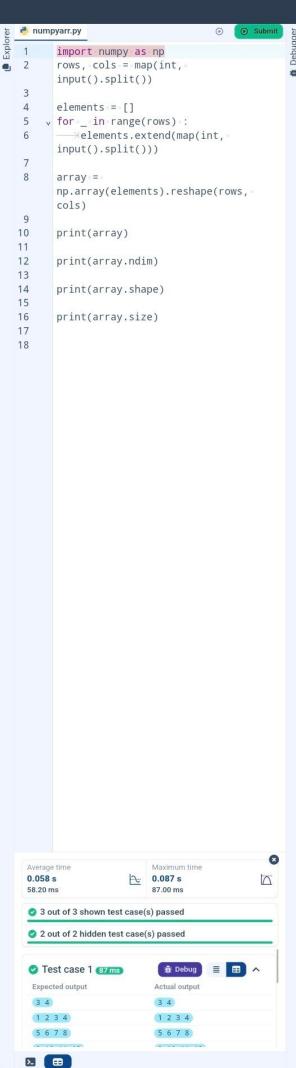
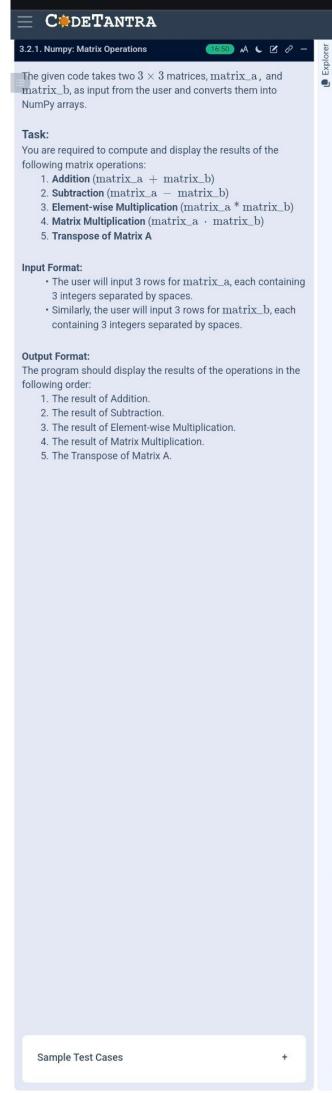
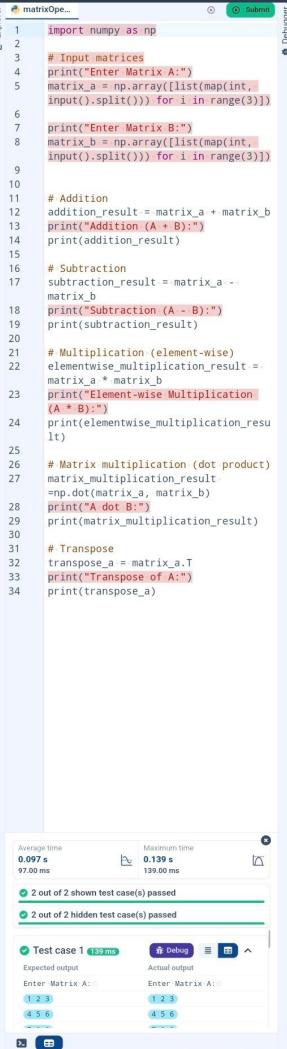


