Login - Codetantra - Course

CODETANTRA Home 202401090042@mitaoe.ac.in Support Logout

1.2.1. Pass or Fail

53:27

Write a Python program that accepts the number of courses and the marks of a student in those courses.
The grade is determined based on the aggregate percentage:

- If the aggregate percentage is greater than 75, the grade is Distinction.
- If the aggregate percentage is greater than or equal to 60 but less than 75, the grade is First Division.
- If the aggregate percentage is greater than or equal to 50 but less than 60, the grade is Second Division.
- If the aggregate percentage is greater than or equal to 40 but less than 50, the grade is Third Division.

Input Format:
The first input will be an integer n, the number of courses.
The second input will be n integers representing the marks of the student in each of the n courses, separated by a space.

Output Format:
If the student passes all courses:

- Print the aggregate percentage (rounded to two decimal places).
- Print the grade based on the aggregate percentage.

Sample Test Cases +

Explorer

passorFa...

Submit

Debugger

```
1  n=int(input())
2  marks=list(map(int,input().split()))
3  if all(m >= 40 for m in marks):
4      agg=sum(marks)/n
5      print("Aggregate Percentage:",'%.2f'%agg)
6      if agg >= 75:
7          print("Grade: Distinction")
8      elif agg >= 60 and agg < 75:
9          print("Grade: First Division")
10     elif agg >= 50 and agg < 60:
```

Average time 0.008 s 8.25 ms

Maximum time 0.009 s 9.00 ms

2 out of 2 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 8 ms Debug

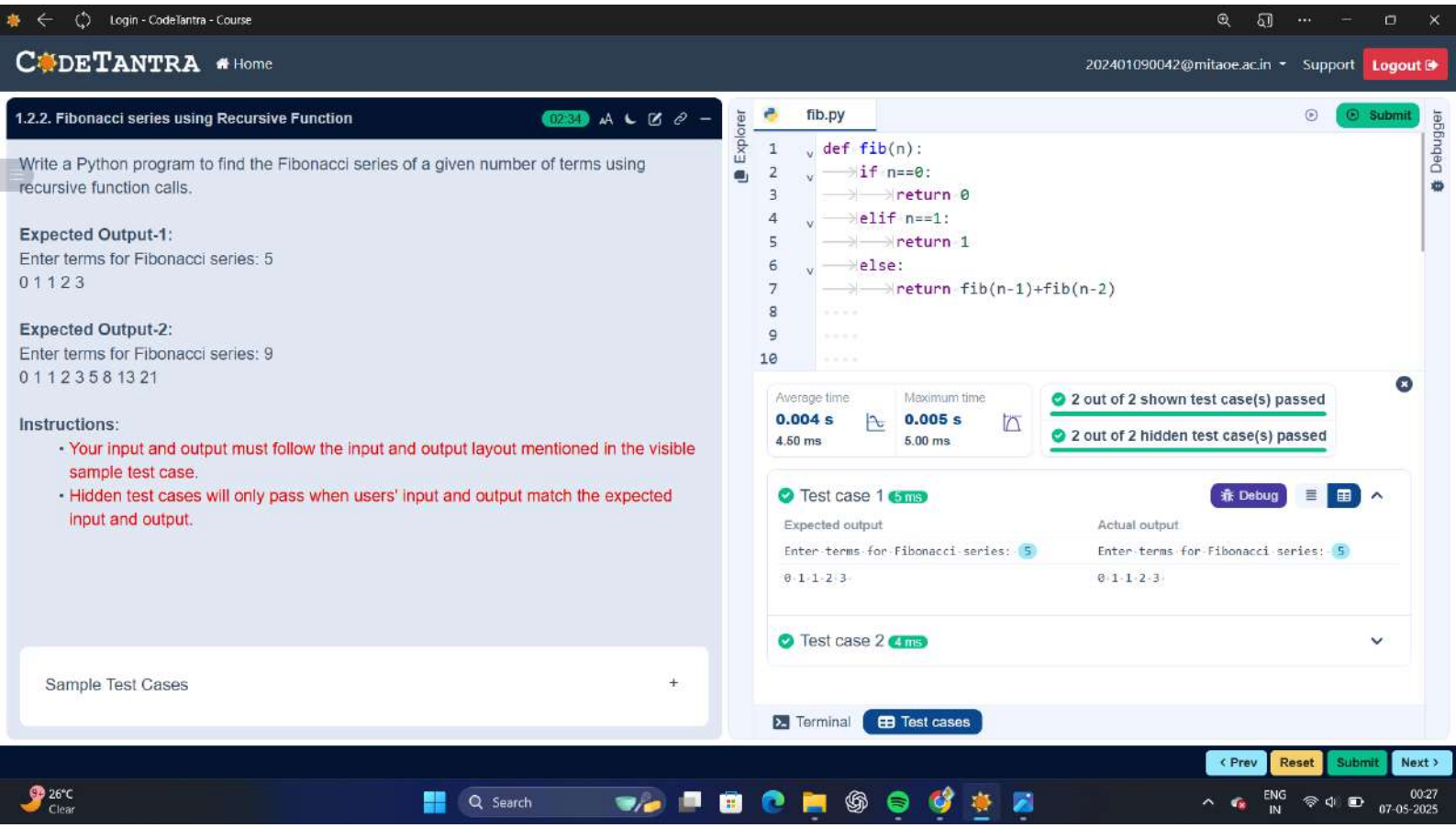
Expected output	Actual output
5	5
56 78 97 86 93	56 78 97 86 93
Aggregate Percentage: 82.00	Aggregate Percentage: 82.00
Grade: Distinction	Grade: Distinction

