vishnu-265-lab7

September 4, 2023

[16]:
```
#Create two 3×3 matrices using the random function in Numpy and perform the␣
 ↪following operations.
#è Dot Product (dot)
#è Product (prod)
#è Multiplication (multiply)


import numpy as np

# Create two 3x3 random matrices
matrix1 = np.random.randint(10,size=(3,3))
matrix2 = np.random.randint(10, size=(3,3))

# Display the matrices
print("Matrix 1:")
print(matrix1)
print("\nMatrix 2:")
print(matrix2)

# Perform the operations

# 1. Product (prod)
product_result = np.prod(matrix1)
print("\nProduct of Matrix 1:")
print(product_result)

# 2. Element-wise Multiplication (multiply)
multiply_result = np.multiply(matrix1, matrix2)
print("\nElement-wise Multiplication:")
print(multiply_result)

# 3. Dot Product (dot)
dot_product_result = np.dot(matrix1, matrix2)
print("\nDot Product:")
print(dot_product_result)
```

```
Matrix 1:
[[2 2 5]
```

```
 [2 6 6]
 [5 2 1]]

Matrix 2:
[[4 0 0]
 [7 1 0]
 [6 0 1]]

Product of Matrix 1:
14400

Element-wise Multiplication:
[[ 8  0  0]
 [14  6  0]
 [30  0  1]]

Dot Product:
[[52  2  5]
 [86  6  6]
 [40  2  1]]
```

[9]:
```python
#Perform the following set operations using the Numpy functions.
# Union
# Intersection
# Set difference
# XOR

import numpy as np

# Create two NumPy arrays as sets
set1 = np.array([1, 2, 3, 4, 5])
set2 = np.array([3, 4, 5, 6, 7])

# Perform set operations

# Union
union_result = np.union1d(set1, set2)
print("Union:")
print(union_result)

# Intersection
intersection_result = np.intersect1d(set1, set2)
print("\nIntersection:")
print(intersection_result)

# Set Difference
set_difference_result = np.setdiff1d(set1, set2)
```

```
print("\nSet Difference (set1 - set2):")
print(set_difference_result)

# XOR (Exclusive OR)
xor_result = np.setxor1d(set1, set2)
print("\nXOR (Exclusive OR):")
print(xor_result)
```

```
Union:
[1 2 3 4 5 6 7]

Intersection:
[3 4 5]

Set Difference (set1 - set2):
[1 2]

XOR (Exclusive OR):
[1 2 6 7]
```

[7]:
```
#. Create a 1D array using Random function and perform the following operations.
# Cumulative sum
# Cumulative Product
# Discrete difference (with n=3)
# Find the unique elements from the array

import numpy as np

array = np.random.randint(10,size=6)

print("Original Array:")
print(array)

cumulative_sum = np.cumsum(array)

cumulative_product = np.cumprod(array)

discrete_difference = np.diff(array, n=3)

unique_elements = np.unique(array)
print("\nCumulative Sum:")
print(cumulative_sum)

print("\nCumulative Product:")
print(cumulative_product)

print("\nDiscrete Difference (n=3):")
```

```
print(discrete_difference)

print("\nUnique Elements:")
print(unique_elements)
```

```
Original Array:
[3 5 1 0 5 7]

Cumulative Sum:
[ 3  8  9  9 14 21]

Cumulative Product:
[ 3 15 15  0  0  0]

Discrete Difference (n=3):
[ 9  3 -9]

Unique Elements:
[0 1 3 5 7]
```

[10]:
```
# Create two 1D array and perform the Addition using zip(), add()
# and user defined function (frompyfunc())

import numpy as np

# Step 2: Create two 1D arrays
array1 = np.array([1, 2, 3, 4, 5])
array2 = np.array([10, 20, 30, 40, 50])

# Step 3: Perform addition using zip()
result_zip = [a + b for a, b in zip(array1, array2)]

# Step 4: Perform addition using numpy.add()
result_np_add = np.add(array1, array2)

# Step 5: Perform addition using numpy.frompyfunc()
def add_elements(a, b):
    return a + b

add_func = np.frompyfunc(add_elements, 2, 1)
result_frompyfunc = add_func(array1, array2)

# Display the results
print("Using zip():", result_zip)
print("Using numpy.add():", result_np_add)
print("Using numpy.frompyfunc():", result_frompyfunc)
```

```
Using zip(): [11, 22, 33, 44, 55]
```

```
Using numpy.add(): [11 22 33 44 55]
Using numpy.frompyfunc(): [11 22 33 44 55]
```

[11]:
```python
#Find the LCM (Least Common Multiple) and GCD (Greatest Common Divisor)
# of an array of elements using reduce().


from functools import reduce
import math

elements = [12, 18, 24, 36]

def calculate_lcm(x, y):
    return x * y // math.gcd(x, y)

lcm_result = reduce(calculate_lcm, elements)

gcd_result = reduce(math.gcd, elements)

print("Elements:", elements)
print("LCM:", lcm_result)
print("GCD:", gcd_result)
```

```
Elements: [12, 18, 24, 36]
LCM: 72
GCD: 6
```