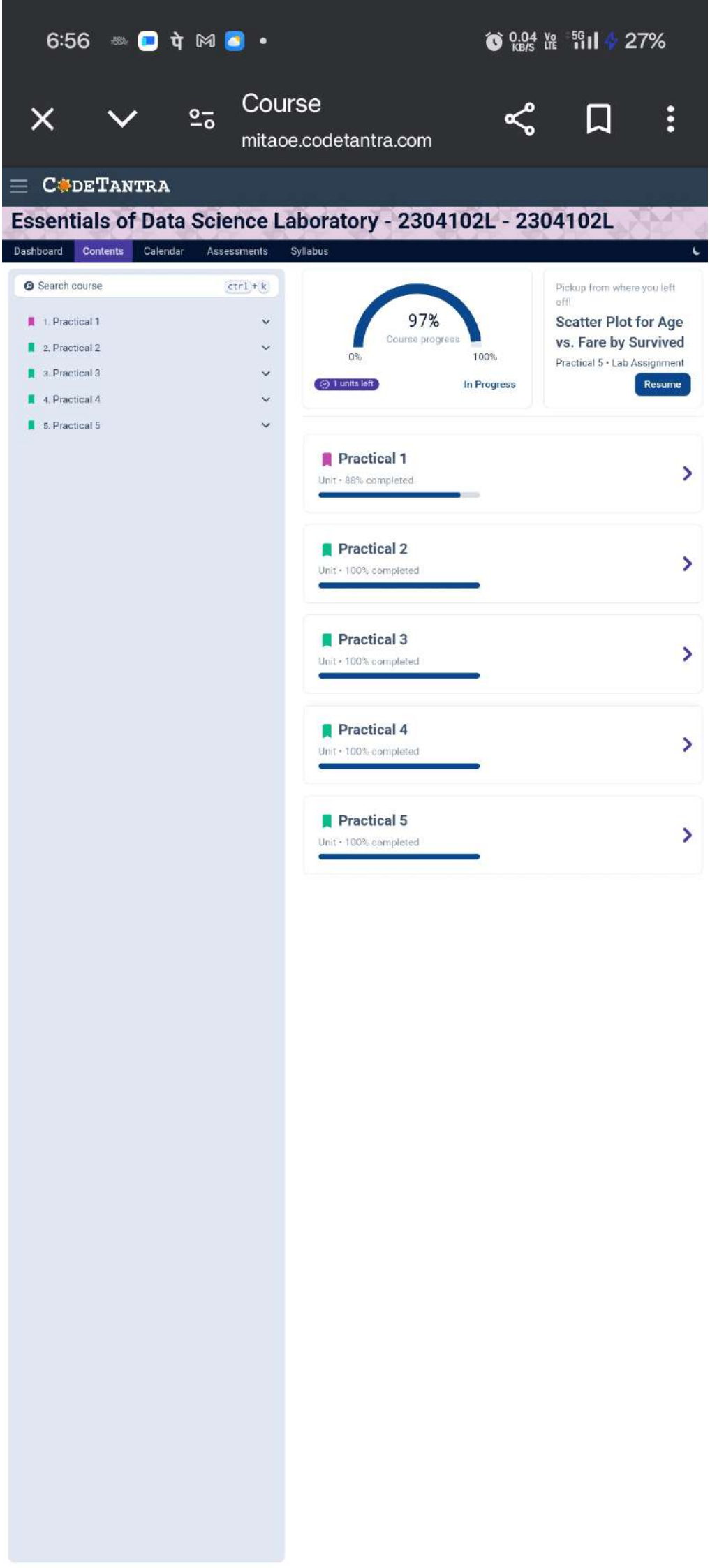

6:56
27%
Course
mitaoe.codetantra.com
CODETANTRA
Essentials of Data Science Laboratory - 2304102L - 2304102L
Dashboard
Contents
Calendar
Assessments
Syllabus
Search course
ctrl + k
1. Practical 1
2. Practical 2
3. Practical 3
4. Practical 4
5. Practical 5
97%
Course progress
0%
100%
1 units left
In Progress
Pickup from where you left off!
Scatter Plot for Age vs. Fare by Survived
Practical 5 • Lab Assignment
Resume
Practical 1
Unit • 88% completed
Practical 2
Unit • 100% completed
Practical 3
Unit • 100% completed
Practical 4
Unit • 100% completed
Practical 5
Unit • 100% completed

1.1.1. Calcula... 05:55

Write a program that accepts the mass of an object (in kilograms) and its velocity (in meters per second), then calculates and displays the momentum of the object. The momentum p is calculated using the formula:

$$p =$$

where:
m is the mass of the object (in kilograms).
v is the velocity of the object (in meters per second).

Input Format:
A single floating-point number representing the mass of the object in kilograms.
A single floating-point number representing the velocity of the object in meters per second.

Output Format:
The output will display calculated momentum with appropriate units (kgm/s) (rounded up to 2 decimal places).