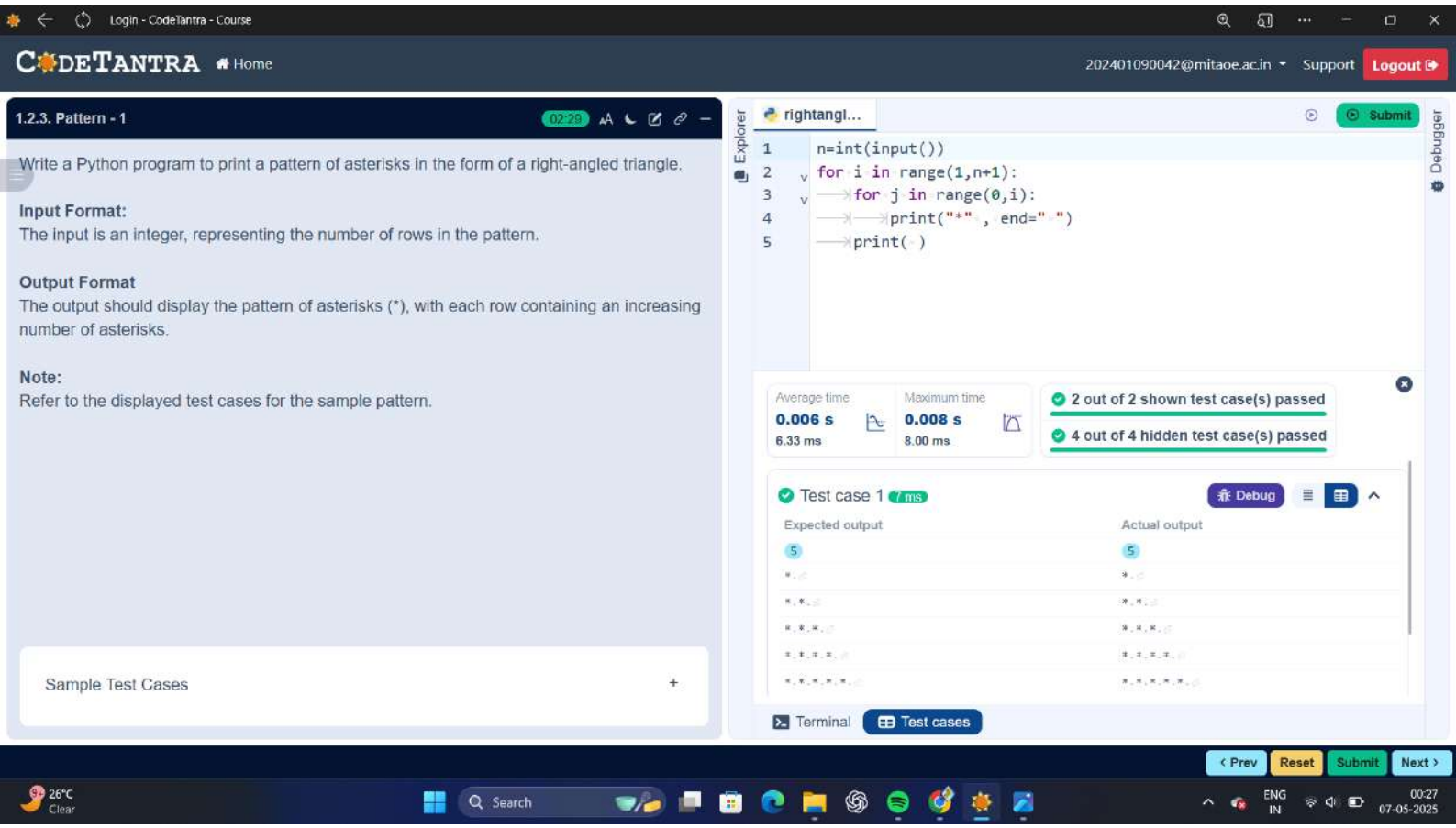
Test case 2 9 ms

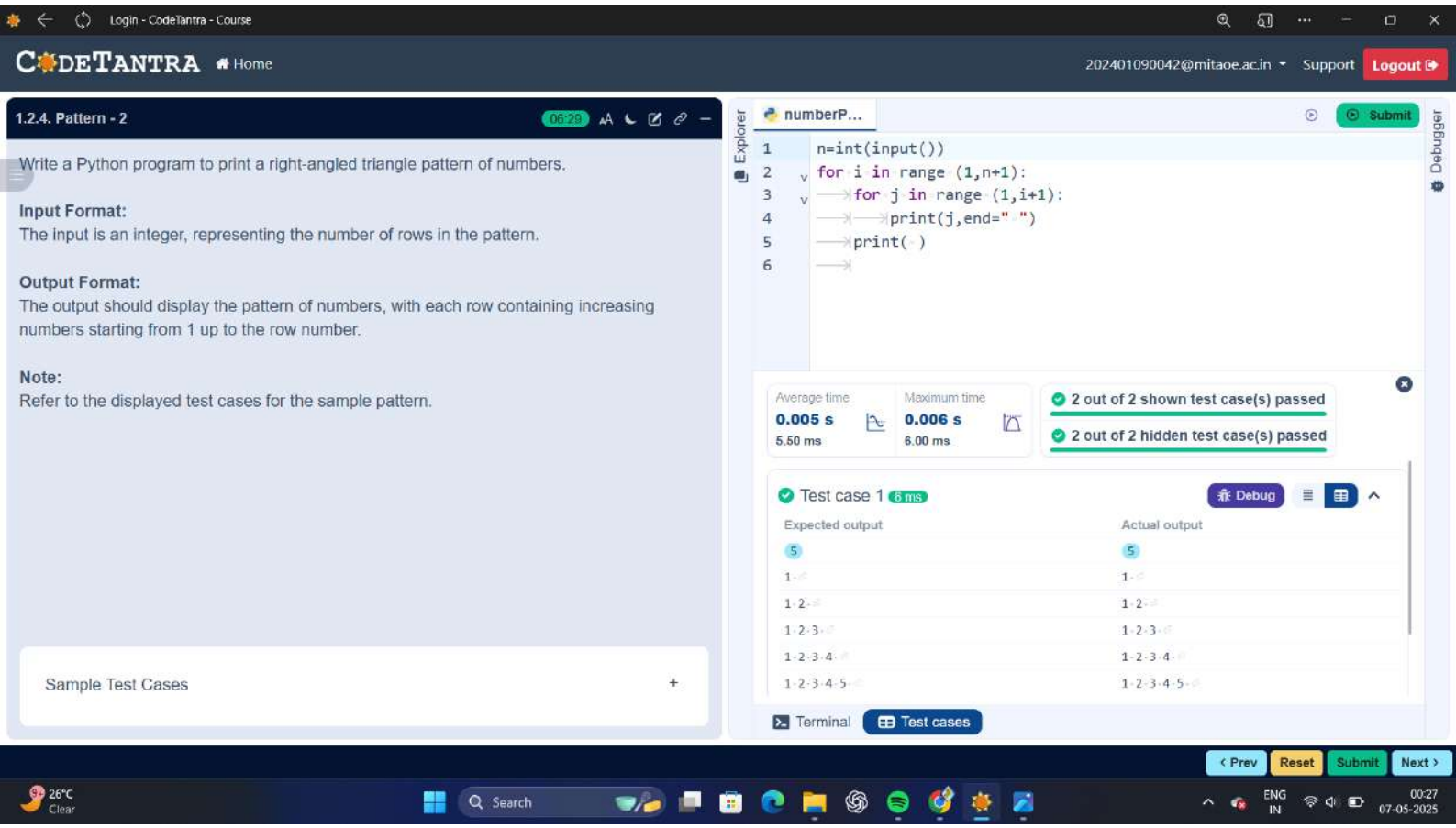
Terminal Test cases

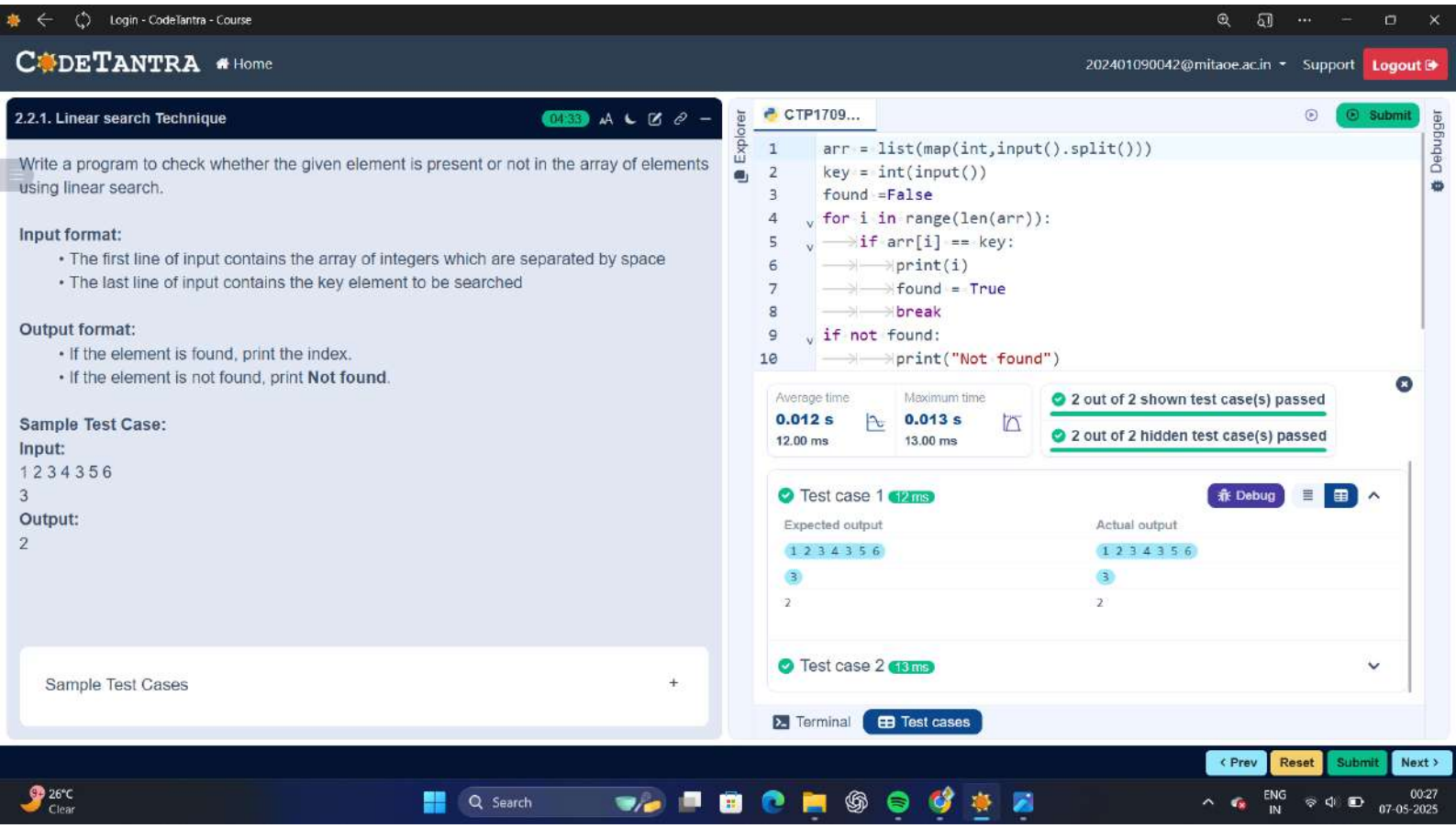
‹ Prev Reset Submit Next ›

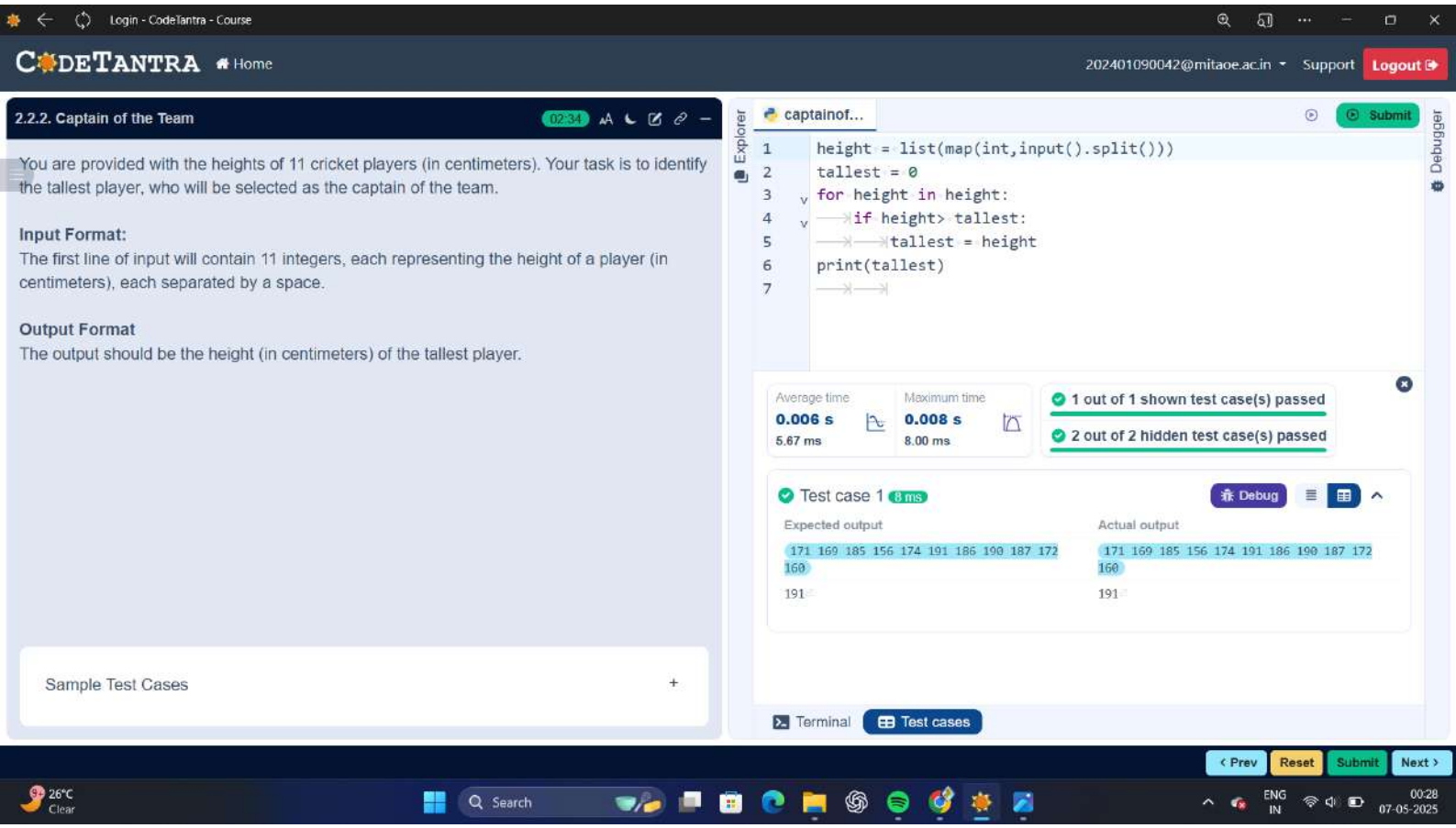
26°C Clear Search ENG IN 00:26 07-05-2025

