EDS Laboratory Assignments

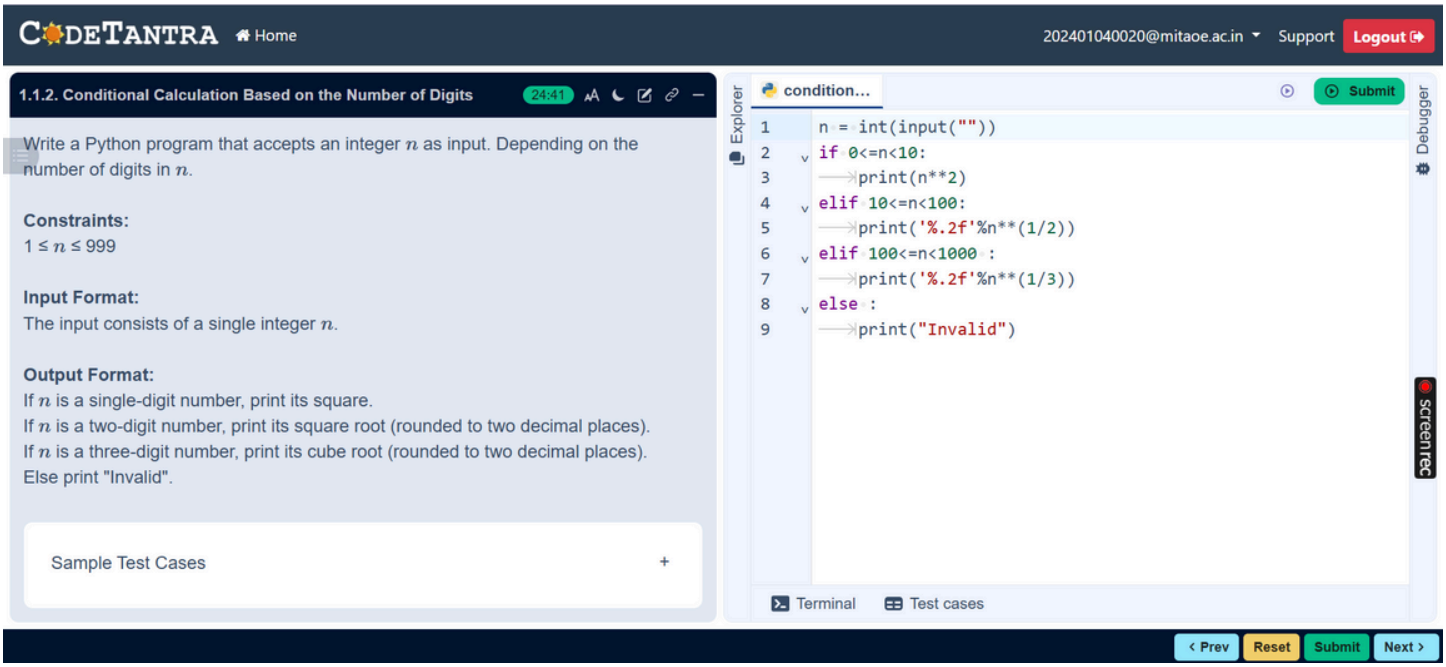
Name : Shreya Abhay Bhagat

Division : CS3

Roll No. : 83

PRN : 202401040020

Practical 1 Lab Assignment 1 :



CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

1.1.3. Age and Salary Calculation

29:43

Write a Python program that reads the birth date and salary of employees.

Input Format:
The input consists of:
A string representing the birth date of the employee in the format $DD-MM-YYYY$.
A floating-point number representing the salary of the employee in rupees.

Output Format:
The output should include:
The age of the employee.
The salary of the employee in dollars.

Note:
1INR=0.012USD

Sample Test Cases +

Explorer | birthDate... | Submit | Debugger

```python
from datetime import datetime

def calculate_age(birthdate):
    date_object = datetime.strptime(birthdate, "%d-%m-%Y")
    today = datetime.today()
    if (today.month, today.day) <(date_object.month, date_object.day):
        age = today.year-date_object.year- ((today.month, today.day) <(date_object.month, date_object.day))
        return age
    elif((today.month, today.day) > (date_object.month, date_object.day)):
        age = today.year-date_object.year- ((today.month, today.day) > (date_object.month, date_object.day))
        return age


def convert_salary_to_dollars(salary_in_rupees):
```

Terminal | Test cases

screenrec

‹ Prev | Reset | Submit | Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

1.1.3. Age and Salary Calculation 29:43

Write a Python program that reads the birth date and salary of employees.

Input Format:
The input consists of:
A string representing the birth date of the employee in the format
$DD-MM-YYYY$.
A floating-point number representing the salary of the employee in rupees.

Output Format:
The output should include:
The age of the employee.
The salary of the employee in dollars.

Note:
1INR=0.012USD

Sample Test Cases +

Explorer | birthDate... | Submit | Debugger

```python
    today.day) > (date_object.month, date_object.day))
        return age


def convert_salary_to_dollars(salary_in_rupees):
    salary_in_dollars = salary_in_rupees*0.012
    return salary_in_dollars


birthdate = input()
salary_in_rupees = float(input())
age = calculate_age(birthdate)
salary_in_dollars = convert_salary_to_dollars(salary_in_rupees)
print(f"Age: {age}")
print(f"Salary in dollars: {salary_in_dollars:.2f}")
```

Terminal | Test cases

screenrec

‹ Prev | Reset | Submit | Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

1.1.4. Reverse a Number

07:46

You are given an integer number. Your task is to reverse the digits of the number and print the reversed number.

Input Format
The input is an integer.

Output Format
Print a single integer which is the reversed number.

Sample Test Cases +

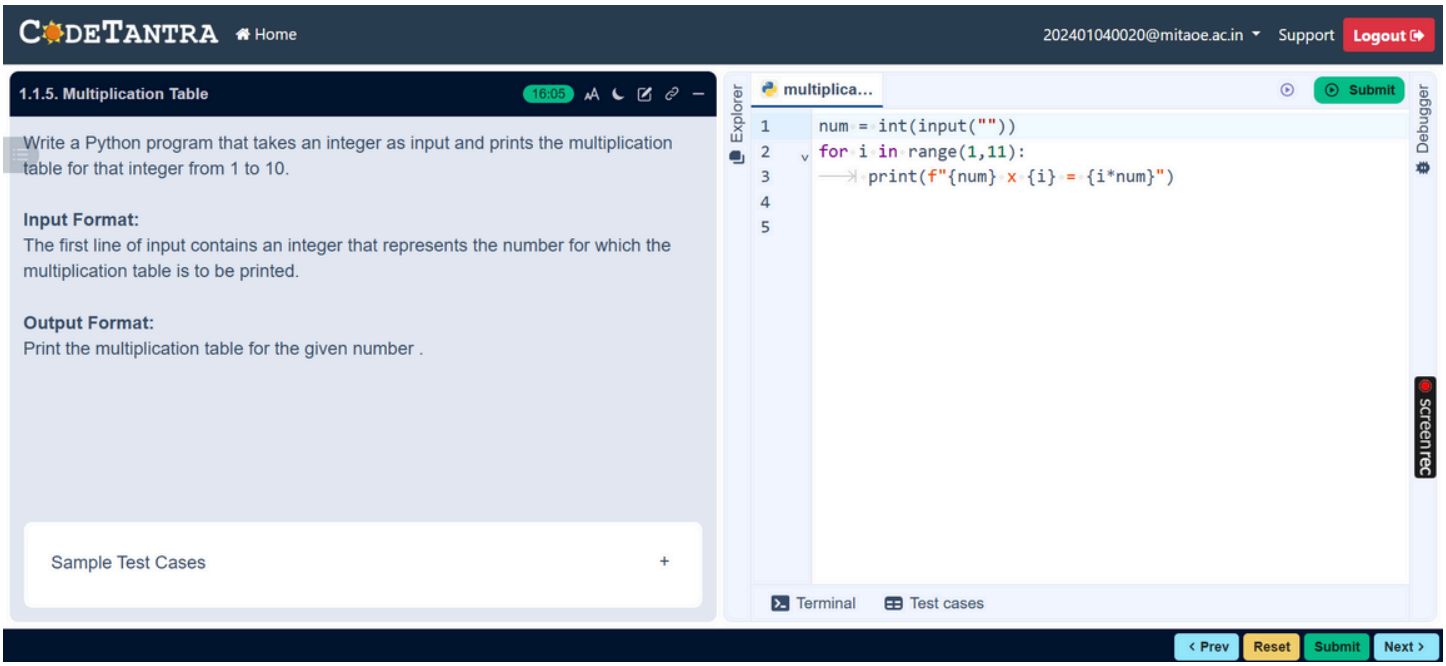
Explorer | reverseN... | Submit | Debugger

```python
n = int(input(""))
sum = 0
while(n>0):
    rem = n%10
    sum = sum*10 + rem
    n = n//10
print(sum)
```

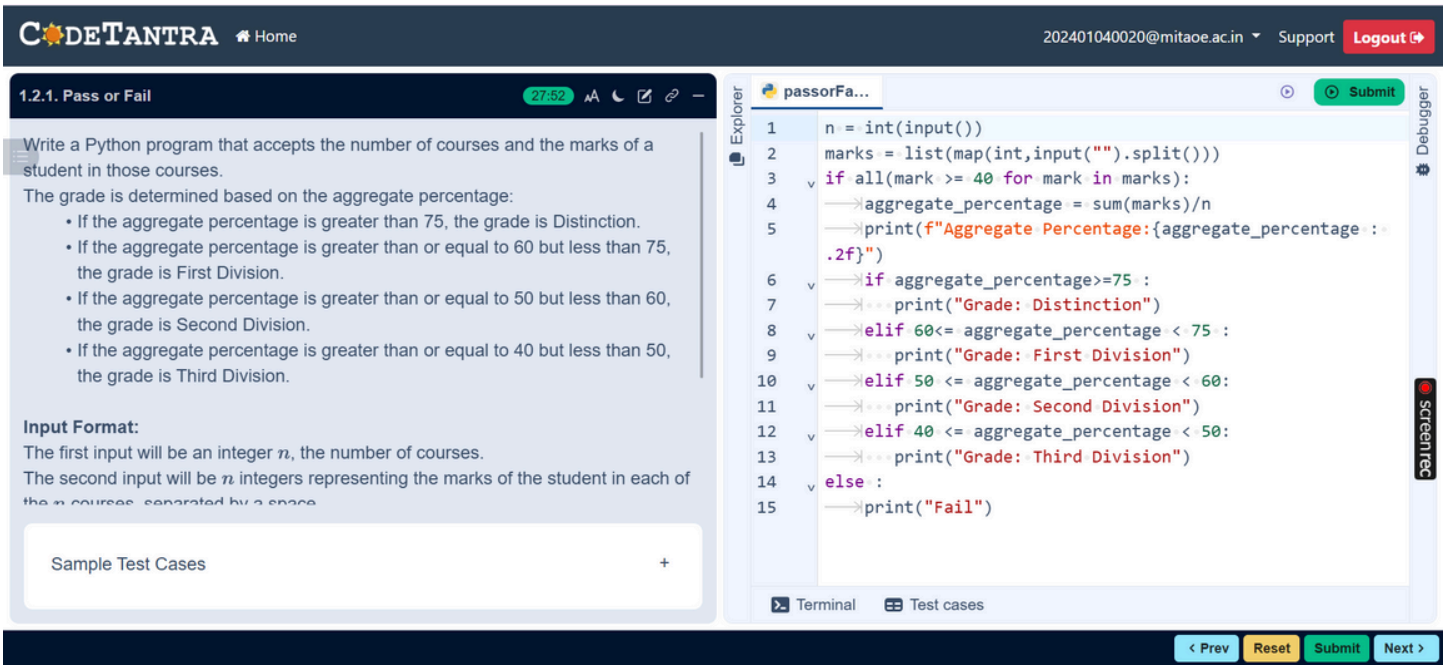
Terminal | Test cases

screenrec

‹ Prev | Reset | Submit | Next ›



Practical 1 Lab Assignment 2 :



CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

1.2.2. Fibonacci series using Recursive Function

04:14

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

Sample Test Cases +

Explorer | fib.py | Submit | Debugger

```
def fib(n):
    if(n<=1):
        return(n)
    else:
        return fib(n-1) + fib(n-2)



n=int(input("Enter terms for Fibonacci series: "))
for i in range (n):
    print(fib(i),end=" ")
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

1.2.3. Pattern - 1

25:50

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

```
r = int(input())
for i in range(r):
    for j in range(i+1):
        print("* ", end="")
    print()
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

