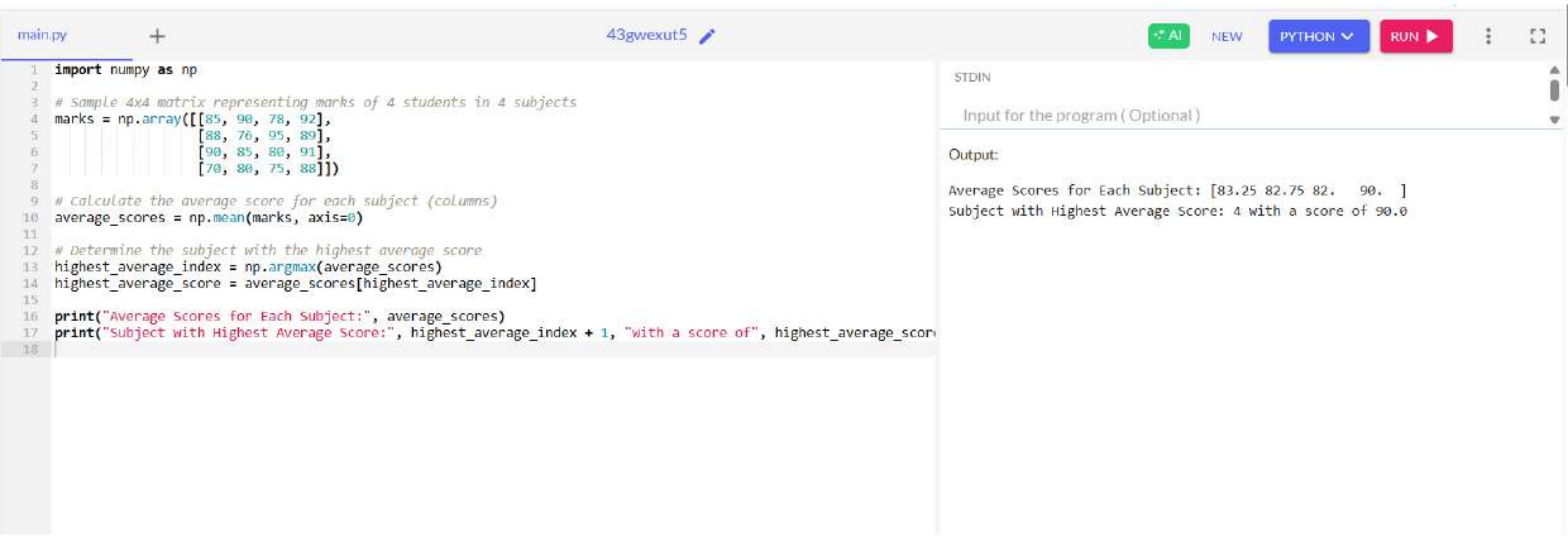
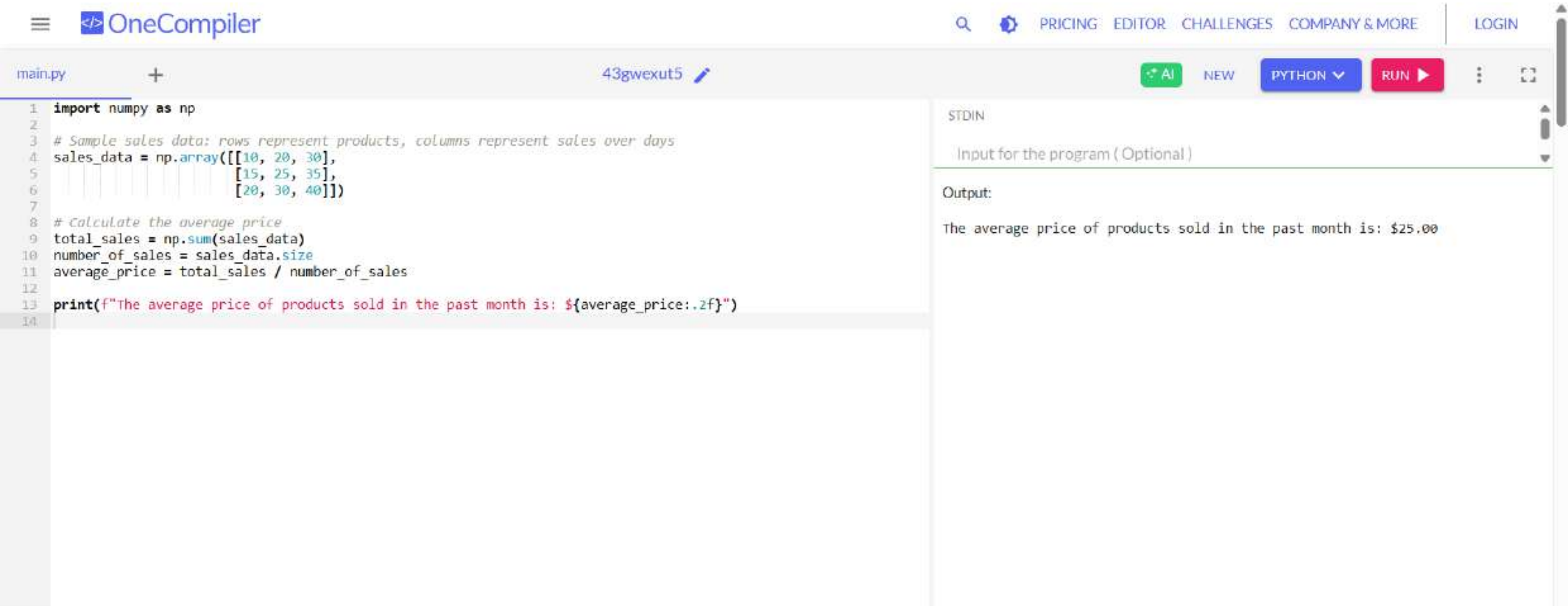


```python
import numpy as np

# Sample 4x4 matrix representing marks of 4 students in 4 subjects
marks = np.array([[85, 90, 78, 92],
                  [88, 76, 95, 89],
                  [90, 85, 80, 91],
                  [70, 80, 75, 88]])

# Calculate the average score for each subject (columns)
average_scores = np.mean(marks, axis=0)

# Determine the subject with the highest average score
highest_average_index = np.argmax(average_scores)
highest_average_score = average_scores[highest_average_index]

print("Average Scores for Each Subject:", average_scores)
print("Subject with Highest Average Score:", highest_average_index + 1, "with a score of", highest_average_scor
```