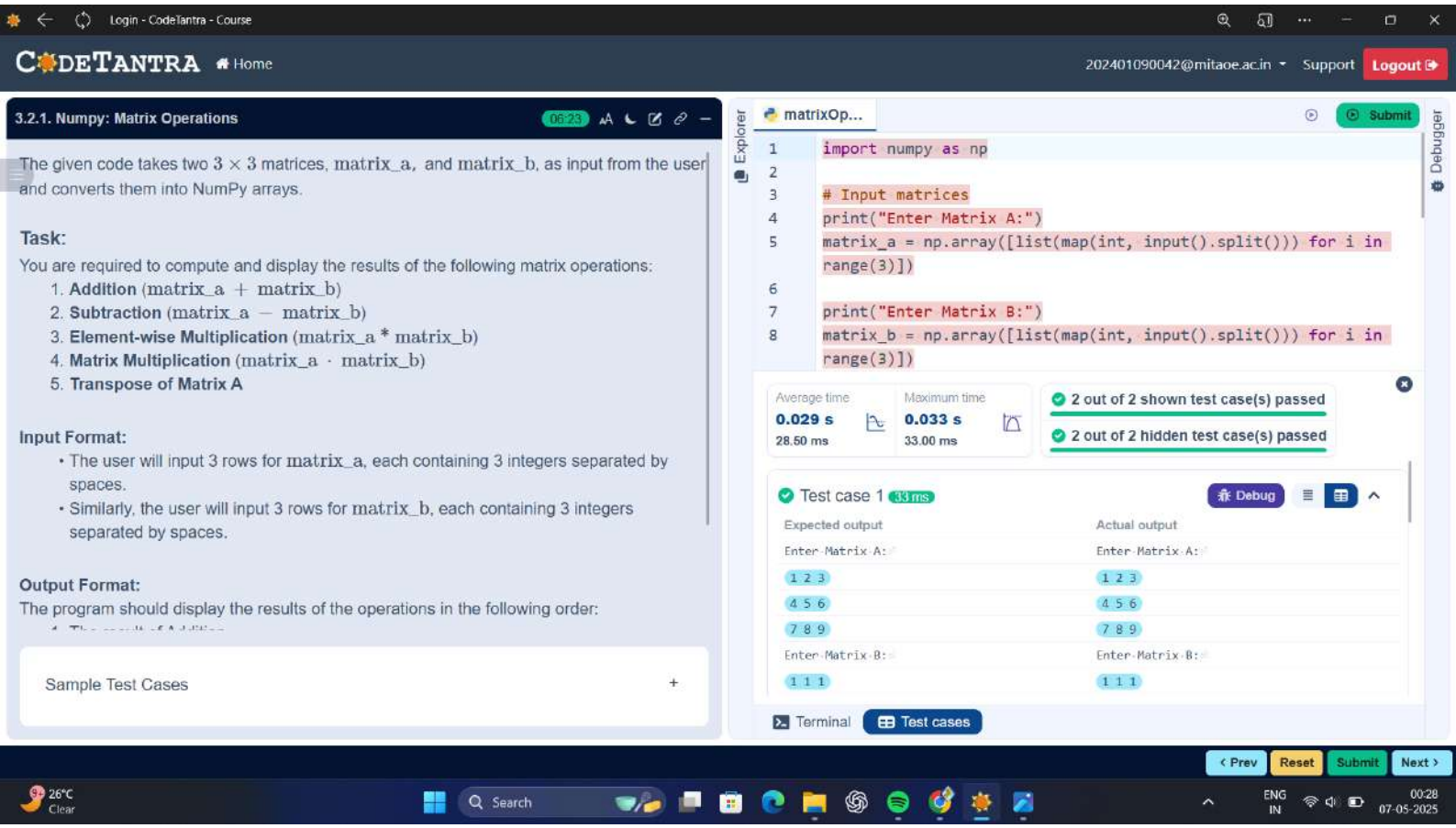

Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
1.2.2. Fibonacci series using Recursive Function
02:34
Write a Python program to find the Fibonacci series of a given number of terms using recursive function calls.
Expected Output-1:
Enter terms for Fibonacci series: 5
0 1 1 2 3
Expected Output-2:
Enter terms for Fibonacci series: 9
0 1 1 2 3 5 8 13 21
Instructions:
• Your input and output must follow the input and output layout mentioned in the visible sample test case.
• Hidden test cases will only pass when users' input and output match the expected input and output.
Sample Test Cases
Explorer
fib.py
Submit
Debugger
def fib(n):
if n==0:
return 0
elif n==1:
return 1
else:
return fib(n-1)+fib(n-2)
Average time
0.004 s
4.50 ms
Maximum time
0.005 s
5.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1 5 ms
Debug
Expected output
Enter terms for Fibonacci series: 5
0 1 1 2 3
Actual output
Enter terms for Fibonacci series: 5
0 1 1 2 3
Test case 2 4 ms
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:27
07-05-2025

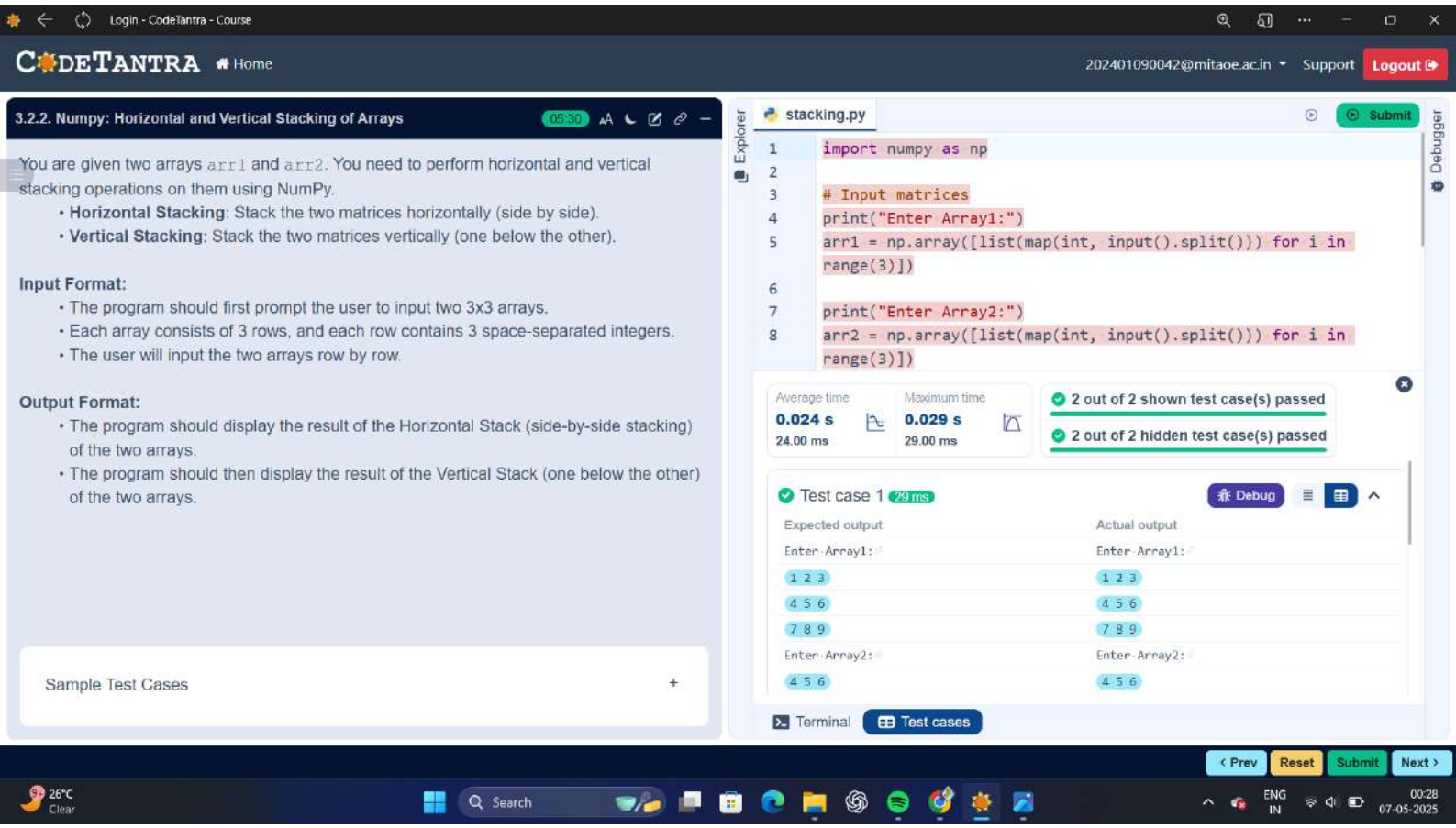

Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
1.2.3. Pattern - 1
02:20
Write a Python program to print a pattern of asterisks in the form of a right-angled triangle.
Input Format:
The input is an integer, representing the number of rows in the pattern.
Output Format
The output should display the pattern of asterisks (*), with each row containing an increasing number of asterisks.
Note:
Refer to the displayed test cases for the sample pattern.
Sample Test Cases
Explorer
rightangl...
Submit
Debugger
1 n=int(input())
2 for i in range(1,n+1):
3 for j in range(0,i):
4 print("*", end="")
5 print()
Average time
0.006 s
6.33 ms
Maximum time
0.008 s
8.00 ms
2 out of 2 shown test case(s) passed
4 out of 4 hidden test case(s) passed
Test case 1
7 ms
Debug
Expected output
Actual output
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
00:27
07-05-2025

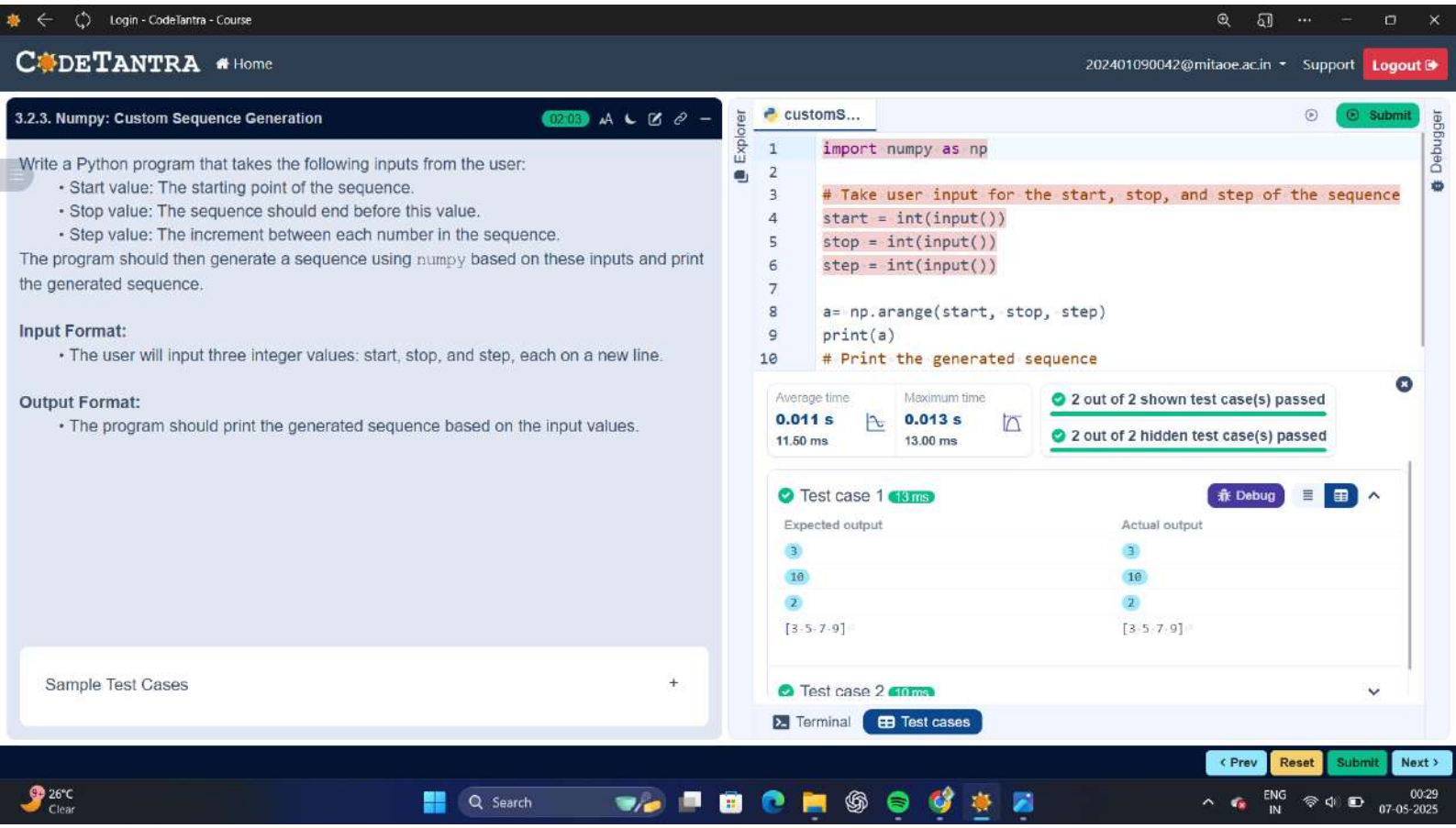

Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
1.2.4. Pattern - 2
06:20
Write a Python program to print a right-angled triangle pattern of numbers.
Input Format:
The input is an integer, representing the number of rows in the pattern.
Output Format:
The output should display the pattern of numbers, with each row containing increasing numbers starting from 1 up to the row number.
Note:
Refer to the displayed test cases for the sample pattern.
Sample Test Cases
Explorer
numberP...
Submit
Debugger
1 n=int(input())
2 for i in range (1,n+1):
3 for j in range (1,i+1):
4 print(j,end=" ")
5 print( )
6
Average time
0.005 s
5.50 ms
Maximum time
0.006 s
6.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1 6 ms
Debug
Expected output
Actual output
5
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
00:27
07-05-2025

