CODETANTRA Home 202401090132@mitaoe.ac.in Support Logout

3.2.3. Numpy: Custom Sequence Generation

01:11

Write a Python program that takes the following inputs from the user:

- Start value: The starting point of the sequence.
- Stop value: The sequence should end before this value.
- Step value: The increment between each number in the sequence.

The program should then generate a sequence using `numpy` based on these inputs and print the generated sequence.

Input Format:

- The user will input three integer values: start, stop, and step, each on a new line.

Output Format:

- The program should print the generated sequence based on the input values.

Sample Test Cases +

Explorer

customS...

Submit

Debugger

```
1  import numpy as np
2
3  # Take user input for the start, stop, and step of the sequence
4  start = int(input())
5  stop = int(input())
6  step = int(input())
7
8  # Generate the sequence using np.arange()
9  a= np.arange(start,stop, step)
10 print(a)
11 # Print the generated sequence
12
```

Terminal Test cases

< Prev Reset Submit Next >

3.2.2. Numpy: Horizontal and Vertical Stacking of Arrays

00:47

You are given two arrays `arr1` and `arr2`. You need to perform horizontal and vertical stacking operations on them using NumPy.

- **Horizontal Stacking**: Stack the two matrices horizontally (side by side).
- **Vertical Stacking**: Stack the two matrices vertically (one below the other).

Input Format:

- The program should first prompt the user to input two 3x3 arrays.
- Each array consists of 3 rows, and each row contains 3 space-separated integers.
- The user will input the two arrays row by row.

Output Format:

- The program should display the result of the Horizontal Stack (side-by-side stacking) of the two arrays.
- The program should then display the result of the Vertical Stack (one below the other) of the two arrays.

Sample Test Cases +

Explorer

stacking.py

Submit

```
1  import numpy as np
2
3  # Input matrices
4  print("Enter Array1:")
5  arr1 = np.array([list(map(int, input().split())) for i in range(3)])
6
7  print("Enter Array2:")
8  arr2 = np.array([list(map(int, input().split())) for i in range(3)])
9
10 # Perform horizontal stacking (hstack)
11 a=np.hstack((arr1,arr2))
12 print("Horizontal Stack:")
13 print(a)
14
15 print("Vertical Stack:")
16 b=np.vstack((arr1,arr2))
17 print(b)
18
19
20
```

Debugger

Terminal Test cases

3.2.1. Numpy: Matrix Operations

01:57

The given code takes two 3×3 matrices, $\mathbf{matrix\_a}$, and $\mathbf{matrix\_b}$, as input from the user and converts them into NumPy arrays.

Task:

You are required to compute and display the results of the following matrix operations:

1. **Addition** ($\mathbf{matrix\_a} + \mathbf{matrix\_b}$)
2. **Subtraction** ($\mathbf{matrix\_a} - \mathbf{matrix\_b}$)
3. **Element-wise Multiplication** ($\mathbf{matrix\_a} * \mathbf{matrix\_b}$)
4. **Matrix Multiplication** ($\mathbf{matrix\_a} \cdot \mathbf{matrix\_b}$)
5. **Transpose of Matrix A**

Input Format:

- The user will input 3 rows for $\mathbf{matrix\_a}$, each containing 3 integers separated by spaces.
- Similarly, the user will input 3 rows for $\mathbf{matrix\_b}$, each containing 3 integers separated by spaces.

Output Format:

The program should display the results of the operations in the following order:

1. The result of Addition.
2. The result of Subtraction.
3. The result of Element-wise Multiplication

Sample Test Cases +

Explorer

matrixOp...

Submit

Debugger

```
1  import numpy as np
2
3  # Input matrices
4  print("Enter Matrix A:")
5  matrix_a = np.array([list(map(int, input().split())) for i in
   range(3)])
6
7  print("Enter Matrix B:")
8  matrix_b = np.array([list(map(int, input().split())) for i in
   range(3)])
9
10
11 # Addition
12 print("Addition (A + B):")
13 print(matrix_a + matrix_b)
14 # Subtraction
15 print("Subtraction (A - B):")
16 print(matrix_a - matrix_b)
17 # Multiplication (element-wise)
18 print("Element-wise Multiplication (A * B):")
19 print(matrix_a*matrix_b)
20 # Matrix multiplication (dot product)
21 print("A dot B:")
22 print(np.dot(matrix_a, matrix_b))
23 # Transpose
```

Terminal Test cases

CODETANTRA Home 202401090132@mitaoe.ac.in Support Logout

3.2.7. Student Data Analysis and Operations

00:49

Write a Python program that takes the file name of a CSV file containing student details, including roll numbers and their marks in three subjects as input, reads the data, and performs the following operations:

- **Print all student details**: Display the complete details of all students, including roll numbers and marks for all subjects.
- **Find total students**: Determine the total number of students in the dataset.
- **Print all student roll numbers**: Extract and print the roll numbers of all students.
- **Print Subject 1 marks**: Extract and print the marks of all students in Subject 1.
- **Find minimum marks in Subject 2**: Identify the lowest marks in Subject 2.
- **Find maximum marks in Subject 3**: Identify the highest marks in Subject 3.
- **Print all subject marks**: Display the marks of all students for each subject.
- **Find total marks of students**: Compute the total marks for each student across all subjects.
- **Find the average marks of each student**: Compute the average marks for each student.
- **Find average marks of each subject**: Compute the average marks for all students in each subject.
- **Find average marks of Subject 1 and Subject 2**: Compute the average marks for Subject 1 and Subject 2.
- **Find average marks of Subject 1 and Subject 3**: Compute the average marks for Subject 1 and Subject 3.
- **Find the roll number of the student with maximum marks in Subject 3**: Identify the student with the highest marks in Subject 3 and print their roll number.