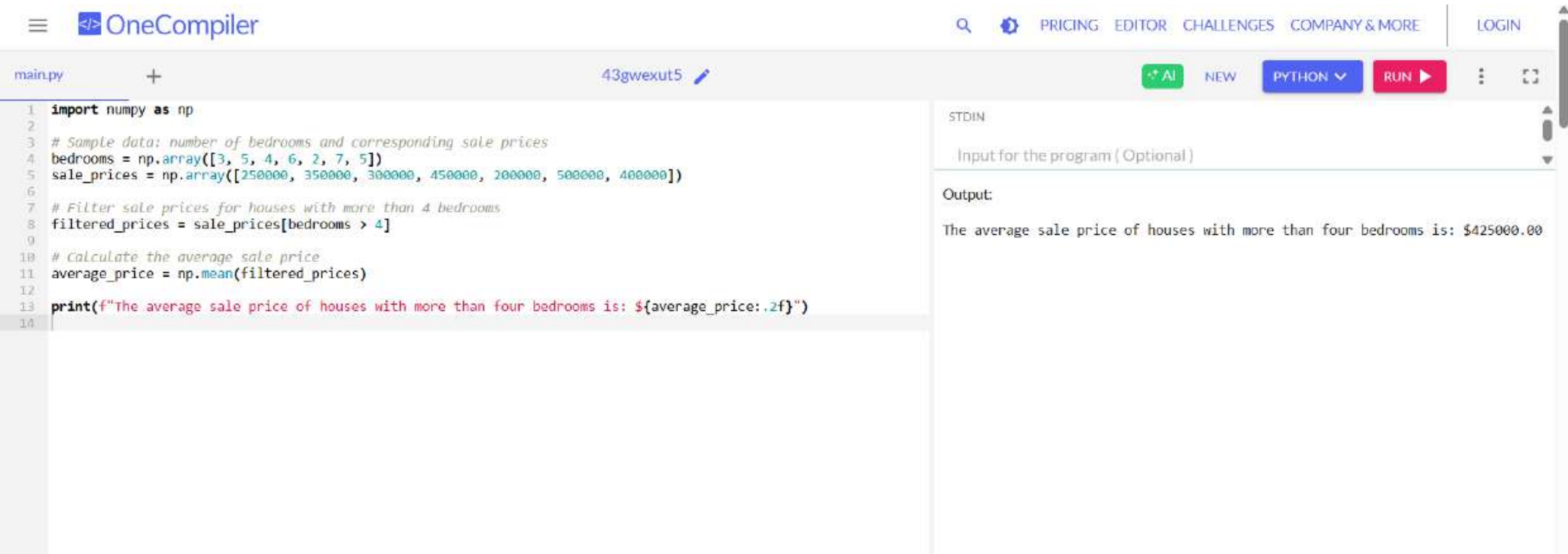
```
Average Scores for Each Subject: [83.25 82.75 82.   90.  ]
Subject with Highest Average Score: 4 with a score of 90.0
```



```python
import numpy as np

# Sample sales data: rows represent products, columns represent sales over days
sales_data = np.array([[10, 20, 30],
                       [15, 25, 35],
                       [20, 30, 40]])

# Calculate the average price
total_sales = np.sum(sales_data)
number_of_sales = sales_data.size
average_price = total_sales / number_of_sales

print(f"The average price of products sold in the past month is: ${average_price:.2f}")
```