Sample Test Cases

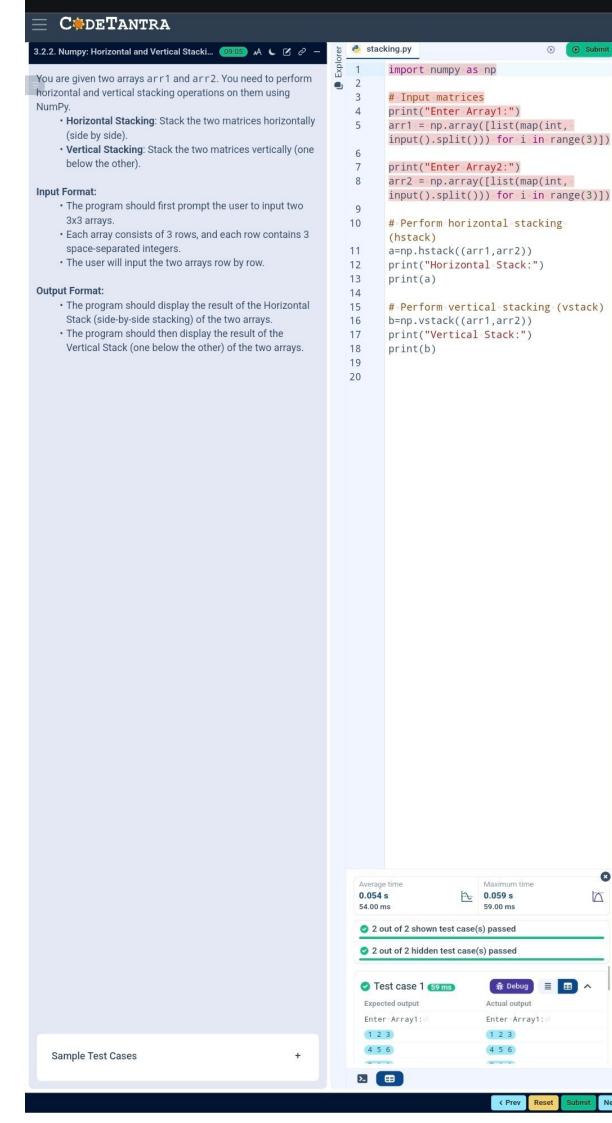


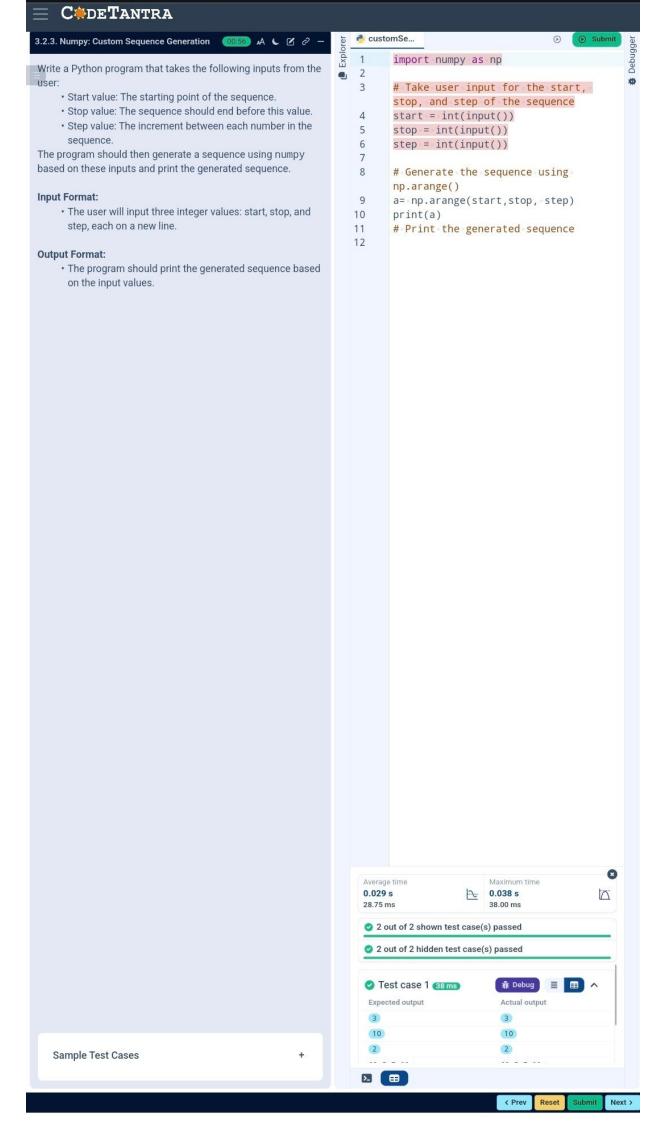
< Prev





< Prev







You are given two arrays A and B. Your task is to complete th

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:

