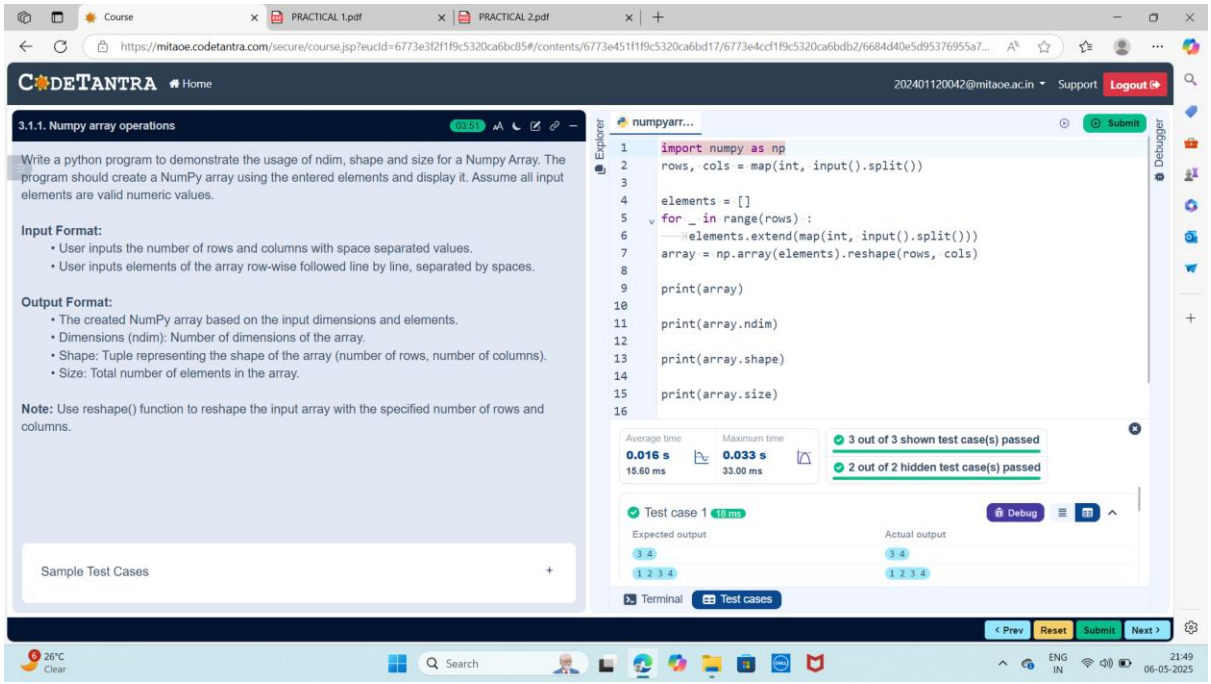
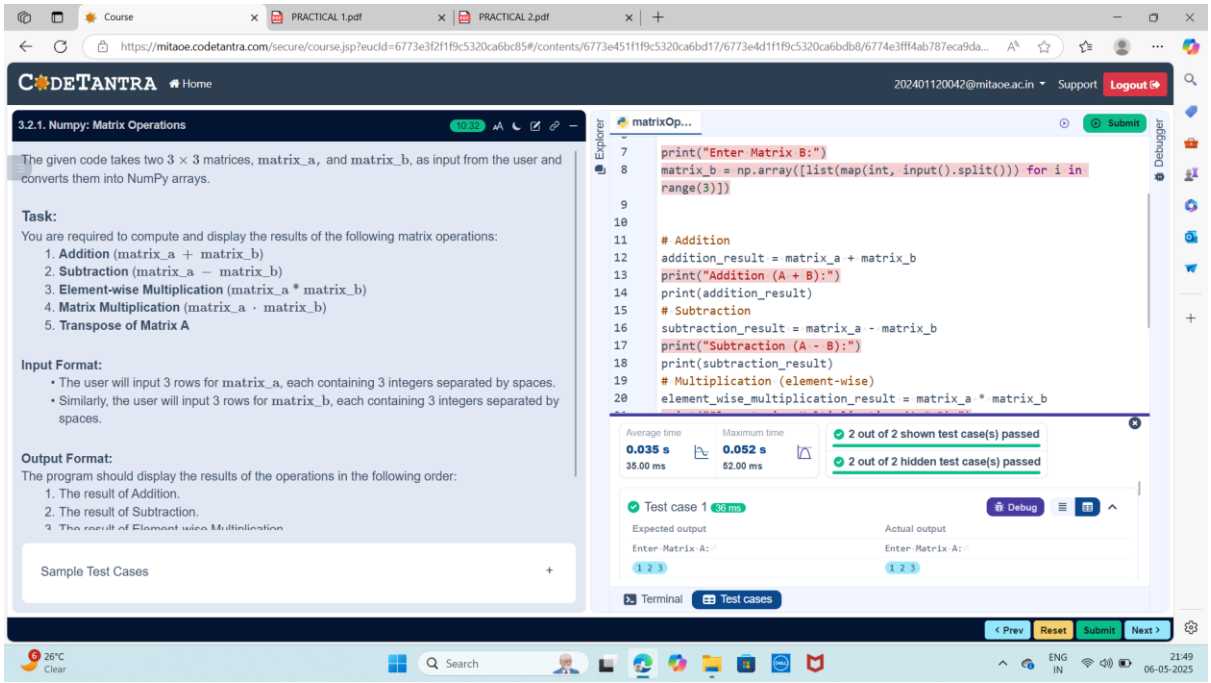
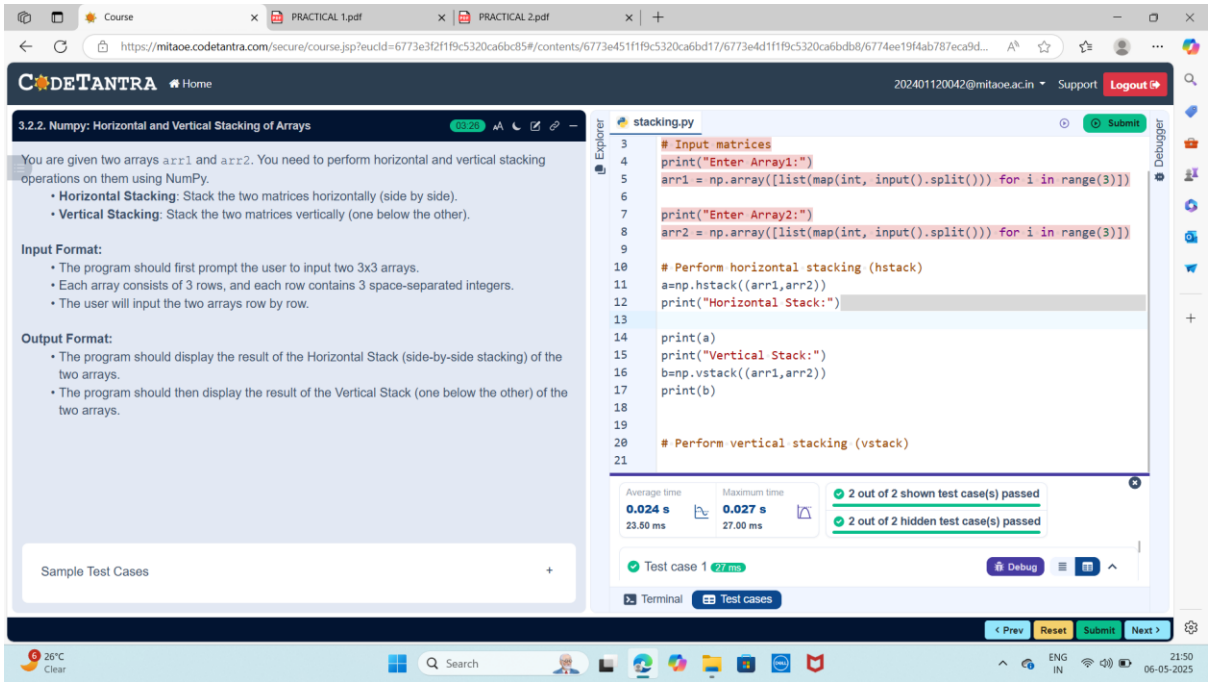
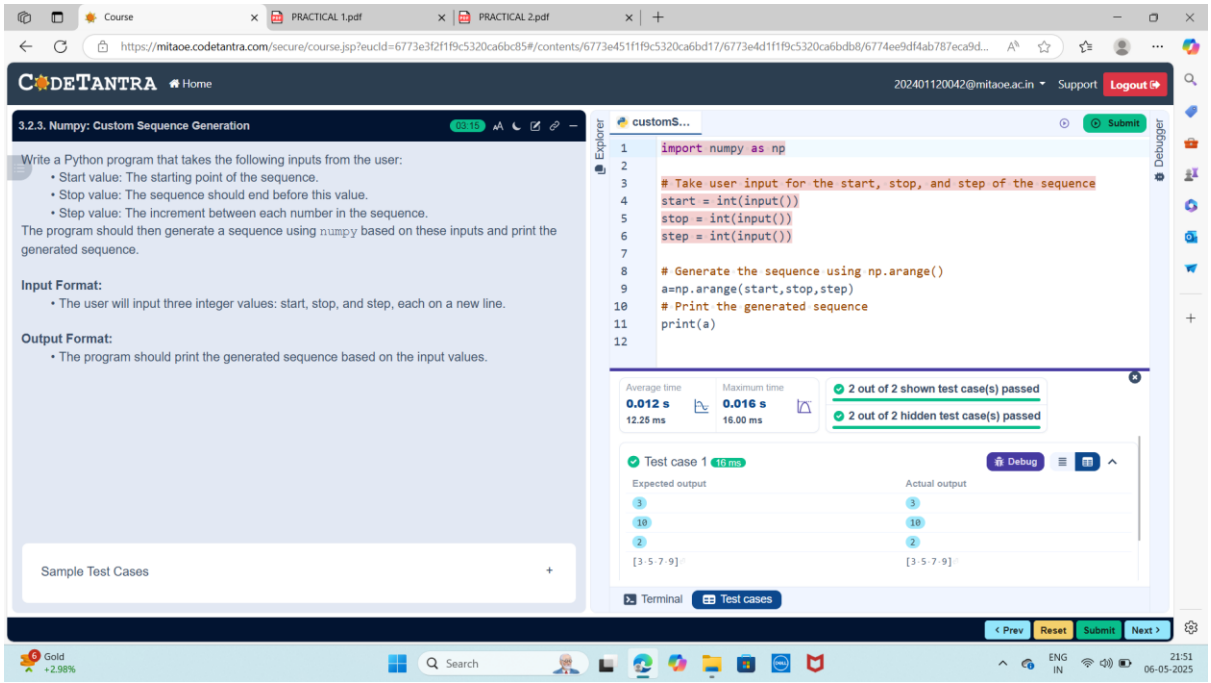
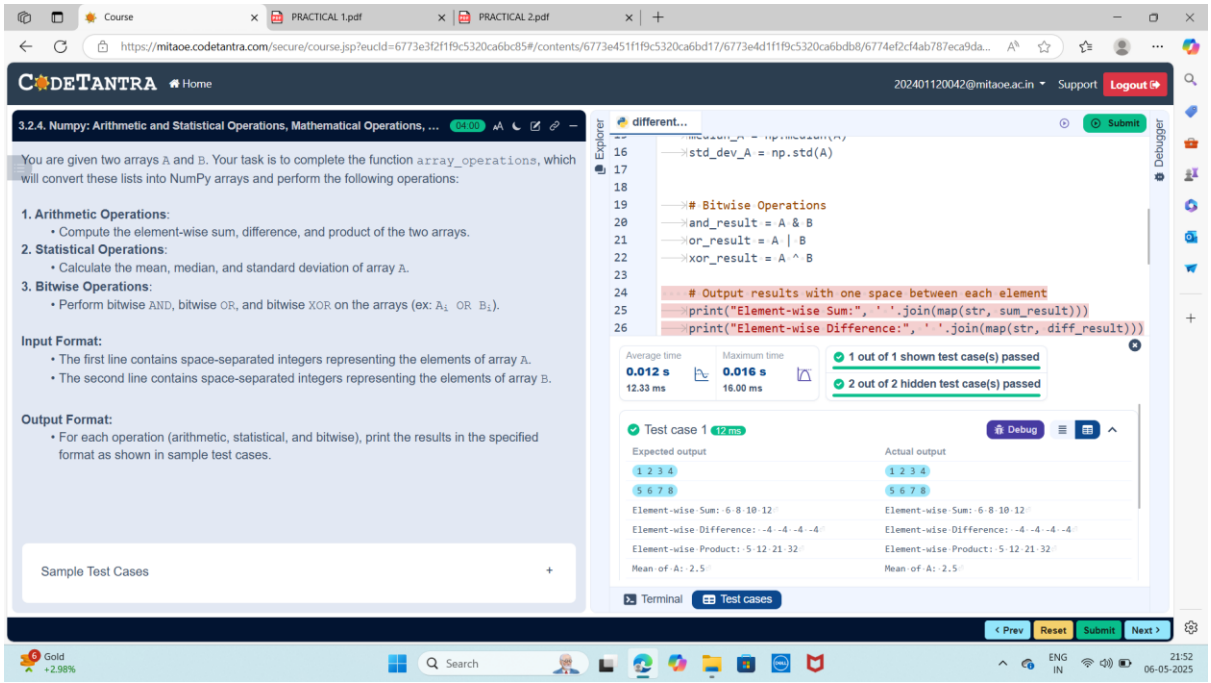
PRACTICAL 3

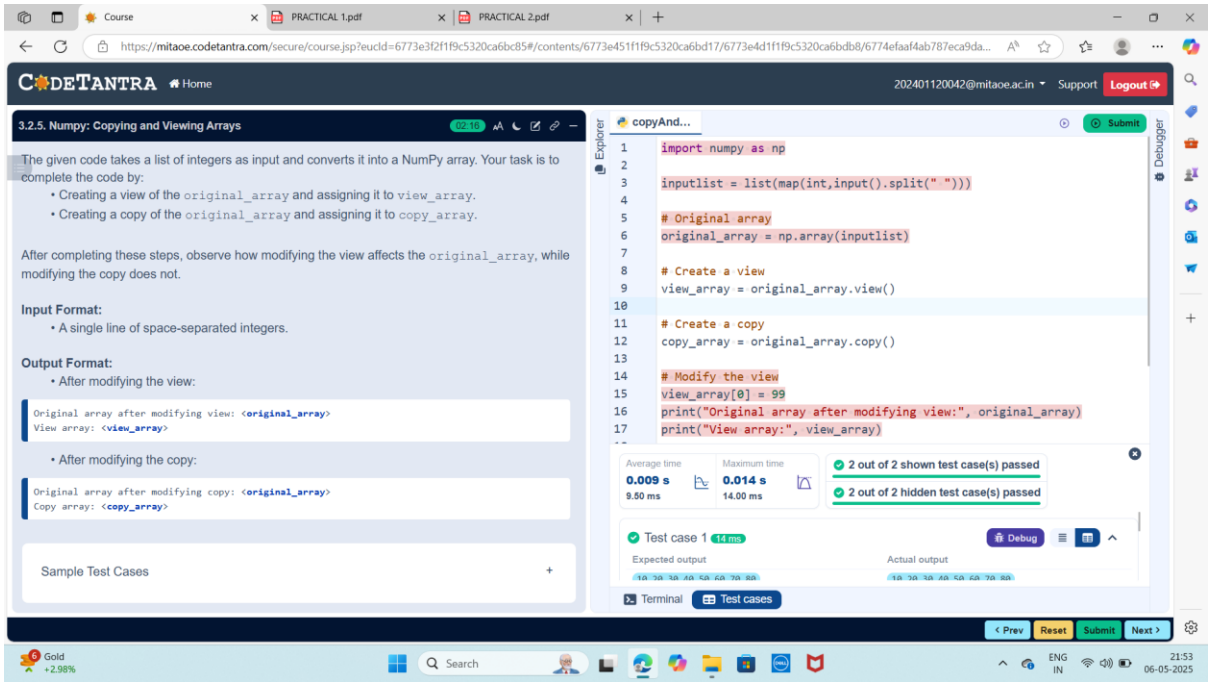

Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.1.1. Numpy array operations
Write a python program to demonstrate the usage of ndim, shape and size for a Numpy Array. The program should create a NumPy array using the entered elements and display it. Assume all input elements are valid numeric values.
Input Format:
• User inputs the number of rows and columns with space separated values.
• User inputs elements of the array row-wise followed line by line, separated by spaces.
Output Format:
• The created NumPy array based on the input dimensions and elements.
• Dimensions (ndim): Number of dimensions of the array.
• Shape: Tuple representing the shape of the array (number of rows, number of columns).
