

1. You are working on a project that involves analyzing student performance data for a class of 32 students. The data is stored in a NumPy array named `student_scores`, where each row represents a student and each column represents a different subject. The subjects are arranged in the following order: Math, Science, English, and History. Your task is to calculate the average score for each subject and identify the subject with the highest average score. Question: How would you use NumPy arrays to calculate the average score for each subject and determine the subject with the highest average score? Assume 4x4 matrix that stores marks of each student in given order.

```
1 import numpy as np
2 student_scores = np.array([
3     [85, 78, 92, 88],
4     [76, 85, 83, 91],
5     [90, 88, 79, 85],
6     [70, 75, 88, 80]
7 ])
8 subject_averages = np.mean(student_scores, axis=0)
9 subjects = ["Math", "Science", "English", "History"]
10 max_avg_index = np.argmax(subject_averages)
11 highest_subject = subjects[max_avg_index]
12 highest_avg_score = subject_averages[max_avg_index]
13 print("Average scores for each subject:")
14 for i in range(len(subjects)):
15     print(f"{subjects[i]}: {subject_averages[i]:.2f}")
16 print(f"\nSubject with the highest average score: {highest_subject}
17     ({highest_avg_score:.2f})")
```

Average scores for each subject:
Math: 80.25
Science: 81.50
English: 85.50
History: 86.00

Subject with the highest average score: History (86.00)

=== Code Execution Successful ===

2. Scenario: You are a data analyst working for a company that sells products online. You have been tasked with analyzing the sales data for the past month. The data is stored in a NumPy array. Question: How would you find the average price of all the products sold in the past month? Assume 3x3 matrix with each row representing the sales for a different product

```
1 import numpy as np
2
3 sales_data = np.array([
4     [100, 150, 200],
5     [80, 120, 160],
6     [90, 110, 130]
7 ])
8
9 all_prices = sales_data.flatten()
10
11 average_price = np.mean(all_prices)
12
13 print("All product prices from sales:")
14 print(all_prices)
15
16 print(f"\nAverage price of all products sold: {average_price:.2f}")
17
```

All product prices from sales:
[100 150 200 80 120 160 90 110 130]

Average price of all products sold: 126.67

=== Code Execution Successful ===

3. Scenario: You are working on a project that involves analyzing a dataset containing information about houses in a neighborhood. The dataset is stored in a CSV file, and you have imported it into a NumPy array named `house_data`. Each row of the array represents a house, and the columns contain various features such as the number of bedrooms, square footage, and sale price.

Question: Using NumPy arrays and operations, how would you find the average sale price of houses with more than four bedrooms in the neighborhood?

```
1 import numpy as np
2
3 house_data = np.array([
4     [3, 1800, 250000],
5     [5, 2400, 450000],
6     [6, 3000, 520000],
7     [4, 2000, 300000],
8     [7, 3500, 600000],
9     [2, 1500, 200000],
10    [5, 2800, 480000],
11    [6, 3200, 550000]
12 ])
13
14 houses_with_more_than_4_bedrooms = house_data[house_data[:, 0] > 4]
15
16 sale_prices = houses_with_more_than_4_bedrooms[:, 2]
17
18 average_sale_price = np.mean(sale_prices)
19
20 print("Average sale price of houses with more than 4 bedrooms:", round
      (average_sale_price, 2))
21
```

Average sale price of houses with more than 4 bedrooms: 520000.0

=== Code Execution Successful ===

4. Scenario: You are working on a project that involves analyzing the sales performance of a company over the past four quarters. The quarterly sales data is stored in a NumPy array named `sales_data`, where each element represents the sales amount for a specific quarter. Your task is to calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter.

Question: Using NumPy arrays and arithmetic operations calculate the total sales for the year and determine the percentage increase in sales from the first quarter to the fourth quarter?

```
main.py  [ ] [ ] [ ] Share Run Output
1 import numpy as np
2
3 sales_data = np.array([100000, 125000, 140000, 180000])
4
5 total_sales = np.sum(sales_data)
6
7 percentage_increase = ((sales_data[3] - sales_data[0]) / sales_data[0]) * 100
8
9 print("Total sales for the year:", total_sales)
10 print("Percentage increase from Q1 to Q4:", round(percentage_increase, 2), "%")
11
```

Total sales for the year: 545000

Percentage increase from Q1 to Q4: 80.0 %

=== Code Execution Successful ===

5. Scenario: You are a data analyst working for a car manufacturing company. As part of your analysis, you have a dataset containing information about the fuel efficiency of different car models. The dataset is stored in a NumPy array named `fuel_efficiency`, where each element represents the fuel efficiency (in miles per gallon) of a specific car model. Your task is to calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models.