1.2.4. Pattern - 2

01:30

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

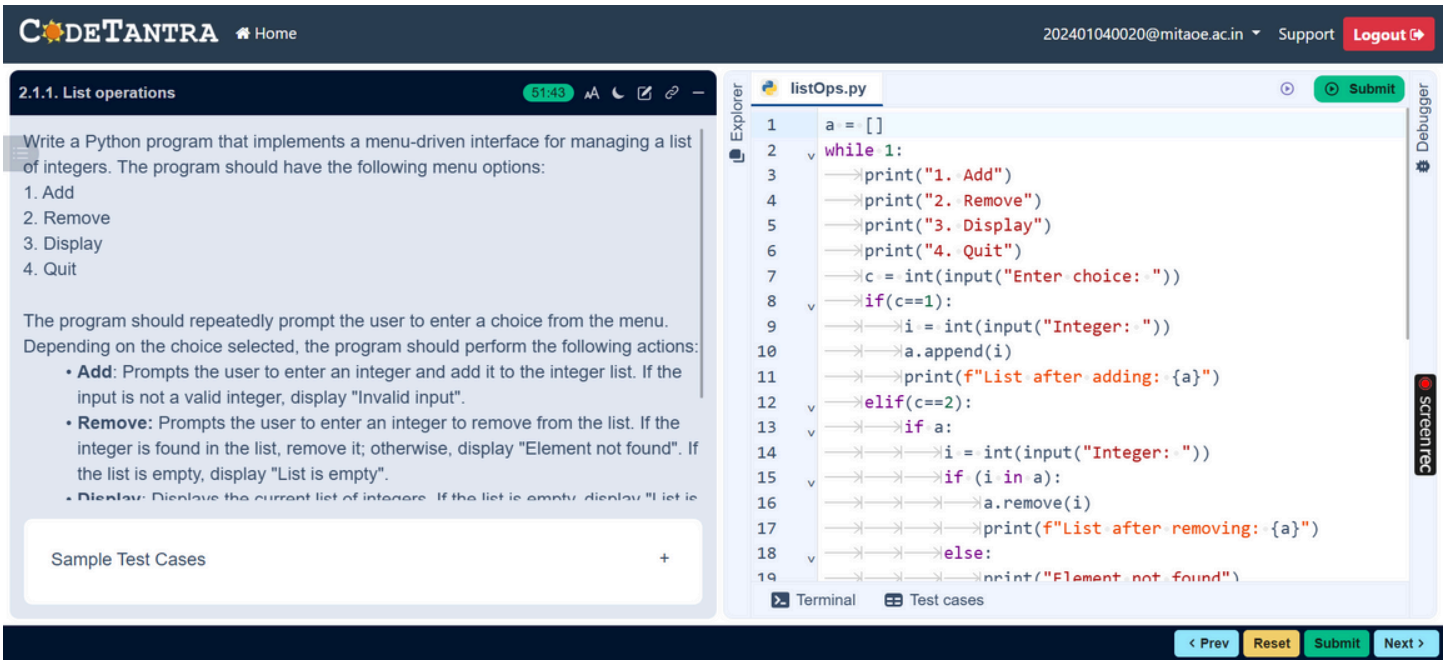
```
r = int(input())
for i in range(r):
    r = 1
    for j in range(i+1):
        print(r , end=" ")
        r=r+1
    print()
```

