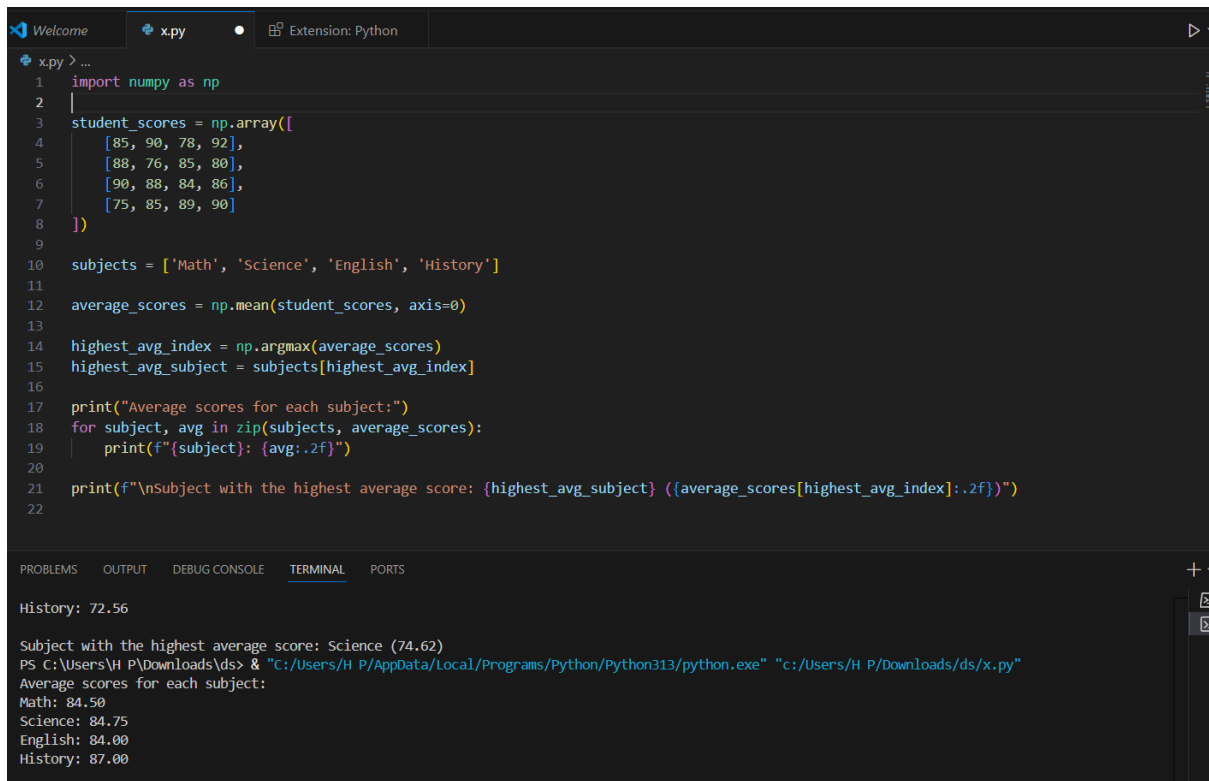


EXP-1



```
1 import numpy as np
2
3 student_scores = np.array([
4     [85, 90, 78, 92],
5     [88, 76, 85, 80],
6     [90, 88, 84, 86],
7     [75, 85, 89, 90]
8 ])
9
10 subjects = ['Math', 'Science', 'English', 'History']
11
12 average_scores = np.mean(student_scores, axis=0)
13
14 highest_avg_index = np.argmax(average_scores)
15 highest_avg_subject = subjects[highest_avg_index]
16
17 print("Average scores for each subject:")
18 for subject, avg in zip(subjects, average_scores):