Sample Test Cases +

Explorer

calculateM... Submit

```
1 m=float(input())
2 v=float(input())
3 p=m*v
4 print("%.2fkgm/s"%p)
```

Debugger

< Prev Reset Submit Next >

1.1.2. Conditi... 29:58

Write a Python program that accepts an integer n as input. Depending on the number of digits in n.

Constraints:
$1 \le n \le 999$

Input Format:
The input consists of a single integer n.

Output Format:
If n is a single-digit number, print its square.
If n is a two-digit number, print its square root (rounded to two decimal places).
If n is a three-digit number, print its cube root (rounded to two decimal places).
Else print "Invalid".

Sample Test Cases +

Explorer

conditional... Submit

```
1  n=int(input())
2  if (n>=0 and n<=9):
3      print(n*n)
4  elif (n>=10 and n<=99):
5      q=n**0.5
6      print('%0.2f'%q)
7  elif (n >= 100 and n<=999):
8      r=n**(1/3)
9      print('%.2f'%r)
10 else:
11     print("Invalid")
```

Debugger

1.1.4. Reverse a Number

14:51

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

reverseNu...

Submit

```
1 num=int(input())
2 n=str(num)
3 print(n[::-1])
```

Debugger

1.1.5. Multiplication Table

04:00

Write a Python program that takes an integer as input and prints the multiplication table for that integer from 1 to 10.

Input Format:
The first line of input contains an integer that represents the number for which the multiplication table is to be printed.

Output Format:
Print the multiplication table for the given number .

Sample Test Cases +

Explorer

multiplicat...

Submit

```
1  i=int(input())
2  n=1
3  while n<=10:
4      print(i,"x",n,"=",i*n)
5      n+=1
```

Debugger

1.2.1. Pass o... 49:23

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.

If the student fails any course (marks < 40 in any course), print:

- "Fail".

Sample Test Cases +

Explorer

passorFail.... Submit

Debugger

```python
def calculate_grade(num_courses, marks):
    if any(mark < 40 for mark in marks):
        return "Fail"

    total_marks = sum(marks)
    aggregate_percentage = (total_marks / (num_courses * 100)) * 100

    if aggregate_percentage > 75:
        grade = "Distinction"
    elif 60 <= aggregate_percentage <= 75:
        grade = "First Division"
    elif 50 <= aggregate_percentage < 60:
        grade = "Second Division"
    elif 40 <= aggregate_percentage < 50:
        grade = "Third Division"
    else:
        grade = "Fail"

    return f"Aggregate Percentage: {aggregate_percentage:.2f}\nGrade: {grade}"


num_courses = int(input())
marks = list(map(int, input().split()))


result = calculate_grade(num_courses, marks)
print(result)
```

Essentials of Data Science Laboratory - 2304102L

Pin

Search course ctrl + k

1.2.2. Fibona...

02:17

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

```
1  def fib(i):
2      if(i==0):
3          return 0
4      elif(i==1):
5          return 1
6      else:
7          return fib(i-1)+fib(i-2)
8
9
10
11
12
13
14
15
16
17
18     n=int(input("Enter terms for Fibonacci series: "))
19     for i in range(n):
20         print(fib(i),end=" ")
```

< Prev | Reset | Submit | Next >

1.2.3. Pattern... 01:21

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

rightangle... Submit

```
1  number = int(input())
2  for i in range(1,number+1):
3      print("* " * i)
```

Debugger

< Prev Reset Submit Next >

1.2.4. Pattern... 04:43

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

numberPat... Submit

```
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()
```

Debugger

< Prev Reset Submit Next >

2.1.1. List op... 55:50

Write a Python program that implements a menu-driven interface for managing a list of integers. The program should have the following menu options:
1. Add
2. Remove
3. Display
4. Quit

The program should repeatedly prompt the user to enter a choice from the menu. Depending on the choice selected, the program should perform the following actions:

- **Add**: Prompts the user to enter an integer and add it to the integer list. If the input is not a valid integer, display "Invalid input".
- **Remove:** Prompts the user to enter an integer to remove from the list. If the integer is found in the list, remove it; otherwise, display "Element not found". If the list is empty, display "List is empty".
- **Display**: Displays the current list of integers. If the list is empty, display "List is empty".
- **Quit**: Exits the program.
- The program should handle invalid menu choices by displaying "Invalid choice".Ensure that the program continues to prompt the user until they choose to quit (option 4).

Sample Test Cases +

Explorer

listOps.py Submit

```
list=[]
while True:

    print("1. Add")
    print("2. Remove")
    print("3. Display")
    print("4. Quit")

    n=int(input("Enter choice: "))

    if n==1:

        i=int(input("Integer: "))

        list.append(i)

        print(f"List after adding: {list}")
    elif n==2:
        if list==[]:

            print("List is empty")
        else:

            i=int(input("Integer: "))

            val=True
            for j in list:
                if j==i:

                    list.remove(j)
                    print(f"List after removing: {list}")
                    val=False
                    break
            if val:

                print("Element not found")



    elif n==3:
        if list==[]:

            print("List is empty")
        else:

            print(list)
    elif n==4:
        break
    else:

        print("Invalid choice")
```