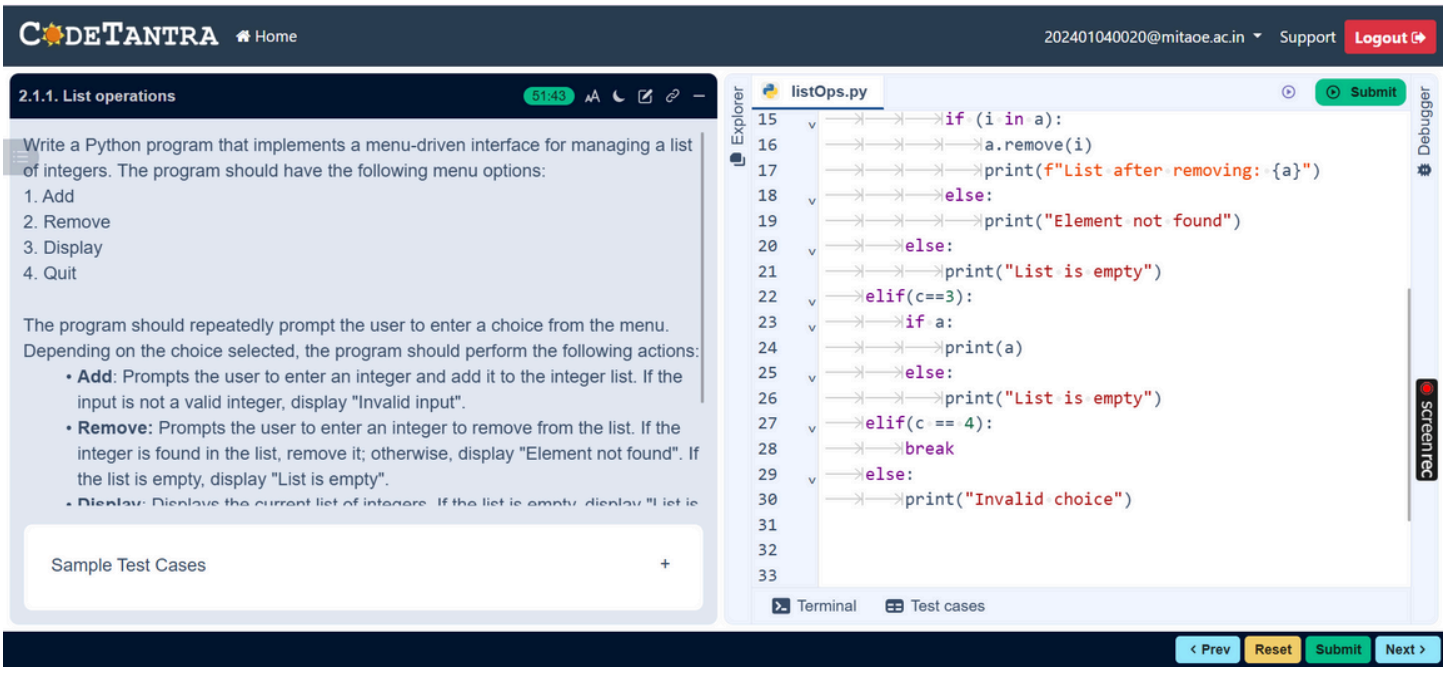
Terminal | Test cases

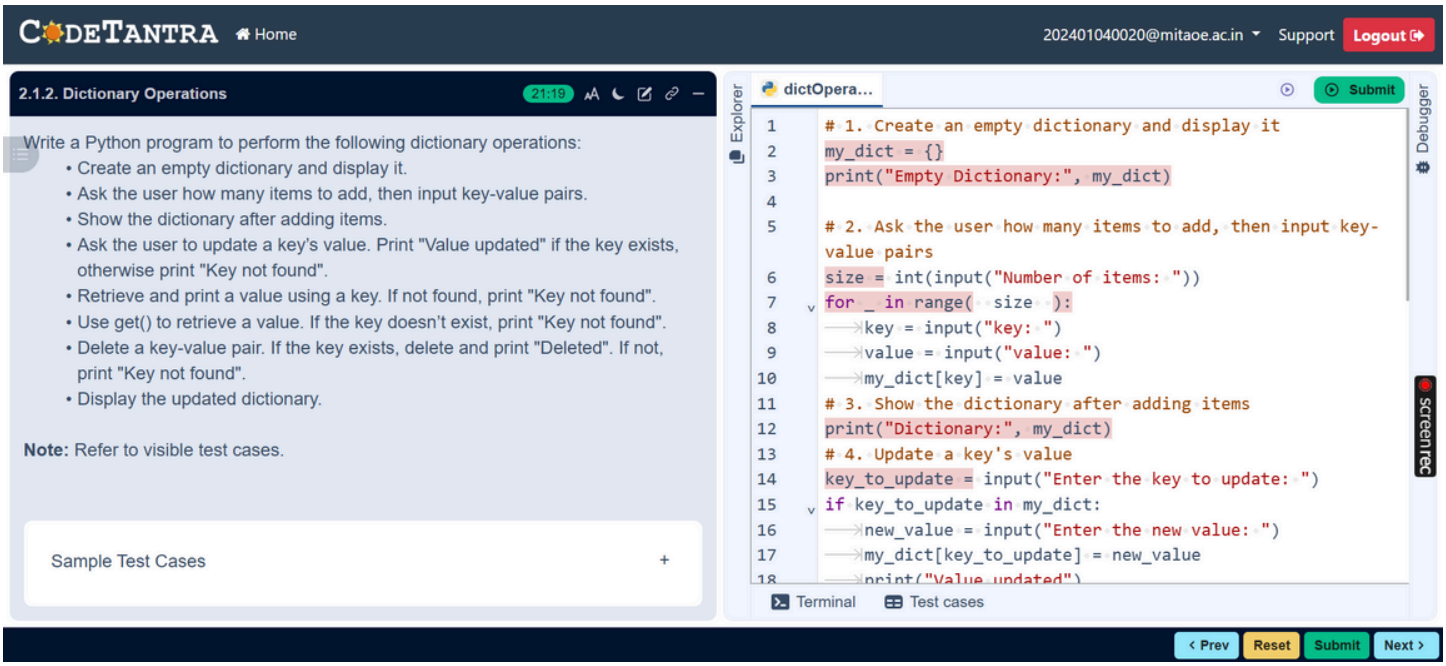
screenrec

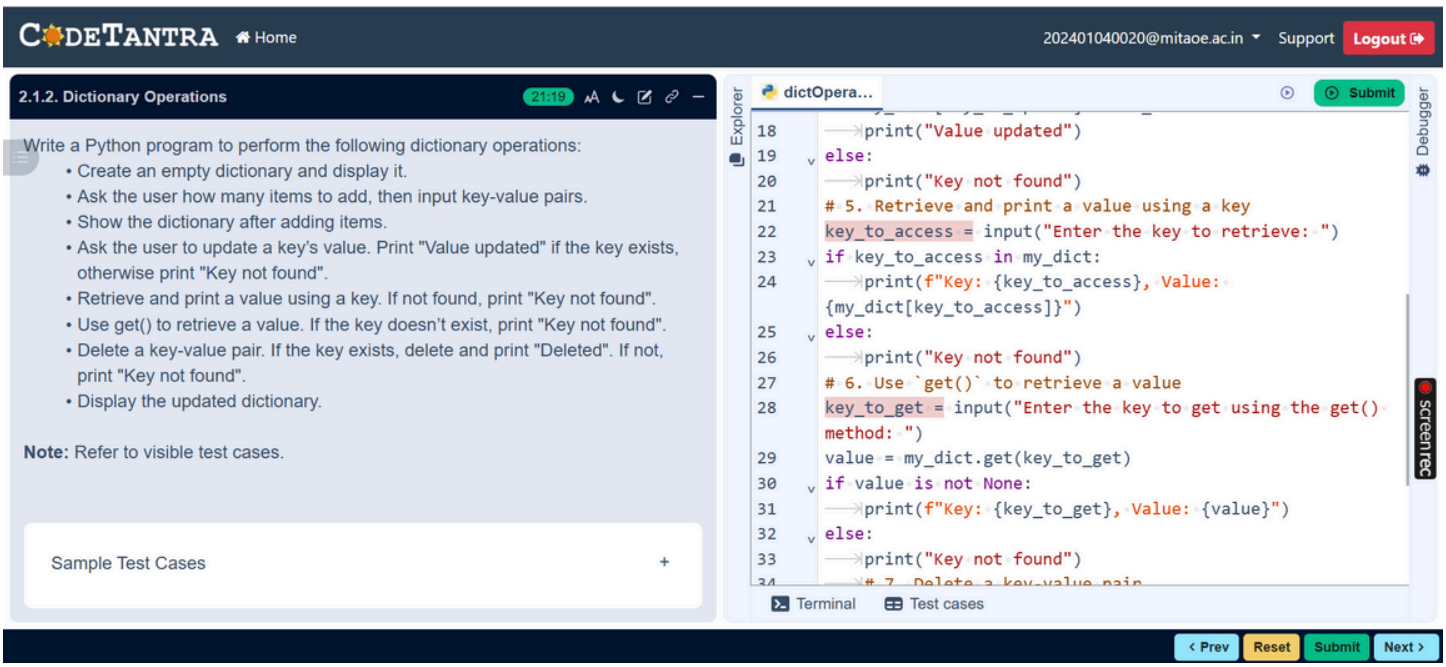
< Prev | Reset | Submit | Next >

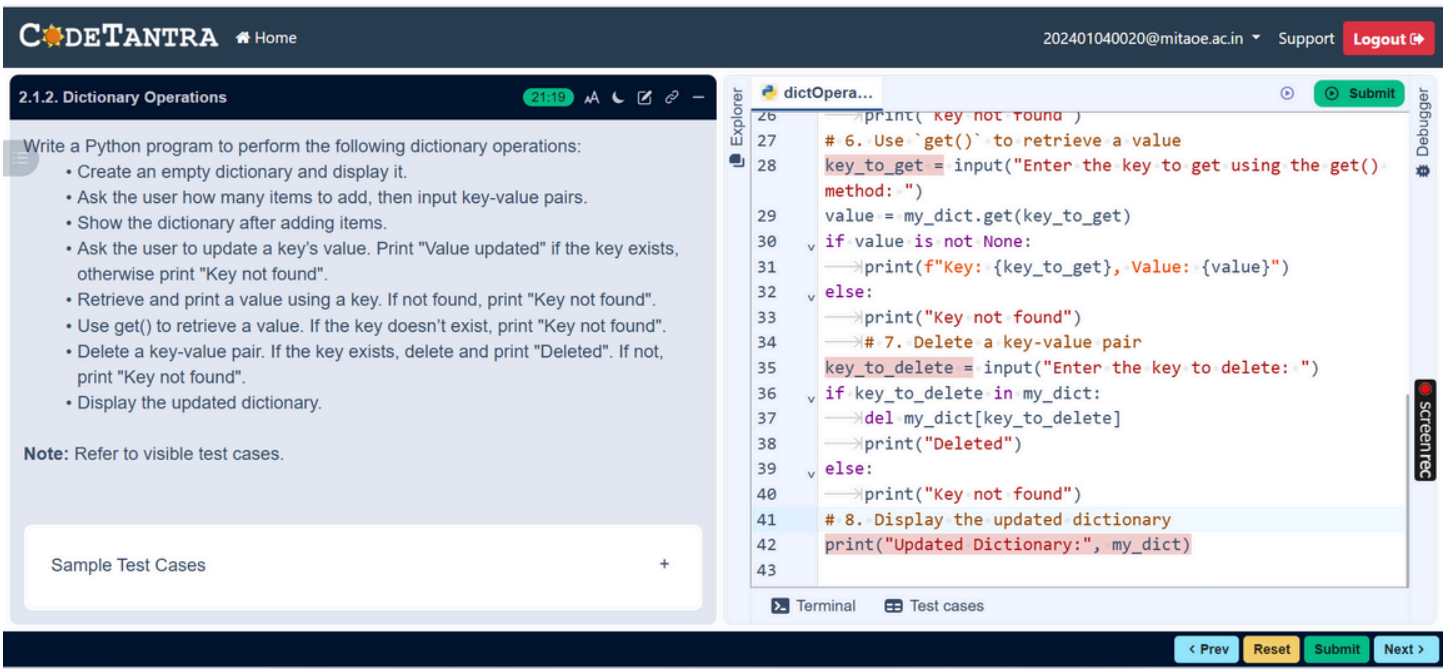
Practical 2 Lab Assignment 1 :



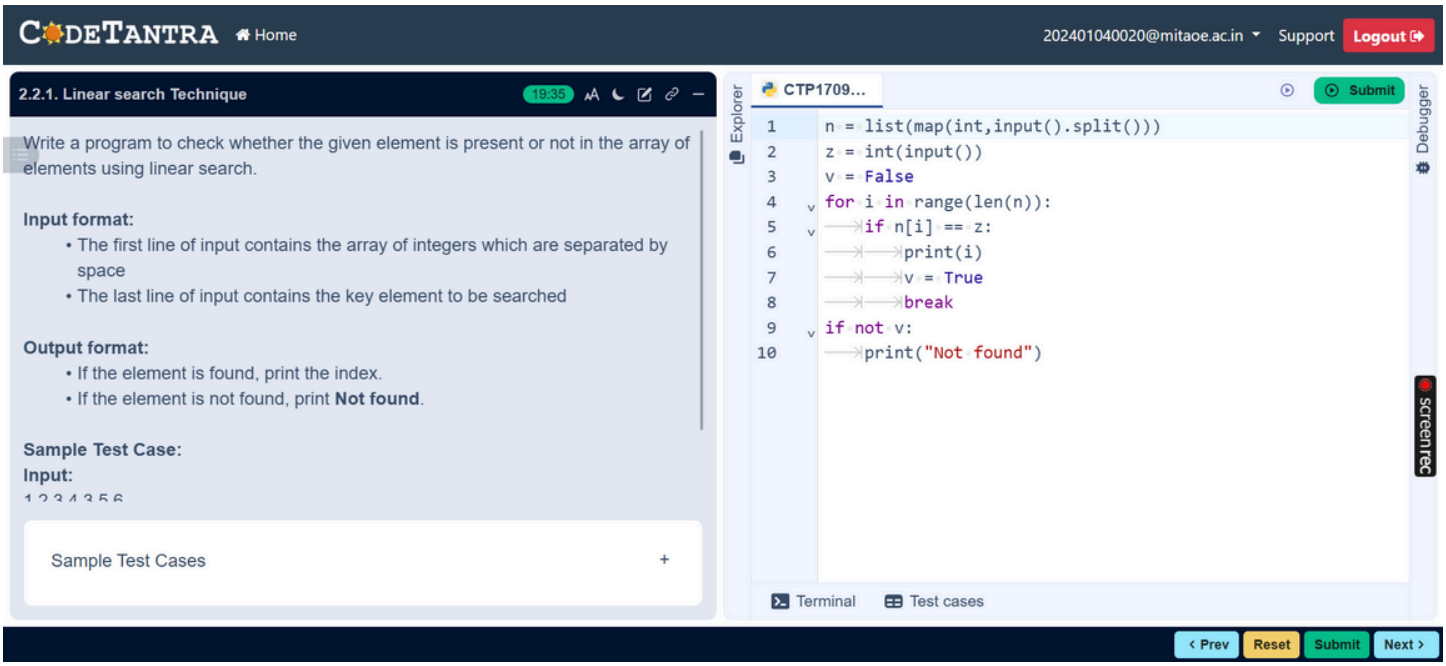


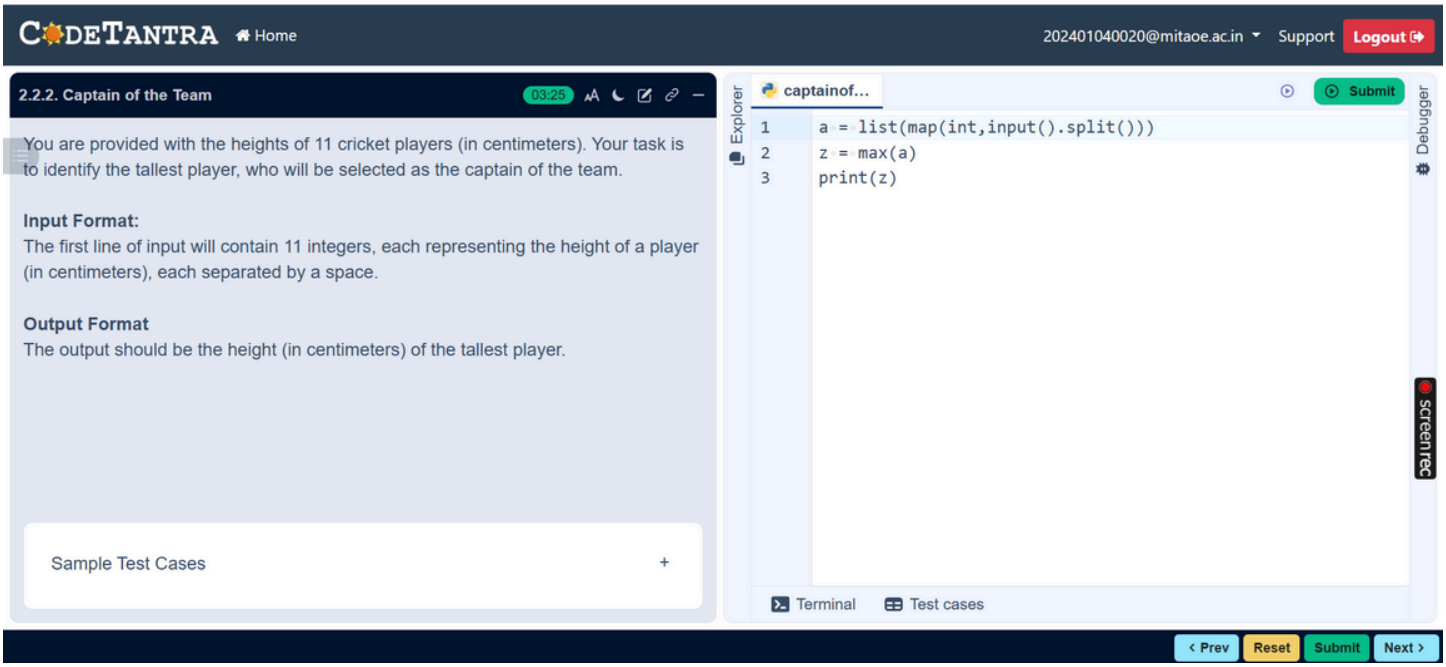

CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
2.1.2. Dictionary Operations
21:19
Write a Python program to perform the following dictionary operations:
• Create an empty dictionary and display it.
• Ask the user how many items to add, then input key-value pairs.
• Show the dictionary after adding items.
• Ask the user to update a key's value. Print "Value updated" if the key exists, otherwise print "Key not found".
• Retrieve and print a value using a key. If not found, print "Key not found".
• Use get() to retrieve a value. If the key doesn't exist, print "Key not found".
• Delete a key-value pair. If the key exists, delete and print "Deleted". If not, print "Key not found".
• Display the updated dictionary.
Note: Refer to visible test cases.
Sample Test Cases
Explorer
dictOpera...
Submit
Debugger
screenrec
# 1. Create an empty dictionary and display it
my_dict = {}
print("Empty Dictionary:", my_dict)
# 2. Ask the user how many items to add, then input key-value pairs
size = int(input("Number of items: "))
for _ in range(size):
key = input("key: ")
value = input("value: ")
my_dict[key] = value
# 3. Show the dictionary after adding items
print("Dictionary:", my_dict)
# 4. Update a key's value
key_to_update = input("Enter the key to update: ")
if key_to_update in my_dict:
new_value = input("Enter the new value: ")
my_dict[key_to_update] = new_value
print("Value updated")
Terminal
Test cases
Prev
Reset
Submit
Next


CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
2.1.2. Dictionary Operations
21:19
Write a Python program to perform the following dictionary operations:
• Create an empty dictionary and display it.
• Ask the user how many items to add, then input key-value pairs.
• Show the dictionary after adding items.
• Ask the user to update a key's value. Print "Value updated" if the key exists, otherwise print "Key not found".
• Retrieve and print a value using a key. If not found, print "Key not found".
• Use get() to retrieve a value. If the key doesn't exist, print "Key not found".
• Delete a key-value pair. If the key exists, delete and print "Deleted". If not, print "Key not found".
• Display the updated dictionary.
Note: Refer to visible test cases.
Sample Test Cases
Explorer
dictOpera...
Submit
Debugger
screenrec
print("Value updated")
else:
print("Key not found")
# 5. Retrieve and print a value using a key
key_to_access = input("Enter the key to retrieve: ")
if key_to_access in my_dict:
print(f"Key: {key_to_access}, Value: {my_dict[key_to_access]}")
else:
print("Key not found")
# 6. Use `get()` to retrieve a value
key_to_get = input("Enter the key to get using the get() method: ")
value = my_dict.get(key_to_get)
if value is not None:
print(f"Key: {key_to_get}, Value: {value}")
else:
print("Key not found")
Terminal
Test cases
Prev
Reset
Submit
Next