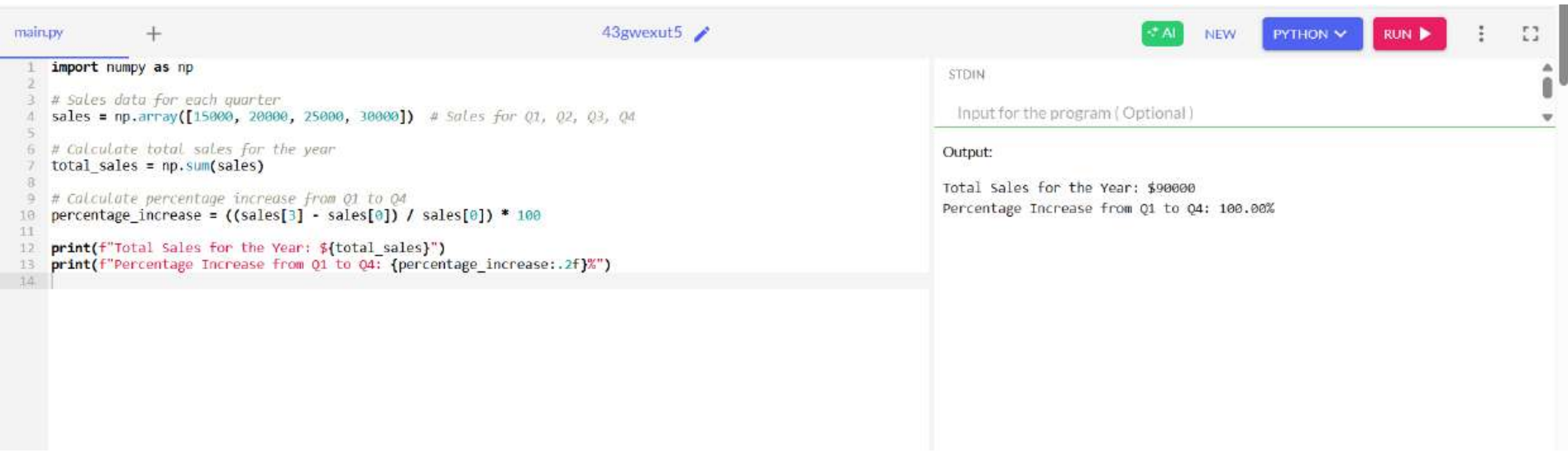
Output:

```
The average price of products sold in the past month is: $25.00
```



```python
import numpy as np

# Sample data: number of bedrooms and corresponding sale prices
bedrooms = np.array([3, 5, 4, 6, 2, 7, 5])
sale_prices = np.array([250000, 350000, 300000, 450000, 200000, 500000, 400000])

# Filter sale prices for houses with more than 4 bedrooms
filtered_prices = sale_prices[bedrooms > 4]

# Calculate the average sale price
average_price = np.mean(filtered_prices)

print(f"The average sale price of houses with more than four bedrooms is: ${average_price:.2f}")

```

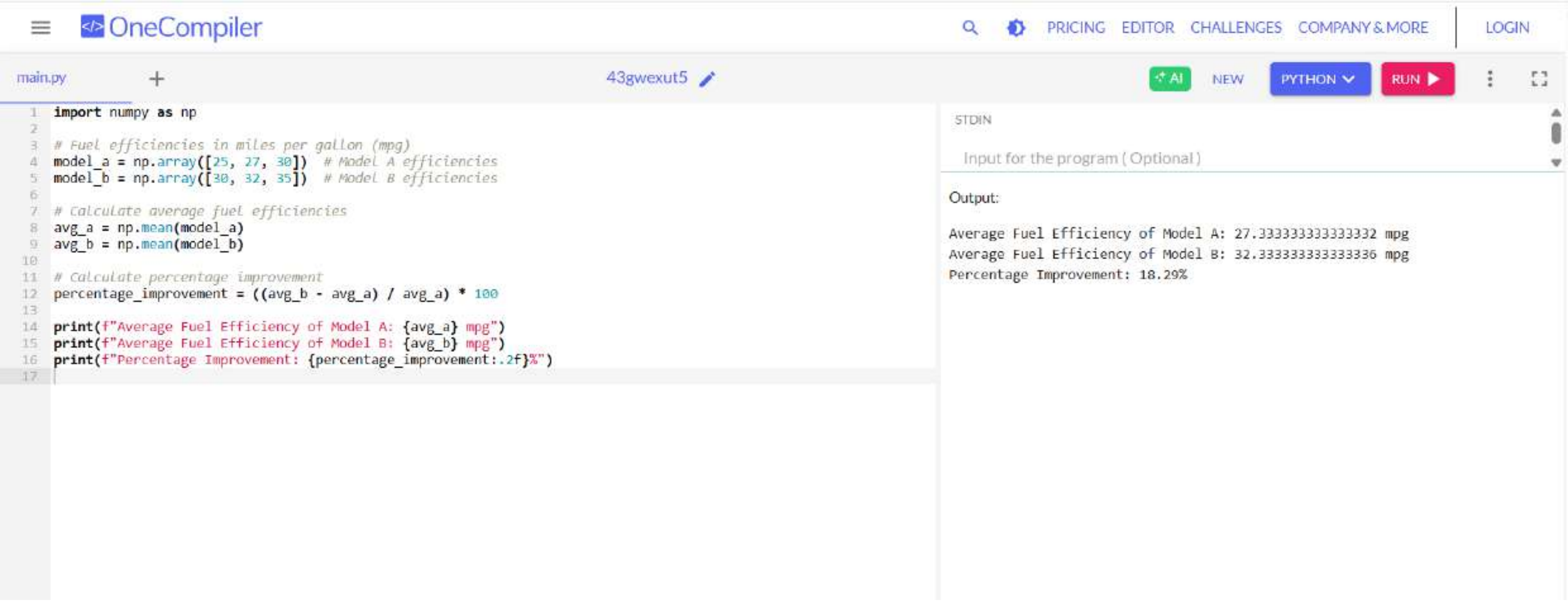
```
The average sale price of houses with more than four bedrooms is: $425000.00
```



```python
import numpy as np

# Sales data for each quarter
sales = np.array([15000, 20000, 25000, 30000])  # Sales for Q1, Q2, Q3, Q4

# Calculate total sales for the year
total_sales = np.sum(sales)

# Calculate percentage increase from Q1 to Q4
percentage_increase = ((sales[3] - sales[0]) / sales[0]) * 100

print(f"Total Sales for the Year: ${total_sales}")
print(f"Percentage Increase from Q1 to Q4: {percentage_increase:.2f}%")
```

Output:

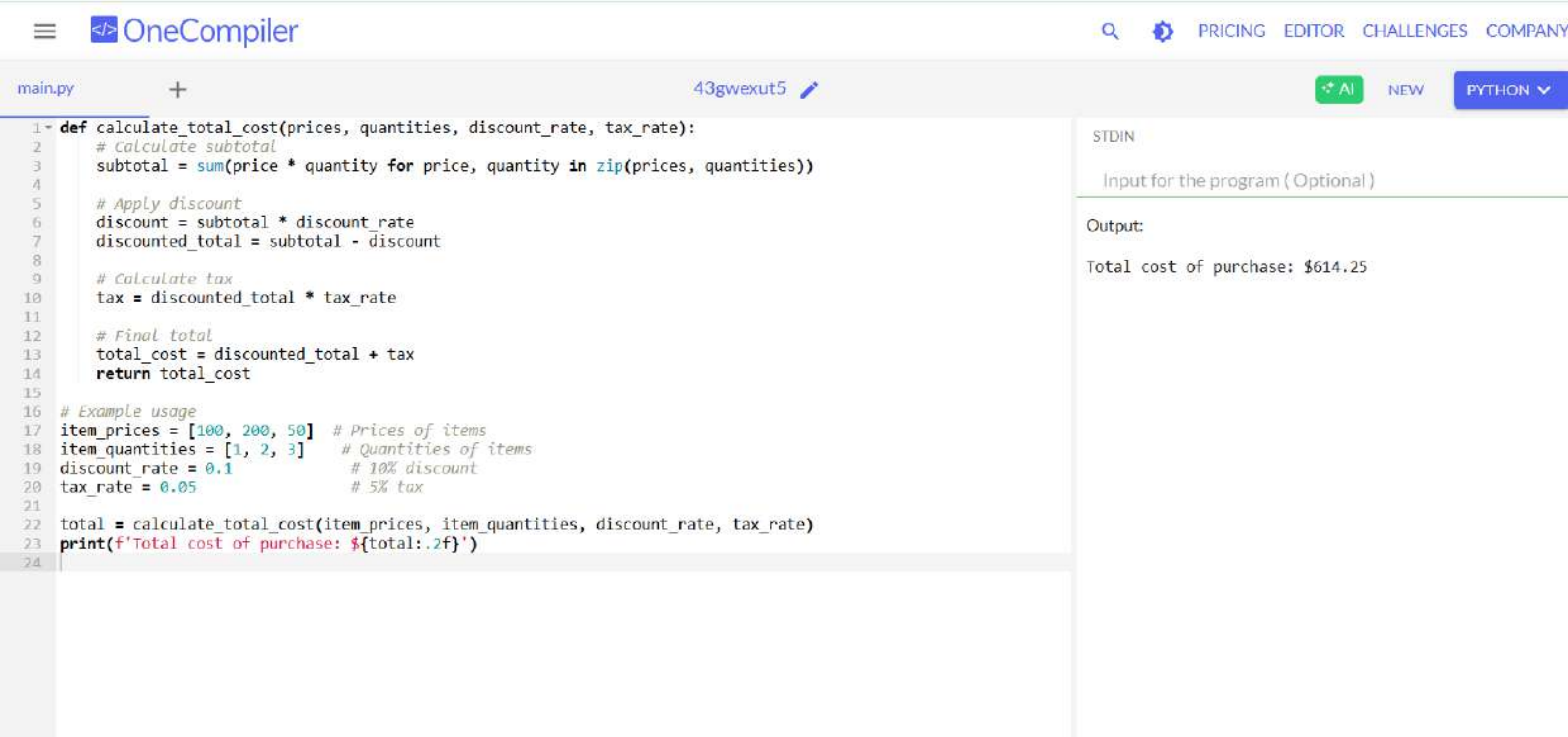
```
Total Sales for the Year: $90000
Percentage Increase from Q1 to Q4: 100.00%
```



```python
import numpy as np

# Fuel efficiencies in miles per gallon (mpg)
model_a = np.array([25, 27, 30])  # Model A efficiencies
model_b = np.array([30, 32, 35])  # Model B efficiencies

# Calculate average fuel efficiencies
avg_a = np.mean(model_a)
avg_b = np.mean(model_b)

# Calculate percentage improvement
percentage_improvement = ((avg_b - avg_a) / avg_a) * 100

print(f"Average Fuel Efficiency of Model A: {avg_a} mpg")
print(f"Average Fuel Efficiency of Model B: {avg_b} mpg")
print(f"Percentage Improvement: {percentage_improvement:.2f}%")
```