19     print(f'{subject}: {avg:.2f}')
20
21 print(f"\nSubject with the highest average score: {highest_avg_subject} ({average_scores[highest_avg_index]:.2f})")
22
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

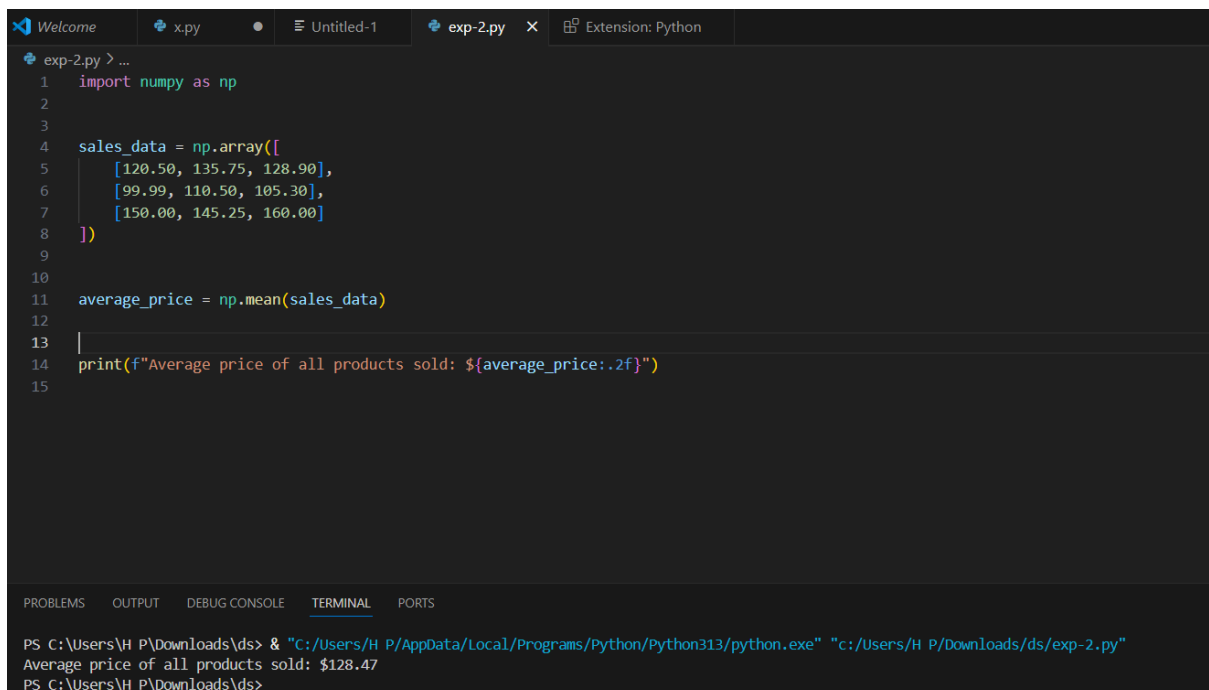
History: 72.56

Subject with the highest average score: Science (74.62)

PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/H P/Downloads/ds/x.py"

Average scores for each subject:
Math: 84.50
Science: 84.75
English: 84.00
History: 87.00

EXP-2