Debugger


6:58
5.00 KB/S
28%
CODETANTRA
Essentials of Data Science Laboratory - 2304102L
Pin
Search course
ctrl + k
1. Practical 1
1.1. Practice Lab Assignment
1.2. Lab Assignment
1.2.1. Pass or Fail
1.2.2. Fibonacci series using Recursive Func...
1.2.3. Pattern - 1
1.2.4. Pattern - 2
2. Practical 2
2.1. Practice Lab Assignment
2.1.1. List operations
2.1.2. Dictionary Operations
2.2. Lab Assignment
3. Practical 3
4. Practical 4
5. Practical 5
2.1.2. Diction...
17:09
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
dictOperati...
Submit
Debugger
1 def main():
2 # Create an empty dictionary
3 dictionary = {}
4 print(f"Empty Dictionary: {dictionary}")
5
6 # Ask the user how many items to add
7 num_items = int(input("Number of items: "))
8
9 # Input key-value pairs
10 for _ in range(num_items):
11 key = input("key: ")
12 value = input("value: ")
13 dictionary[key] = value
14
15 # Show the dictionary after adding items
16 print(f"Dictionary: {dictionary}")
17
18 # Update a key's value
19 update_key = input("Enter the key to update: ")
20 if update_key in dictionary:
21 new_value = input("Enter the new value: ")
22 dictionary[update_key] = new_value
23 print("Value updated")
24 else:
25 print("Key not found")
26
27 # Retrieve and print a value using a key
28 retrieve_key = input("Enter the key to retrieve: ")
29 if retrieve_key in dictionary:
30 print(f"Key: {retrieve_key}, Value: {dictionary[retrieve_key]}")
31 else:
32 print("Key not found")
33
< Prev
Reset
Submit
Next >