3.2.4. Numpy: Arithmetic and Statistical Ope... 02:03 🗚 🕻 🗹 🔗 -

### 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:  $A_{\hat{1}}$  OR  $B_{\hat{1}}$ ).

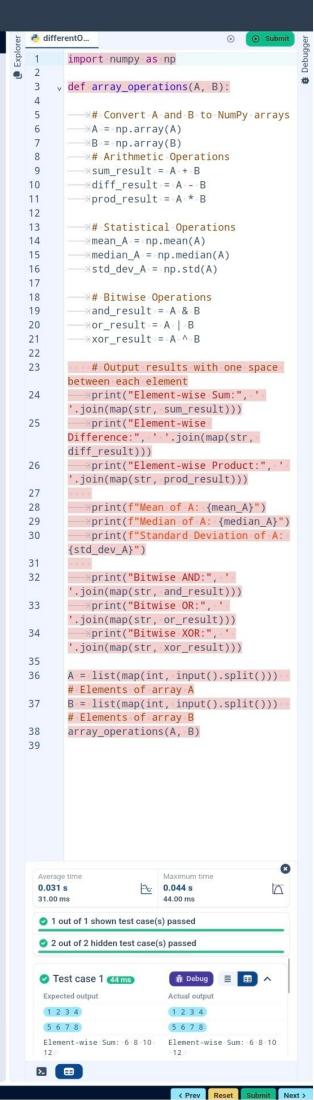
# **Input Format:**

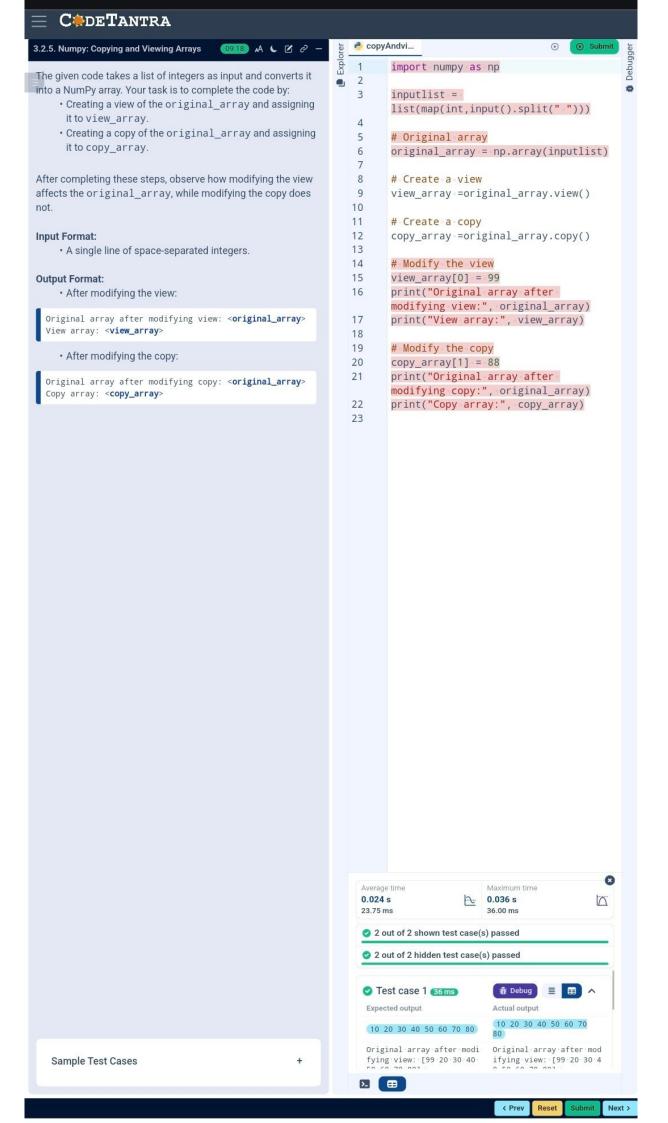
- The first line contains space-separated integers representing the elements of array A.
- The second line contains space-separated integers representing the elements of array B.

## **Output Format:**

Sample Test Cases

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





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:

3.2.6. Numpy: Searching, Sorting, Counting, ... 02:0

• 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.

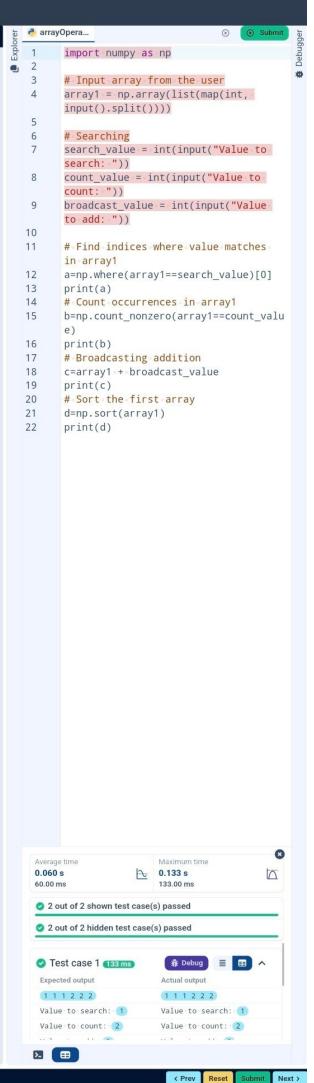
#### Input Format:

- 1. A single line containing space-separated integers representing array1.
- 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:

- 1. The indices where search\_value occurs in array1.
- 2. The count of occurrences of count\_value in array1.
- The array after adding the broadcast\_value to each element.
- 4. The sorted array.

Sample Test Cases



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:

3.2.7. Student Data Analysis and Operations (48:28) A 🕻 🗹 🔗 -

- 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.
- Find the roll number of the student with minimum marks in Subject 2: Identify the student with the lowest marks in Subject 2 and print their roll number.
- Find the roll number of students who scored 24 marks in Subject 2: Identify students who obtained exactly 24 marks in Subject 2 and print their roll numbers.
- Find the count of students who got less than 40 marks in Subject 1: Count the number of students who scored less than 40 marks in Subject 1.
- Find the count of students who got more than 90 marks in Subject 2: Count the number of students who scored more than 90 marks in Subject 2.
- Find the count of students who scored >=90 in each subject: Count the number of students who scored 90 or more marks in each subject.
- Find the count of subjects in which each student scored >=90: Determine how many subjects each student scored 90 or more marks in.
- Print Subject 1 marks in ascending order: Sort and print the marks of students in Subject 1 in ascending order.
- Print students who scored between 50 and 90 in Subject
   Display students who scored marks between 50 and
   In Subject 1.
- Find index positions of students who scored 79 in Subject 1: Identify the index positions of students who scored exactly 79 marks in Subject 1.

**Note:** Fill in the missing code to perform the above-mentioned operations.

Sample Test Cases

```
🥙 Operations...
Explorer
         import numpy as np
   1
   2
•
                                                   4
   3
         a = np.loadtxt("Sample.csv",
         delimiter=',', skiprows=1)
   4
         print("All student Details:\n", a )
   5
   6
         # 2. print total students
         print("Total Students:", a.shape[0])
   8
   9
         # 3. Print all student Roll numbers
  10
  11
         print("All Student Roll Nos",
         a[:,0] )
   12
  13
         # 4. Print subject 1 marks
         print("Subject 1 Marks",a[:,1] )
  14
  15
  16
         # 5. print minimum marks of Subject 2
  17
         print("Min marks in Subject 2",
         np.min(a[:,2]) ----)
  18
   19
         # 6. print maximum marks of Subject 3
  20
         print("Max marks in Subject 3",
         np.max(a[:,3]) ----)
  21
         # 7. Print All subject marks
  22
  23
         print("All subject marks:",
         a[:,1:] --)
  24
  25
         # 8. print Total marks of
         print("Total Marks",
  26
         np.sum(a[:,1:],axis=1) ----)
  27
         # 9. print average marks of each
  28
         student
  29
         avg = np.mean(a[:,1:],axis=1)
  30
         print(np.round(avg,1))
  31
  32
         # 10. print average marks of each
         subject
  33
         print("Average Marks of each
         subject", np.mean(a[:,1:],axis=0)
          . .)
  34
  35
         # 11. print average marks of S1 and
  36
         print("Average Marks of S1 and S2",
          np.mean(a[:,1:3],axis=0)
  37
  38
         # 12. print average marks of S1 and
  39
         print("Average Marks of S1 and S3",
            np.mean(a[:,[1,3]],axis=0)
  40
  41
         # 13. print Roll number who got
         maximum marks in Subject 3
  42
         i = np.argmax(a[:.31)
                           Maximum time
    0.090 s
                          0.090 s
                                              10
                          90.00 ms
    1 out of 1 shown test case(s) passed
    Test case 1 90 ms
                            🏦 Debug
                                    Expected output
                           Actual output
     All student Details:
                           All student Details:
                            · [ [ 301 . · · 67 . · · 77 . · · 88 . 1
      [[301. 67. 77. 88.]
      [302. 78. 88. 77.]
                            [302. 78. 88. 77.]
      [303. 45. 56. 89.]
                            [303. 45. 56. 89.]
    № 🖽
```

< Prev

Reset