```
1 import numpy as np
2
3
4 sales_data = np.array([
5     [120.50, 135.75, 128.90],
6     [99.99, 110.50, 105.30],
7     [150.00, 145.25, 160.00]
8 ])
9
10
11 average_price = np.mean(sales_data)
12
13
14 print(f"Average price of all products sold: ${average_price:.2f}")
15
```

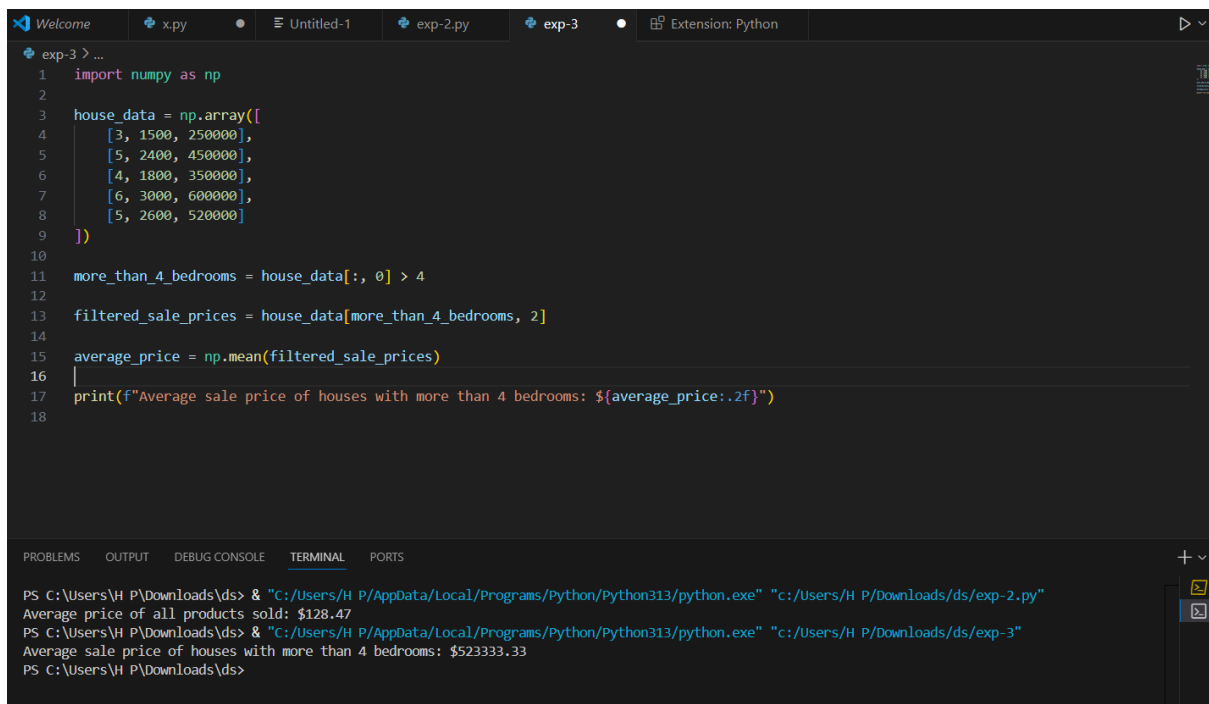
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/H P/Downloads/ds/exp-2.py"

Average price of all products sold: \$128.47

PS C:\Users\H P\Downloads\ds>

EXP-3



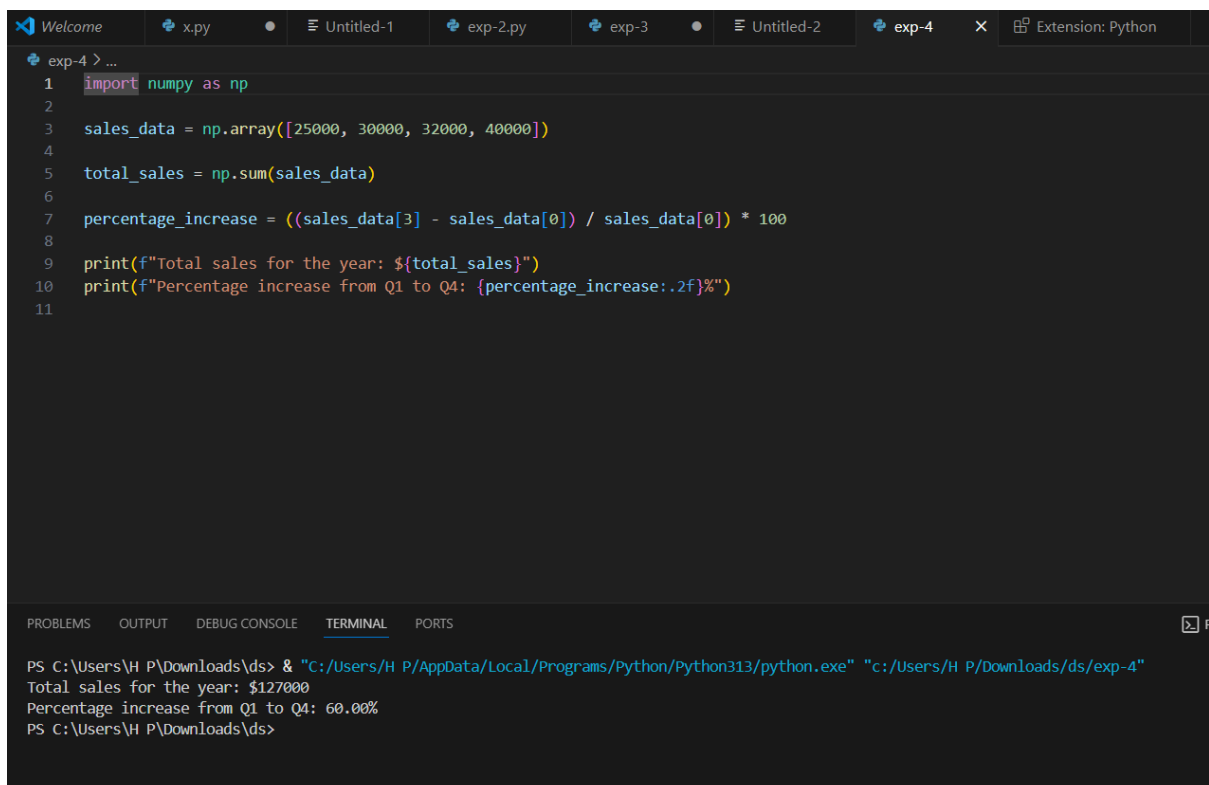
The screenshot shows the Visual Studio Code editor with a Python file named `exp-3.py`. The script uses NumPy to filter house data based on the number of bedrooms and calculate the average sale price. The terminal output shows the execution of the script, displaying the average price of all products and the average price of houses with more than 4 bedrooms.