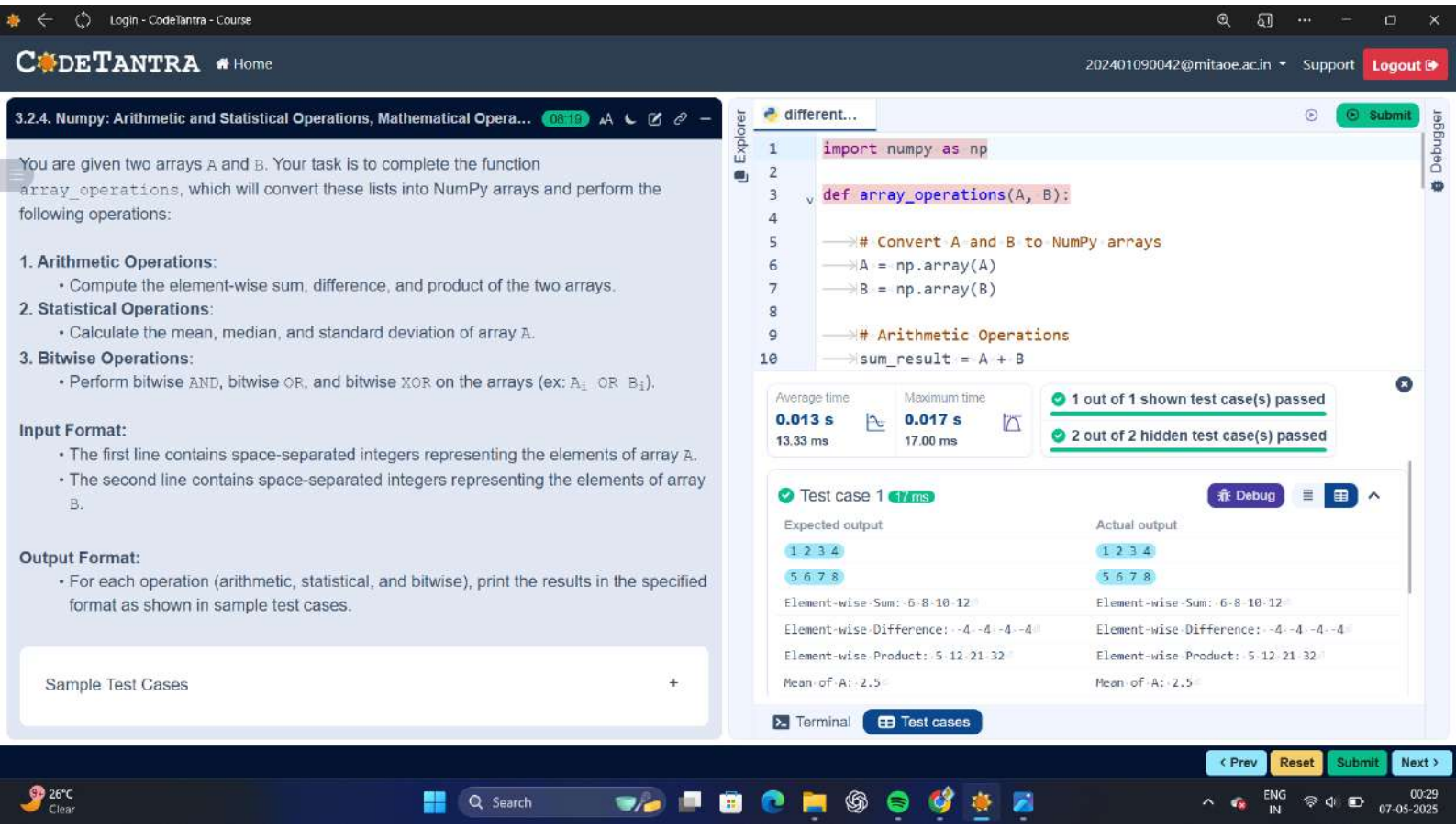

Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
2.2.1. Linear search Technique
04:33
Write a program to check whether the given element is present or not in the array of elements using linear search.
Input format:
• The first line of input contains the array of integers which are separated by space
• The last line of input contains the key element to be searched
Output format:
• If the element is found, print the index.
• If the element is not found, print Not found.
Sample Test Case:
Input:
1 2 3 4 3 5 6
3
Output:
2
Sample Test Cases
Explorer
CTP1709...
Submit
Debugger
1 arr = list(map(int,input().split()))
2 key = int(input())
3 found =False
4 for i in range(len(arr)):
5 if arr[i] == key:
6 print(i)
7 found = True
8 break
9 if not found:
10 print("Not found")
Average time
0.012 s
12.00 ms
Maximum time
0.013 s
13.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
12 ms
Debug
Expected output
1 2 3 4 3 5 6
3
2
Actual output
1 2 3 4 3 5 6
3
2
Test case 2
13 ms
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:27
07-05-2025

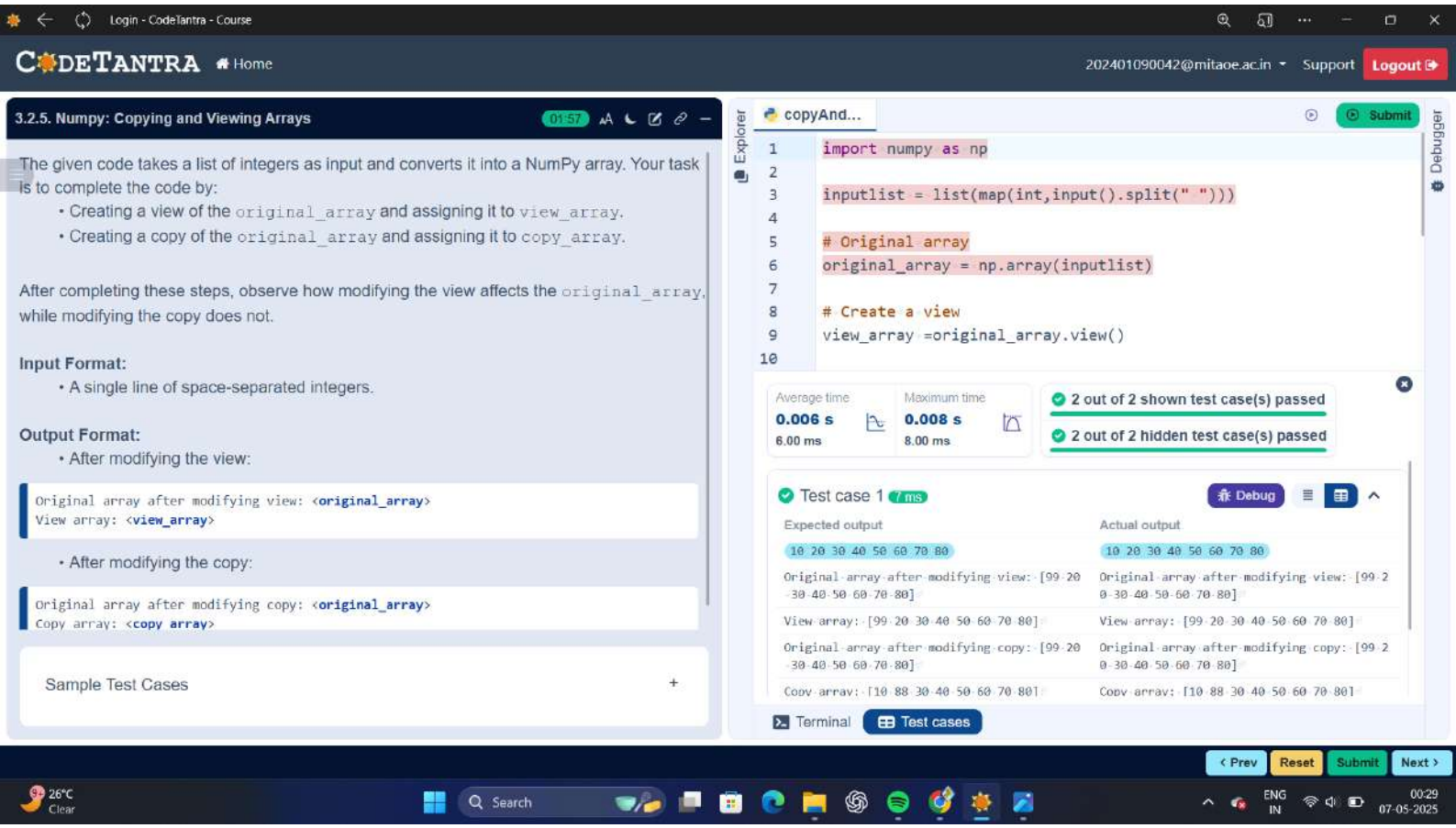

Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
2.2.2. Captain of the Team
02:34
You are provided with the heights of 11 cricket players (in centimeters). Your task is to identify the tallest player, who will be selected as the captain of the team.
Input Format:
The first line of input will contain 11 integers, each representing the height of a player (in centimeters), each separated by a space.
Output Format
The output should be the height (in centimeters) of the tallest player.
Sample Test Cases
Explorer
captainof...
Submit
Debugger
1 height = list(map(int,input().split()))
2 tallest = 0
3 for height in height:
4 if height> tallest:
5 tallest = height
6 print(tallest)
7
Average time
0.006 s
5.67 ms
Maximum time
0.008 s
8.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
8 ms
Debug
Expected output
171 169 185 156 174 191 186 190 187 172 160
191
Actual output
171 169 185 156 174 191 186 190 187 172 160
191
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:28
07-05-2025