• Size: Total number of elements in the array.
Note: Use reshape() function to reshape the input array with the specified number of rows and columns.
Sample Test Cases
numpyarr...
Submit
import numpy as np
rows, cols = map(int, input().split())
elements = []
for _ in range(rows) :
elements.extend(map(int, input().split()))
array = np.array(elements).reshape(rows, cols)
print(array)
print(array.ndim)
print(array.shape)
print(array.size)
Average time
0.016 s
15.60 ms
Maximum time
0.033 s
33.00 ms
3 out of 3 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
18 ms
Debug
Expected output
Actual output
Terminal
Test cases
Prev
Reset
Submit
Next

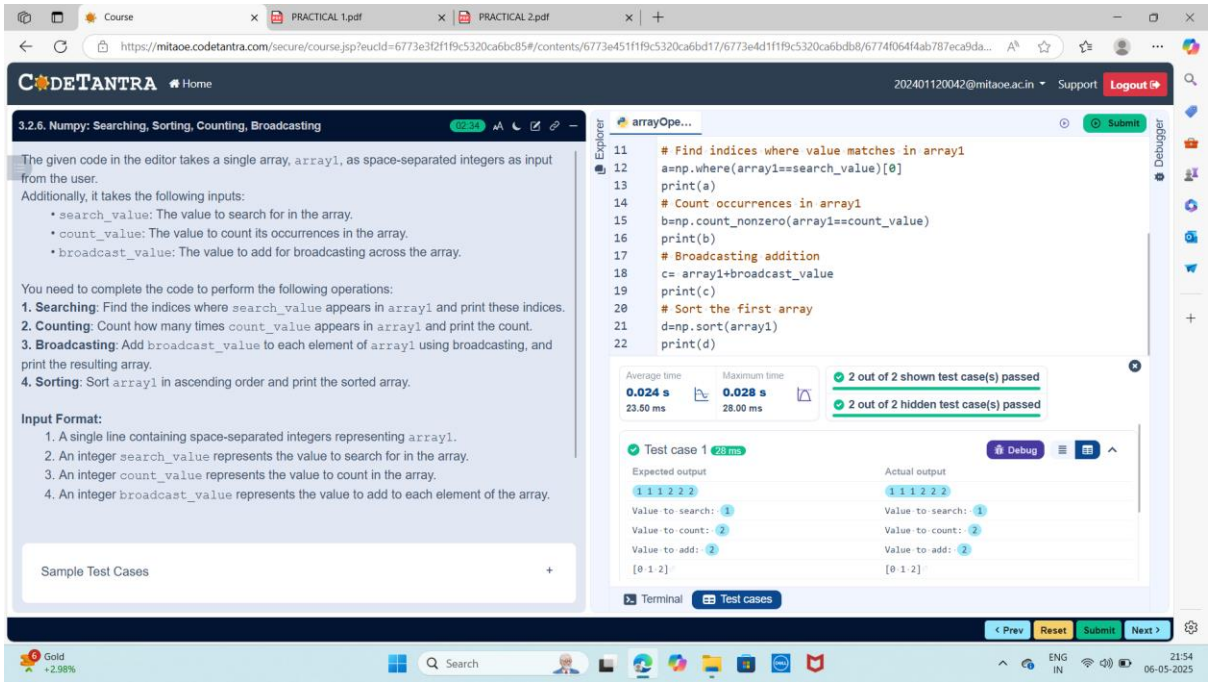

Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.2.1. Numpy: Matrix Operations
The given code takes two 3 × 3 matrices, matrix_a, and matrix_b, as input from the user and converts them into NumPy arrays.
Task:
You are required to compute and display the results of the following matrix operations:
1. Addition (matrix_a + matrix_b)
2. Subtraction (matrix_a − matrix_b)
3. Element-wise Multiplication (matrix_a * matrix_b)
4. Matrix Multiplication (matrix_a · matrix_b)
5. Transpose of Matrix A
Input Format:
• The user will input 3 rows for matrix_a, each containing 3 integers separated by spaces.
• Similarly, the user will input 3 rows for 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.
Sample Test Cases
matrixOp...