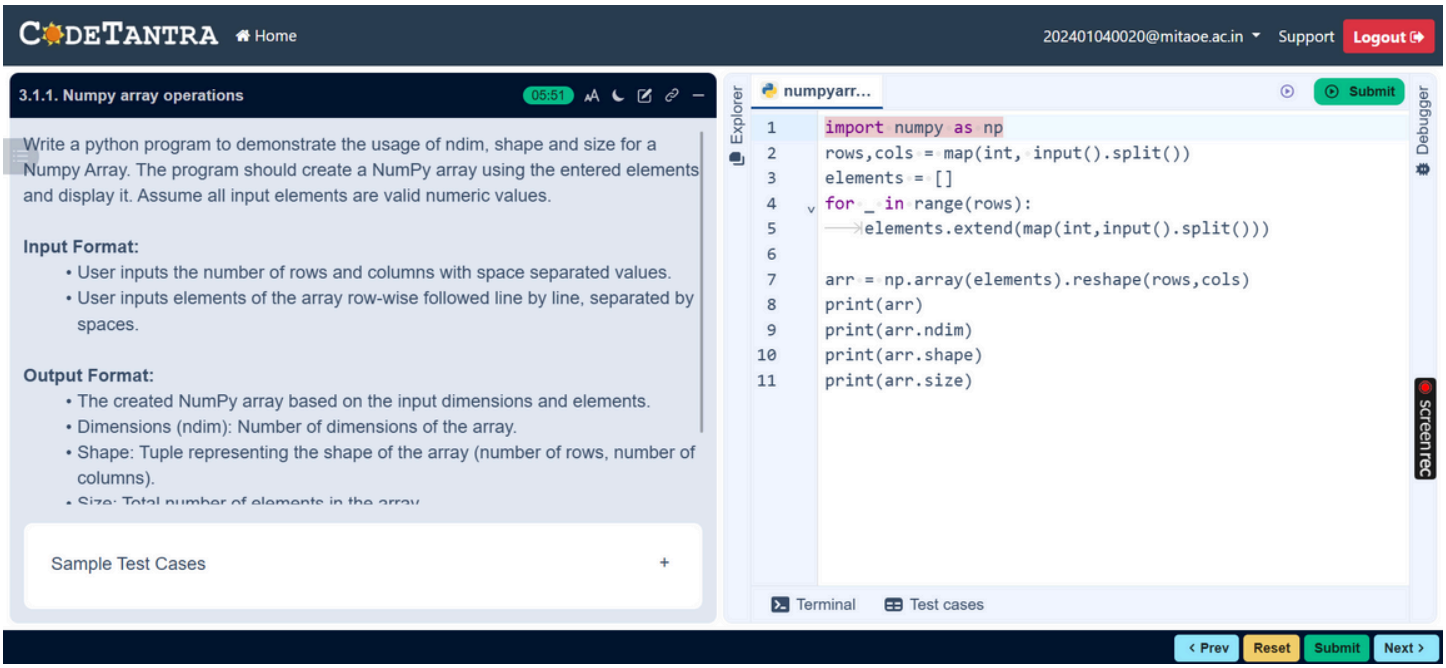

CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
2.1.2. Dictionary Operations
21:19
Write a Python program to perform the following dictionary operations:
• Create an empty dictionary and display it.
• Ask the user how many items to add, then input key-value pairs.
• Show the dictionary after adding items.
• Ask the user to update a key's value. Print "Value updated" if the key exists, otherwise print "Key not found".
• Retrieve and print a value using a key. If not found, print "Key not found".
• Use get() to retrieve a value. If the key doesn't exist, print "Key not found".
• Delete a key-value pair. If the key exists, delete and print "Deleted". If not, print "Key not found".
• Display the updated dictionary.
Note: Refer to visible test cases.
Sample Test Cases
Explorer
dictOpera...
Submit
Debugger
screenrec
print("Key not found")
# 6. Use `get()` to retrieve a value
key_to_get = input("Enter the key to get using the get() method: ")
value = my_dict.get(key_to_get)
if value is not None:
print(f"Key: {key_to_get}, Value: {value}")
else:
print("Key not found")
# 7. Delete a key-value pair
key_to_delete = input("Enter the key to delete: ")
if key_to_delete in my_dict:
del my_dict[key_to_delete]
print("Deleted")
else:
print("Key not found")
# 8. Display the updated dictionary
print("Updated Dictionary:", my_dict)
Terminal
Test cases
Prev
Reset
Submit
Next

Practical 2 Assignment 2 :





Practical 3 Assignment 1 :



Practical 3 Assignment 2 :

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

3.2.1. Numpy: Matrix Operations

24:46

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

Sample Test Cases +

Explorer matrixOp... Submit Debugger

```
matrix_b = np.array([list(map(int, input().split())) for
i in range(3)])

# Addition
print("Addition (A + B):")
print(np.add(matrix_a,matrix_b))
# Subtraction
print("Subtraction (A - B):")
print(np.subtract(matrix_a,matrix_b))
# Multiplication (element-wise)
print("Element-wise Multiplication (A * B):")
print(np.multiply(matrix_a,matrix_b))
# Matrix multiplication (dot product)
print("A dot B:")
print(np.dot(matrix_a,matrix_b))
# Transpose
print("Transpose of A:")
print(matrix_a.T)
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

3.2.2. Numpy: Horizontal and Vertical Stacking of Arrays

11:42

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.

Sample Test Cases +

Explorer stacking.py Submit Debugger

```
import numpy as np

# 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)
print("Horizontal Stack:")
x = np.hstack((arr1,arr2))
print(x)
# Perform vertical stacking (vstack)
print("Vertical Stack:")
y=np.vstack((arr1, arr2))
print(y)
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

3.2.3. Numpy: Custom Sequence Generation

04:26

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 +

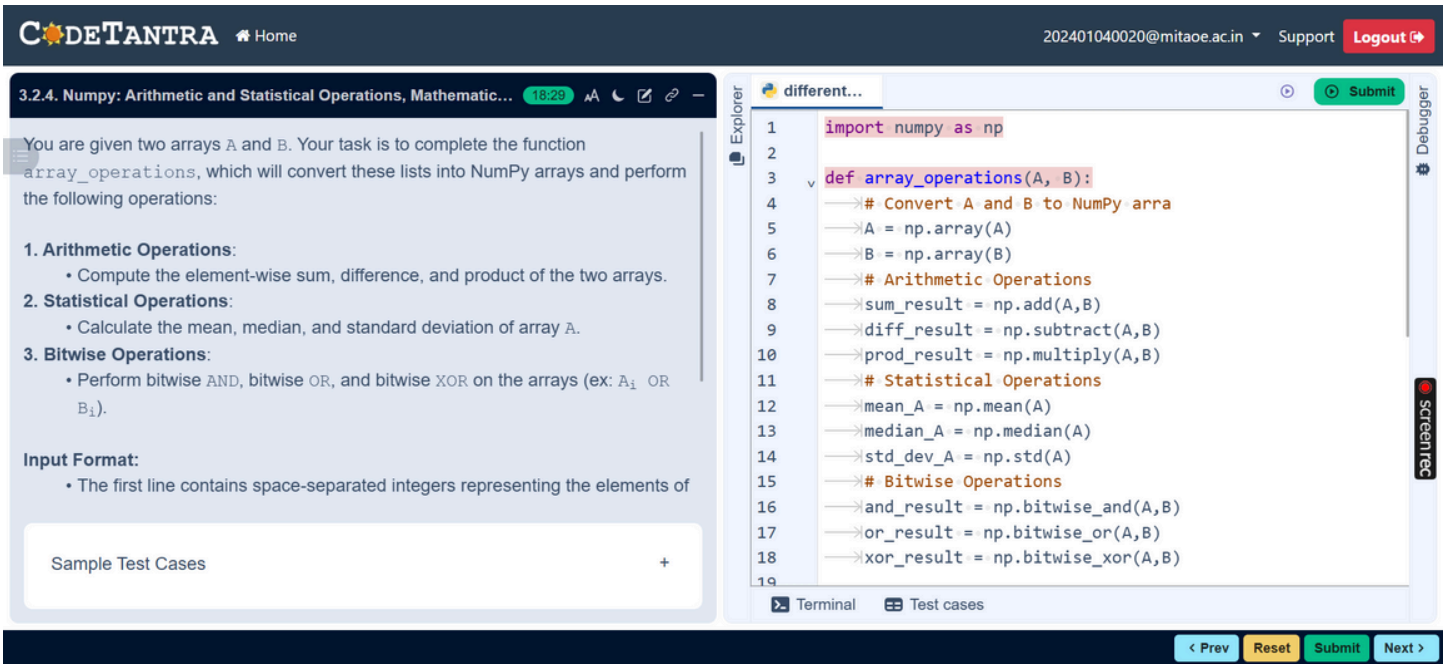
Explorer customS... Submit Debugger

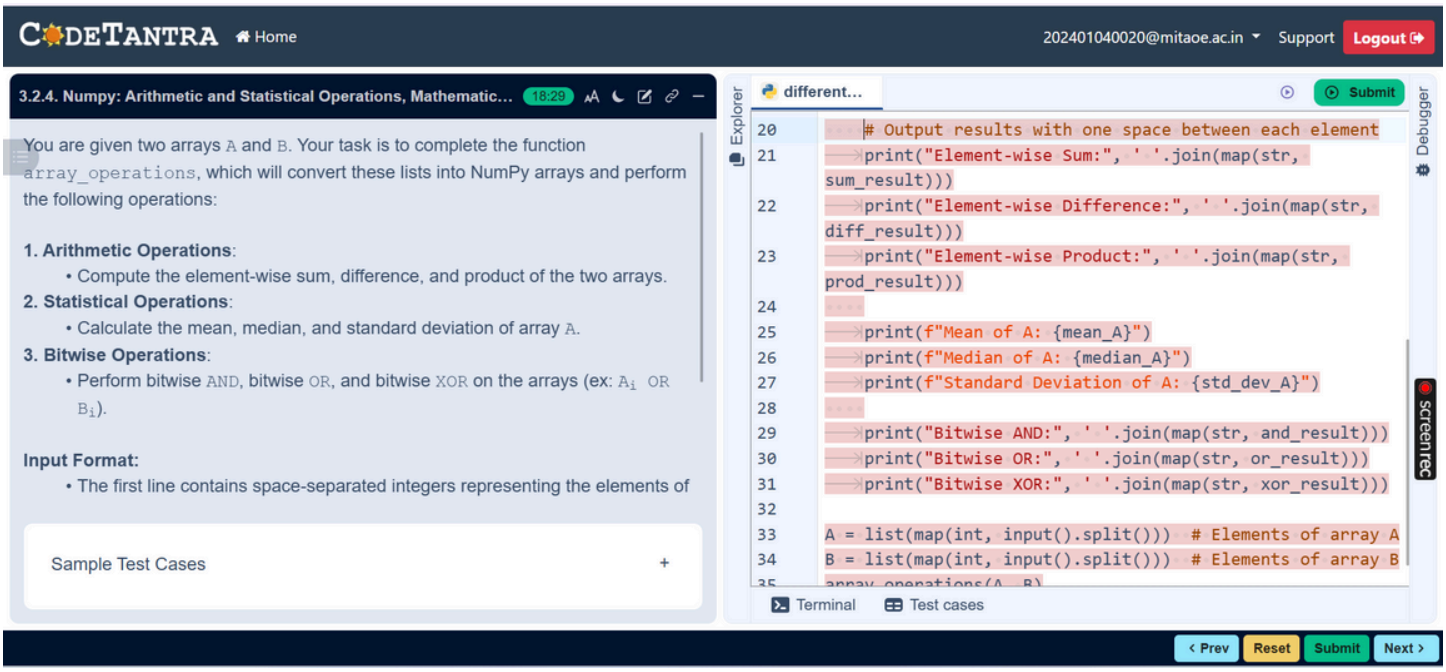
```
import numpy as np

# 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)
```

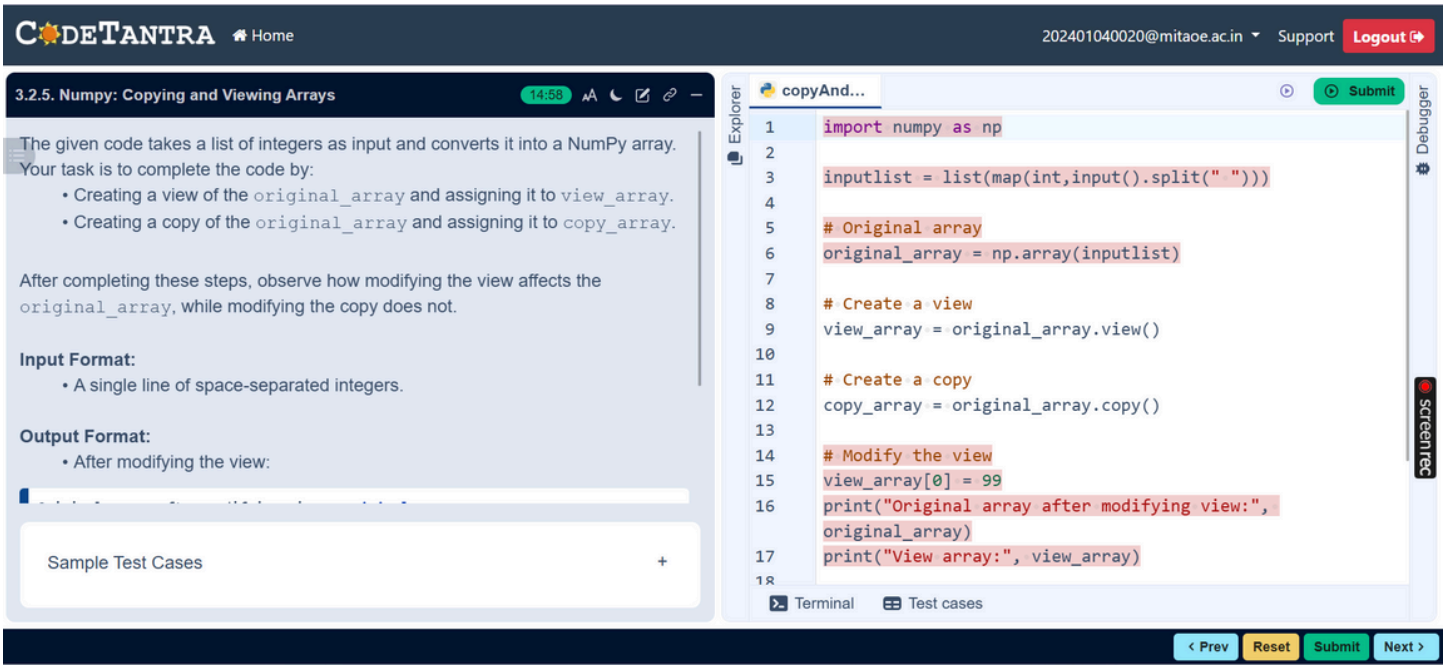
Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›


CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
3.2.4. Numpy: Arithmetic and Statistical Operations, Mathematic...
18:29
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
Sample Test Cases
Explorer
different...
Submit
Debugger
import numpy as np
def array_operations(A, B):
# Convert A and B to NumPy arra
A = np.array(A)
B = np.array(B)
# Arithmetic Operations
sum_result = np.add(A,B)
diff_result = np.subtract(A,B)
prod_result = np.multiply(A,B)
# Statistical Operations
mean_A = np.mean(A)
median_A = np.median(A)
std_dev_A = np.std(A)
# Bitwise Operations
and_result = np.bitwise_and(A,B)
or_result = np.bitwise_or(A,B)
xor_result = np.bitwise_xor(A,B)
Terminal
Test cases
screenrec
‹ Prev
Reset
Submit
Next ›


CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
3.2.4. Numpy: Arithmetic and Statistical Operations, Mathematic...
18:29
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
Sample Test Cases
Explorer
different...
Submit
Debugger
# 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)))
print("Element-wise Product:", ' '.join(map(str, prod_result)))
print(f"Mean of A: {mean_A}")
print(f"Median of A: {median_A}")
print(f"Standard Deviation of A: {std_dev_A}")
print("Bitwise AND:", ' '.join(map(str, and_result)))
print("Bitwise OR:", ' '.join(map(str, or_result)))
print("Bitwise XOR:", ' '.join(map(str, xor_result)))
A = list(map(int, input().split())) # Elements of array A
B = list(map(int, input().split())) # Elements of array B
Terminal
Test cases
screenrec
‹ Prev
Reset
Submit
Next ›


CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
3.2.5. Numpy: Copying and Viewing Arrays
14:58
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:
Sample Test Cases
Explorer
copyAnd...
Submit
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)
Terminal
Test cases
screenrec
‹ Prev
Reset
Submit
Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

3.2.5. Numpy: Copying and Viewing Arrays

14:58

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:

Sample Test Cases +

Explorer copyAnd... Submit Debugger

```
# 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)

# Modify the copy
copy_array[1] = 88
print("Original array after modifying copy:", original_array)
print("Copy array:", copy_array)
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

3.2.6. Numpy: Searching, Sorting, Counting, Broadcasting

12:33

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.

Sample Test Cases +

Explorer arrayOpe... Submit Debugger

```
import numpy as np

# Input array from the user
array1 = np.array(list(map(int, input().split())))

# Searching
search_value = int(input("Value to search: "))
count_value = int(input("Value to count: "))
broadcast_value = int(input("Value to add: "))

# Find indices where value matches in array1
z = np.where(array1==search_value)
print(z[0])
# Count occurrences in array1
c = np.count_nonzero(array1 == count_value)
print(c)
# Broadcasting addition
broadcasted_array = array1 + broadcast_value
print(broadcasted_array)
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

3.2.6. Numpy: Searching, Sorting, Counting, Broadcasting 12:33

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.

Sample Test Cases +

Explorer arrayOpe... Submit Debugger

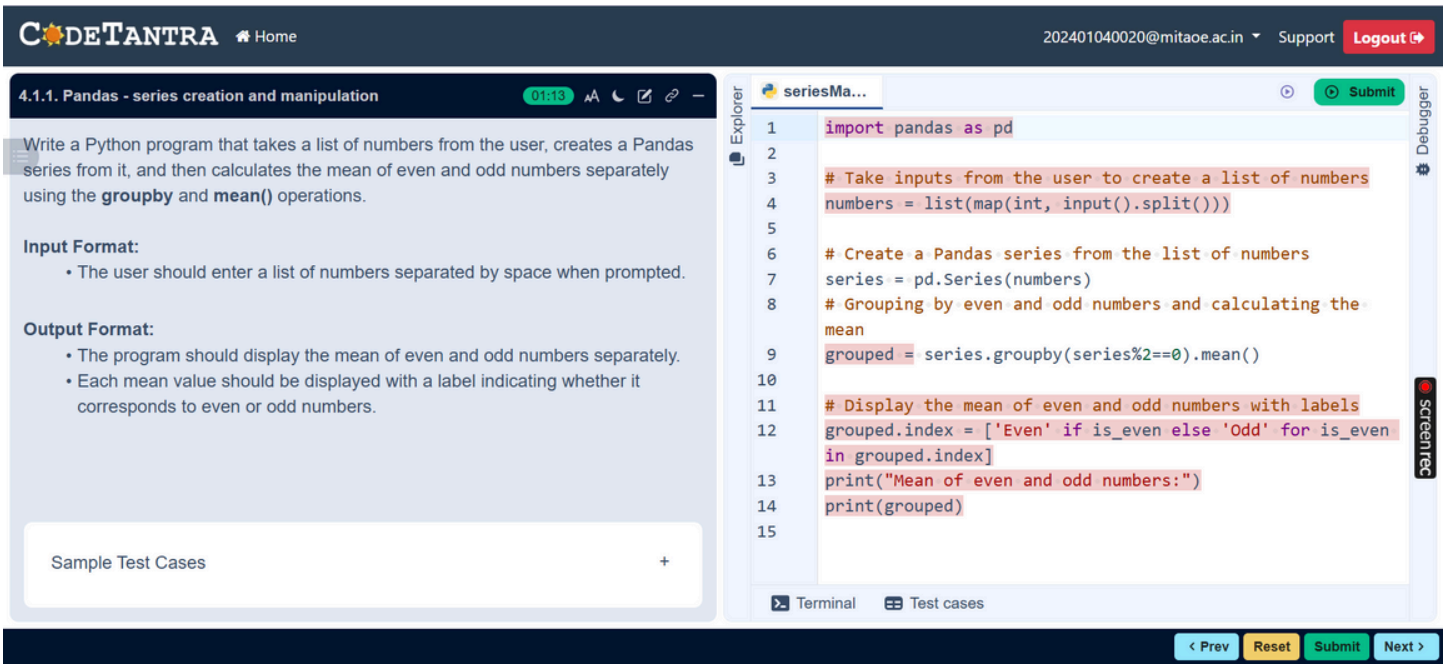
```
array1 = np.array(list(map(int, input().split())))

# Searching
search_value = int(input("Value to search: "))
count_value = int(input("Value to count: "))
broadcast_value = int(input("Value to add: "))

# Find indices where value matches in array1
z = np.where(array1==search_value)
print(z[0])
# Count occurrences in array1
c = np.count_nonzero(array1 == count_value)
print(c)
# Broadcasting addition
broadcasted_array = array1 + broadcast_value
print(broadcasted_array)
# Sort the first array
sorted_arr = np.sort(array1)
print(sorted_arr)
```

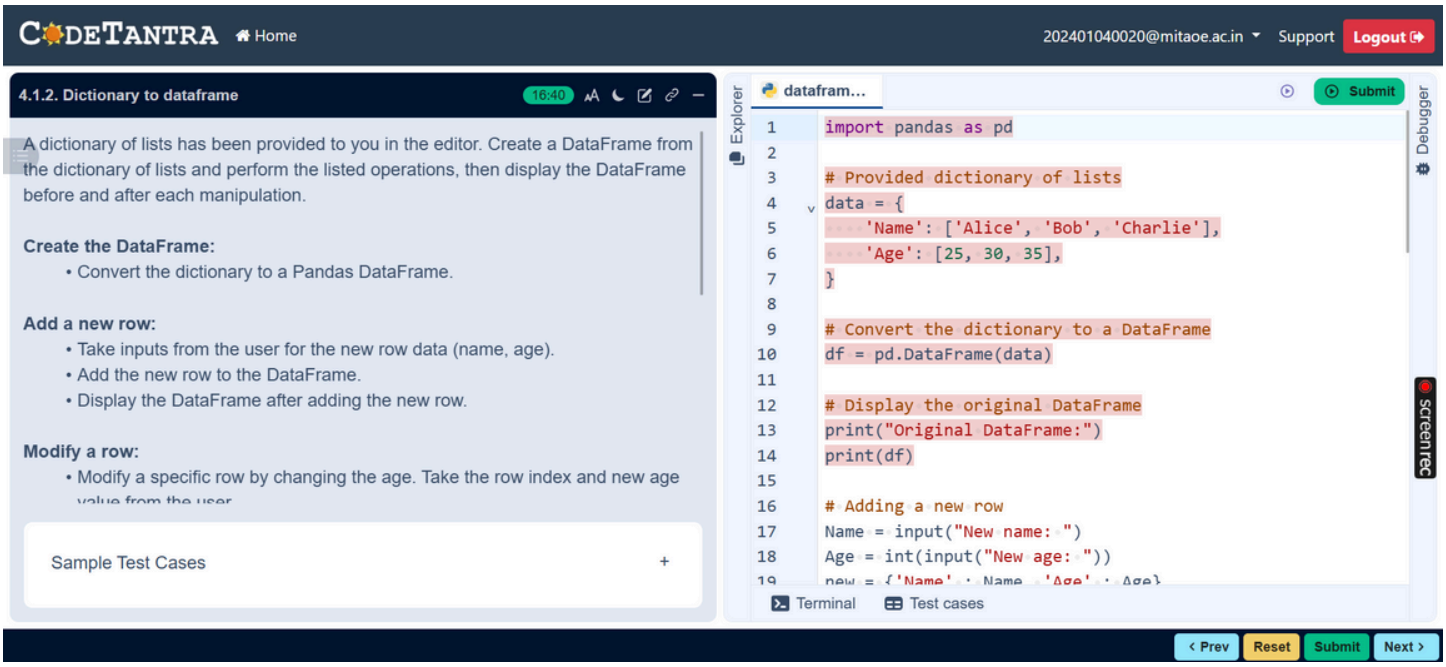
Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

Practical 4 Assignment 1 :





4.1.2. Dictionary to dataframe

16:40

A dictionary of lists has been provided to you in the editor. Create a DataFrame from the dictionary of lists and perform the listed operations, then display the DataFrame before and after each manipulation.

Create the DataFrame:

- Convert the dictionary to a Pandas DataFrame.

Add a new row:

- Take inputs from the user for the new row data (name, age).
- Add the new row to the DataFrame.
- Display the DataFrame after adding the new row.

Modify a row:

- Modify a specific row by changing the age. Take the row index and new age value from the user

Sample Test Cases +

Explorer | dataframe... | Submit | Debugger

```
19  new = {'Name' : Name, 'Age' : Age}
20  df = df.append(new,ignore_index=True)
21  # Display the DataFrame after adding a new row
22  print("After adding a row:\n",df)
23  index=int(input("Index of row to modify: "))
24  age=int(input("New age: "))
25  df.at[index,'Age']=age
26  # Modifying a row
27
28
29  # Display the DataFrame after modifying a row
30  print("After modifying a row:")
31  print(df)
32
33  # Deleting a row
34  delt=int(input("Index of row to delete: "))
35  df=df.drop(delt).reset_index(drop=True)
36
37  # Display the DataFrame after deleting a row
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.1.2. Dictionary to dataframe 16:40

A dictionary of lists has been provided to you in the editor. Create a DataFrame from the dictionary of lists and perform the listed operations, then display the DataFrame before and after each manipulation.

Create the DataFrame:
- Convert the dictionary to a Pandas DataFrame.

Add a new row:
- Take inputs from the user for the new row data (name, age).
- Add the new row to the DataFrame.
- Display the DataFrame after adding the new row.

Modify a row:
- Modify a specific row by changing the age. Take the row index and new age value from the user

Sample Test Cases +

Explorer | dataframe... | Submit | Debugger

```
36
37  # Display the DataFrame after deleting a row
38  print("After deleting a row:")
39  print(df)
40
41  # Adding a new column
42  gen=input("Enter genders separated by space: ").split()
43  df['Gender']=gen
44
45
46
47  # Display the DataFrame after adding a new column
48  print("After adding a new column:")
49  print(df)
50
51  # Modifying a column
52  df['Name']=df['Name'].str.upper()
53  # Display the DataFrame after modifying a column
54  print("After modifying a column:")
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.1.2. Dictionary to dataframe 16:40

A dictionary of lists has been provided to you in the editor. Create a DataFrame from the dictionary of lists and perform the listed operations, then display the DataFrame before and after each manipulation.

Create the DataFrame:
- Convert the dictionary to a Pandas DataFrame.

Add a new row:
- Take inputs from the user for the new row data (name, age).
- Add the new row to the DataFrame.
- Display the DataFrame after adding the new row.