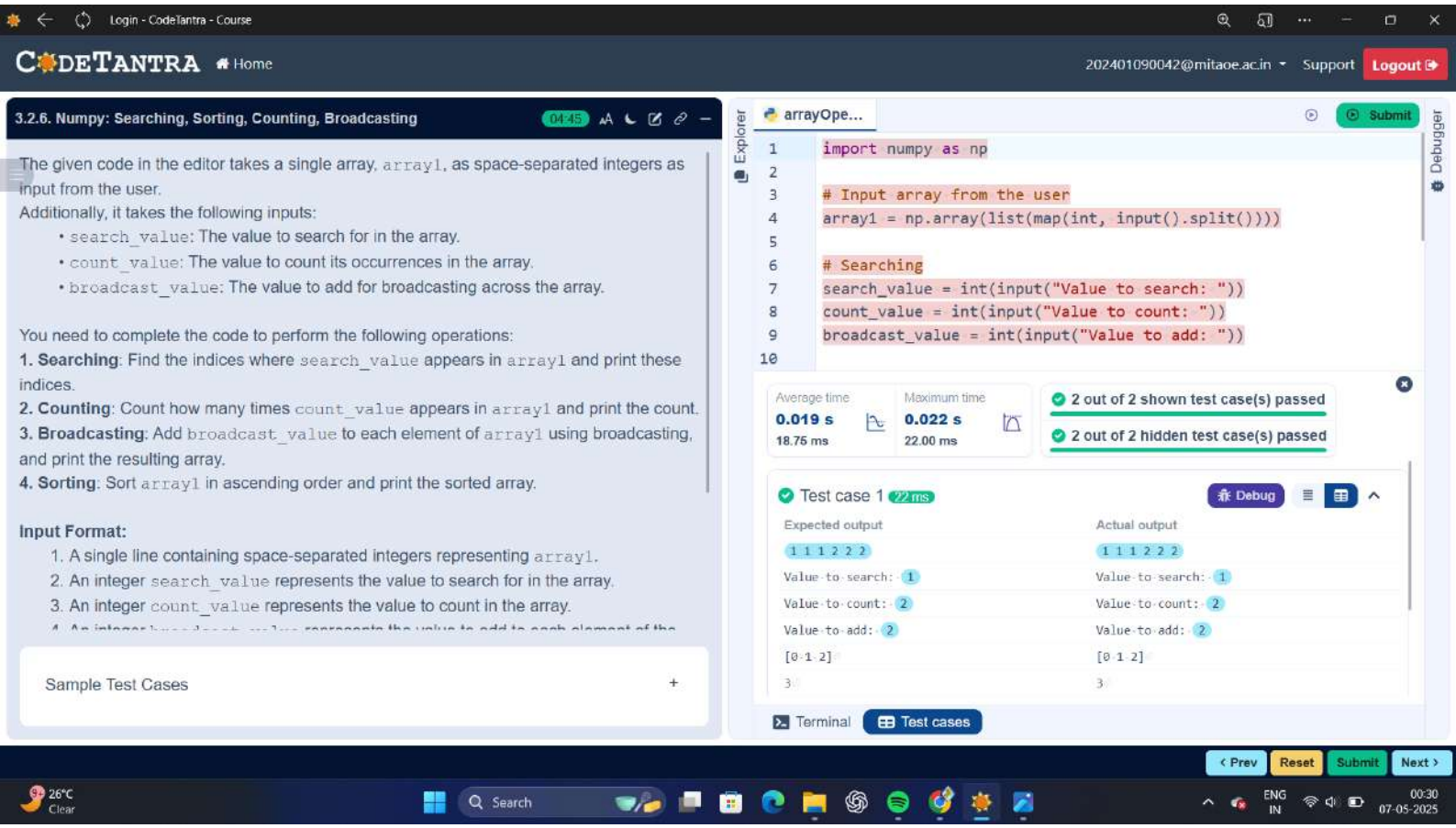

Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
3.2.1. Numpy: Matrix Operations
06:23
The given code takes two 3 × 3 matrices, matrix_a, and matrix_b, as input from the user and converts them into NumPy arrays.
Task:
You are required to compute and display the results 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:
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)])
Average time
0.029 s
28.50 ms
Maximum time
0.033 s
33.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
43 ms
Debug
Expected output
Enter Matrix A:
1 2 3
4 5 6
7 8 9
Enter Matrix B:
1 1 1
Actual output
Enter Matrix A:
1 2 3
4 5 6
7 8 9
Enter Matrix B:
1 1 1
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:28
07-05-2025

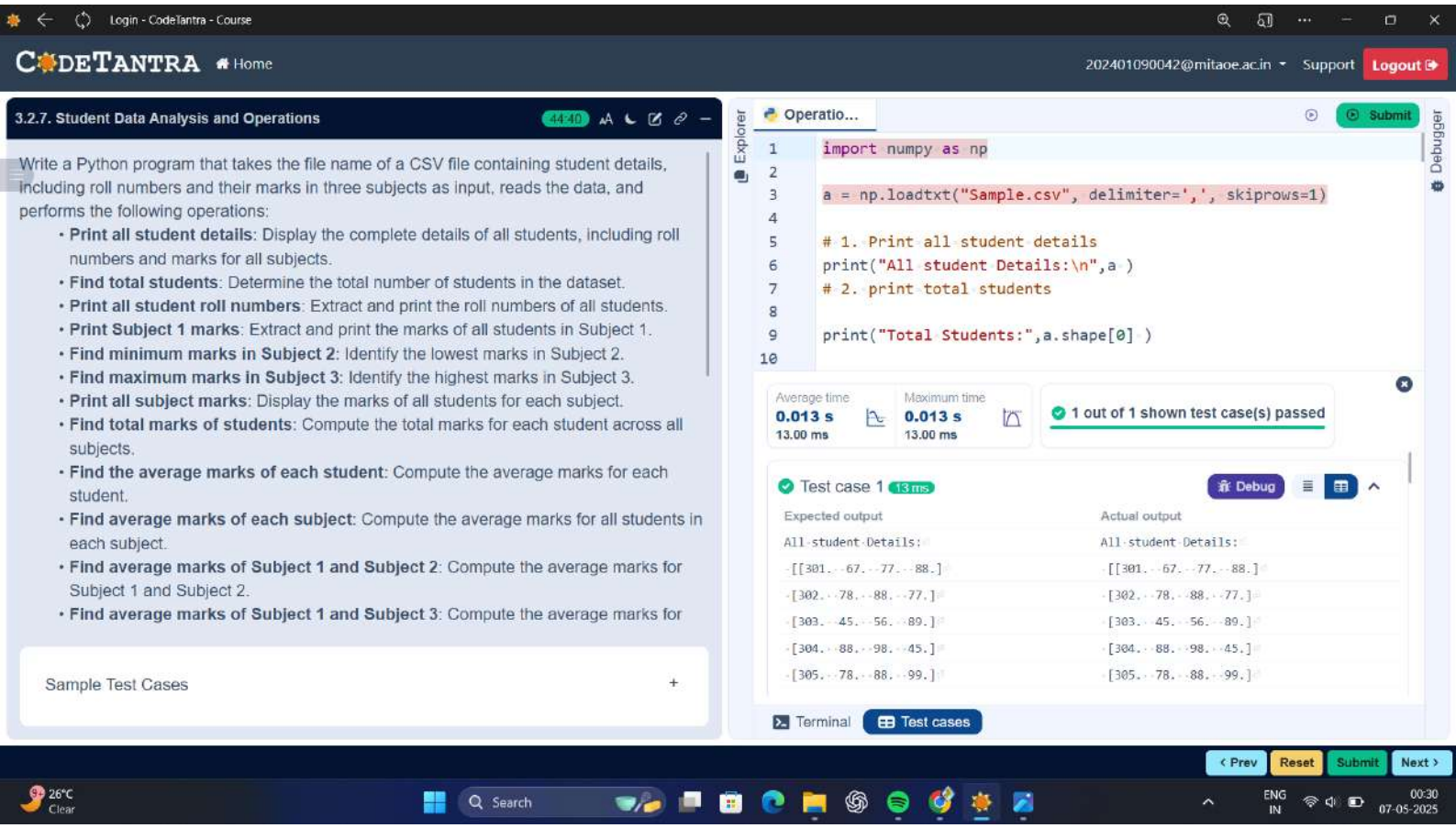

Login - Codelantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
3.2.2. Numpy: Horizontal and Vertical Stacking of Arrays
05:30
You are given two arrays arr1 and arr2. You need to perform horizontal and vertical stacking operations on them using NumPy.
• Horizontal Stacking: Stack 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
Debugger
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)])
Average time
0.024 s
24.00 ms
Maximum time
0.029 s
29.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1 29 ms
Debug
Expected output
Enter Array1:
1 2 3
4 5 6
7 8 9
Enter Array2:
4 5 6
Actual output
Enter Array1:
1 2 3
4 5 6
7 8 9
Enter Array2:
4 5 6
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:28
07-05-2025


Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
3.2.3. Numpy: Custom Sequence Generation
02:03
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
Explorer
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 a= np.arange(start, stop, step)
9 print(a)
10 # Print the generated sequence
Average time
0.011 s
11.50 ms
Maximum time
0.013 s
13.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1 13 ms
Debug
Expected output
3
10
2
[3 5 7 9]
Actual output
3
10
2
[3 5 7 9]
Test case 2 10 ms
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:29
07-05-2025


Login - Codelantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
3.2.4. Numpy: Arithmetic and Statistical Operations, Mathematical Opera...
08:10
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
Debugger
1 import numpy as np
2
3 def array_operations(A, B):
4
5 # Convert A and B to NumPy arrays
6 A = np.array(A)
7 B = np.array(B)
8
9 # Arithmetic Operations
10 sum_result = A + B
Average time
0.013 s
13.33 ms
Maximum time
0.017 s
17.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
17 ms
Debug
Expected 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
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 ›
26°C
Clear
Search
ENG
IN
00:29
07-05-2025


Login - Codelantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
3.2.5. Numpy: Copying and Viewing Arrays
01:57
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
Average time
0.006 s
6.00 ms
Maximum time
0.008 s
8.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
7 ms
Debug
Expected output
10 20 30 40 50 60 70 80
Original array after modifying view: [99 20 30 40 50 60 70 80]
View array: [99 20 30 40 50 60 70 80]
Original array after modifying copy: [99 20 30 40 50 60 70 80]
Copy array: [10 88 30 40 50 60 70 80]
Actual output
10 20 30 40 50 60 70 80
Original array after modifying view: [99 20 30 40 50 60 70 80]
View array: [99 20 30 40 50 60 70 80]
Original array after modifying copy: [99 20 30 40 50 60 70 80]
Copy array: [10 88 30 40 50 60 70 80]
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
00:29
07-05-2025


Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
3.2.6. Numpy: Searching, Sorting, Counting, Broadcasting
04:45
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.
Sample Test Cases
Explorer
arrayOpe...
Submit
Debugger
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
Average time
0.019 s
18.75 ms
Maximum time
0.022 s
22.00 ms
2 out of 2 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1 22 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]
3
Actual output
1 1 1 2 2 2
Value to search: 1
Value to count: 2
Value to add: 2
[0 1 2]
3
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:30
07-05-2025


Login - Codelantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
3.2.7. Student Data Analysis and Operations
44:40
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
Sample Test Cases
Explorer
Operatio...
Submit
Debugger
1 import numpy as np
2
3 a = np.loadtxt("Sample.csv", delimiter=',', skiprows=1)
4
5 # 1. Print all student details
6 print("All student Details:\n",a)
7 # 2. print total students
8
9 print("Total Students:",a.shape[0])
10
Average time
0.013 s
13.00 ms
Maximum time
0.013 s
13.00 ms
1 out of 1 shown test case(s) passed
Test case 1
13 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
00:30
07-05-2025

Login - Codelantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

4.2.1. Month with the Highest Total Sales

12:12

Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:

- The CSV file contains the columns: `Date`, `Product`, `Quantity`, `Price`, and `City`.
- Group the data by Month and calculate the total sales for each month.
- Find the month with the highest total sales and display it.
- Also, display the total sales for the best month.

Sample Data:

```
Date,Product,Quantity,Price,City
2025-01-01,Product A,5,20,New York
2025-01-01,Product B,3,15,Los Angeles
2025-01-02,Product A,7,20,New York
2025-01-02,Product C,4,30,Chicago
2025-01-03,Product B,2,15,Chicago
2025-01-03,Product A,8,20,Los Angeles
2025-01-04,Product C,6,30,New York
2025-01-04,Product B,5,15,Los Angeles
2025-01-05,Product A,3,20,Chicago
2025-01-05,Product C,10,30,Los Angeles
```

Sample Test Cases +

Explorer

monthFor... sales_dat...