2.2.1. Linear ... 00:40

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 +

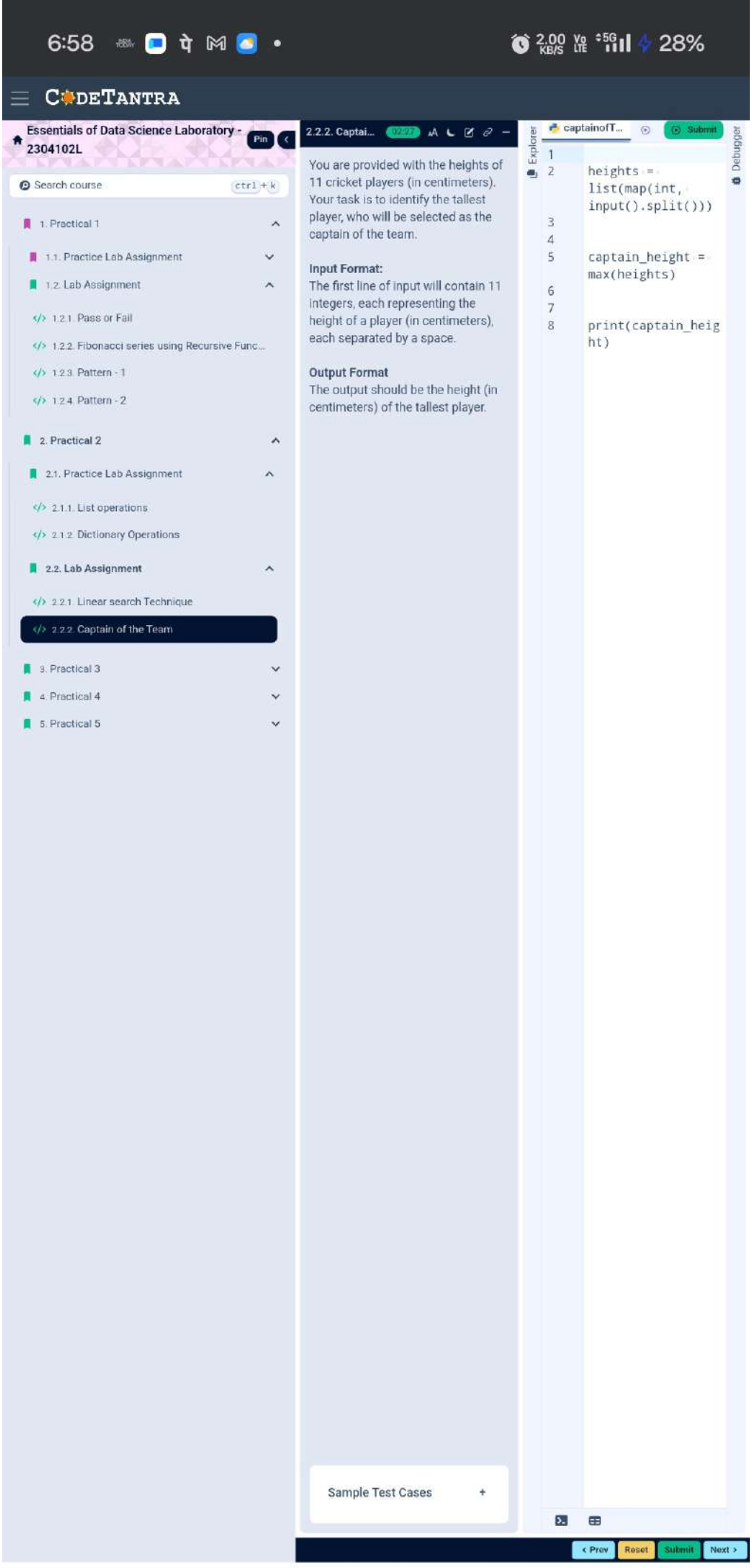
Explorer

CTP17092... Submit

```
1  list=list(map(int, input().split()))
2  n=int(input())
3  for i in list:
4      if(i==n):
5          print(list.index(i))
6          break
7
8  else:
9      print("Not found")
```

Debugger

< Prev Reset Submit Next >


6:58
2.00 KB/S
28%
CODETANTRA
Essentials of Data Science Laboratory - 2304102L
Pin
Search course
ctrl + k
1. Practical 1
1.1. Practice Lab Assignment
1.2. Lab Assignment
1.2.1. Pass or Fail
1.2.2. Fibonacci series using Recursive Func...
1.2.3. Pattern - 1
1.2.4. Pattern - 2
2. Practical 2
2.1. Practice Lab Assignment
2.1.1. List operations
2.1.2. Dictionary Operations
2.2. Lab Assignment
2.2.1. Linear search Technique
2.2.2. Captain of the Team
3. Practical 3
4. Practical 4
5. Practical 5
2.2.2. Captai...
02:27
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
captainofT...
Submit
Debugger
1
2 heights = list(map(int, input().split()))
3
4
5 captain_height = max(heights)
6
7
8 print(captain_height)
< Prev
Reset
Submit
Next >

3.1.1. Numpy... 24:20

Write a python program to demonstrate the usage of ndim, shape and size for a Numpy Array. The program should create a NumPy array using the entered elements and display it. Assume all input elements are valid numeric values.

Input Format:

- User inputs the number of rows and columns with space separated values.
- User inputs elements of the array row-wise followed line by line, separated by spaces.

Output Format:

- The created NumPy array based on the input dimensions and elements.
- Dimensions (ndim): Number of dimensions of the array.
- Shape: Tuple representing the shape of the array (number of rows, number of columns).
- Size: Total number of elements in the array.

Note: Use reshape() function to reshape the input array with the specified number of rows and columns.

Sample Test Cases +

Explorer

numpyarr.py

Submit

Debugger

```
1  import numpy as np
2
3  rows,cols=map(int,input().split())
4
5  elements=[]
6  for _ in range(rows):
7      elements.extend(map(int,input().split()))
8  array=np.array(elements).reshape(rows,cols)
9
10 print(array)
11
12 print(array.ndim)
13 print(array.shape)
14 print(array.size)
```

< Prev Reset Submit Next >

3.2.1. Numpy... 77/79

The given code takes two 3×3 matrices, `matrix_a`, and `matrix_b`, as input from the user and converts them into NumPy arrays.

Task:

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

1. **Addition** (`matrix_a + matrix_b`)
2. **Subtraction** (`matrix_a - matrix_b`)
3. **Element-wise Multiplication** (`matrix_a * matrix_b`)
4. **Matrix Multiplication** (`matrix_a · matrix_b`)
5. **Transpose of Matrix A**

Input Format:

- The user will input 3 rows for `matrix_a`, each containing 3 integers separated by spaces.
- Similarly, the user will input 3 rows for `matrix_b`, each containing 3 integers separated by spaces.

Output Format:

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

1. The result of Addition.
2. The result of Subtraction.
3. The result of Element-wise Multiplication.
4. The result of Matrix Multiplication.
5. The Transpose of Matrix A.

Sample Test Cases +

Explorer

matrixOpe... Submit

Debugger

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

3.2.2. Numpy...

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

```python
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)
a=np.hstack((arr1,arr2))
print("Horizontal Stack:")
print(a)

print("Vertical Stack:")
b=np.vstack((arr1,arr2))
print(b)


# Perform vertical stacking (vstack)

```

3.2.3. Numpy... 05:27

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

customSe... Submit

Debugger

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

3.2.4. Numpy... 01:29

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 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

differentO... Submit

Debugger

```python
import numpy as np

def array_operations(A, B):

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

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

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

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

< Prev Reset Submit Next >

3.2.5. Numpy... 07:22

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 array: <view_array>
```

- After modifying the copy:

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

Sample Test Cases +

Explorer

copyAndvi... Submit

Debugger

```python
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)

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

< Prev Reset Submit Next >

3.2.6. Numpy... 04:21

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

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

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

Input Format:

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

Output Format:

1. The indices where `search_value` occurs in `array1`.
2. The count of occurrences of `count_value` in `array1`.
3. The array after adding the `broadcast_value` to each element.
4. The sorted array.

Sample Test Cases +

Explorer

arrayOpera... 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
a=np.where(array1==search_value)[0]
print(a)
# Count occurrences in array1
b=np.count_nonzero(array1==count_value)
print(b)
# Broadcasting addition
c=array1+broadcast_value
print(c)
# Sort the first array
d=np.sort(array1)
print(d)
```