```
exp-3 > ...
1 import numpy as np
2
3 house_data = np.array([
4     [3, 1500, 250000],
5     [5, 2400, 450000],
6     [4, 1800, 350000],
7     [6, 3000, 600000],
8     [5, 2600, 520000]
9 ])
10
11 more_than_4_bedrooms = house_data[:, 0] > 4
12
13 filtered_sale_prices = house_data[more_than_4_bedrooms, 2]
14
15 average_price = np.mean(filtered_sale_prices)
16 |
17 print(f"Average sale price of houses with more than 4 bedrooms: ${average_price:.2f}")
18
```

Terminal Output:

```
PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "C:/Users/H P/Downloads/ds/exp-2.py"
Average price of all products sold: $128.47
PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "C:/Users/H P/Downloads/ds/exp-3"
Average sale price of houses with more than 4 bedrooms: $523333.33
PS C:\Users\H P\Downloads\ds>
```

EXP-4



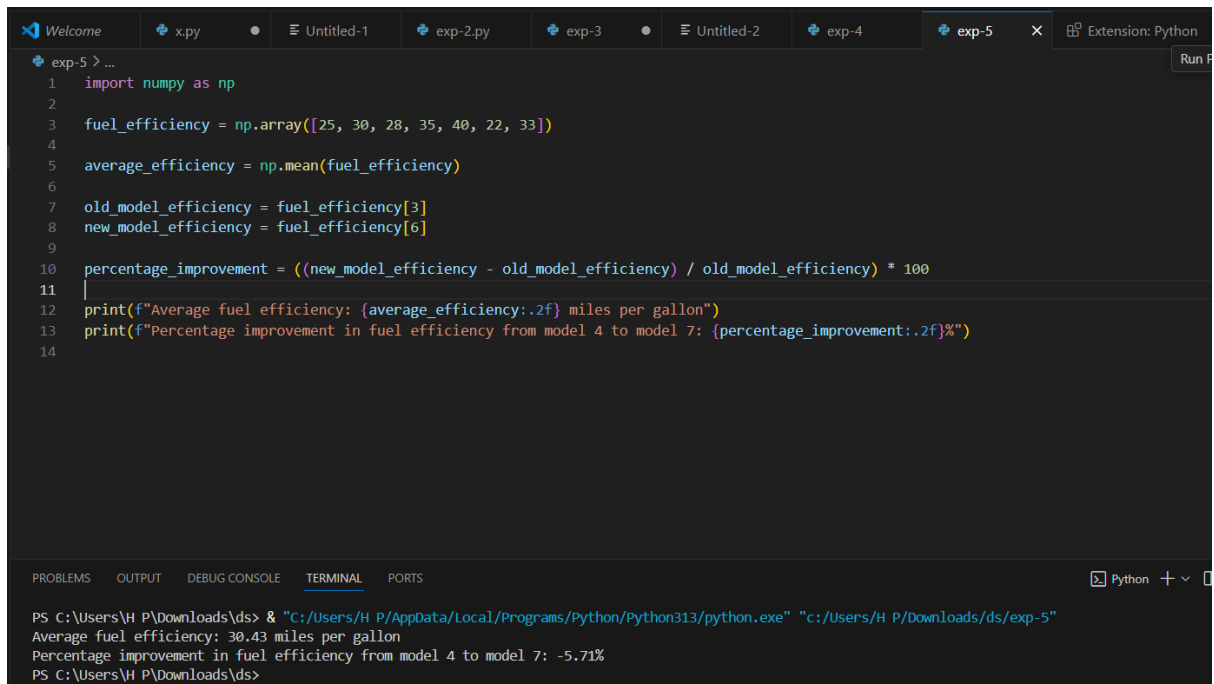
The screenshot shows the Visual Studio Code editor with a Python file named `exp-4.py`. The script uses NumPy to calculate the total sales and the percentage increase from Q1 to Q4. The terminal output shows the execution of the script, displaying the total sales and the percentage increase.