Submit

Debugger

```
1  import pandas as pd
2
3  # Prompt the user for the file name
4  file_name = input()
5
6  # Load the data
7  df = pd.read_csv(file_name)
8  df['Date'] = pd.to_datetime(df['Date'])
9
10 #Extract the month from the Date column
```

Average time: 0.031 s (30.67 ms) Maximum time: 0.068 s (68.00 ms)

1 out of 1 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 68 ms Debug

Expected output	Actual output
sales_data.csv	sales_data.csv
Best month: 2025-01	Best month: 2025-01
Total sales: $1210.00	Total sales: $1210.00

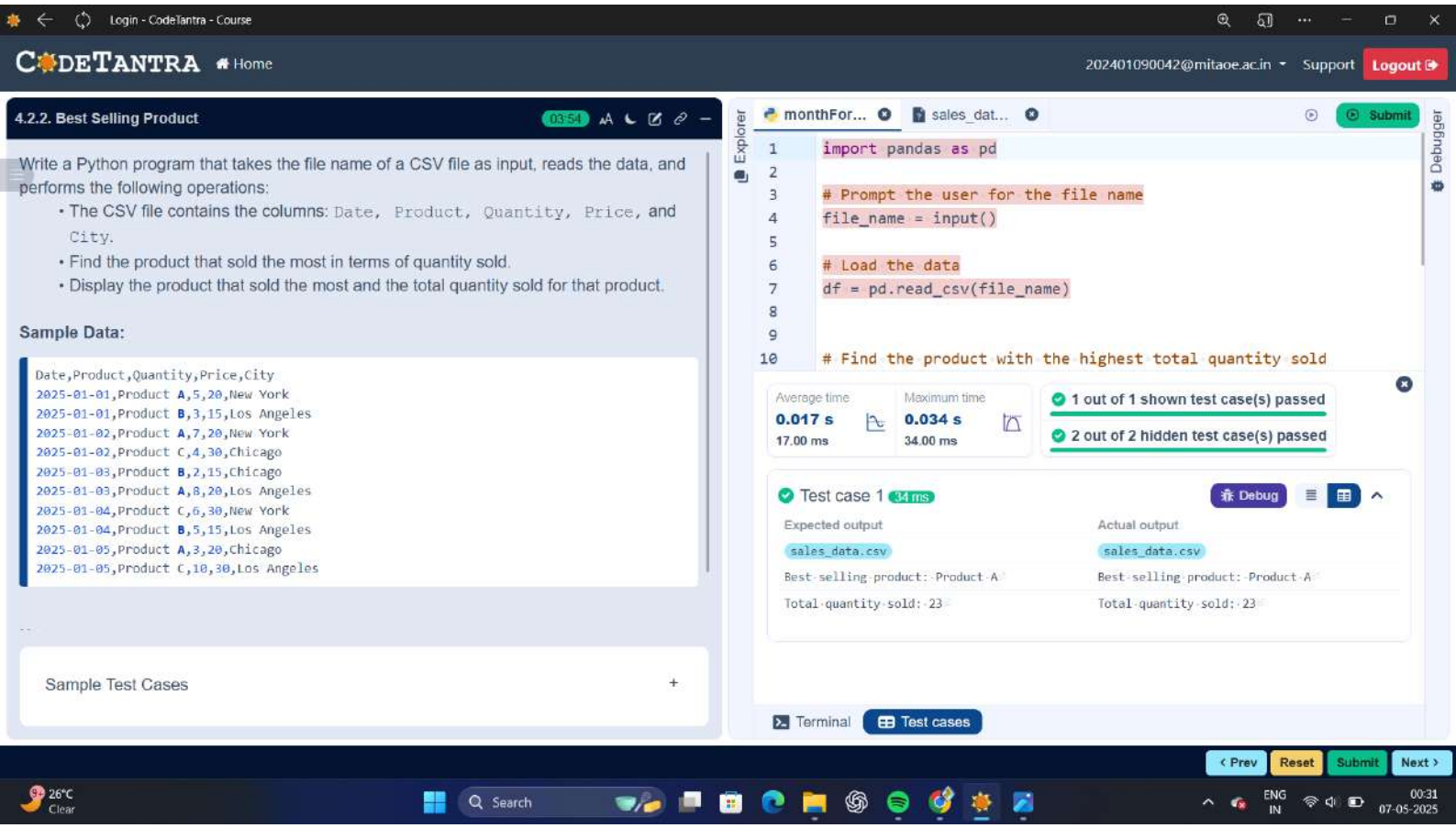
Terminal Test cases

‹ Prev Reset Submit Next ›

26°C Clear

Search

ENG IN 00:30 07-05-2025


Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
4.2.2. Best Selling Product
03:54
Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:
• The CSV file contains the columns: Date, Product, Quantity, Price, and City.
• Find the product that sold the most in terms of quantity sold.
• Display the product that sold the most and the total quantity sold for that product.
Sample Data:
Date,Product,Quantity,Price,City
2025-01-01,Product A,5,20,New York
2025-01-01,Product B,3,15,Los Angeles
2025-01-02,Product A,7,20,New York
2025-01-02,Product C,4,30,Chicago
2025-01-03,Product B,2,15,Chicago
2025-01-03,Product A,8,20,Los Angeles
2025-01-04,Product C,6,30,New York
2025-01-04,Product B,5,15,Los Angeles
2025-01-05,Product A,3,20,Chicago
2025-01-05,Product C,10,30,Los Angeles
Sample Test Cases
Explorer
monthFor...
sales_dat...
Submit
Debugger
import pandas as pd
# Prompt the user for the file name
file_name = input()
# Load the data
df = pd.read_csv(file_name)
# Find the product with the highest total quantity sold
Average time
0.017 s
17.00 ms
Maximum time
0.034 s
34.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
34 ms
Debug
Expected output
sales_data.csv
Best selling product: Product A
Total quantity sold: 23
Actual output
sales_data.csv
Best selling product: Product A
Total quantity sold: 23
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:31
07-05-2025

Login - Codelantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

4.2.3. City that Sold the Most Products

03:30

Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:

- The CSV file contains the columns: `Date`, `Product`, `Quantity`, `Price`, and `City`.
- Group the data by `City` and calculate the total quantity of products sold for each city.
- Find the city that sold the most products (based on the total quantity sold).

Sample Data:

```
Date,Product,Quantity,Price,City
2025-01-01,Product A,5,20,New York
2025-01-01,Product B,3,15,Los Angeles
2025-01-02,Product A,7,20,New York
2025-01-02,Product C,4,30,Chicago
2025-01-03,Product B,2,15,Chicago
2025-01-03,Product A,8,20,Los Angeles
2025-01-04,Product C,6,30,New York
2025-01-04,Product B,5,15,Los Angeles
2025-01-05,Product A,3,20,Chicago
2025-01-05,Product C,10,30,Los Angeles
```

Sample Test Cases +

Explorer

monthFor... sales_dat...

Submit

Debugger

```
1  import pandas as pd
2
3  # Prompt the user for the file name
4  file_name = input()
5
6  # Load the data
7  df = pd.read_csv(file_name)
8
9  # write the code..
10 city_sales = df.groupby("City")["Quantity"].sum()
```

Average time 0.016 s 16.00 ms

Maximum time 0.033 s 33.00 ms

1 out of 1 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 43 ms Debug

Expected output

sales_data.csv

City sold the most products: Los Angeles

Actual output

sales_data.csv

City sold the most products: Los Angeles

Terminal Test cases

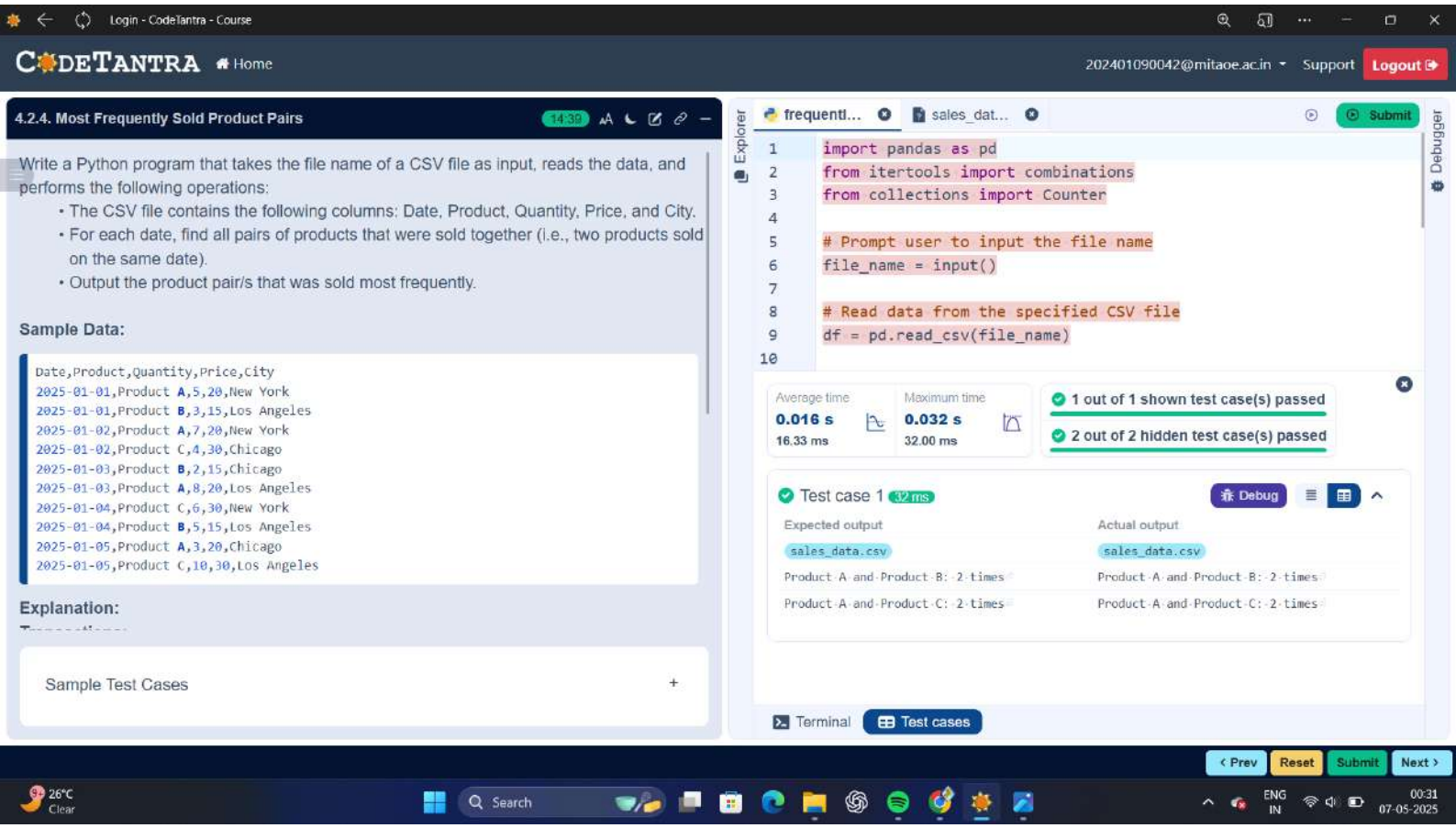
‹ Prev Reset Submit Next ›

26°C Clear

Search

ENG IN

00:31 07-05-2025


Login - Codetantra - Course
CODETANTRA
Home
202401090042@mitaoe.ac.in
Support
Logout
4.2.4. Most Frequently Sold Product Pairs
14:30
Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:
• The CSV file contains the following columns: Date, Product, Quantity, Price, and City.
• For each date, find all pairs of products that were sold together (i.e., two products sold on the same date).
• Output the product pair/s that was sold most frequently.
Sample Data:
Date,Product,Quantity,Price,City
2025-01-01,Product A,5,20,New York
2025-01-01,Product B,3,15,Los Angeles
2025-01-02,Product A,7,20,New York
2025-01-02,Product C,4,30,Chicago
2025-01-03,Product B,2,15,Chicago
2025-01-03,Product A,8,20,Los Angeles
2025-01-04,Product C,6,30,New York
2025-01-04,Product B,5,15,Los Angeles
2025-01-05,Product A,3,20,Chicago
2025-01-05,Product C,10,30,Los Angeles
Explanation:
Sample Test Cases
Explorer
frequenti...
sales_dat...
Submit
Debugger
1 import pandas as pd
2 from itertools import combinations
3 from collections import Counter
4
5 # Prompt user to input the file name
6 file_name = input()
7
8 # Read data from the specified CSV file
9 df = pd.read_csv(file_name)
10
Average time
0.016 s
16.33 ms
Maximum time
0.032 s
32.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
42 ms
Debug
Expected output
sales_data.csv
Product A and Product B: 2 times
Product A and Product C: 2 times
Actual output
sales_data.csv
Product A and Product B: 2 times
Product A and Product C: 2 times
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
00:31
07-05-2025

Login - Codelantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

4.2.4. Most Frequently Sold Product Pairs

14:30

Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:

- The CSV file contains the following columns: Date, Product, Quantity, Price, and City.
- For each date, find all pairs of products that were sold together (i.e., two products sold on the same date).
- Output the product pair/s that was sold most frequently.