< Prev Reset Submit Next >

3.2.7. Studen... 35:00

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

- **Print all student details**: Display the complete details of all students, including roll numbers and marks for all subjects.
- **Find total students**: Determine the total number of students in the dataset.
- **Print all student roll numbers**: Extract and print the roll numbers of all students.
- **Print Subject 1 marks**: Extract and print the marks of all students in Subject 1.
- **Find minimum marks in Subject 2**: Identify the lowest marks in Subject 2.
- **Find maximum marks in Subject 3**: Identify the highest marks in Subject 3.
- **Print all subject marks**: Display the marks of all students for each subject.
- **Find total marks of students**: Compute the total marks for each student across all subjects.
- **Find the average marks of each student**: Compute the average marks for each student.
- **Find average marks of each subject**: Compute the average marks for all students in each subject.
- **Find average marks of Subject 1 and Subject 2**: Compute the average marks for Subject 1 and Subject 2.
- **Find average marks of Subject 1 and Subject 3**: Compute the average marks for Subject 1 and Subject 3.
- **Find the roll number of the student with maximum marks in Subject 3**: Identify the student with the highest marks in Subject 3 and print their roll number.
- **Find the roll number of the student with minimum marks in Subject 2**: Identify the student with the lowest marks in Subject 2 and print their roll number.
- **Find the roll number of students who scored 24 marks in Subject 2**: Identify students who obtained exactly 24 marks in Subject 2 and print their roll numbers.
- **Find the count of students who got less than 40 marks in Subject 1**: Count the number of students who scored less than 40 marks in Subject 1.

Sample Test Cases +

Explorer

Operations... 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
8   # 2. print total students
9
10  print("Total Students:",len(a))
11
12  # 3. Print all student Roll numbers
13  print("All Student Roll Nos",a[:,0])
14
15  # 4. Print subject 1 marks
16  print("Subject 1 Marks", a[:,1])
17
18  # 5. print minimum marks of Subject 2
19  print("Min marks in Subject 2", a[:,2].min())
20
21  # 6. print maximum marks of Subject 3
22  print("Max marks in Subject 3", a[:,3].max())
23
24  # 7. Print All subject marks
25  a1=np.delete(a,0,1)
26  print("All subject marks:", a1)
27
28  # 8. print Total marks of students
29  print("Total Marks", a[:,1]+a[:,2]+a[:,3])
30
31
32  # 9. print average marks of each student
33  a2=np.array(a[:,1]+a[:,2]+a[:,3])
34  A=np.around(a2/3,1)
35  print(A)
36
37  # 10. print average marks
```

< Prev Reset Submit Next >

4.1.1. Pandas... 19:22

Write a Python program that takes a list of numbers from the user, creates a Pandas series from it, and then calculates the mean of even and odd numbers separately using the **groupby** and **mean()** operations.

Input Format:

- The user should enter a list of numbers separated by space when prompted.

Output Format:

- The program should display the mean of even and odd numbers separately.
- Each mean value should be displayed with a label indicating whether it corresponds to even or odd numbers.

Sample Test Cases +

Explorer

seriesMani... Submit

Debugger

```python
import pandas as pd

# Take inputs from the user to create a list of numbers
numbers = list(map(int, input().split()))

# Creggggate a Pandas series from the list of numbers
S=pd.Series(numbers)

# Grouping by even and odd numbers and calculating the mean
grouped = S.groupby(S%2==0).mean()

# Display the mean of even and odd numbers with labels
grouped.index = ['Even' if is_even else 'Odd' for is_even in grouped.index]
print("Mean of even and odd numbers:")
print(grouped)
```

< Prev Reset Submit Next >

4.1.2. Diction... 55:38

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.
- Display the DataFrame after modifying the row.

Delete a row:
- Take the row index to be deleted from the user.
- Remove the specified row.
- Display the DataFrame after deleting the row.

Add a new column:
- Add a column **Gender** with values taken from the user.
- Display the DataFrame after adding the new column.

Modify a column:
- Convert names to uppercase.
- Display the DataFrame after modifying the column.

Delete a column:
- Remove the **Age** column.
- Display the DataFrame after deleting the column.

Sample Test Cases +

Explorer

dataframe... Submit

```python
import pandas as pd

# Provided dictionary of lists
data = {
    'Name': ['Alice', 'Bob', 'Charlie'],
    'Age': [25, 30, 35],
}

# Convert the dictionary to a DataFrame
df = pd.DataFrame(data)

# Display the original DataFrame
print("Original DataFrame:")
print(df)

# Adding a new row
name=input("New name: ")
age=int(input("New age: "))
Data={
    'Name': [name],
    'Age': [age]
}
D=pd.DataFrame(Data)

df=pd.concat([df,D],ignore_index=True)


