1. Create a NumPy array with numbers from 1 to 20

python Copy Edit

```python
import numpy as np

arr = np.arange(1, 21)  # Creates an array from 1 to 20
print(arr)
```

2. Create a 5x5 array of ones

python Copy Edit

```python
ones_array = np.ones((5, 5))  # Creates a 5x5 array filled with ones
print(ones_array)
```

3. Create an array of 10 random numbers between 0 and 1

python Copy Edit

```python
random_array = np.random.rand(10)  # Generates 10 random numbers between 0 and 1
print(random_array)
```

Let me know if you need any explanations! 😊

1. Create Two Random Arrays and Multiply Them

python

```python
import numpy as np

# Creating two random arrays of size 3x3
array1 = np.random.randint(1, 10, (3, 3))
array2 = np.random.randint(1, 10, (3, 3))

# Multiplying the arrays element-wise
result = array1 * array2

print("Array 1:\n", array1)
print("Array 2:\n", array2)
print("Multiplication Result:\n", result)
```

2. Find the Sum, Mean, and Maximum of a Random Array

python Copy

```python
# Creating a random array of size 5x5
random_array = np.random.randint(1, 100, (5, 5))

# Computing sum, mean, and max
array_sum = np.sum(random_array)
array_mean = np.mean(random_array)
array_max = np.max(random_array)

print("Random Array:\n", random_array)
print("Sum:", array_sum)
print("Mean:", array_mean)
print("Max:", array_max)
```

1. Create an array of numbers from 1 to 50 and print the first 10 numbers

python Copy Edit

```python
import numpy as np

# Creating an array from 1 to 50
arr = np.arange(1, 51)

# Printing the first 10 numbers
print("First 10 numbers:", arr[:10])
```

---

2. Print only the even numbers from the array

python Copy Edit

```python
# Filtering even numbers
even_numbers = arr[arr % 2 == 0]

print("Even numbers:", even_numbers)
```

1. Extract the Middle 2x2 Section of a 4x4 Matrix

To extract the middle 2x2 section from a 4x4 matrix, you can use slicing. Here's how

```python
import numpy as np

# Creating a 4x4 matrix
matrix = np.array([[1, 2, 3, 4],
                   [5, 6, 7, 8],
                   [9, 10, 11, 12],
                   [13, 14, 15, 16]])

# Extracting the middle 2x2 section
middle_section = matrix[1:3, 1:3]

print("Matrix:\n", matrix)
print("Middle 2x2 Section:\n", middle_section)
```

2. Reverse the Order of Rows in a Matrix Using Slicing

To reverse the order of rows in a matrix, you can use slicing with `[::-1]`:

```python
# Reversing the order of rows
reversed_matrix = matrix[::-1]

print("Reversed Matrix:\n", reversed_matrix)
```

1. Sort the Array Column-Wise

To sort the array column-wise (i.e., sort each column independently), you can use `np.sort()` with the `axis=0` argument.

python Copy Edit

```python
import numpy as np

# Creating a 4x4 random matrix
array = np.random.randint(1, 100, (4, 4))

# Sorting column-wise
sorted_array = np.sort(array, axis=0)

print("Original Array:\n", array)
print("Column-wise Sorted Array:\n", sorted_array)
```

2. Filter Numbers that are Even and Greater than 30

You can use a Boolean mask to filter numbers that meet both conditions (even and grea

