.arange(0, 10, 2) # Output: array([0, 2, 4, 6, 8])

**Q4. What's the difference between np.array, np.linspace, and np.arange?**

* np.array() → convert list to array
* np.arange(start, stop, step) → evenly spaced values
* np.linspace(start, stop, num) → fixed number of values between a range

**🐼 Pandas – Questions & Answers**

**Q1. What is Pandas used for?**  
**A:** Data manipulation, cleaning, and analysis using Series and DataFrame.

**Q2. How do you read a CSV file?**

import pandas as pd

df = pd.read\_csv('data.csv')

**Q3. How do you get basic info about a DataFrame?**

df.head(), df.info(), df.describe()

**Q4. How to filter rows where a column equals a value?**

df[df['column\_name'] == 'value']

**Q5. How do you group data and get the mean?**

df.groupby('Category')['Sales'].mean()

**📊 Matplotlib – Questions & Answers**

**Q1. How do you create a simple line plot?**

import matplotlib.pyplot as plt

plt.plot([1, 2, 3], [4, 5, 6])

plt.show()

**Q2. How to add labels and title?**

plt.xlabel('X-axis')

plt.ylabel('Y-axis')

plt.title('My Plot')

**🌈 Seaborn – Questions & Answers**

**Q1. What is Seaborn and how is it different from Matplotlib?**  
**A:** Seaborn is built on top of Matplotlib, and it provides a high-level API for statistical visualizations with better default themes.

**Q2. How do you create a boxplot and heatmap?**

import seaborn as sns

sns.boxplot(data=df, x='Category', y='Sales')

sns.heatmap(df.corr(), annot=True)

**🔍 EDA (Exploratory Data Analysis) – Questions & Answers**

**Q1. What are the main steps in EDA?**

1. Understand the data (types, nulls, shape)
2. Univariate analysis
3. Bivariate/multivariate analysis
4. Outlier detection
5. Missing value treatment

**Q2. How to check missing values in a DataFrame?**

df.isnull().sum()

**Q3. How to find correlations between features?**

df.corr()

**📈 Data Analysis – Questions & Answers**

**Q1. How to summarize numerical columns?**

df.describe()

**Q2. How to find top 10 products by sales?**

df.groupby('Product Name')['Sales'].sum().sort\_values(ascending=False).head(10)

**Q3. How to detect outliers using IQR?**

Q1 = df['Sales'].quantile(0.25)

Q3 = df['Sales'].quantile(0.75)

IQR = Q3 - Q1

outliers = df[(df['Sales'] < (Q1 - 1.5 \* IQR)) | (df['Sales'] > (Q3 + 1.5 \* IQR))]

**🔰 Beginner-Level Questions**

**1. What is Pandas?**

**Answer**: Pandas is a Python library for data manipulation and analysis. It provides data structures like Series and DataFrame.

**2. What are Series and DataFrame in Pandas?**

* **Series**: A one-dimensional labeled array.
* **DataFrame**: A two-dimensional labeled data structure (like a table in Excel or SQL).

import pandas as pd

s = pd.Series([10, 20, 30])

df = pd.DataFrame({'A': [1, 2], 'B': [3, 4]})

**3. How do you read a CSV or Excel file using Pandas?**

pd.read\_csv("file.csv")

pd.read\_excel("file.xlsx")

**4. How do you check for missing values?**

df.isnull().sum()

**5. How do you get summary statistics of a DataFrame?**

df.describe()

**⚙️ Intermediate-Level Questions**

**6. How do you filter rows in a DataFrame?**

df[df['Sales'] > 1000]

**7. How do you group data in Pandas?**

df.groupby('Category')['Profit'].sum()

**8. How do you sort a DataFrame?**

df.sort\_values(by='Sales', ascending=False)

**9. How do you merge/join DataFrames?**

pd.merge(df1, df2, on='Customer ID', how='inner')

**10. How do you apply a function to a column?**

df['Discount\_10'] = df['Discount'].apply(lambda x: x \* 1.1)

**🧠 Advanced-Level Questions**

**11. What is the difference between loc and iloc?**

* loc: Label-based indexing.
* iloc: Integer-location based indexing.

df.loc[0:2, ['Sales']] # by label

df.iloc[0:2, [1]] # by position

**12. How do you handle missing data?**

* Drop rows: df.dropna()
* Fill values: df.fillna(0)

**13. What’s the difference between drop, pop, and del in Pandas?**

* drop(): Returns a new DataFrame.
* pop(): Removes and returns the column.
* del: Deletes column in place.