Question: How would you use NumPy arrays and arithmetic operations to calculate the average fuel efficiency and determine the percentage improvement in fuel efficiency between two car models?

main.py	Output
<pre>1 import numpy as np 2 3 fuel_efficiency = np.array([22.5, 25.0, 27.5, 30.0, 35.0, 40.0]) 4 5 average_efficiency = np.mean(fuel_efficiency) 6 7 model_old = fuel_efficiency[1] 8 model_new = fuel_efficiency[5] 9 10 percentage_improvement = ((model_new - model_old) / model_old) * 100 11 12 print("Average fuel efficiency:", round(average_efficiency, 2), "mpg") 13 print("Percentage improvement from Model 2 to Model 6:", round((percentage_improvement, 2), "%")</pre>	<pre>Average fuel efficiency: 30.0 mpg Percentage improvement from Model 2 to Model 6: 60.0 % === Code Execution Successful ===</pre>

6. Scenario: You are a cashier at a grocery store and need to calculate the total cost of a customer's purchase, including applicable discounts and taxes. You have the item prices and quantities in separate lists, and the discount and tax rates are given as percentages. Your task is to calculate the total cost for the customer.

Question: Use arithmetic operations to calculate the total cost of a customer's purchase, including discounts and taxes, given the item prices, quantities, discount rate, and tax rate?

main.py	Output
<pre>1 import numpy as np 2 prices = np.array([100.0, 50.0, 30.0]) 3 quantities = np.array([2, 3, 1]) 4 5 discount_rate = 10 6 tax_rate = 5 7 8 subtotal = np.sum(prices * quantities) 9 10 discount_amount = (discount_rate / 100) * subtotal 11 subtotal_after_discount = subtotal - discount_amount 12 13 tax_amount = (tax_rate / 100) * subtotal_after_discount 14 total_cost = subtotal_after_discount + tax_amount 15 16 print("Subtotal:", round(subtotal, 2)) 17 print("Discount:", round(discount_amount, 2)) 18 print("Subtotal after discount:", round(subtotal_after_discount, 2)) 19 print("Tax:", round(tax_amount, 2)) 20 print("Total cost to customer:", round(total_cost, 2)) 21</pre>	<pre>Subtotal: 380.0 Discount: 38.0 Subtotal after discount: 342.0 Tax: 17.1 Total cost to customer: 359.1 === Code Execution Successful ===</pre>

7. Scenario: You are working as a data analyst for an e-commerce company. You have been given a dataset containing information about customer orders, stored in a Pandas DataFrame named `order_data`. The DataFrame has columns for customer ID, order date, product name, and order quantity. Your task is to analyze the data and answer specific questions about the orders.

Question: Using Pandas DataFrame operations, how would you find the following information from the `order_data` DataFrame:

1. The total number of orders made by each customer.
2. The average order quantity for each product.

3. The earliest and latest order dates in the dataset.

```
main.py  Run  Output

1 import pandas as pd
2 data = {
3     'Customer_ID': [101, 102, 101, 103, 102, 104, 101],
4     'Order_Date': pd.to_datetime([
5         '2023-01-10', '2023-01-12', '2023-01-15',
6         '2023-01-18', '2023-01-20', '2023-01-25',
7         '2023-01-28'
8     ]),
9     'Product_Name': ['Laptop', 'Mouse', 'Laptop', 'Keyboard', 'Mouse',
10        'Monitor', 'Laptop'],
11     'Order_Quantity': [1, 2, 1, 1, 3, 2, 1]
12 }
13 order_data = pd.DataFrame(data)
14 orders_per_customer = order_data.groupby('Customer_ID').size()
15 avg_quantity_per_product = order_data.groupby('Product_Name')['Order_Quantity']
16     .mean()
17 earliest_order = order_data['Order_Date'].min()
18 latest_order = order_data['Order_Date'].max()
19 print("1. Total number of orders per customer:")
20 print(orders_per_customer, "\n")
21 print("2. Average order quantity per product:")
22 print(avg_quantity_per_product, "\n")
23 print(earliest_order)
24 print(latest_order)
```

```
1. Total number of orders per customer:
Customer_ID
101    3
102    2
103    1
104    1
dtype: int64

2. Average order quantity per product:
Product_Name
Keyboard    1.0
Laptop     1.0
Monitor     2.0
Mouse      2.5
Name: Order_Quantity, dtype: float64

3. Earliest order date: 2023-01-10 00:00:00
4. Latest order date: 2023-01-28 00:00:00

=== Code Execution Successful ===
```

8. Scenario: You are a data scientist working for a company that sells products online. You have been tasked with analyzing the sales data for the past month. The data is stored in a Pandas data frame.