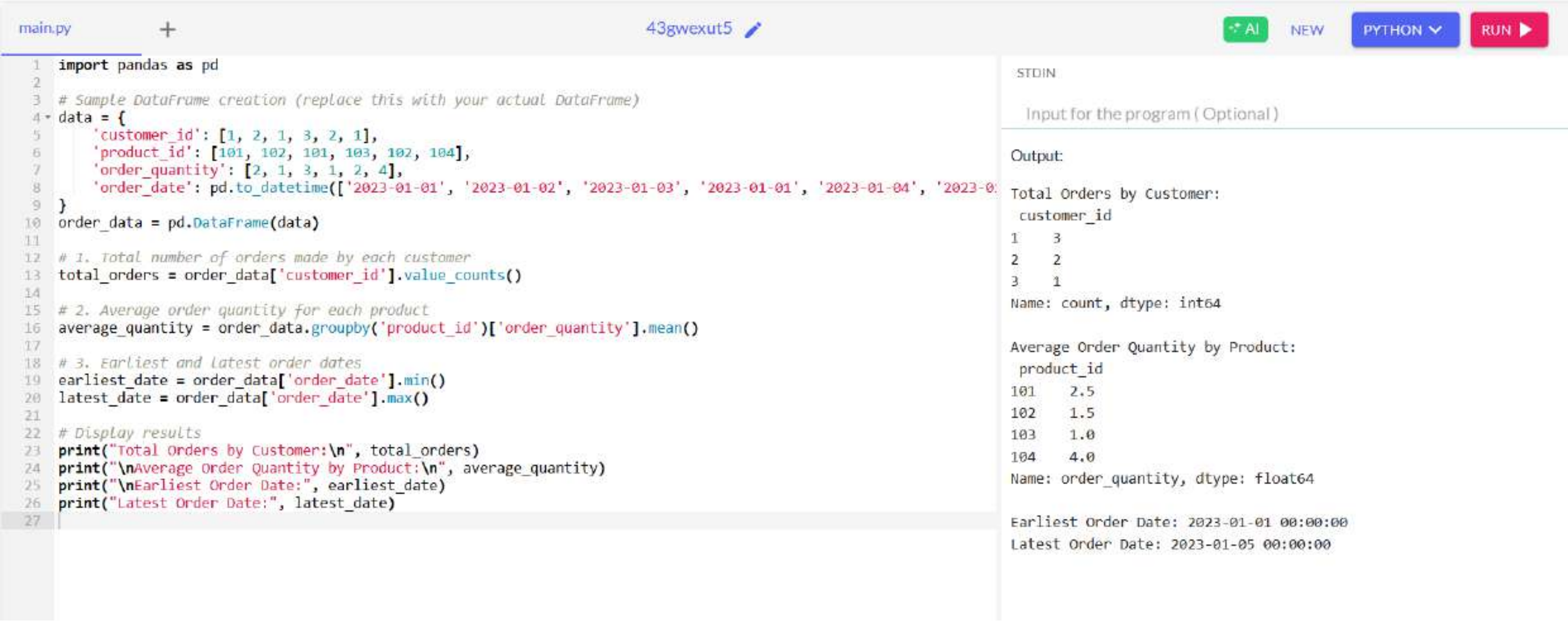
Output:

```
Average Fuel Efficiency of Model A: 27.333333333333332 mpg
Average Fuel Efficiency of Model B: 32.333333333333336 mpg
Percentage Improvement: 18.29%
```



```python
def calculate_total_cost(prices, quantities, discount_rate, tax_rate):
    # Calculate subtotal
    subtotal = sum(price * quantity for price, quantity in zip(prices, quantities))

    # Apply discount
    discount = subtotal * discount_rate
    discounted_total = subtotal - discount

    # Calculate tax
    tax = discounted_total * tax_rate

    # Final total
    total_cost = discounted_total + tax
    return total_cost

# Example usage
item_prices = [100, 200, 50]  # Prices of items
item_quantities = [1, 2, 3]    # Quantities of items
discount_rate = 0.1           # 10% discount
tax_rate = 0.05               # 5% tax

total = calculate_total_cost(item_prices, item_quantities, discount_rate, tax_rate)
print(f'Total cost of purchase: ${total:.2f}')
```

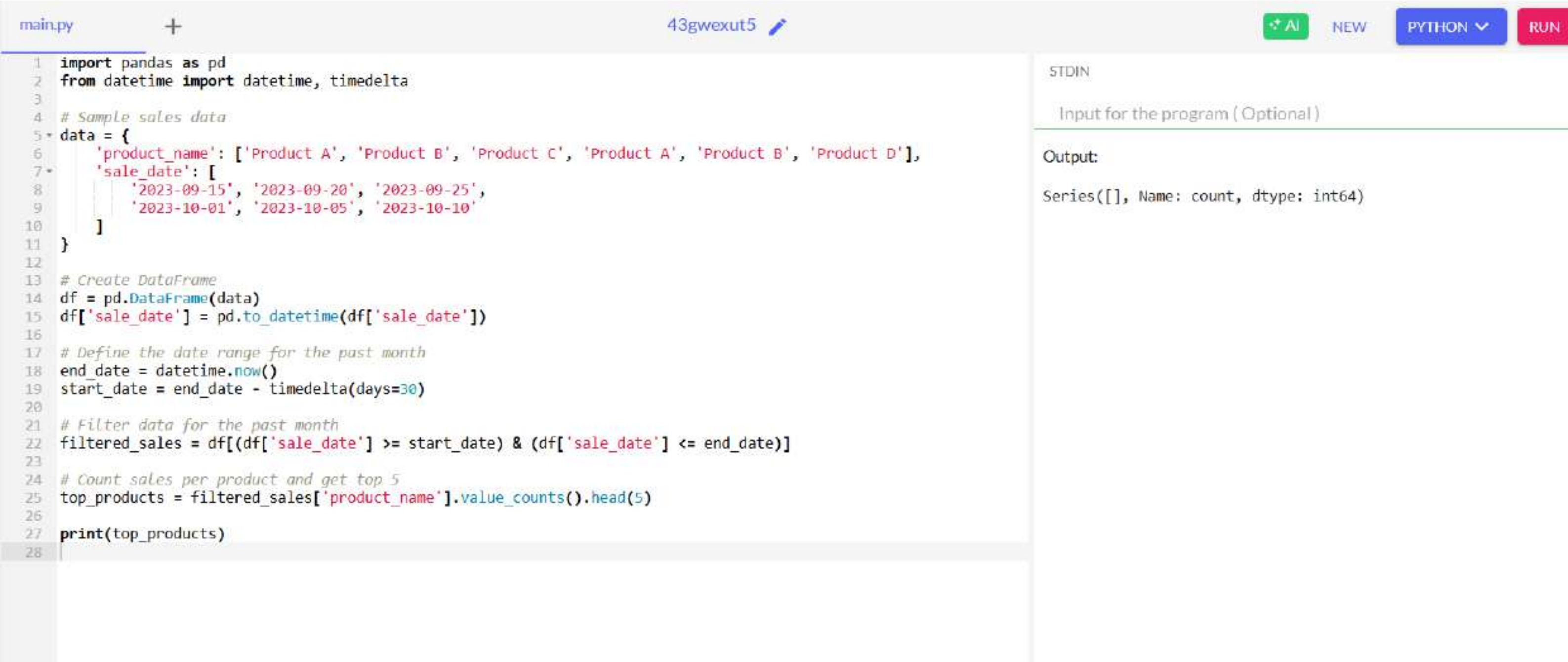
```
Total cost of purchase: $614.25
```



```python
import pandas as pd

# Sample DataFrame creation (replace this with your actual DataFrame)
data = {
    'customer_id': [1, 2, 1, 3, 2, 1],
    'product_id': [101, 102, 101, 103, 102, 104],
    'order_quantity': [2, 1, 3, 1, 2, 4],
    'order_date': pd.to_datetime(['2023-01-01', '2023-01-02', '2023-01-03', '2023-01-01', '2023-01-04', '2023-0
}
order_data = pd.DataFrame(data)

# 1. Total number of orders made by each customer
total_orders = order_data['customer_id'].value_counts()

# 2. Average order quantity for each product
average_quantity = order_data.groupby('product_id')['order_quantity'].mean()

# 3. Earliest and latest order dates
earliest_date = order_data['order_date'].min()
latest_date = order_data['order_date'].max()

# Display results
print("Total Orders by Customer:\n", total_orders)
print("\nAverage Order Quantity by Product:\n", average_quantity)
print("\nEarliest Order Date:", earliest_date)
print("Latest Order Date:", latest_date)
```