Submit
print("Enter Matrix B:")
matrix_b = np.array([list(map(int, input().split())) for i in range(3)])
# Addition
addition_result = matrix_a + matrix_b
print("Addition (A + B):")
print(addition_result)
# Subtraction
subtraction_result = matrix_a - matrix_b
print("Subtraction (A - B):")
print(subtraction_result)
# Multiplication (element-wise)
element_wise_multiplication_result = matrix_a * matrix_b
Average time
0.035 s
35.00 ms
Maximum time
0.052 s
52.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
36 ms
Debug
Expected output
Actual output
Terminal
Test cases
Prev
Reset
Submit
Next

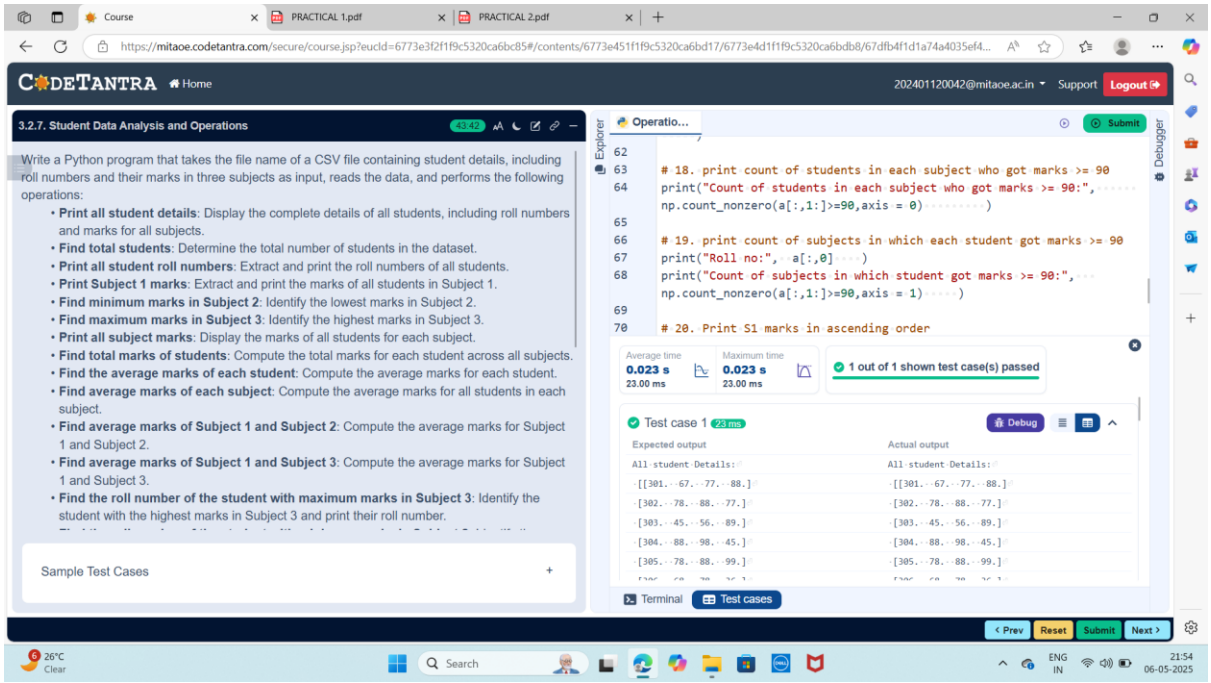

Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.2.2. Numpy: Horizontal and Vertical Stacking of Arrays
You are given two arrays arr1 and arr2. You need to perform horizontal and vertical stacking operations on them using NumPy.
• Horizontal Stacking: 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
stacking.py
Submit
# Input matrices
print("Enter Array1:")
arr1 = np.array([list(map(int, input().split())) for i in range(3)])
print("Enter Array2:")
arr2 = np.array([list(map(int, input().split())) for i in range(3)])
# Perform horizontal stacking (hstack)
a=np.hstack((arr1,arr2))
print("Horizontal Stack:")
print(a)
print("Vertical Stack:")
b=np.vstack((arr1,arr2))
print(b)
# Perform vertical stacking (vstack)
Average time
0.024 s
23.50 ms
Maximum time
0.027 s
27.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
Debug
Terminal
Test cases
Prev
Reset
Submit
Next
Search
ENG
IN
21:50
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.2.3. Numpy: Custom Sequence Generation
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
customS...
