```
import numpy as np

# Sample data: response times in milliseconds
response_times = [120, 150, 200, 300, 250, 100, 170, 180, 220, 90]

# Calculate the 25th, 50th, and 75th percentiles
percentiles = np.percentile(response_times, [25, 50, 75])

# Display the results
print(f"25th percentile: {percentiles[0]} ms")
print(f"50th percentile (median): {percentiles[1]} ms")
print(f"75th percentile: {percentiles[2]} ms")
```

```
25th percentile: 127.5 ms
50th percentile (median): 175.0 ms
75th percentile: 215.0 ms
```

```
import numpy as np

# Sample data: recovery times in days
recovery_times = [5, 7, 8, 6, 7, 9, 10, 6, 5, 8, 7, 6, 9, 10, 8, 6, 5, 9, 7, 8]

# Calculate the 10th, 50th, and 90th percentiles
percentiles = np.percentile(recovery_times, [10, 50, 90])

# Display the results
print(f"10th percentile: {percentiles[0]} days")
print(f"50th percentile (median): {percentiles[1]} days")
print(f"90th percentile: {percentiles[2]} days")
```

```
10th percentile: 5.0 days
50th percentile (median): 7.0 days
90th percentile: 9.100000000000001 days
```

```
import numpy as np
from scipy import stats

# Sample data: purchase amounts in dollars
purchase_amounts = [50, 20, 30, 20, 50, 70, 50, 80, 30, 20, 50, 30, 20, 50, 70, 50, 80, 20]

# Calculate the mean (average) purchase amount
mean_purchase_amount = np.mean(purchase_amounts)

# Identify the mode of the purchase amounts
mode_purchase_amount = stats.mode(purchase_amounts)

# Display the results
print(f"Mean (average) purchase amount: ${mean_purchase_amount:.2f}")
```

```
Mean (average) purchase amount: $43.89
```

```
import numpy as np

# Sample data: monthly expenses of different departments (in dollars)
# Each row represents a department, and each column represents a month
expenses = np.array([
    [2000, 2100, 2200, 2300, 2400],  # Department 1
    [1500, 1600, 1700, 1800, 1900],  # Department 2
    [3000, 3100, 3200, 3300, 3400]   # Department 3
])

# Calculate the variance of the monthly expenses for each department
variance = np.var(expenses, axis=1)

# Calculate the covariance matrix of the monthly expenses
covariance_matrix = np.cov(expenses)

# Display the results
print("Variance of monthly expenses for each department:")
print(variance)

print("\nCovariance matrix of monthly expenses:")
print(covariance_matrix)
```

```
Variance of monthly expenses for each department:
[20000. 20000. 20000.]

Covariance matrix of monthly expenses:
[[25000. 25000. 25000.]
 [25000. 25000. 25000.]
 [25000. 25000. 25000.]]
```

```
import numpy as np
from scipy import stats

# Sample data: daily temperatures in degrees Celsius
temperatures = np.array([22, 24, 21, 23, 25, 30, 19, 31, 28, 20, 24, 22, 25, 30, 29, 27, 32, 26, 23

# Calculate the variance of the temperatures
variance = np.var(temperatures)

# Calculate the Z-scores for identifying outliers
z_scores = np.abs(stats.zscore(temperatures))

# Define a threshold for identifying outliers (e.g., Z-score > 2)
threshold = 2
outliers = np.where(z_scores > threshold)

# Display the results
print(f"Variance of temperatures: {variance:.2f}")

print("\nPotential outliers (indices and values):")
for index in outliers[0]:
    print(f"Index: {index}, Temperature: {temperatures[index]}")
```

```
Variance of temperatures: 13.49

Potential outliers (indices and values):
```

```
import numpy as np

# Sample data: daily temperature readings for each city over a year
# Each row represents a city, and each column represents a day
# For example: temperatures[0] is the data for City 1, temperatures[1] is the data for City 2, and sc
temperatures = np.array([
    [15, 16, 17, 16, 15, 14, 13, 14, 15, 16, 17, 18] * 30,  # City 1
    [10, 12, 13, 15, 16, 17, 18, 17, 16, 15, 14, 13] * 30,  # City 2
    [20, 22, 24, 26, 27, 28, 30, 29, 28, 27, 25, 23] * 30   # City 3
])

# Calculate the mean temperature for each city
mean_temperatures = np.mean(temperatures, axis=1)

# Calculate the standard deviation of temperature for each city
std_devs = np.std(temperatures, axis=1)

# Determine the city with the highest temperature range
temp_ranges = np.ptp(temperatures, axis=1)  # ptp is "peak-to-peak" (max - min)
city_highest_range = np.argmax(temp_ranges)

# Find the city with the most consistent temperature (lowest standard deviation)
city_most_consistent = np.argmin(std_devs)

# Display the results
print("Mean temperatures for each city:")
for i, mean_temp in enumerate(mean_temperatures, start=1):
    print(f"City {i}: {mean_temp:.2f}°C")

print("\nStandard deviation of temperatures for each city:")
for i, std_dev in enumerate(std_devs, start=1):
    print(f"City {i}: {std_dev:.2f}°C")

print(f"\nCity with the highest temperature range: City {city_highest_range + 1}")

print(f"City with the most consistent temperature (lowest standard deviation): City {city_most_consis
```

```
Mean temperatures for each city:
City 1: 15.50°C
City 2: 14.67°C
City 3: 25.75°C

Standard deviation of temperatures for each city:
City 1: 1.38°C
City 2: 2.25°C
City 3: 2.89°C

City with the highest temperature range: City 3
City with the most consistent temperature (lowest standard deviation): City 1
```

```
import numpy as np

# Sample data: daily sales figures over the past month
# Replace this list with your actual daily sales data
daily_sales = np.array([200, 220, 210, 230, 250, 240, 260, 250, 270, 280,
                        290, 300, 310, 320, 330, 340, 350, 360, 370, 380,
                        390, 400, 410, 420, 430, 440, 450, 460, 470, 480,
                        490])

# Calculate the variance of the daily sales
variance = np.var(daily_sales)

# Display the result
print(f"Variance of daily sales: {variance:.2f}")
```

```
Variance of daily sales: 7531.74
```