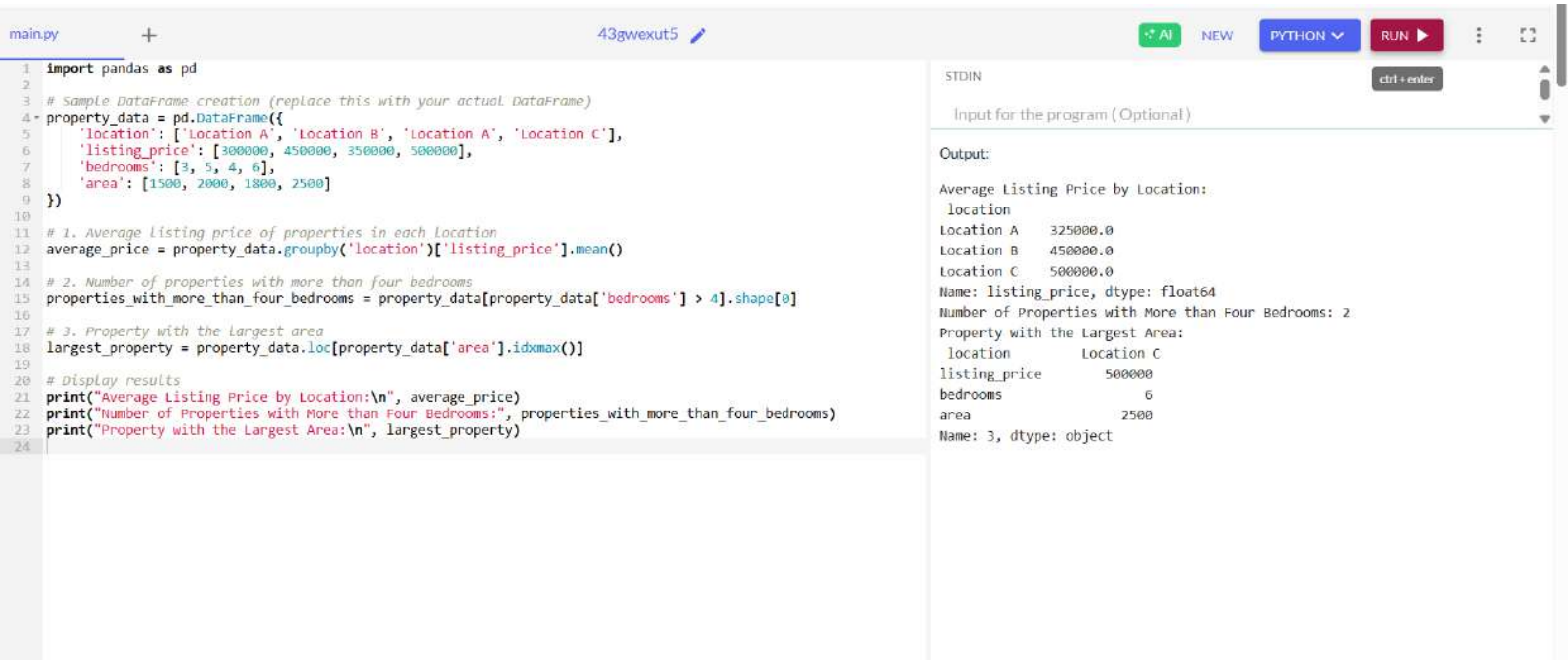
Output:

```
Total Orders by Customer:
 customer_id
1    3
2    2
3    1
Name: count, dtype: int64

Average Order Quantity by Product:
 product_id
101    2.5
102    1.5
103    1.0
104    4.0
Name: order_quantity, dtype: float64

Earliest Order Date: 2023-01-01 00:00:00
Latest Order Date: 2023-01-05 00:00:00
```