Submit
import numpy as np
# Take user input for the start, stop, and step of the sequence
start = int(input())
stop = int(input())
step = int(input())
# Generate the sequence using np.arange()
a=np.arange(start,stop,step)
# Print the generated sequence
print(a)
Average time
0.012 s
12.25 ms
Maximum time
0.016 s
16.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
Debug
Expected output
Actual output
3
10
2
[3 5 7 9]
Terminal
Test cases
Prev
Reset
Submit
Next
Search
ENG
IN
21:51
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.2.4. Numpy: Arithmetic and Statistical Operations, Mathematical Operations, ...
04:00
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: Ai OR Bi).
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
different...
Submit
Explorer
Debugger
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)))
print("Element-wise Difference:", ' '.join(map(str, diff_result)))
Average time
0.012 s
12.33 ms
Maximum time
0.016 s
16.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
12 ms
Debug
Expected output
Actual output
1 2 3 4
5 6 7 8
Element-wise Sum: 6 8 10 12
Element-wise Difference: -4 -4 -4 -4
Element-wise Product: 5 12 21 32
Mean of A: 2.5
Terminal
Test cases
Prev
Reset
Submit
Next
Gold
+2.98%
Search
ENG
IN
21:52
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.2.5. Numpy: Copying and Viewing Arrays
02:16
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
copyAnd...
Submit
Explorer
Debugger
import numpy as np
inputlist = list(map(int,input().split(" ")))
# Original array
original_array = np.array(inputlist)
# Create a view
view_array = original_array.view()
# Create a copy
copy_array = original_array.copy()
# Modify the view
view_array[0] = 99
print("Original array after modifying view:", original_array)
print("View array:", view_array)
Average time
0.009 s
9.50 ms
Maximum time
0.014 s
14.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
14 ms
Debug
Expected output
Actual output
Terminal
Test cases
Prev
Reset
Submit
Next
Gold
+2.98%
Search
ENG
IN
21:53
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
https://mitaoe.codetantra.com/secure/course.jsp?eucId=6773e3f2f1f9c5320ca6bc85#/contents/6773e451f1f9c5320ca6bd17/6773e4d1f1f9c5320ca6bdb8/6774f064f4ab787eca9da...
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.2.6. Numpy: Searching, Sorting, Counting, Broadcasting
02:34
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.
Sample Test Cases
Explorer
arrayOpe...
Submit
Debugger
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)
Average time 0.024 s 23.50 ms
Maximum time 0.028 s 28.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1 28 ms
Debug
Expected output
1 1 1 2 2 2
Value to search: 1
Value to count: 2
Value to add: 2
[0 1 2]
Actual output
1 1 1 2 2 2
Value to search: 1
Value to count: 2
Value to add: 2
[0 1 2]
Terminal
Test cases
Prev
Reset
Submit
Next
Gold +2.98%
Search
ENG IN
21:54 06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
https://mitaoe.codetantra.com/secure/course.jsp?eucId=6773e3f2f1f9c5320ca6bc85#/contents/6773e451f1f9c5320ca6bd17/6773e4d1f1f9c5320ca6bdb8/67dfb4f1d1a74a4035ef4...
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
3.2.7. Student Data Analysis and Operations
43:42
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
Debugger
62
63 # 18. print count of students in each subject who got marks >= 90
64 print("Count of students in each subject who got marks >= 90:",
np.count_nonzero(a[:,1:]>=90,axis = 0))
65
66 # 19. print count of subjects in which each student got marks >= 90
67 print("Roll no:",a[:,0])
68 print("Count of subjects in which student got marks >= 90:",
np.count_nonzero(a[:,1:]>=90,axis = 1))
69
70 # 20. Print S1 marks in ascending order
Average time 0.023 s 23.00 ms
Maximum time 0.023 s 23.00 ms
1 out of 1 shown test case(s) passed
Test case 1 28 ms
Debug
Expected output
All student Details:
[[301. 67. 77. 88.]
[302. 78. 88. 77.]
[303. 45. 56. 89.]
[304. 88. 98. 45.]
[305. 78. 88. 99.]
Actual output
All student Details:
[[301. 67. 77. 88.]
[302. 78. 88. 77.]
[303. 45. 56. 89.]
[304. 88. 98. 45.]
[305. 78. 88. 99.]
Terminal
Test cases
Prev
Reset
Submit
Next
26°C Clear
Search
ENG IN
21:54 06-05-2025