Modify a row:
- Modify a specific row by changing the age. Take the row index and new age value from the user

Sample Test Cases +

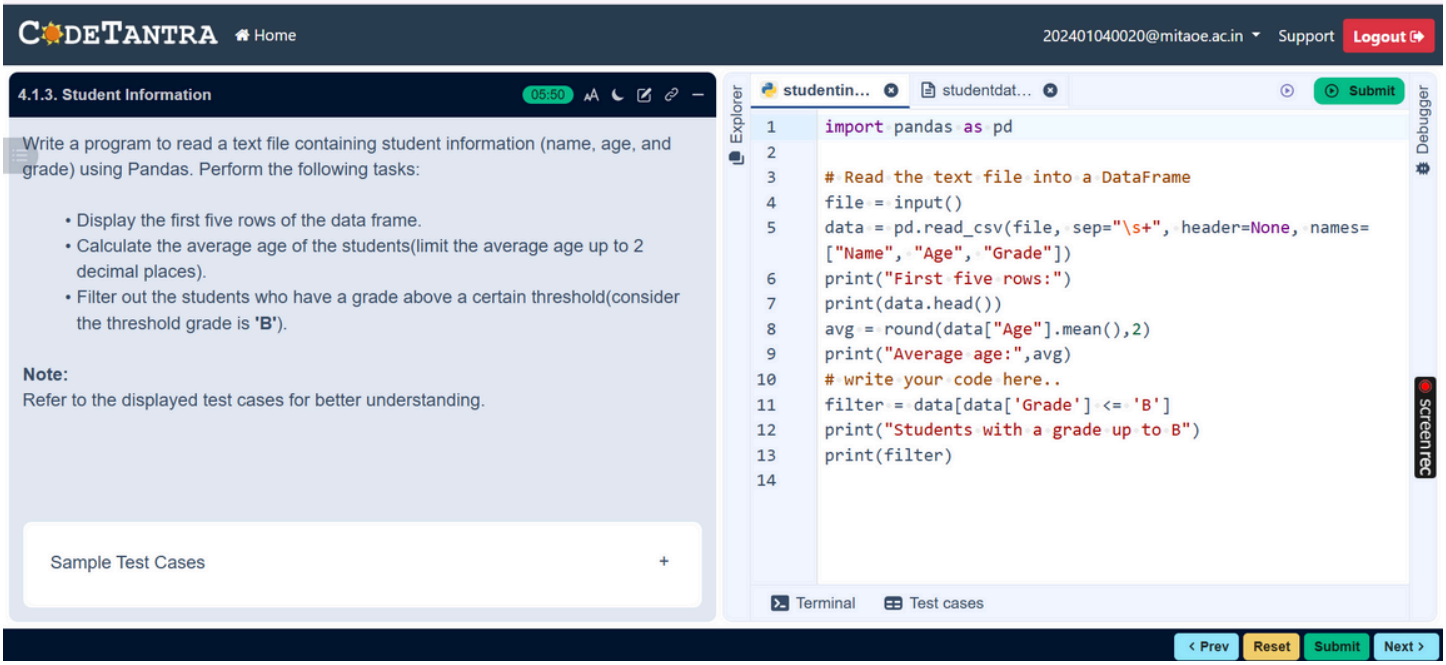
Explorer | dataframe... | Submit | Debugger

```
44
45
46
47  # Display the DataFrame after adding a new column
48  print("After adding a new column:")
49  print(df)
50
51  # Modifying a column
52  df['Name']=df['Name'].str.upper()
53  # Display the DataFrame after modifying a column
54  print("After modifying a column:")
55  print(df)
56
57  # Deleting a column
58  df=df.drop(columns=['Age'])
59  # Display the DataFrame after deleting a column
60  print("After deleting a column:")
61  print(df)
62
```

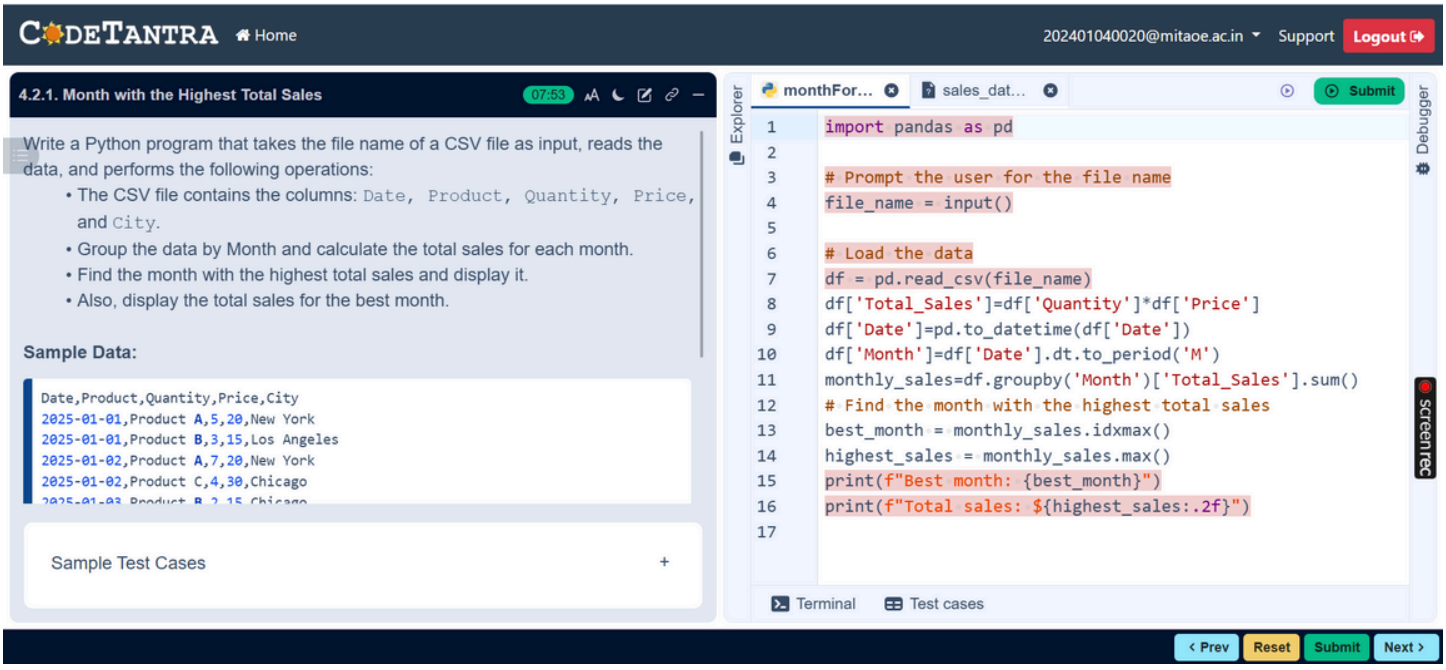
Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >



Practical 4 Assignment 2 :



CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.2. Best Selling Product

23:52

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
```

Sample Test Cases +

Explorer | monthFor... | sales_dat... | Submit | Debugger

```python
import pandas as pd

# Prompt the user for the file name
file_name = input()

# Load the data
df = pd.read_csv(file_name)
product_sales=df.groupby('Product')
['Quantity'].sum().reset_index()

# Find the product with the highest total quantity sold
best_product =
product_sales.loc[product_sales['Quantity'].idxmax()]
highest_quantity = product_sales.max()
print(f"Best selling product: {best_product['Product']}")
print(f"Total quantity sold: {best_product['Quantity']}")
"""
# Display the result
print(f"Best selling product: {best_product}")
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.3. City that Sold the Most Products

05:58

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
```

Sample Test Cases +

Explorer | monthFor... | sales_dat... | Submit | Debugger

```python
import pandas as pd

# Prompt the user for the file name
file_name = input()

# Load the data
df = pd.read_csv(file_name)
city_wise_sell=df.groupby('City')
['Quantity'].sum().reset_index()
# write the code..
best_city=city_wise_sell.loc[city_wise_sell['Quantity'].id
xmax()]
"""
# Display the result
print(f"City sold the most products: {best_city}")
"""
print(f"City sold the most products: {best_city['City']}")
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.4. Most Frequently Sold Product Pairs

10:12

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 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
```

Sample Test Cases +

Explorer | frequentl... | sales_dat... | Submit | Debugger

```python
import pandas as pd
from itertools import combinations
from collections import Counter

# Prompt user to input the file name
file_name = input()

# Read data from the specified CSV file
df = pd.read_csv(file_name)

# write the code
grouped=df.groupby('Date')['Product'].apply(list)
# create list
product_combinations=[]
for products in grouped:
    product_combinations.extend(combinations(sorted(set(produc
ts)),2))
combinations_count=Counter(product_combinations)
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.4. Most Frequently Sold Product Pairs

10:12

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 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
```

Sample Test Cases +

Explorer | frequentl... | sales_dat... | Submit | Debugger

```
# Read data from the specified CSV file
df = pd.read_csv(file_name)

# write the code
grouped=df.groupby('Date')['Product'].apply(list)
# create list
product_combinations=[]
for products in grouped:
    product_combinations.extend(combinations(sorted(set(products)),2))
combinations_count=Counter(product_combinations)
max_count=combinations_count.most_common(1)[0][1]
# Output the most frequent product pairs
for combo, count in combinations_count.items():
    if count==max_count:
        print(f"{combo[0]} and {combo[1]}: {count} times")
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.5. Titanic Dataset Analysis and Data Cleaning

23:58

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

Sample Test Cases +

Explorer | titanicDat... | Submit | Debugger

```
import pandas as pd
import numpy as np

# Load the Titanic dataset
data = pd.read_csv('Titanic-Dataset.csv')

# 1. Display the first 5 rows of the dataset
print(data.head())

# 2. Display the last 5 rows of the dataset
print(data.tail())

# 3. Get the shape of the dataset
print(data.shape)

# 4. Get a summary of the dataset (info)
print(data.info())

# 5. Get basic statistics of the dataset
```

Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.5. Titanic Dataset Analysis and Data Cleaning

23:58

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

Sample Test Cases +

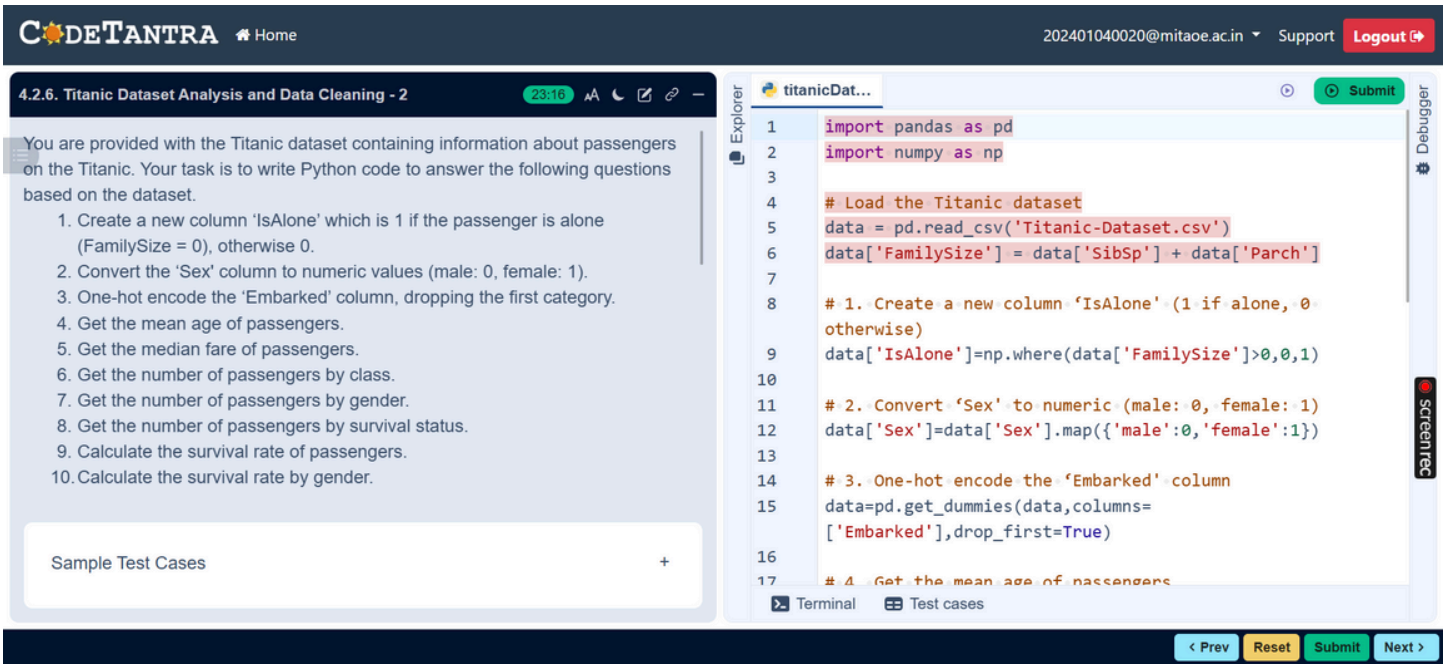
Explorer | titanicDat... | Submit | Debugger