# Display the DataFrame after adding a new row
print("After adding a row:\n",df)

# Modifying a row
a=int(input("Index of row to modify: "))
A=int(input("New age: "))
df.at[a,'Age']=A

# Display the DataFrame after modifying a row
print("After modifying a row:")
print(df)

# Deleting a row
n=int(input("Index of row to delete: "))
df=df.drop(index=n).reset_index(drop=True)
# Display the DataFrame after deleting row
```

Debugger

< Prev Reset Submit Next >

4.1.3. Studen... 08:12

Write a program to read a text file containing student information (name, age, and grade) using Pandas. Perform the following tasks:

- Display the first five rows of the data frame.
- Calculate the average age of the students(limit the average age up to 2 decimal places).
- Filter out the students who have a grade above a certain threshold(consider the threshold grade is 'B').

Note:
Refer to the displayed test cases for better understanding.

Sample Test Cases +

Explorer

studentinf... Submit

```
1  import pandas as pd
2
3  # Read the text file into a DataFrame
4  file = input()
5  data = pd.read_csv(file, sep="\s+", header=None, names=["Name", "Age", "Grade"])
6  print("First five rows:")
7  print(data.head())
8  avg= round(data['Age'].mean(),2)
9  print("Average age:",avg)
10 print("Students with a grade up to B")
11 print(data[data['Grade'].str.strip().str.upper()<='B'])
12
13
14 # write your code here..
15
16
```

Debugger

< Prev Reset Submit Next >

4.2.1. Month ... 01:39

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,Cit
2025-01-01,Product A,5,20,New Y
2025-01-01,Product B,3,15,Los A
2025-01-02,Product A,7,20,New Y
2025-01-02,Product C,4,30,Chica
2025-01-03,Product B,2,15,Chica
2025-01-03,Product A,8,20,Los A
2025-01-04,Product C,6,30,New Y
2025-01-04,Product B,5,15,Los A
2025-01-05,Product A,3,20,Chica
2025-01-05,Product C,10,30,Los
```

Note:
The data cannot be displayed in the file. You can refer to the sample data provided for insights.

Sample Test Cases +

Explorer

monthForS... 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)

df['Sales']=df['Quantity']*df['Price']
df['Month']=df['Date'].str[:7]
monthly_sales=df.groupby('Month')['Sales'].sum()
# Find the month with the highest total sales
best_month=monthly_sales.idxmax()
highest_sales = monthly_sales.max()

print(f"Best month: {best_month}")
print(f"Total sales: ${highest_sales:.2f}")
```

< Prev Reset Submit Next >

4.2.2. Best S... 01:39

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,Cit
2025-01-01,Product A,5,20,New \
2025-01-01,Product B,3,15,Los A
2025-01-02,Product A,7,20,New \
2025-01-02,Product C,4,30,Chica
2025-01-03,Product B,2,15,Chica
2025-01-03,Product A,8,20,Los A
2025-01-04,Product C,6,30,New \
2025-01-04,Product B,5,15,Los A
2025-01-05,Product A,3,20,Chica
2025-01-05,Product C,10,30,Los
```

Note:
The data cannot be displayed in the file. You can refer to the sample data provided for insights.

Sample Test Cases +

Explorer

monthForS... Submit

```python
import pandas as pd

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

# Load the data
df = pd.read_csv(file_name)

sold=df.groupby('Product')['Quantity'].sum()
# Find the product with the highest total quantity sold
best_product = sold.idxmax()
highest_quantity = sold.max()

# Display the result
print(f"Best selling product: {best_product}")
print(f"Total quantity sold: {highest_quantity}")

```

Debugger

< Prev Reset Submit Next >

4.2.3. City th... 01:39

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,Cit
2025-01-01,Product A,5,20,New \
2025-01-01,Product B,3,15,Los A
2025-01-02,Product A,7,20,New \
2025-01-02,Product C,4,30,Chica
2025-01-03,Product B,2,15,Chica
2025-01-03,Product A,8,20,Los A
2025-01-04,Product C,6,30,New \
2025-01-04,Product B,5,15,Los A
2025-01-05,Product A,3,20,Chica
2025-01-05,Product C,10,30,Los
```

Note:
The data cannot be displayed in the file. You can refer to the sample data provided for insights.

Sample Test Cases +

Explorer

monthForS... Submit

```python
import pandas as pd

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

# Load the data
df = pd.read_csv(file_name)

# write the code..
city=df.groupby('City')['Quantity'].sum()
best_city=city.idxmax()
# Display the result
print(f"City sold the most products: {best_city}")
```

Debugger

< Prev Reset Submit Next >

4.2.4. Most F... 07:40

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,Cit
2025-01-01,Product A,5,20,New \
2025-01-01,Product B,3,15,Los A
2025-01-02,Product A,7,20,New \
2025-01-02,Product C,4,30,Chica
2025-01-03,Product B,2,15,Chica
2025-01-03,Product A,8,20,Los A
2025-01-04,Product C,6,30,New \
2025-01-04,Product B,5,15,Los A
2025-01-05,Product A,3,20,Chica
2025-01-05,Product C,10,30,Los
```

Explanation:

Transactions:

- **2025-01-01**: Product A, Product B
- **2025-01-02**: Product A, Product C
- **2025-01-03**: Product B, Product A
- **2025-01-04**: Product C, Product B
- **2025-01-05**: Product A, Product C

Now, let's count how often the pairs of products appear together:

- **Product A and Product B**: Appear in transactions on 2025-01-01 and 2025-01-03.
- **Product A and Product C**: Appear in transactions on 2025-01-02 and 2025-01-05.
- **Product B and Product C**: Appears in transactions on 2025-01-04.