```
exp-4 > ...
1 import numpy as np
2
3 sales_data = np.array([25000, 30000, 32000, 40000])
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(f"Total sales for the year: ${total_sales}")
10 print(f"Percentage increase from Q1 to Q4: {percentage_increase:.2f}%")
11
```

Terminal Output:

```
PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "C:/Users/H P/Downloads/ds/exp-4"
Total sales for the year: $127000
Percentage increase from Q1 to Q4: 60.00%
PS C:\Users\H P\Downloads\ds>
```

EXP-5



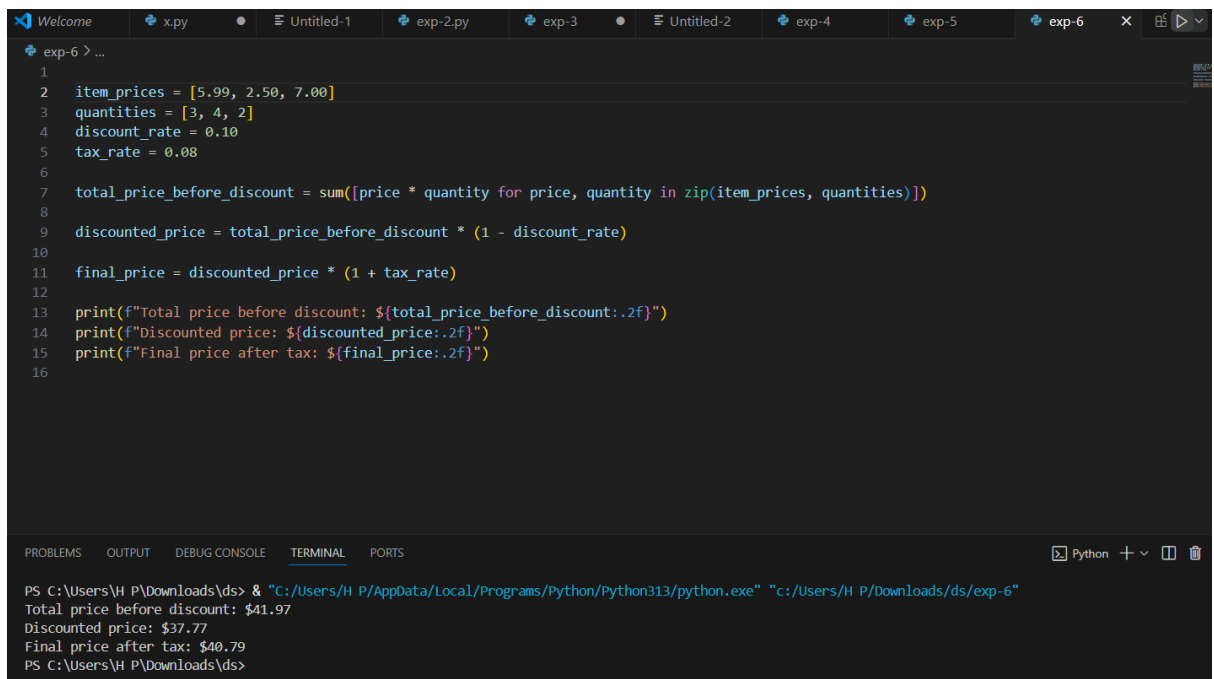
The screenshot shows a VS Code editor with a Python file named `exp-5.py`. The script calculates the average fuel efficiency from a list of values and then compares two specific models to find the percentage improvement. The terminal output shows the results of running the script.