**14. What’s the use of pivot\_table() in Pandas?**

Creates summary tables with aggregation:

df.pivot\_table(index='Region', values='Profit', aggfunc='sum')

**15. How do you find correlation between columns?**

df.corr()

**16. How to speed up Pandas operations?**

* Use **vectorized** operations.
* Avoid loops.
* Use **categorical** data for text-heavy columns.

**17. What’s the difference between apply() and map()?**

* map(): Works on Series.
* apply(): Works on DataFrame or Series (more flexible).

**18. How to get top N records by group?**

df.groupby('Category').apply(lambda x: x.nlargest(3, 'Profit'))

**🔰 Beginner-Level Questions**

**1. What is NumPy? Why is it used?**

**Answer**: NumPy (Numerical Python) is a fundamental Python library for numerical computations. It provides fast and memory-efficient array operations, and it is widely used for data analysis, scientific computing, and machine learning.

**2. What is the difference between a Python list and a NumPy array?**

| **Feature** | **Python List** | **NumPy Array** |
| --- | --- | --- |
| Speed | Slower | Faster (C-based) |
| Type | Heterogeneous | Homogeneous |
| Operations | Element-wise loops | Vectorized ops |

**3. How do you create a NumPy array?**

import numpy as np

a = np.array([1, 2, 3])

**4. How to create arrays with default values (zeros, ones, etc.)?**

np.zeros((2, 3)) # 2x3 array of zeros

np.ones(5) # 1D array of ones

np.eye(3) # 3x3 Identity matrix

np.full((2, 2), 7) # 2x2 filled with 7

**5. How do you check the shape and size of an array?**

a.shape # Dimensions

a.size # Total elements

a.ndim # Number of dimensions

**⚙️ Intermediate-Level Questions**

**6. How do you generate random numbers in NumPy?**

np.random.rand(2, 3) # Uniform [0, 1)

np.random.randint(1, 10) # Integer from [1, 10)

**7. How do you reshape an array?**

a = np.array([1, 2, 3, 4, 5, 6])

a.reshape((2, 3))

**8. What is broadcasting in NumPy?**

**Answer**: Broadcasting is NumPy’s ability to perform arithmetic operations on arrays of different shapes.

a = np.array([1, 2, 3])

b = 2

print(a + b) # [3, 4, 5]

**9. How do you perform element-wise operations?**

a = np.array([1, 2, 3])

b = np.array([4, 5, 6])

print(a + b)

print(a \* b)

**10. How do you filter elements in a NumPy array?**

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

a[a > 3] # [4, 5]

**🧠 Advanced-Level Questions**

**11. What’s the difference between copy() and view()?**

* copy(): Creates a new array with a copy of the data.
* view(): Creates a new array object that looks at the same data.

a = np.array([1, 2, 3])

b = a.copy()

c = a.view()

**12. Explain np.where() with an example.**

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

np.where(a > 25, 'Yes', 'No') # ['No', 'No', 'Yes', 'Yes']

**13. How to compute statistical operations in NumPy?**

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

a.mean(), a.std(), a.var(), np.median(a)

**14. What is np.linspace() and np.arange()?**

* np.arange(start, stop, step) – Like Python range()
* np.linspace(start, stop, num) – Evenly spaced values between two numbers

np.arange(1, 10, 2) # [1 3 5 7 9]

np.linspace(0, 1, 5) # [0. 0.25 0.5 0.75 1.]

**15. How do you handle NaN or inf values in NumPy arrays?**

np.isnan(a), np.isinf(a)

np.nan\_to\_num(a) # Replaces NaN with 0

* **Find the second-largest value in a NumPy array**

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

np.sort(arr)[-2]

* **Matrix multiplication**

np.dot(a, b) # or a @ b