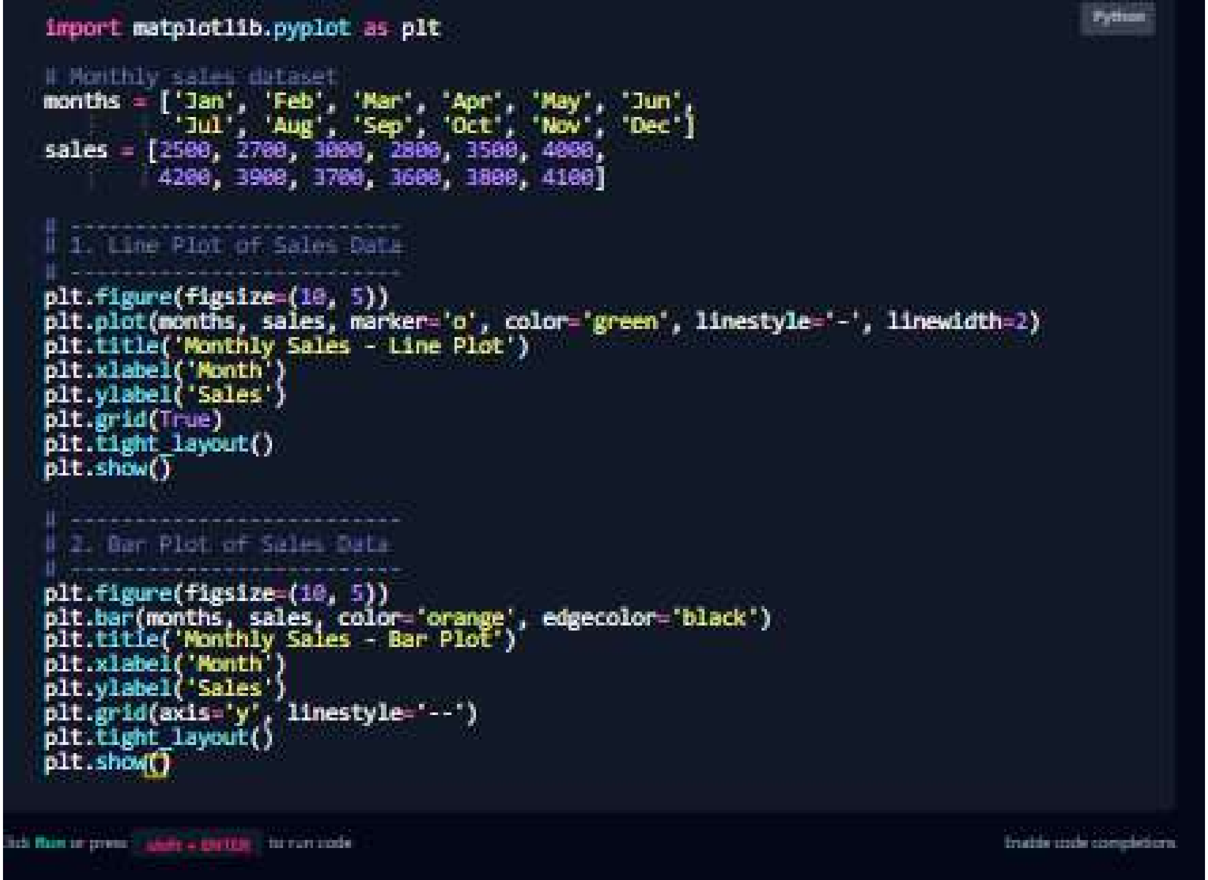
```python
import pandas as pd
from datetime import datetime, timedelta

# Sample sales data
data = {
    'product_name': ['Product A', 'Product B', 'Product C', 'Product A', 'Product B', 'Product D'],
    'sale_date': [
        '2023-09-15', '2023-09-20', '2023-09-25',
        '2023-10-01', '2023-10-05', '2023-10-10'
    ]
}

# Create DataFrame
df = pd.DataFrame(data)
df['sale_date'] = pd.to_datetime(df['sale_date'])

# Define the date range for the past month
end_date = datetime.now()
start_date = end_date - timedelta(days=30)

# Filter data for the past month
filtered_sales = df[(df['sale_date'] >= start_date) & (df['sale_date'] <= end_date)]

# Count sales per product and get top 5
top_products = filtered_sales['product_name'].value_counts().head(5)

print(top_products)
```

```
Series([], Name: count, dtype: int64)
```



