Python + Applied Python Test

Section 1: Python

Q1. Write a Python program to calculate the final price of a product after applying a series of discounts based on the following conditions:

- If the product price is above Rs1000, apply a 10% discount.
- If the customer is a member, apply an additional 5% discount.
- If the purchase is made during a sale period, apply an additional 7% discount.

Note: Discount should be applicable on resultant price of previous discount.

Test Case:

Input: Product price = 1500

Is Customer member = Yes

Sale period = No

Output: Final Price = 1282.5

Q2. Write a Python program to find the **sum of all prime numbers** in a given range.

Test Case:

Input: 1 10

Output: 26 (because 2, 3, 5, 7, 9 are prime numbers so, 2 + 3 + 5 + 7 + 9 = 26)

Q3: Input: n = 5,

Output:

*								*
*	*						*	*
*		*				*		*
*			*		*			*
*	*	*	*	*	*	*	*	*

Q4: Write a program to calculate and print the sum of elements in each list of lists and add the resultant values. Without using sum() function.

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

Output: [12, 15, 14]

41 → *(12+15+14)*

Q5: Write a Python program to reverse the keys and values of a given dictionary using dictionary comprehension.

Dictionary: {'x': 1, 'y': 2, 'z': 3}

Expected Output: {1: 'x', 2: 'y', 3: 'z'}.

Section 2: NumPy

Q1:

Write a python program which takes dictionary as an input and convert it into a numpy array

Q2.

Create 3x3 2D NumPy array and then perform basic given operations:

- Delete the second column.
- Add new row [10, 20, 30].
- Insert a new column [11, 22, 33] at index 1.

Q3.

Write a NumPy program to create a 4x4 array with random values and find the sum of each row.

Q4.

Given a 2D NumPy array, replace all elements that are **greater than the mean of the array** with 1, and all elements **less than or equal to the mean** with 0.
Test Case:

Input: np.array([[4, 8, 12], [2, 6, 10], [1, 5, 9]])

Expected Output:

[[0 1 1]

[0 0 1]

[0 0 0]]

Q5:

Write a Python program to find **the total number of even numbers** present in a given 2D NumPy array.
Test Case:

Input: np.array([[5, 8, 11], [6, 9, 14], [2, 3, 7]])

Expected Output: 3 # (8, 6, 14, 2)

Q6:

You are given a space-separated list of numbers. Convert it into a **NumPy array of integers**, sort it in **descending order**, and return the array.
Test Case:

Input: "9 3 8 2 7 6"

Expected Output: [9 8 7 6 3 2]

Q7:

Given two NumPy arrays, return a new array that contains **the minimum value at each index** from the two given arrays.
Test Case:

Input:

a = np.array([9, 3, 6, 4, 2])

b = np.array([5, 7, 1, 8, 10])

Expected Output: [5 3 1 4 2]

Section 3: Pandas

Dataset: sns.load_dataset("taxi")

1. Extract and display the **hour of the day** from the pickup timestamp and determine the most common hour for rides.
2. Identify the top **three most popular drop-off locations** based on frequency.

3. Compute the **average tip percentage** for each payment method and determine which method yields the highest tips.
4. Find the ride with the **highest fare per mile** and display its details.
5. Filter and display only the trips that **started and ended in different boroughs**.
6. Fill missing values in categorical columns with the most frequently occurring value for each column.
7. Group the taxi dataset by "pickup_borough" and find the **average fare** for each borough.

Section 4: Visualization (Matplotlib and Seaborn)

NOTE: Understand the given data by plotting all these graphs and make insights from those graphs. Write the conclusion of each graph

1. Plot a time-series graph showing **the total revenue per day** over the dataset period.
2. Display the relationship between **distance and total fare** using a scatter plot.
3. Create a multi-panel plot (use subplots) showing separate distributions for **distance, fare, and tip**.
4. Plot a bar chart comparing **the number of rides per payment method**.

5. Create a pie chart showing the **proportion of total revenue contributed by each borough**.
6. Generate a box plot of fare values to identify **potential outliers**.
7. Display a heatmap showing the correlation between **distance, fare, tip, and total cost**.
8. Show the **variation in tip amounts across different boroughs** using a violin plot.
9. Plot the **density of ride distances** to understand its distribution.
10. Create a regression plot to examine the **relationship between distance and total fare**.
11. Display a count plot of **the number of rides for each hour of the day**.
12. Compare fare distributions across **different boroughs** using a box plot.
13. Show how **the number of passengers affects the total fare** using a swarm plot.
14. Create a strip plot showing the **variation in tip percentage by payment method**.
15. Use a pair plot to analyse relationships between **distance, fare, and tip** values.