Most Frequent Product Combinations:

- **Product A and Product B** (2 times)
- **Product A and Product C** (2 times)

Note:

The data cannot be displayed in the file. You can refer to the sample data provided for insights.

Sample Test Cases +

Explorer

frequently... Submit

```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)

count_pairs=Counter()
# write the code

for date,group in df.groupby('Date'):
    products=group['Product']
    for pair in combinations(sorted(products),2):
        count_pairs[pair] +=1
# Output the most frequent product pairs.


best_count= max(count_pairs.values())
best_pair= [pair for pair,val in count_pairs.items() if val==best_count ]
for pair in best_pair:
    print(f"{pair[0]} and {pair[1]}: {best_count} times")
```

Debugger

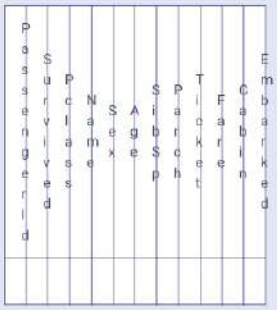
< Prev Reset Submit Next >

4.2.5. Titanic ... 04:11

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

titanicData... Submit

```python
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(5))

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

# 3. Get the shape of the dataset
print('(891, 12)')

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

# 5. Get basic statistics of the dataset
print(data.describe())

# 6. Check for missing values

print(data.isnull().sum())
# 7. Fill missing values in the 'Age' column with the median age


# 8. Fill missing values in the 'Embarked' column with the mode


# 9. Drop the 'Cabin' column due to many missing values


# 10. Create a new column 'FamilySize' by adding 'SibSp' and 'Parch'


```

Debugger

< Prev Reset Submit Next >

CODETANTRA

Essentials of Data Science Laboratory - 2304102L

Pin

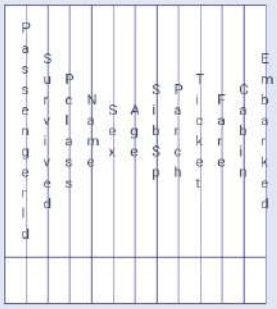
Search course ctrl + k

4.2.6. Titanic ... 03:45

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

titanicData... Submit

```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']

# 1. Create a new column 'IsAlone' (1 if alone, 0 otherwise)
def familysize(size):
    if size == 0:
        return 1
    else:
        return 0
data['IsAlone'] = data['FamilySize'].apply(familysize)


# 2. Convert 'Sex' to numeric (male: 0, female: 1)
def gender(sex):
    if sex == 'male':
        return 0
    else:
        return 1
data['Sex'] = data['Sex'].apply(gender)


# 3. One-hot encode the 'Embarked' column

# 4. Get the mean age of passengers
mean_val = data['Age'].mean()
print(mean_val)


# 5. Get the median fare of passengers
median_val = data['Fare'].median()
print(median_val)


# 6. Get the number of passengers by class
print(data['Pclass'].value_counts())

# 7. Get the
```

Debugger

< Prev Reset Submit Next >

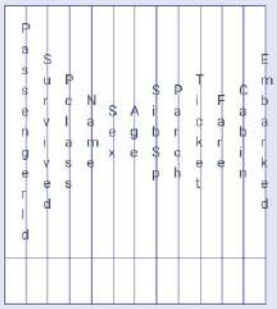
4.2.7. Titanic ...

01:57

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

titanicData... Submit

```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
grouped = data.groupby('Pclass')['Survived'].mean()
print(grouped)
# 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())

# 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')
```

Debugger

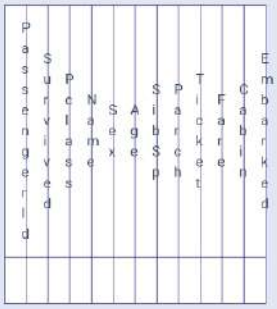
< Prev Reset Submit Next >

4.2.8. Titanic ... 08:59

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Brad
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