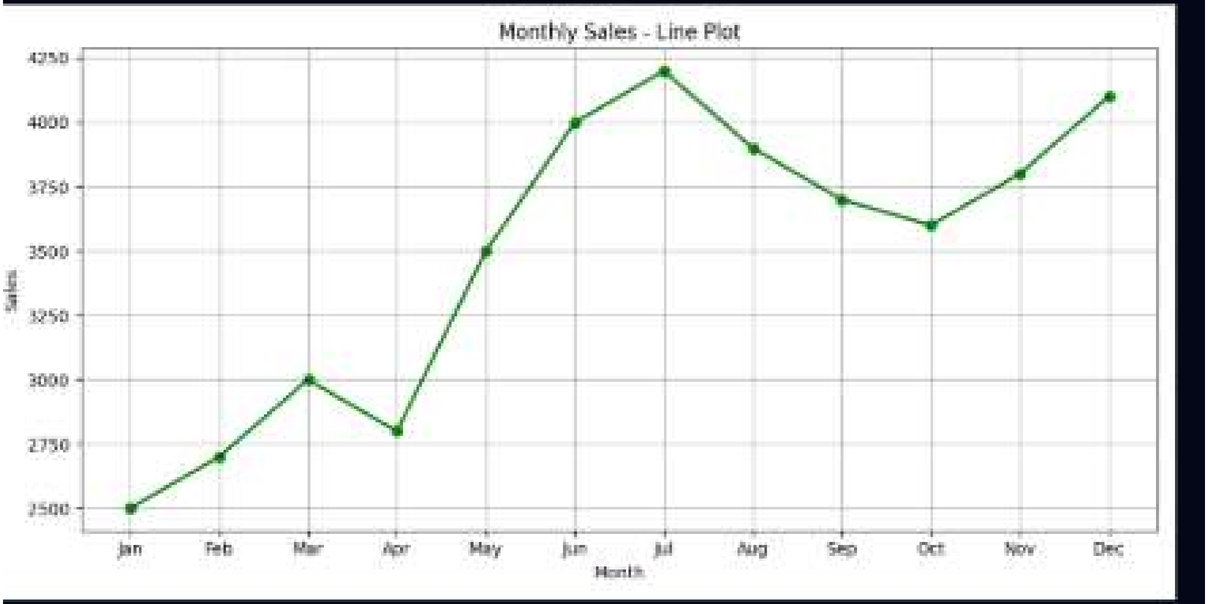
```python
import pandas as pd

# Sample DataFrame creation (replace this with your actual DataFrame)
property_data = pd.DataFrame({
    'location': ['Location A', 'Location B', 'Location A', 'Location C'],
    'listing_price': [300000, 450000, 350000, 500000],
    'bedrooms': [3, 5, 4, 6],
    'area': [1500, 2000, 1800, 2500]
})

# 1. Average listing price of properties in each location
average_price = property_data.groupby('location')['listing_price'].mean()

# 2. Number of properties with more than four bedrooms
properties_with_more_than_four_bedrooms = property_data[property_data['bedrooms'] > 4].shape[0]

# 3. Property with the largest area
largest_property = property_data.loc[property_data['area'].idxmax()]

# Display results
print("Average Listing Price by Location:\n", average_price)
print("Number of Properties with More than Four Bedrooms:", properties_with_more_than_four_bedrooms)
print("Property with the Largest Area:\n", largest_property)
```

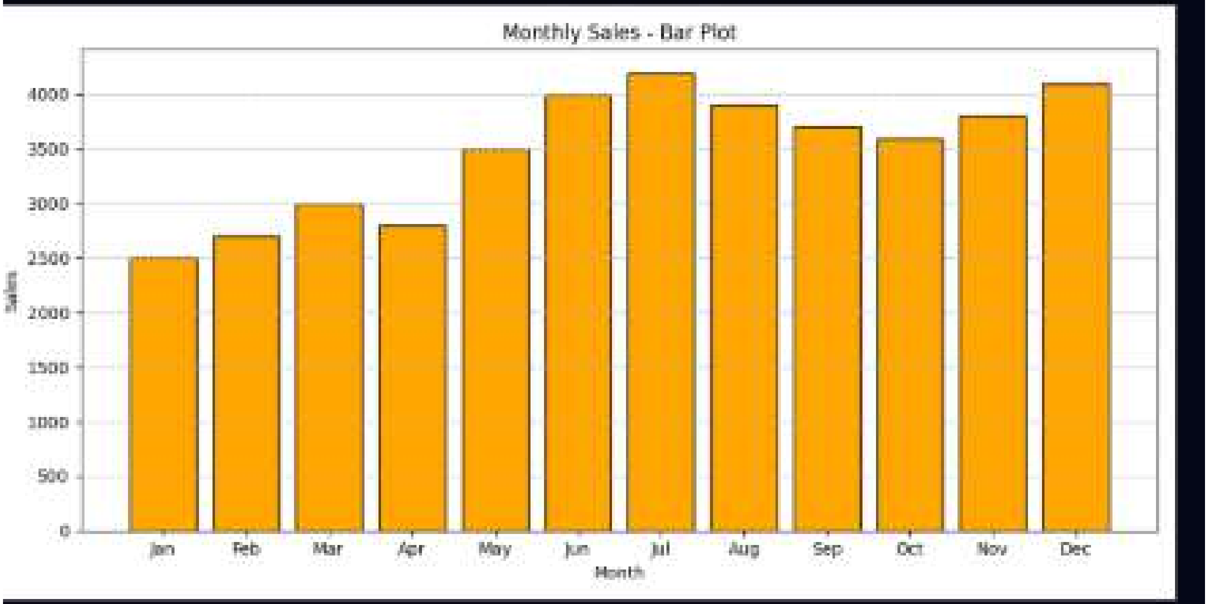
```
Average Listing Price by Location:
 location
Location A    325000.0
Location B    450000.0
Location C    500000.0
Name: listing_price, dtype: float64
Number of Properties with More than Four Bedrooms: 2
Property with the Largest Area:
 location         Location C
listing_price        500000
bedrooms                  6
area                   2500
Name: 3, dtype: object
```



```python
import matplotlib.pyplot as plt

# Monthly sales dataset
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
          'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
sales = [2500, 2700, 3000, 2800, 3500, 4000,
         4200, 3900, 3700, 3600, 3800, 4100]

# ---------------------------
# 1. Line Plot of Sales Data
# ---------------------------
plt.figure(figsize=(10, 5))
plt.plot(months, sales, marker='o', color='green', 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 Sales Data
# ---------------------------
plt.figure(figsize=(10, 5))
plt.bar(months, sales, color='orange', edgecolor='black')
plt.title('Monthly Sales - Bar Plot')
plt.xlabel('Month')
plt.ylabel('Sales')
plt.grid(axis='y', linestyle='--')
plt.tight_layout()
plt.show()
```

lick Run or press shift + ENTER to run code
Enable code completions


Monthly Sales - Line Plot
Sales
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month


Monthly Sales - Bar Plot
Sales
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month