Sample Data:

```
Date,Product,Quantity,Price,City
2025-01-01,Product A,5,20,New York
2025-01-01,Product B,3,15,Los Angeles
2025-01-02,Product A,7,20,New York
2025-01-02,Product C,4,30,Chicago
2025-01-03,Product B,2,15,Chicago
2025-01-03,Product A,8,20,Los Angeles
2025-01-04,Product C,6,30,New York
2025-01-04,Product B,5,15,Los Angeles
2025-01-05,Product A,3,20,Chicago
2025-01-05,Product C,10,30,Los Angeles
```

Explanation:

Sample Test Cases +

Explorer

frequenti... sales_dat... Submit

```
1  import pandas as pd
2  from itertools import combinations
3  from collections import Counter
4
5  # Prompt user to input the file name
6  file_name = input()
7
8  # Read data from the specified CSV file
9  df = pd.read_csv(file_name)
10
```

Debugger

Average time 0.016 s 16.33 ms

Maximum time 0.032 s 32.00 ms

1 out of 1 shown test case(s) passed

2 out of 2 hidden test case(s) passed

Test case 1 42 ms Debug

Expected output	Actual output
sales_data.csv	sales_data.csv
Product A and Product B: 2 times	Product A and Product B: 2 times
Product A and Product C: 2 times	Product A and Product C: 2 times

Terminal Test cases

‹ Prev Reset Submit Next ›

26°C Clear

Search

ENG IN 00:31 07-05-2025

Login - Codelantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

4.2.5. Titanic Dataset Analysis and Data Cleaning

10:14

You are provided with the Titanic dataset containing information about passengers on the Titanic. Your task is to write Python code to answer the following questions based on the dataset. For each question, perform necessary data cleaning, transformations, and calculations as required.

1. Display the first 5 rows of the dataset.
2. Display the last 5 rows of the dataset.
3. Get the shape of the dataset (number of rows and columns).
4. Get a summary of the dataset (using .info()).
5. Get basic statistics (mean, standard deviation, etc.) of the dataset using .describe().
6. Check for missing values and display the count of missing values for each column.
7. Fill missing values in the 'Age' column with the median age.
8. Fill missing values in the 'Embarked' column with the most frequent value (mode).
9. Drop the 'Cabin' column due to many missing values.
10. Create a new column, 'FamilySize' by adding the 'SibSp' and 'Parch' columns.

The Titanic dataset contains columns as shown below,

Pass	Survi	Pcl	Na	Se	Ag	Sib	Pa	Tic	Far	Ca	Embar

Sample Test Cases +

Explorer

titanicDat...

Submit

Debugger

```
1  import pandas as pd
2  import numpy as np
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # 1. Display the first 5 rows of the dataset
8  print(data.head())
9
10 # 2. Display the last 5 rows of the dataset
```

Average time 0.183 s 183.00 ms

Maximum time 0.183 s 183.00 ms

1 out of 1 shown test case(s) passed

Test case 1 183 ms Debug

Expected output

```
   PassengerId  Survived  Pclass  ...     Fare Cabin  Embarked
0            1         0       3  ...   7.2500   NaN         S
1            2         1       1  ...  71.2833   C85         C
2            3         1       3  ...   7.9250   NaN         S
```

Actual output

```
   PassengerId  Survived  Pclass  ...     Fare Cabin  Embarked
0            1         0       3  ...   7.2500   NaN         S
1            2         1       1  ...  71.2833   C85         C
2            3         1       3  ...   7.9250   NaN         S
```

Terminal Test cases

‹ Prev Reset Submit Next ›

26°C Clear

Search

ENG IN 00:32 07-05-2025

Login - Codetantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

4.2.6. Titanic Dataset Analysis and Data Cleaning - 2

36:44

You are provided with the Titanic dataset containing information about passengers on the Titanic. Your task is to write Python code to answer the following questions based on the dataset.

1. Create a new column 'IsAlone' which is 1 if the passenger is alone (FamilySize = 0), otherwise 0.
2. Convert the 'Sex' column to numeric values (male: 0, female: 1).
3. One-hot encode the 'Embarked' column, dropping the first category.
4. Get the mean age of passengers.
5. Get the median fare of passengers.
6. Get the number of passengers by class.
7. Get the number of passengers by gender.
8. Get the number of passengers by survival status.
9. Calculate the survival rate of passengers.
10. Calculate the survival rate by gender.

The Titanic dataset contains columns as shown below,

Pa ss en ger	Su rvi ve	Pcl as s	Na me	Se x	Ag e	Sib Sp	Pa rch	Tic ket	Far e	Ca bin	Em bar ke

Sample Test Cases +

Explorer

titanicDat...

Submit

Debugger

```
1  import pandas as pd
2  import numpy as np
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6  data['FamilySize'] = data['SibSp'] + data['Parch']
7
8  data['IsAlone'] = np.where(data['FamilySize'] == 0, 1, 0)
9
10 # 2. Convert 'Sex' to numeric (male: 0, female: 1)
```

Average time 0.061 s 61.00 ms

Maximum time 0.061 s 61.00 ms

1 out of 1 shown test case(s) passed

Test case 1 61 ms Debug

Expected output	Actual output
29.69911764705882	29.69911764705882
14.4542	14.4542
3 491	3 491
1 216	1 216
2 184	2 184
Name: Pclass, dtype: int64	Name: Pclass, dtype: int64

Terminal Test cases

‹ Prev Reset Submit Next ›

26°C Clear

Search

ENG IN 00:32 07-05-2025

Login - Codelantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

4.2.7. Titanic Dataset Analysis and Data Cleaning - 3

13:44

You are provided with the Titanic dataset containing information about passengers on the Titanic. Your task is to write Python code to answer the following questions based on the dataset.

1. Calculate the survival rate by class.
2. Calculate the survival rate by embarkation location (Embarked_S).
3. Calculate the survival rate by family size (FamilySize).
4. Calculate the survival rate by being alone (IsAlone).
5. Get the average fare by passenger class (Pclass).
6. Get the average age by passenger class (Pclass).
7. Get the average age by survival status (Survived).
8. Get the average fare by survival status (Survived).
9. Get the number of survivors by class (Pclass).
10. Get the number of non-survivors by class (Pclass).

The Titanic dataset contains columns as shown below,

Passenger	Survive	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarke

Sample Test Cases +

Explorer

titanicDat...

Submit

Debugger

```
1  import pandas as pd
2  import numpy as np
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6  data['FamilySize'] = data['SibSp'] + data['Parch']
7  data['IsAlone'] = np.where(data['FamilySize'] > 0, 0, 1)
8  data = pd.get_dummies(data, columns=['Embarked'],
   drop_first=True)
9
```

Average time 0.077 s 77.00 ms

Maximum time 0.077 s 77.00 ms

1 out of 1 shown test case(s) passed

Test case 1 77 ms Debug

Expected output	Actual output
Pclass	Pclass
1 0.629630	1 0.629630
2 0.472826	2 0.472826
3 0.242363	3 0.242363
Name: Survived, dtype: float64	Name: Survived, dtype: float64
Embarked_S	Embarked_S

Terminal Test cases

‹ Prev Reset Submit Next ›

26°C Clear

Search

ENG IN 00:32 07-05-2025

Login - Codelantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

4.2.8. Titanic Dataset Analysis and Data Cleaning - 4

32:16

You are provided with the Titanic dataset containing information about passengers on the Titanic. Your task is to write Python code to answer the following questions based on the dataset.

1. Get the number of survivors by gender (Sex).
2. Get the number of non-survivors by gender (Sex).
3. Get the number of survivors by embarkation location (Embarked_S).
4. Get the number of non-survivors by embarkation location (Embarked_S).
5. Calculate the percentage of children (Age < 18) who survived.
6. Calculate the percentage of adults (Age >= 18) who survived.
7. Get the median age of survivors.
8. Get the median age of non-survivors.
9. Get the median fare of survivors.
10. Get the median fare of non-survivors.

The Titanic dataset contains columns as shown below,

Passenger	Survive	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarke

Sample Test Cases +

Explorer

titanicDat...

Submit

Debugger

```
1  import pandas as pd
2  import numpy as np
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6  data = pd.get_dummies(data, columns=['Embarked'],
   drop_first=True)
7
8
9  # 1. Get the number of survivors by gender
```

Average time 0.056 s 56.00 ms

Maximum time 0.056 s 56.00 ms

1 out of 1 shown test case(s) passed

Test case 1 56 ms Debug

Expected output	Actual output
female 233	female 233
male 109	male 109
Name: Sex, dtype: int64	Name: Sex, dtype: int64
male 468	male 468
female 81	female 81
Name: Sex, dtype: int64	Name: Sex, dtype: int64

Terminal Test cases

‹ Prev Reset Submit Next ›

26°C Clear

Search

ENG IN 00:33 07-05-2025

Login - Codetantra - Course

CODETANTRA Home 202401090042@mitaoe.ac.in Support Logout

5.2.1. Titanic Dataset

44:13

Write a Python program to analyze and visualize data from the Titanic dataset based on the following instructions:

Dataset Information:

The dataset is stored in a CSV file named `titanic.csv` and has been loaded using the `pandas` library. It contains the following columns:

- `Pclass`: Passenger class (1 = First, 2 = Second, 3 = Third).
- `Gender`: Gender of the passenger (male/female).
- `Age`: Age of the passenger.
- `Survived`: Survival status (0 = Did not survive, 1 = Survived).
- `Fare`: Ticket fare paid by the passenger.

Visualization:

To represent these trends, you will create 5 visualizations using Matplotlib. The visualizations should be arranged in a 3x2 grid (3 rows and 2 columns).

Visualization Details:

Sample Test Cases +

Explorer

titanicDat...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset from the CSV file
5  df = pd.read_csv('titanic.csv')
6
7  # Set up the figure for 5 subplots
8  fig, axes = plt.subplots(3, 2, figsize=(12, 12))
9
10 # write the code..
```

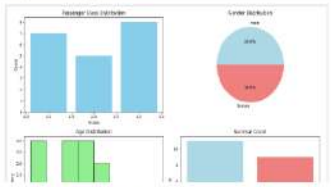
Average time 0.970 s 970.00 ms

Maximum time 0.970 s 970.00 ms

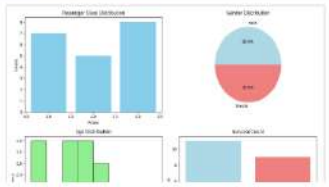
1 out of 1 shown test case(s) passed

Test case 1 970 ms Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

24°C Partly cloudy

Search

ENG IN 00:33 07-05-2025

Login - Codelantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

5.2.2. Histogram of passenger information of Titanic

03:08

Write a Python code to plot a histogram for the distribution of the 'Age' column from the Titanic dataset. The histogram should display the frequency of different age ranges with the following specifications:

1. Use **30 bins** for the histogram.
2. Set the **edge color** of the bars to **black** (k).
3. Label the x-axis as **'Age'** and the y-axis as **'Frequency'**.
4. Add the title **"Age Distribution"** to the histogram.

The Titanic dataset contains columns as shown below,

Passengerld	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Data:

-

Sample Test Cases +

Explorer

Histogra...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

Average time 0.244 s 244.00 ms

Maximum time 0.244 s 244.00 ms

1 out of 1 shown test case(s) passed

Test case 1 244 ms

Debug

Expected output

Actual output

Terminal Test cases

‹ Prev Reset Submit Next ›

24°C Partly cloudy

Search

ENG IN

00:33 07-05-2025

Login - Codetantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

5.2.3. Bar plot of survival rate of passengers

06:32

Write a Python code to plot a bar chart that shows the count of passengers who survived and did not survive in the Titanic dataset. The chart should display the following specifications:

1. Use the '**Survived**' column to show the count of survivors (0 = Did not survive, 1 = Survived).
2. Set the chart type to '**bar**'.
3. Add the title "**Survival Count**" to the chart.
4. Label the x-axis as '**Survived**' and the y-axis as '**Count**'.

The Titanic dataset contains columns as shown below,

Passengerld	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Data:

-

Sample Test Cases +

Explorer

BarPlotOf...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

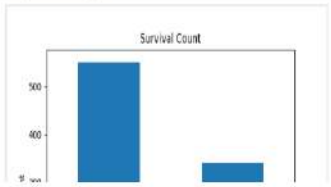
Average time 0.289 s 289.00 ms

Maximum time 0.289 s 289.00 ms

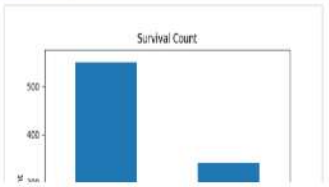
1 out of 1 shown test case(s) passed

Test case 1 289 ms Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

24°C Partly cloudy

Search

ENG IN

00:34 07-05-2025

Login - Codetantra - Course

CODETANTRA Home 202401090042@mitaoe.ac.in Support Logout

5.2.4. Bar Plot for Survival by Gender

08:01

Write a Python code to plot a stacked bar chart that shows the count of passengers who survived and did not survive, grouped by gender, in the Titanic dataset. The chart should display the following specifications:

1. Group the data by the **'Sex'** column, then use the **value_counts()** function to count the occurrences of survivors (0 = Did not survive, 1 = Survived) for each gender.
2. Use a **stacked bar chart** to display the survival counts.
3. Add the title **"Survival by Gender"** to the chart.
4. Label the x-axis as **'Gender'** and the y-axis as **'Count'**.
5. The legend should indicate **'Not Survived'** and **'Survived'**.

The Titanic dataset contains columns as shown below,

Passengerld	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Test Cases +

Explorer

BarPlotOf...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

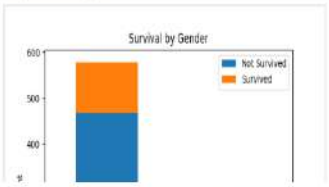
Average time 0.312 s 312.00 ms

Maximum time 0.312 s 312.00 ms

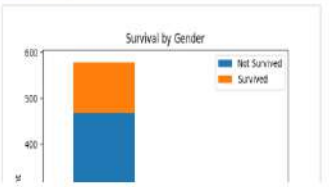
1 out of 1 shown test case(s) passed

Test case 1 312 ms Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

Gold +2.98%

Search

ENG IN 00:34 07-05-2025

Login - Codetantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

5.2.5. Bar Plot for Survival by Pclass

06:44

Write a Python code to plot a stacked bar chart that shows the count of passengers who survived and did not survive, grouped by passenger class (**Pclass**), in the Titanic dataset. The chart should display the following specifications:

1. Group the data by the **Pclass** column and count the number of survivors (0 = Did not survive, 1 = Survived) for each class using **value_counts()**.
2. Use a **stacked bar chart** to display the survival counts.
3. Add the title **"Survival by Pclass"** to the chart.
4. Label the x-axis as **'Pclass'** and the y-axis as **'Count'**.
5. The legend should indicate **'Not Survived'** and **'Survived'**.

The Titanic dataset contains columns as shown below,

Passengerld	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Test Cases +

Explorer

BarPlotOf...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

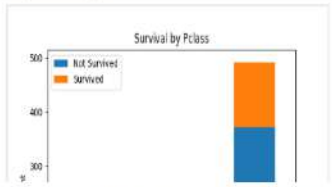
Average time 0.349 s 349.00 ms

Maximum time 0.349 s 349.00 ms

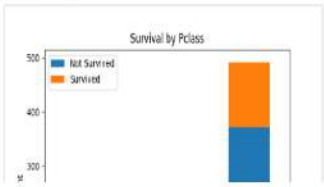
1 out of 1 shown test case(s) passed

Test case 1 349 ms Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

Gold +2.98%

Search

ENG IN 00:34 07-05-2025

Login - Codetantra - Course

CODETANTRA Home 202401090042@mitaoe.ac.in Support Logout

5.2.6. Bar Plot for Survival by Embarked

05:25

Write a Python code to plot a stacked bar chart showing the survival count for passengers based on their embarkation location in the Titanic dataset.
The chart should display the following specifications:

1. Use the **Embarked** column to determine the embarkation location. After converting this column into dummy variables (using **pd.get_dummies()**), plot the survival count based on the **Embarked_Q** column (representing passengers who embarked from Queenstown) in relation to survival.
2. Set the chart type to 'bar' and make it stacked.
3. Add the title **"Survival by Embarked "** to the chart.
4. Label the x-axis as **'Embarked'** and the y-axis as **'Count'**.
5. Include a legend to distinguish between survivors and non-survivors (label the legend as **'Survived'** and **'Not Survived'**).

The Titanic dataset contains columns as shown below,

Passengerld	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Test Cases +

Explorer

BarPlotOf...

Submit

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

Debugger

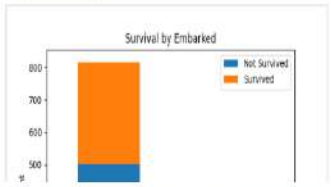
Average time 0.317 s 317.00 ms

Maximum time 0.317 s 317.00 ms

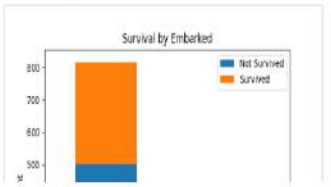
1 out of 1 shown test case(s) passed

Test case 1 317 ms Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

Gold +2.98%

Search

ENG IN 00:35 07-05-2025

Login - Codetantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

5.2.7. Box plot for Age Distribution

03:35

Write a Python code to plot a boxplot that shows the distribution of the 'Age' column from the Titanic dataset across different passenger classes. The boxplot should display the following specifications:

1. Use the **Pclass** column to group the data for the boxplot.
2. Set the title of the plot to **"Age by Pclass"**.
3. Remove the default subtitle with **plt.suptitle('')**.
4. Label the x-axis as **'Pclass'** and the y-axis as **'Age'**.

The Titanic dataset contains columns as shown below,

Passengerld	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Data:

Sample Test Cases +

Explorer

BoxPlotF...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

Average time 0.259 s 259.00 ms

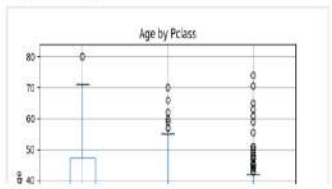
Maximum time 0.259 s 259.00 ms

1 out of 1 shown test case(s) passed

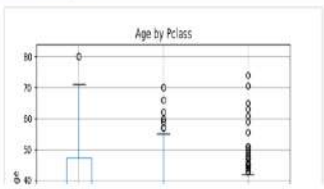
Test case 1 259 ms

Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

Gold +2.98%

Search

ENG IN 00:35 07-05-2025

Login - Codetantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

5.2.8. Box Plot for Age by Survived

09:18

Write a Python code to plot a boxplot that shows the distribution of the 'Age' column from the Titanic dataset based on whether passengers survived or not. The boxplot should display the following specifications:

1. Use the **Survived** column to group the data for the boxplot (0 = Did not survive, 1 = Survived).
2. Set the title of the plot to **"Age by Survival"**.
3. Remove the default subtitle with **plt.suptitle('')**.
4. Label the x-axis as **'Survived'** and the y-axis as **'Age'**.

The Titanic dataset contains columns as shown below,

Passenger Id	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Data:

Sample Test Cases +

Explorer

BoxPlotF...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

Average time 0.271 s 271.00 ms

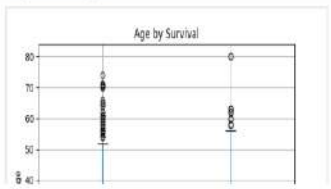
Maximum time 0.271 s 271.00 ms

1 out of 1 shown test case(s) passed

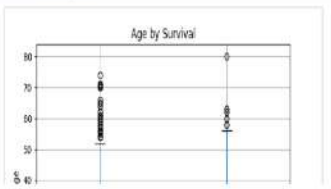
Test case 1 271 ms

Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

Gold +2.98%

Search

ENG IN 00:35 07-05-2025

Login - Codetantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

5.2.9. Box Plot for Fare by Pclass

03:24

Write a Python code to plot a boxplot that shows the distribution of the 'Fare' column from the Titanic dataset based on the passenger class (Pclass). The boxplot should display the following specifications:

1. Use the **Pclass** column to group the data for the boxplot.
2. Set the title of the plot to **"Fare by Pclass"**.
3. Remove the default subtitle with **plt.suptitle('')**.
4. Label the x-axis as **'Pclass'** and the y-axis as **'Fare'**.

The Titanic dataset contains columns as shown below,

Passengerld	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Data:

-

Sample Test Cases +

Explorer

BoxPlotF...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

Average time 0.254 s 254.00 ms

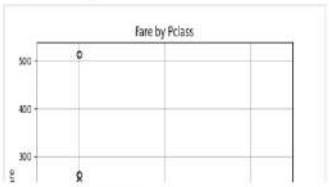
Maximum time 0.254 s 254.00 ms

1 out of 1 shown test case(s) passed

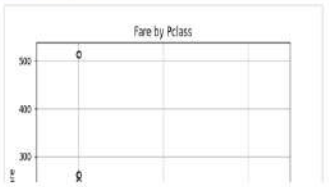
Test case 1 254 ms

Debug

Expected output



Actual output



Terminal Test cases

‹ Prev Reset Submit Next ›

Gold +2.98%

Search

ENG IN 00:36 07-05-2025

Login - Codetantra - Course

CODETANTRA Home

202401090042@mitaoe.ac.in Support Logout

5.2.10. Scatter Plot for Age vs. Fare

05:02

Write a Python code to plot a scatter plot showing the relationship between the 'Age' and 'Fare' columns in the Titanic dataset. The scatter plot should display the following specifications:

1. Use the **Age** column for the x-axis and the **Fare** column for the y-axis.
2. Set the title of the plot to **"Age vs. Fare"**.
3. Label the x-axis as **'Age'** and the y-axis as **'Fare'**.

The Titanic dataset contains columns as shown below,

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Data:

```
PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked
```

Sample Test Cases +

Explorer

AgeFareS...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

Average time 0.224 s 224.00 ms

Maximum time 0.224 s 224.00 ms

1 out of 1 shown test case(s) passed

Test case 1 224 ms Debug

Expected output

Actual output

Terminal Test cases

‹ Prev Reset Submit Next ›

Gold +2.98%

Search

ENG IN 00:36 07-05-2025

Login - Codetantra - Course

CODETANTRA Home 202401090042@mitaoe.ac.in Support Logout

5.2.11. Scatter Plot for Age vs. Fare by Survived

05:54

Write a Python code to plot a scatter plot showing the relationship between the 'Age' and 'Fare' columns in the Titanic dataset, with points color-coded by survival status. The scatter plot should display the following specifications:

1. Use the **Age** column for the x-axis and the **Fare** column for the y-axis.
2. Color the points based on the **Survived** column: **Red** for passengers who did not survive (**Survived = 0**). **Blue** for passengers who survived (**Survived = 1**).
3. Set the title of the plot to **"Age vs. Fare by Survival"**.
4. Label the x-axis as **'Age'** and the y-axis as **'Fare'**.

The Titanic dataset contains columns as shown below,

Passenger Id	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

Sample Data:

Sample Test Cases +

Explorer

AgeFareS...

Submit

Debugger

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median(), inplace=True)
9  data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
10 data.drop('Cabin', axis=1, inplace=True)
```

Average time 0.241 s 241.00 ms

Maximum time 0.241 s 241.00 ms

1 out of 1 shown test case(s) passed

Test case 1 241 ms Debug

Expected output

Actual output

Terminal Test cases

‹ Prev Reset Submit

Gold +2.98%

Search

ENG IN 00:36 07-05-2025