```
exp-5 > ...
1  import numpy as np
2
3  fuel_efficiency = np.array([25, 30, 28, 35, 40, 22, 33])
4
5  average_efficiency = np.mean(fuel_efficiency)
6
7  old_model_efficiency = fuel_efficiency[3]
8  new_model_efficiency = fuel_efficiency[6]
9
10 percentage_improvement = ((new_model_efficiency - old_model_efficiency) / old_model_efficiency) * 100
11 |
12 print(f"Average fuel efficiency: {average_efficiency:.2f} miles per gallon")
13 print(f"Percentage improvement in fuel efficiency from model 4 to model 7: {percentage_improvement:.2f}%")
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + -

```
PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/H P/Downloads/ds/exp-5"
Average fuel efficiency: 30.43 miles per gallon
Percentage improvement in fuel efficiency from model 4 to model 7: -5.71%
PS C:\Users\H P\Downloads\ds>
```

EXP-6



The screenshot shows a VS Code editor with a Python file named `exp-6.py`. The script calculates the total price of items after applying a discount and then adding tax. The terminal output shows the results of running the script.

```
exp-6 > ...
1
2  item_prices = [5.99, 2.50, 7.00]
3  quantities = [3, 4, 2]
4  discount_rate = 0.10
5  tax_rate = 0.08
6
7  total_price_before_discount = sum([price * quantity for price, quantity in zip(item_prices, quantities)])
8
9  discounted_price = total_price_before_discount * (1 - discount_rate)
10
11 final_price = discounted_price * (1 + tax_rate)
12
13 print(f"Total price before discount: ${total_price_before_discount:.2f}")
14 print(f"Discounted price: ${discounted_price:.2f}")
15 print(f"Final price after tax: ${final_price:.2f}")
16
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + -

```
PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/H P/Downloads/ds/exp-6"
Total price before discount: $41.97
Discounted price: $37.77
Final price after tax: $40.79
PS C:\Users\H P\Downloads\ds>
```

EXP-7

```
exp-7 > ...
1 import pandas as pd
2
3 data = {
4     'customer_id': [101, 102, 103, 101, 102, 104],
5     'order_date': ['2025-04-01', '2025-04-02', '2025-04-03', '2025-04-04', '2025-04-05', '2025-04-06'],
6     'product_name': ['Laptop', 'Smartphone', 'Tablet', 'Laptop', 'Smartphone', 'Tablet'],
7     'order_quantity': [1, 2, 1, 2, 1, 3]
8 }
9 order_data = pd.DataFrame(data)
10
11 order_data['order_date'] = pd.to_datetime(order_data['order_date'])
12
13 total_orders_by_customer = order_data.groupby('customer_id').size()
14
15 average_order_quantity_per_product = order_data.groupby('product_name')['order_quantity'].mean()
16
17 earliest_order_date = order_data['order_date'].min()
18 latest_order_date = order_data['order_date'].max()
19
20 print("Total number of orders by each customer:")
21 print(total_orders_by_customer)
22
23 print("\nAverage order quantity for each product:")
24 print(average_order_quantity_per_product)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
dtype: int64