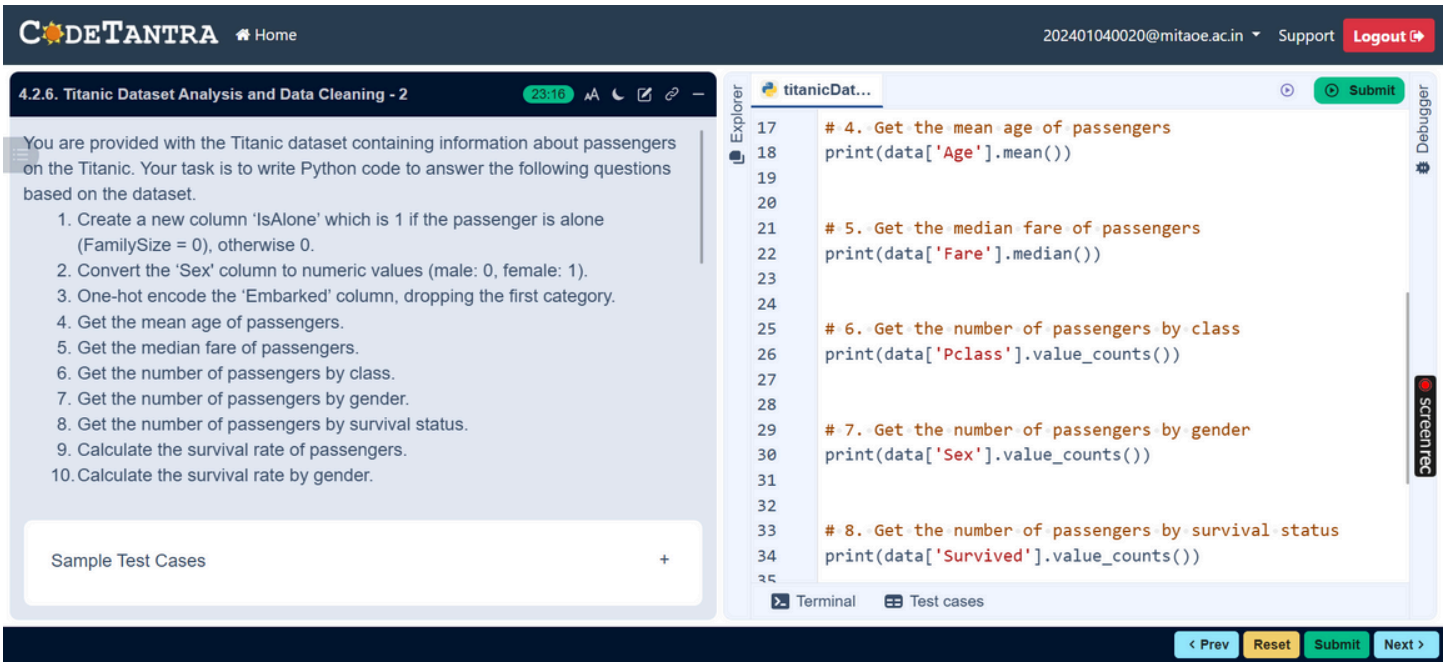
```
# 6. Check for missing values
print(data.isnull().sum())

# 7. Fill missing values in the 'Age' column with the median age
data['Age'].fillna(data['Age'].median(),inplace=True)

# 8. Fill missing values in the 'Embarked' column with the mode
data['Embarked'].fillna(data['Embarked'].mode()[0],inplace=True)

# 9. Drop the 'Cabin' column due to many missing values
data.drop('Cabin',axis=1, inplace=True)

# 10. Create a new column 'FamilySize' by adding 'SibSp' and 'Parch'
data['FamilySize']=data['SibSp']+ data['Parch']
```

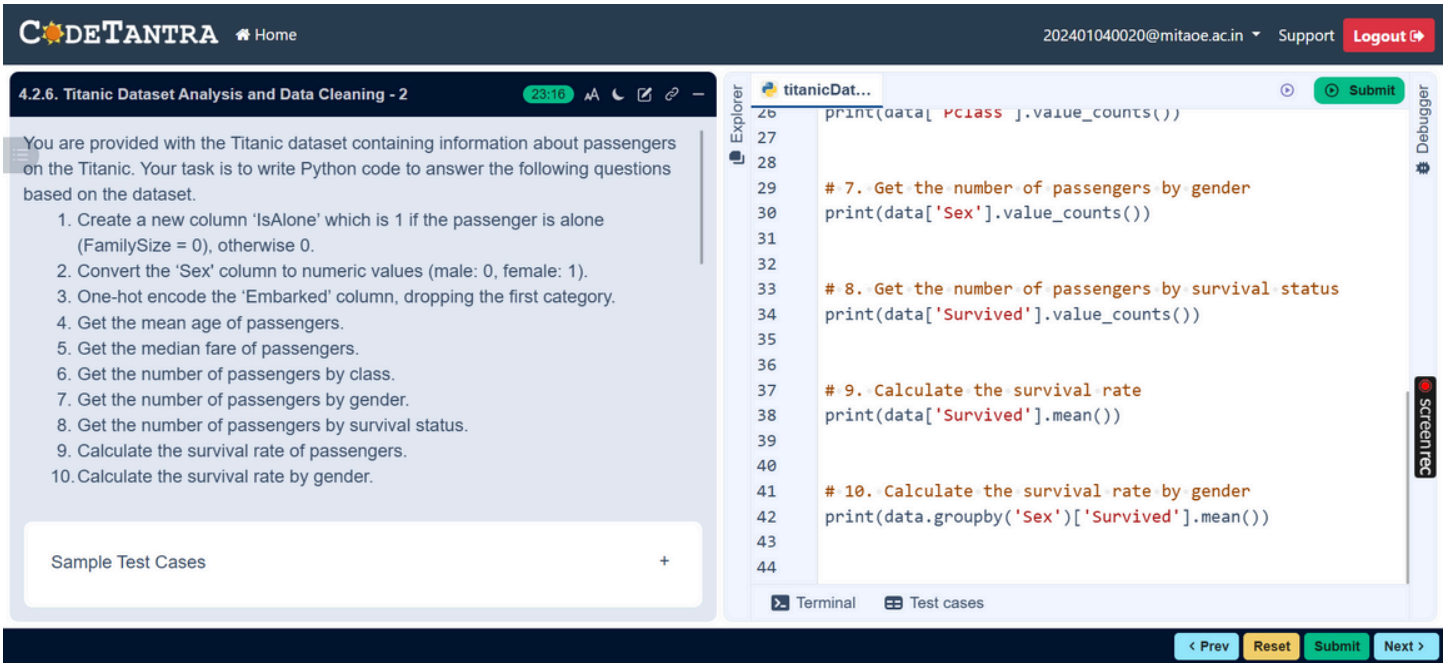
Terminal | Test cases

screenrec

< Prev | Reset | Submit | Next >


CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
4.2.6. Titanic Dataset Analysis and Data Cleaning - 2
23:16
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. 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.
Sample Test Cases
Explorer
titanicDat...
Submit
Debugger
import pandas as pd
import numpy as np
# Load the Titanic dataset
data = pd.read_csv('Titanic-Dataset.csv')
data['FamilySize'] = data['SibSp'] + data['Parch']
1. Create a new column 'IsAlone' (1 if alone, 0 otherwise)
data['IsAlone']=np.where(data['FamilySize']>0,0,1)
2. Convert 'Sex' to numeric (male: 0, female: 1)
data['Sex']=data['Sex'].map({'male':0,'female':1})
# 3. One-hot encode the 'Embarked' column
data=pd.get_dummies(data,columns=['Embarked'],drop_first=True)
# 4. Get the mean age of passengers
Terminal
Test cases
screenrec
Prev
Reset
Submit
Next


CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
4.2.6. Titanic Dataset Analysis and Data Cleaning - 2
23:16
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. 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.
Sample Test Cases
Explorer
titanicDat...
Submit
Debugger
# 4. Get the mean age of passengers
print(data['Age'].mean())
# 5. Get the median fare of passengers
print(data['Fare'].median())
# 6. Get the number of passengers by class
print(data['Pclass'].value_counts())
# 7. Get the number of passengers by gender
print(data['Sex'].value_counts())
# 8. Get the number of passengers by survival status
print(data['Survived'].value_counts())
Terminal
Test cases
screenrec
Prev
Reset
Submit
Next


CODETANTRA
Home
202401040020@mitaoe.ac.in
Support
Logout
4.2.6. Titanic Dataset Analysis and Data Cleaning - 2
23:16
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. 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.
Sample Test Cases
Explorer
titanicDat...
Submit
Debugger
print(data['Pclass'].value_counts())
# 7. Get the number of passengers by gender
print(data['Sex'].value_counts())
# 8. Get the number of passengers by survival status
print(data['Survived'].value_counts())
# 9. Calculate the survival rate
print(data['Survived'].mean())
# 10. Calculate the survival rate by gender
print(data.groupby('Sex')['Survived'].mean())
Terminal
Test cases
screenrec
Prev
Reset
Submit
Next

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.7. Titanic Dataset Analysis and Data Cleaning - 3

21:27

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).

Sample Test Cases +

Explorer titanicDat... Submit Debugger

```python
import pandas as pd
import numpy as np

# Load the Titanic dataset
data = pd.read_csv('Titanic-Dataset.csv')
data['FamilySize'] = data['SibSp'] + data['Parch']
data['IsAlone'] = np.where(data['FamilySize'] > 0, 0, 1)
data = pd.get_dummies(data, columns=['Embarked'], drop_first=True)

# 1. Calculate the survival rate by class
print(data.groupby('Pclass')['Survived'].mean())

# 2. Calculate the survival rate by embarked location
print(data.groupby('Embarked_S')['Survived'].mean())

# 3. Calculate the survival rate by family size
print(data.groupby('FamilySize')['Survived'].mean())
```

Terminal Test cases

screenrec

Prev Reset Submit Next

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.7. Titanic Dataset Analysis and Data Cleaning - 3 21:27

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).

Sample Test Cases +

Explorer titanicDat... Submit Debugger

```python
# 4. Calculate the survival rate by being alone
print(data.groupby('IsAlone')['Survived'].mean())

# 5. Get the average fare by class
print(data.groupby('Pclass')['Fare'].mean())

# 6. Get the average age by class
print(data.groupby('Pclass')['Age'].mean())

# 7. Get the average age by survival status
print(data.groupby('Survived')['Age'].mean())

# 8. Get the average fare by survival status
print(data.groupby('Survived')['Fare'].mean())

# 9. Get the number of survivors by class
print(data[data['Survived']==1]['Pclass'].value_counts())
```

Terminal Test cases

screenrec

Prev Reset Submit Next

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.7. Titanic Dataset Analysis and Data Cleaning - 3 21:27

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).

Sample Test Cases +

Explorer titanicDat... Submit Debugger

```python
print(data.groupby('IsAlone')['Survived'].mean())

# 5. Get the average fare by class
print(data.groupby('Pclass')['Fare'].mean())

# 6. Get the average age by class
print(data.groupby('Pclass')['Age'].mean())

# 7. Get the average age by survival status
print(data.groupby('Survived')['Age'].mean())

# 8. Get the average fare by survival status
print(data.groupby('Survived')['Fare'].mean())

# 9. Get the number of survivors by class
print(data[data['Survived']==1]['Pclass'].value_counts())

# 10. Get the number of non-survivors by class
print(data[data['Survived']==0]['Pclass'].value_counts())
```

Terminal Test cases

screenrec

Prev Reset Submit Next

4.2.8. Titanic Dataset Analysis and Data Cleaning - 4

13:07

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.

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
10 print(data[data['Survived']==1]['Sex'].value_counts())
11
12 # 2. Get the number of non-survivors by gender
13 print(data[data['Survived']==0]['Sex'].value_counts())
14
15 # 3. Get the number of survivors by embarked location
16 print(data[data['Survived']==1]
   ['Embarked_S'].value_counts())
17
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.8. Titanic Dataset Analysis and Data Cleaning - 4 13:07

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.