Sample Test Cases +

Explorer

Operatio...

Submit

```python
import numpy as np

a = np.loadtxt("Sample.csv", delimiter=',', skiprows=1)

# 1. Print all student details
print("All student Details:\n",a)

# 2. print total students
r,c=a.shape
print("Total Students:",r)

# 3. Print all student Roll numbers
print("All Student Roll Nos",a[:,0])

# 4. Print subject 1 marks
print("Subject 1 Marks",a[:,1])

# 5. print minimum marks of Subject 2
print("Min marks in Subject 2",np.min(a[:,2]))

# 6. print maximum marks of Subject 3
print("Max marks in Subject 3",np.max(a[:,3]))

# 7. Print All subject marks
print("All subject marks:",a[:,1:])
```

Debugger

Terminal Test cases

Prev Reset Submit Next

3.2.6. Numpy: Searching, Sorting, Counting, Broadcasting

09:15

The given code in the editor takes a single array, `array1`, as space-separated integers as input from the user.
Additionally, it takes the following inputs:

- `search_value`: The value to search for in the array.
- `count_value`: The value to count its occurrences in the array.
- `broadcast_value`: The value to add for broadcasting across the array.

You need to complete the code to perform the following operations:
1. **Searching**: Find the indices where `search_value` appears in `array1` and print these indices.
2. **Counting**: Count how many times `count_value` appears in `array1` and print the count.
3. **Broadcasting**: Add `broadcast_value` to each element of `array1` using broadcasting, and print the resulting array.
4. **Sorting**: Sort `array1` in ascending order and print the sorted array.

Input Format:

1. A single line containing space-separated integers representing `array1`.
2. An integer `search_value` represents the value to search for in the array.
3. An integer `count_value` represents the value to count in the array.
4. An integer `broadcast_value` represents the value to add to each element of the array.

Output Format:

Sample Test Cases +

Explorer

arrayOpe...

Submit

```
1  import numpy as np
2
3  # Input array from the user
4  array1 = np.array(list(map(int, input().split())))
5
6  # Searching
7  search_value = int(input("Value to search: "))
8  count_value = int(input("Value to count: "))
9  broadcast_value = int(input("Value to add: "))
10
11 # Find indices where value matches in array1
12 a=np.where(array1==search_value)[0]
13 print(a)
14 # Count occurrences in array1
15 b=np.count_nonzero(array1==count_value)
16 print(b)
17 # Broadcasting addition
18 c= array1+broadcast_value
19 print(c)
20 # Sort the first array
21 d= np.sort(array1)
22 print(d)
```

Debugger

Terminal Test cases

3.2.5. Numpy: Copying and Viewing Arrays

10:05

The given code takes a list of integers as input and converts it into a NumPy array. Your task is to complete the code by:

- Creating a view of the `original_array` and assigning it to `view_array`.
- Creating a copy of the `original_array` and assigning it to `copy_array`.

After completing these steps, observe how modifying the view affects the `original_array`, while modifying the copy does not.

Input Format:

- A single line of space-separated integers.

Output Format:

- After modifying the view:

```
Original array after modifying view: <original_array>
View array: <view_array>
```

- After modifying the copy:

```
Original array after modifying copy: <original_array>
Copy array: <copy_array>
```

Sample Test Cases +

Explorer

copyAnd...

Submit

Debugger

```
1  import numpy as np
2
3  inputlist = list(map(int,input().split(" ")))
4
5  # Original array
6  original_array = np.array(inputlist)
7
8  # Create a view
9  view_array = original_array.view()
10
11 # Create a copy
12 copy_array = original_array.copy()
13
14 # Modify the view
15 view_array[0] = 99
16 print("Original array after modifying view:", original_array)
17 print("View array:", view_array)
18
19 # Modify the copy
20 copy_array[1] = 88
21 print("Original array after modifying copy:", original_array)
22 print("Copy array:", copy_array)
23
```

Terminal Test cases

3.2.4. Numpy: Arithmetic and Statistical Operations, Mathematical Operations, Bi... 09:57

You are given two arrays `A` and `B`. Your task is to complete the function `array_operations`, which will convert these lists into NumPy arrays and perform the following operations:

1. **Arithmetic Operations**:
   - Compute the element-wise sum, difference, and product of the two arrays.
2. **Statistical Operations**:
   - Calculate the mean, median, and standard deviation of array `A`.
3. **Bitwise Operations**:
   - Perform bitwise `AND`, bitwise `OR`, and bitwise `XOR` on the arrays (ex: A_i `OR` B_i).

Input Format:

- The first line contains space-separated integers representing the elements of array `A`.
- The second line contains space-separated integers representing the elements of array `B`.

Output Format:

- For each operation (arithmetic, statistical, and bitwise), print the results in the specified format as shown in sample test cases.

Sample Test Cases +

Explorer

different...

Submit

```python
import numpy as np

def array_operations(A, B):

    # Convert A and B to NumPy arrays
    A = np.array(A)
    B = np.array(B)
    # Arithmetic Operations
    sum_result = A + B
    diff_result = A - B
    prod_result = A * B

    # Statistical Operations
    mean_A = np.mean(A)
    median_A = np.median(A)
    std_dev_A = np.std(A)


    # Bitwise Operations
    and_result = A & B
    or_result = A | B
    xor_result = A ^ B

    # Output results with one space between each element
    print("Element-wise Sum:", ' '.join(map(str, sum_result)))
```

Debugger

Terminal Test cases