python C

```python
# Filtering even numbers greater than 30
filtered_array = array[(array % 2 == 0) & (array > 30)]

print("Filtered Numbers (Even and > 30):", filtered_array)
```

```python
import matplotlib.pyplot as plt

# Product names and their prices
products = ['Product A', 'Product B', 'Product C', 'Product D', 'Product E']
prices = [120, 90, 210, 60, 140]

# Create the line plot
plt.plot(products, prices, marker='o', color='blue')

# Customize the plot
plt.title('Product Prices')
plt.xlabel('Products')
plt.ylabel('Price')
plt.grid(True)

# Show the plot
plt.show()
```

```python
import numpy as np

# 1. Store pizza prices in a NumPy array (example prices in dollars)
pizza_prices = np.array([8.99, 12.50, 10.75, 15.00, 9.25])

# 2. Find the cheapest and most expensive pizza
cheapest = np.min(pizza_prices)
most_expensive = np.max(pizza_prices)

# 3. Calculate the average cost per pizza
average_cost = np.mean(pizza_prices)
```

```python
import numpy as np

# 1. Store the last 10 game scores in a NumPy array
scores = np.array([102, 95, 110, 87, 99, 105, 97, 115, 89, 101])

# 2. Find the highest and lowest scores
highest_score = np.max(scores)
lowest_score = np.min(scores)

# 3. Calculate the average score
average_score = np.mean(scores)

# 4. Check if they scored above 100 in any game
scored_above_100 = np.any(scores > 100)
```

```python
import numpy as np

# 1. Store temperatures for 7 days in Celsius
temps_celsius = np.array([22.5, 24.0, 19.8, 21.2, 23.4, 25.1, 20.0])

# 2. Find highest and lowest temperature
highest_temp = np.max(temps_celsius)
lowest_temp = np.min(temps_celsius)

# 3. Calculate average temperature
average_temp = np.mean(temps_celsius)

# 4. Convert temperatures to Fahrenheit
temps_fahrenheit = (temps_celsius * 9/5) + 32
```

```python
import numpy as np

# 1. Store prices of 5 items from 3 supermarkets (rows = items, columns = supermarkets)
prices = np.array([
    [2.99, 3.49, 2.79],  # Item 1
    [1.99, 2.09, 1.95],  # Item 2
    [4.49, 4.29, 4.59],  # Item 3
    [0.99, 1.09, 1.05],  # Item 4
    [5.25, 5.10, 5.35]   # Item 5
])

# 2. Find the cheapest price for each item
cheapest_per_item = np.min(prices, axis=1)

# 3. Find the most expensive price in the entire matrix
most_expensive_price = np.max(prices)

# 4. Calculate the average price of all items
average_price = np.mean(prices)
```

```python
import numpy as np

# 1. Create an array of 6 random numbers between 1 and 50
lottery_numbers = np.random.randint(1, 51, 6)

# 2. Sort the numbers in ascending order
sorted_lottery_numbers = np.sort(lottery_numbers)

# 3. Output numbers bigger than 7 in the array
numbers_above_7 = sorted_lottery_numbers[sorted_lottery_numbers > 7]
```

```python
import numpy as np

# 1. Store the grades of 20 students in a NumPy array
grades = np.array([85, 92, 78, 88, 95, 67, 80, 90, 74, 91, 83, 76, 89, 85, 70, 96, 60, 81, 77, 82

# 2. Find the highest and lowest grade
highest_grade = np.max(grades)
lowest_grade = np.min(grades)

# 3. Calculate the class average
class_average = np.mean(grades)

# 4. Count how many students scored above 80
students_above_80 = np.sum(grades > 80)

# 5. Sort the grades in descending order
sorted_grades = np.sort(grades)[::-1]
```

```python
import numpy as np

# 1. Store the distances from Earth to different planets in a NumPy array (in million km)
planet_distances_km = np.array([57.9, 108.2, 149.6, 227.9, 778.3, 1432.6, 2871.0, 4495.1, 5913.5])

# 2. Find the closest and farthest planet
closest_planet_distance = np.min(planet_distances_km)
farthest_planet_distance = np.max(planet_distances_km)

# 3. Calculate the average distance to the planets
average_distance = np.mean(planet_distances_km)

# 4. Convert all distances to miles (1 km = 0.621 miles)
planet_distances_miles = planet_distances_km * 0.621
```

```python
import numpy as np

# 1. Store the lengths of 5 movies in a NumPy array (in minutes)
movie_lengths = np.array([120, 95, 150, 130, 110])  # Movie lengths in minutes

# 2. Find the longest and shortest movie
longest_movie = np.max(movie_lengths)
shortest_movie = np.min(movie_lengths)

# 3. Calculate the total time for the marathon
total_time_minutes = np.sum(movie_lengths)

# 4. Convert all movie times to hours and minutes format
movie_times_hours_minutes = [(length // 60, length % 60) for length in movie_lengths]
```