Sample Test Cases +

Explorer titanicDat... Submit Debugger

```
18 # 4. Get the number of non-survivors by embarked location
19 print(data[data['Survived']==0]
   ['Embarked_S'].value_counts())
20
21 # 5. Calculate the percentage of children (Age < 18) who
   survived
22 children=data[data['Age']<18]
23 print(children['Survived'].mean())
24 # 6. Calculate the percentage of adults (Age >= 18) who
   survived
25 adults=data[data['Age']>=18]
26 print(adults['Survived'].mean())
27 # 7. Get the median age of survivors
28 print(data[data['Survived']==1]['Age'].median())
29
30 # 8. Get the median age of non-survivors
31 print(data[data['Survived']==0]['Age'].median())
32
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

4.2.8. Titanic Dataset Analysis and Data Cleaning - 4 13:07

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.

Sample Test Cases +

Explorer titanicDat... Submit Debugger

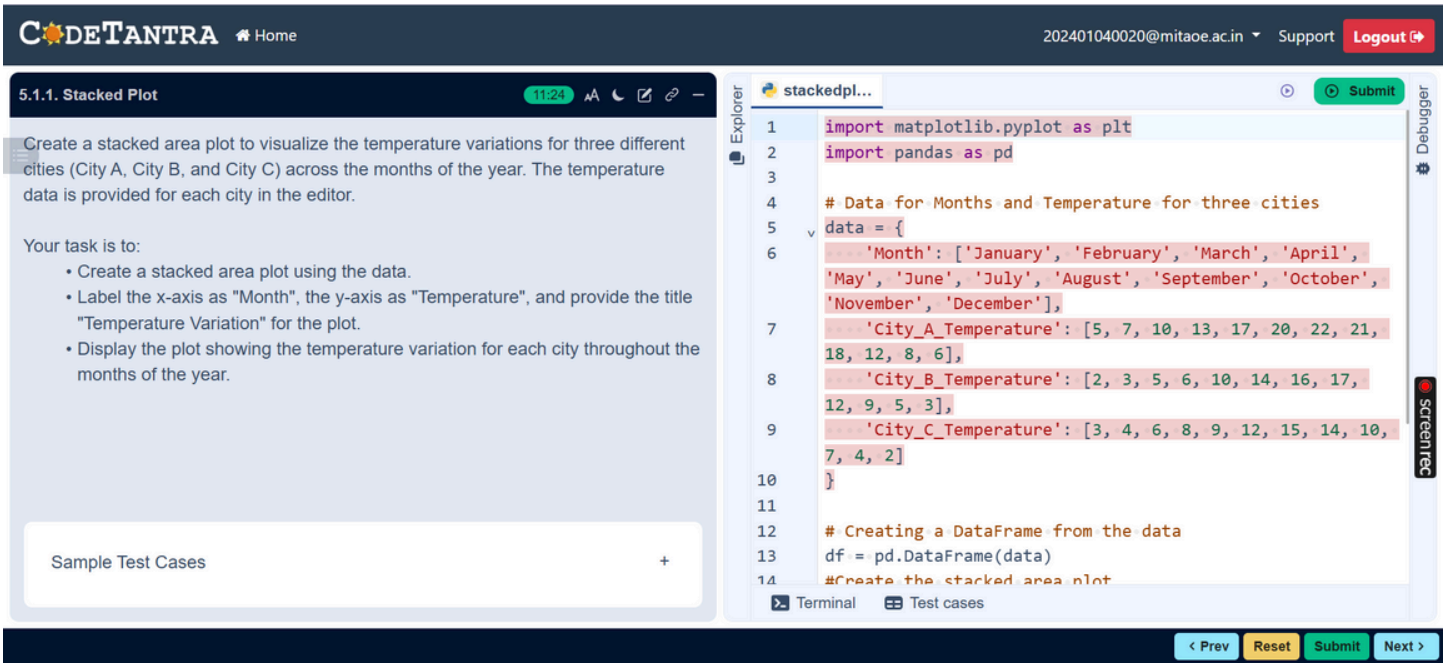
```
22 children=data[data['Age']<18]
23 print(children['Survived'].mean())
24 # 6. Calculate the percentage of adults (Age >= 18) who
   survived
25 adults=data[data['Age']>=18]
26 print(adults['Survived'].mean())
27 # 7. Get the median age of survivors
28 print(data[data['Survived']==1]['Age'].median())
29
30 # 8. Get the median age of non-survivors
31 print(data[data['Survived']==0]['Age'].median())
32
33 # 9. Get the median fare of survivors
34 print(data[data['Survived']==1]['Fare'].median())
35
36 # 10. Get the median fare of non-survivors
37 print(data[data['Survived']==0]['Fare'].median())
38
```

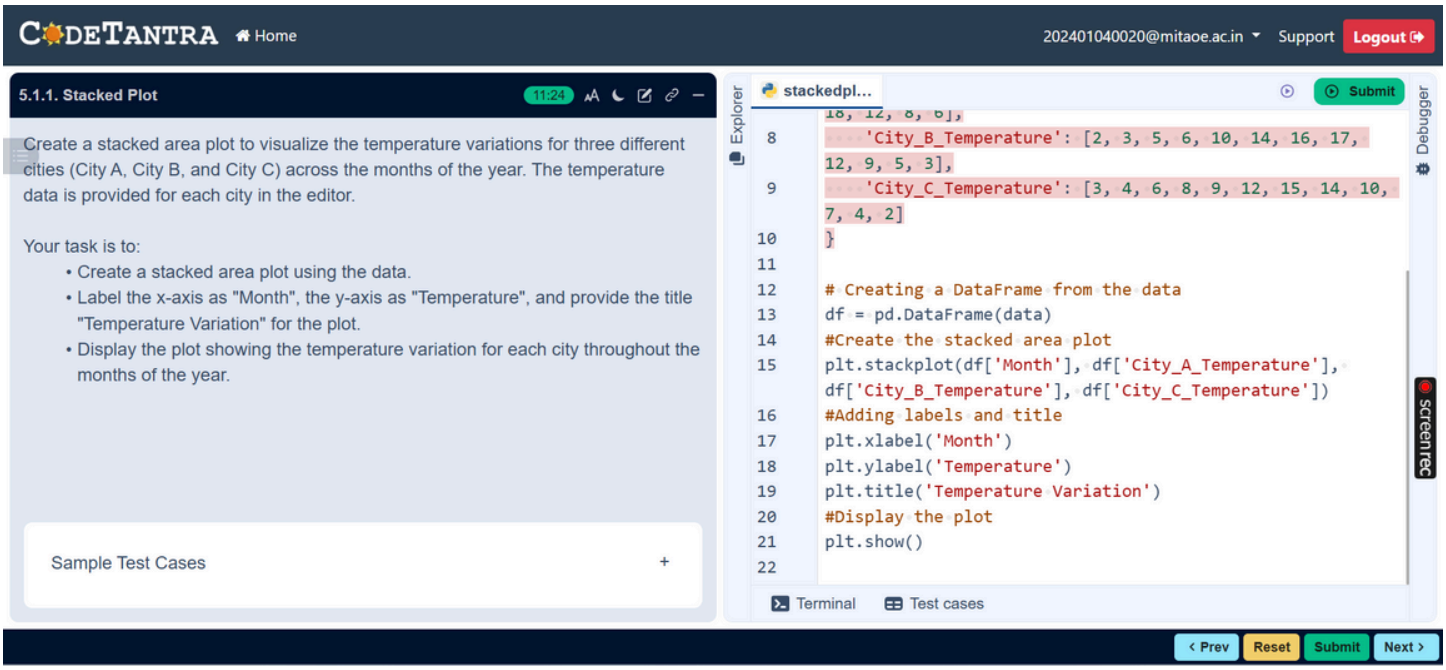
Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

Practical 5 Assignment 1 :





Practical 5 Assignment 2 :

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.2. Histogram of passenger information of Titanic

03:01

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,

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

Sample Test Cases +

Explorer Histogra... Submit Debugger

```
# Data Cleaning
data['Age'].fillna(data['Age'].median(), inplace=True)
data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'], drop_first=True)

# Write your code here for Histogram
plt.figure()
plt.hist(data['Age'], bins=30, edgecolor='k')
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

Terminal Test cases

screenrec

< Prev Reset Submit Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.3. Bar plot of survival rate of passengers

02:30

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,

Passeng	Survi	Pcla	Nam	Sex	Age	SibS	Parc	Ticke	Fare	Cabin	Embar

Sample Test Cases +

Explorer BarPlotOf... Submit Debugger

```
# Data Cleaning
data['Age'].fillna(data['Age'].median(), inplace=True)
data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True)
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'], drop_first=True)

# Write your code here for Bar Plot for Survival Rate
plt.figure()
data['Survived'].value_counts().plot(kind='bar')
plt.title('Survival Count')
plt.xlabel('Survived')
plt.ylabel('Count')
plt.show()
```

Terminal Test cases

screenrec

< Prev Reset Submit Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.4. Bar Plot for Survival by Gender

03:33

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,

P	S		N			Si					E

Sample Test Cases +

Explorer BarPlotOf... Submit Debugger

```
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'], drop_first=True)

# Write your code here for Bar Plot for Survival by Gender
plt.figure()
data.groupby('Sex')['Survived'].value_counts().unstack().plot(kind='bar', stacked=True)
plt.title('Survival by Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.legend(['Not Survived', 'Survived'])
plt.show()
```

Terminal Test cases

screenrec

< Prev Reset Submit Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.5. Bar Plot for Survival by Pclass

05:30

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,

Pas se	Sur	Pc	Na	S	A	Si b	P	Ti	Fa	C	Em

Sample Test Cases +

Explorer BarPlotOf... Submit Debugger

```
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'],
drop_first=True)

# Write your code here for Bar Plot for Survival by Pclass
plt.figure()
data.groupby('Pclass')
['Survived'].value_counts().unstack().plot(kind='bar',
stacked=True)
plt.title('Survival by Pclass')
plt.xlabel('Pclass')
plt.ylabel('Count')
plt.legend(['Not Survived', 'Survived'])
plt.show()
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.6. Bar Plot for Survival by Embarked

03:20

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,

Sample Test Cases +

Explorer BarPlotOf... Submit Debugger

```
# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'],
drop_first=True)

# Write your code here for Bar Plot for Survival by
Embarked
plt.figure()
data.groupby('Embarked_Q')
['Survived'].value_counts().unstack().plot(kind='bar',
stacked=True)
plt.title('Survival by Embarked')
plt.xlabel('Embarked')
plt.ylabel('Count')
plt.legend(['Not Survived', 'Survived'])
plt.show()
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.7. Box plot for Age Distribution

02:09

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,

Pas seng erI	Sur vive d	Pc las s	Na me	Sex	Age	Si b Sp	Par ch	Tic ket	Fa re	Cab in	Em bar k ed

Sample Test Cases +

Explorer BoxPlotF... Submit Debugger

```
data['Embarked'].fillna(data['Embarked'].mode()[0],
inplace=True)
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'],
drop_first=True)

# Write your code here for Box Plot for Age by Pclass
plt.figure()
data.boxplot(column='Age', by='Pclass')
plt.title('Age by Pclass')
plt.suptitle('')
plt.xlabel('Pclass')
plt.ylabel('Age')
plt.show()
```

Terminal Test cases

screenrec

‹ Prev Reset Submit Next ›

5.2.8. Box Plot for Age by Survived

02:16

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,

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

Sample Test Cases +

Explorer BoxPlotF... Submit Debugger

```
inplace=True)
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'],
drop_first=True)

# Write your code here for Box Plot for Age by Survived
plt.figure()
data.boxplot(column='Age', by='Survived')
plt.title('Age by Survival')
plt.suptitle('')
plt.xlabel('Survived')
plt.ylabel('Age')
plt.show()
```

Terminal Test cases

screenrec

< Prev Reset Submit Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.9. Box Plot for Fare by Pclass

02:04

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,

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

Sample Test Cases +

Explorer BoxPlotF... Submit Debugger

```
inplace=True)
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'],
drop_first=True)

# Write your code here for Box Plot for Fare by Pclass
plt.figure()
data.boxplot(column='Fare', by='Pclass')
plt.title('Fare by Pclass')
plt.suptitle('')
plt.xlabel('Pclass')
plt.ylabel('Fare')
plt.show()
```

Terminal Test cases

screenrec

< Prev Reset Submit Next >

CODETANTRA Home 202401040020@mitaoe.ac.in Support Logout

5.2.10. Scatter Plot for Age vs. Fare

02:52

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,

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

Sample Test Cases +

Explorer AgeFareS... Submit Debugger

```
data['Embarked'].fillna(data['Embarked'].mode()[0],
inplace=True)
data.drop('Cabin', axis=1, inplace=True)

# Convert categorical features to numeric
data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
data = pd.get_dummies(data, columns=['Embarked'],
drop_first=True)

# Write your code here for Box Plot for Fare by Pclass
plt.figure()
plt.scatter(data['Age'], data['Fare'])
plt.title('Age vs. Fare')
plt.xlabel('Age')
plt.ylabel('Fare')
plt.show()
```

Terminal Test cases

screenrec

< Prev Reset Submit Next >

5.2.11. Scatter Plot for Age vs. Fare by Survived

03:22

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,

Passeng	Survi	Pcla	Nam	Sex	Age	SibS	Parch	Ticket	Fare	Cabin	Embar

Sample Test Cases +

Explorer

AgeFareS...

Submit

Debugger

```
11
12  # Convert categorical features to numeric
13  data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
14  data = pd.get_dummies(data, columns=['Embarked'],
    drop_first=True)
15
16  # Write your code here for Scatter Plot for Age vs. Fare
    by Survived
17  plt.figure()
18  colors = {0: 'red', 1: 'blue'}
19  plt.scatter(data['Age'], data['Fare'],
    c=data['Survived'].apply(lambda x:colors[x]))
20  plt.title('Age vs. Fare by Survival')
21  plt.xlabel('Age')
22  plt.ylabel('Fare')
23  plt.show()
24
25
26
```

Terminal Test cases

screenrec

< Prev Reset Submit