Average order quantity for each product:
product_name
Laptop      1.5
Smartphone  1.5
Tablet      2.0
Name: order_quantity, dtype: float64
```

EXP-8

```
exp-8 > ...
1 import pandas as pd
2
3 data = {
4     'product_name': ['Laptop', 'Smartphone', 'Tablet', 'Laptop', 'Smartphone', 'Smartwatch', 'Tablet', 'Smartphone', 'Laptop', 'Smartphone'],
5     'quantity_sold': [10, 25, 15, 7, 20, 5, 10, 30, 12, 8]
6 }
7 sales_data = pd.DataFrame(data)
8
9 total_sales_by_product = sales_data.groupby('product_name')['quantity_sold'].sum()
10
11 sorted_sales = total_sales_by_product.sort_values(ascending=False)
12
13 top_5_products = sorted_sales.head(5)
14
15 print("Top 5 products sold the most in the past month:")
16 print(top_5_products)
17
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
PS C:\Users\H P\Downloads\ds> & "C:/Users/H P/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/H P/Downloads/ds/exp-8"
Top 5 products sold the most in the past month:
product_name
Smartphone    75
Laptop        29
Tablet        25
Smartwatch    13
Name: quantity_sold, dtype: int64
PS C:\Users\H P\Downloads\ds>
```

EXP-9

```
exp-2.py  exp-3  Untitled-2  exp-4  exp-5  exp-6  exp-7  exp-8
exp-9 > ...
1  import pandas as pd
2
3  data = {
4      'property_id': [1, 2, 3, 4, 5],
5      'location': ['New York', 'Los Angeles', 'New York', 'San Francisco', 'Los Angeles'],
6      'bedrooms': [3, 5, 4, 6, 2],
7      'area_sqft': [1500, 2500, 1800, 3000, 1200],
8      'listing_price': [600000, 800000, 750000, 1200000, 500000]
9  }
10 property_data = pd.DataFrame(data)
11
12 average_price_by_location = property_data.groupby('location')['listing_price'].mean()
13
14 properties_with_more_than_4_bedrooms = (property_data['bedrooms'] > 4).sum()
15
16 property_with_largest_area = property_data.loc[property_data['area_sqft'].idxmax()]
17
18 print("Average listing price of properties in each location:")
19 print(average_price_by_location)
20
21 print(f"\nNumber of properties with more than four bedrooms: {properties_with_more_than_4_bedrooms}")
22
23 print("\nProperty with the largest area:")
24 print(property_data.loc[property_data['area_sqft'].idxmax()])
Name: listing_price, dtype: float64

Number of properties with more than four bedrooms: 2

Property with the largest area:
property_id      4
location      San Francisco
bedrooms         6
area_sqft       3000
listing_price   1200000
Name: 3, dtype: object
```

EXP-10

```
exp-3  Untitled-2  exp-4  exp-5  exp-6  exp-7  exp-8  exp-9  exp-10  X  55
exp-10
1  import matplotlib.pyplot as plt
2  print ( "" )
3  months = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']
4  sales = [1000, 1500, 1200, 1600, 2000, 1800, 1700, 2200, 2400, 2100, 2300, 2500]
5
6  plt.figure(figsize=(10, 5))
7  plt.plot(months, sales, marker='o', color='b', linestyle='--', linewidth=2, markersize=8)
8  plt.title('Monthly Sales Data (Line Plot)', fontsize=14)
9  plt.xlabel('Month', fontsize=12)
10 plt.ylabel('Sales', fontsize=12)
11 plt.xticks(rotation=45)
12 plt.grid(True)
13 plt.tight_layout()
14 plt.show()
15
16 plt.figure(figsize=(10, 5))
17 plt.bar(months, sales, color='skyblue')
18 plt.title('Monthly Sales Data (Bar Plot)', fontsize=14)
19 plt.xlabel('Month', fontsize=12)
20 plt.ylabel('Sales', fontsize=12)
21 plt.xticks(rotation=45)
22 plt.tight_layout()
23 plt.show()
24

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
```