Question: How would you find the top 5 products that have been sold the most in the past month?

```
main.py  Run  Output

1 import pandas as pd
2
3 data = {
4     'Product_Name': ['Laptop', 'Mouse', 'Keyboard', 'Monitor', 'Mouse', 'Laptop',
5        'Monitor', 'Keyboard', 'Mouse', 'Laptop'],
6     'Quantity_Sold': [3, 5, 2, 4, 3, 2, 3, 1, 2, 4]
7 }
8 sales_df = pd.DataFrame(data)
9
10 total_sales = sales_df.groupby('Product_Name')['Quantity_Sold'].sum()
11
12 top_5_products = total_sales.sort_values(ascending=False).head(5)
13
14 print("Top 5 best-selling products in the past month:")
15 print(top_5_products)
16
```

```
Top 5 best-selling products in the past month:
Product_Name
Mouse      10
Laptop     9
Monitor    7
Keyboard   3
Name: Quantity_Sold, dtype: int64

=== Code Execution Successful ===
```

9. Scenario: You work for a real estate agency and have been given a dataset containing information about properties for sale. The dataset is stored in a Pandas DataFrame named `property_data`. The DataFrame has columns for property ID, location, number of bedrooms, area in square feet, and listing price. Your task is to analyze the data and answer specific questions about the properties.

Question: Using Pandas DataFrame operations, how would you find the following information from the `property_data` DataFrame:

1. The average listing price of properties in each location.
2. The number of properties with more than four bedrooms.
3. The property with the largest area.

main.py

Share

Run

Output

```

1 import pandas as pd
2 data = {
3     'Property_ID': [101, 102, 103, 104, 105, 106],
4     'Location': ['Downtown', 'Uptown', 'Downtown', 'Midtown', 'Uptown',
5                 'Midtown'],
6     'Bedrooms': [3, 5, 4, 2, 6, 4],
7     'Area_sqft': [1500, 2500, 1800, 1300, 2700, 2200],
8     'Listing_Price': [300000, 450000, 350000, 280000, 500000, 400000]
9 }
10 property_data = pd.DataFrame(data)
11 avg_price_per_location = property_data.groupby('Location')['Listing_Price'].mean()
12 num_properties_more_than_4_bed = property_data[property_data['Bedrooms'] > 4].shape[0]
13 property_largest_area = property_data.loc[property_data['Area_sqft'].idxmax()]
14 print("1. Average Listing Price by Location:")
15 print(avg_price_per_location, "\n")
16 print(f"2. Number of properties with more than 4 bedrooms: {num_properties_more_than_4_bed}\n")
17 print("3. Property with the largest area:")
18 print(property_largest_area)

```

1. Average Listing Price by Location:
Location
Downtown 325000.0
Midtown 340000.0
Uptown 475000.0
Name: Listing_Price, dtype: float64

2. Number of properties with more than 4 bedrooms: 2

3. Property with the largest area:
Property_ID 105
Location Uptown
Bedrooms 6
Area_sqft 2700
Listing_Price 500000
Name: 4, dtype: object

=== Code Execution Successful ===

10. Scenario: You are working on a data visualization project and need to create basic plots using Matplotlib. You have a dataset containing the monthly sales data for a company, including the month and corresponding sales values. Your task is to develop a Python program that generates line plots and bar plots to visualize the sales data.
- Question: 1. How would you develop a Python program to create a line plot of the monthly sales data?
- 2: How would you develop a Python program to create a bar plot of the monthly sales data?

```

import matplotlib.pyplot as plt

# Sample sales data
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
          'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
sales = [2500, 2700, 3000, 3200, 2800, 3500, 3700, 3600, 3900, 4100, 4300, 4500]

# 1. Line Plot of Monthly Sales
plt.figure(figsize=(10, 5))
plt.plot(months, sales, marker='o', color='blue', linestyle='--', linewidth=2)
plt.title('Monthly Sales Line Plot')
plt.xlabel('Month')
plt.ylabel('Sales ($)')
plt.grid(True)
plt.tight_layout()
plt.show()

# 2. Bar Plot of Monthly Sales
plt.figure(figsize=(10, 5))
plt.bar(months, sales, color='green')
plt.title('Monthly Sales Bar Plot')
plt.xlabel('Month')
plt.ylabel('Sales ($)')
plt.tight_layout()
plt.show()

```