titanicData... Submit

Explorer 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  # 1. Get the number of survivors by gender
9  print(data[data['Survived']==1]['Sex'].value_counts())
10
11 # 2. Get the number of non-survivors by gender
12 print(data[data['Survived']==0]['Sex'].value_counts())
13
14 # 3. Get the number of survivors by embarked location
15 print(data[data['Survived']==1]['Embarked_S'].value_counts())
16
17 # 4. Get the number of non-survivors by embarked location
18 print(data[data['Survived']==0]['Embarked_S'].value_counts())
19
20 # 5. Calculate the percentage of children (Age < 18) who survived
21 print(data[data['Age']<18]['Survived'].mean())
22
23 # 6. Calculate the percentage of adults (Age >= 18) who survived
24 print(data[data['Age']>17]['Survived'].mean())
25
26 # 7. Get the median age of survivors
27 print(data[data['Survived']==1]['Age'].median())
28
29 # 8. Get the
```

< Prev Reset Submit Next >

5.1.1. Stacke... 04:05

Create a stacked area plot to visualize the temperature variations for three different cities (City A, City B, and City C) across the months of the year. The temperature data is provided for each city in the editor.

Your task is to:

- Create a stacked area plot using the data.
- Label the x-axis as "Month", the y-axis as "Temperature", and provide the title "Temperature Variation" for the plot.
- Display the plot showing the temperature variation for each city throughout the months of the year.

Sample Test Cases +

Explorer

stackedplo... Submit

```python
import matplotlib.pyplot as plt
import pandas as pd

# Data for Months and Temperature for three cities
data = {
    'Month': ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December'],
    'City_A_Temperature': [5, 7, 10, 13, 17, 20, 22, 21, 18, 12, 8, 6],
    'City_B_Temperature': [2, 3, 5, 6, 10, 14, 16, 17, 12, 9, 5, 3],
    'City_C_Temperature': [3, 4, 6, 8, 9, 12, 15, 14, 10, 7, 4, 2]
}

# Write your code...
df=pd.DataFrame(data)
plt.stackplot(df['Month'],df['City_A_Temperature'],df['City_B_Temperature'],df['City_C_Temperature'])

plt.xlabel('Month')

plt.ylabel('Temperature')

plt.title('Temperature Variation')

plt.show()
```

Debugger

Essentials of Data Science Laboratory - 2304102L

Pin

Search course ctrl + k

- 2.1.2. Dictionary Operations
- 2.2. Lab Assignment
  - 2.2.1. Linear search Technique
  - 2.2.2. Captain of the Team
- 3. Practical 3
  - 3.1. Practice Lab Assignment
    - 3.1.1. Numpy array operations
  - 3.2. Lab Assignment
    - 3.2.1. Numpy: Matrix Operations
    - 3.2.2. Numpy: Horizontal and Vertical Stack...
    - 3.2.3. Numpy: Custom Sequence Generation
    - 3.2.4. Numpy: Arithmetic and Statistical Op...
    - 3.2.5. Numpy: Copying and Viewing Arrays
    - 3.2.6. Numpy: Searching, Sorting, Counting, ...
    - 3.2.7. Student Data Analysis and Operations
- 4. Practical 4
  - 4.1. Practice Lab Assignment
    - 4.1.1. Pandas - series creation and manipul...
    - 4.1.2. Dictionary to dataframe
    - 4.1.3. Student Information
  - 4.2. Lab Assignment
    - 4.2.1. Month with the Highest Total Sales
    - 4.2.2. Best Selling Product
    - 4.2.3. City that Sold the Most Products
    - 4.2.4. Most Frequently Sold Product Pairs
    - 4.2.5. Titanic Dataset Analysis and Data Cle...
    - 4.2.6. Titanic Dataset Analysis and Data Cle...
    - 4.2.7. Titanic Dataset Analysis and Data Cle...
    - 4.2.8. Titanic Dataset Analysis and Data Cle...
- 5. Practical 5
  - 5.1. Practice Lab Assignment
    - 5.1.1. Stacked Plot
  - 5.2. Lab Assignment
    - 5.2.1. Titanic Dataset
    - 5.2.2. Histogram of passenger information ...
    - 5.2.3. Bar plot of survival rate of passengers
    - 5.2.4. Bar Plot for Survival by Gender
    - 5.2.5. Bar Plot for Survival by Pclass
    - 5.2.6. Bar Plot for Survival by Embarked
    - 5.2.7. Box plot for Age Distribution
    - 5.2.8. Box Plot for Age by Survived
    - 5.2.9. Box Plot for Fare by Pclass
    - 5.2.10. Scatter Plot for Age vs. Fare
    - 5.2.11. Scatter Plot for Age vs. Fare by Survi...

5.2.1. Titanic ... 05:53

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:

Write the code to create a series of visualizations as follows:

Bar Plot (Pclass Distribution):

- Create a bar plot to show the distribution of passengers across the different passenger classes (`Pclass`).
- Use the color `skyblue` for the bars.
- Title the plot as **"Passenger Class Distribution"**.
- Label the x-axis as **"Pclass"** and the y-axis as **"Count"**.

Pie Chart (Gender Distribution):

- Create a pie chart to display the distribution of male and female passengers.
- Use `lightblue` for males and `lightcoral` for females.
- Include percentages on the slices (use `autopct='%1.1f%%'`).
- Title the plot as **"Gender Distribution"**.

Histogram (Age Distribution):

- Create a histogram to visualize the distribution of passengers' ages.
- Use `lightgreen` for the bars with black edges (`edgecolor = 'black'`).
- Set the number of bins to **8** for the histogram.
- Title the plot as **"Age Distribution"**.
- Label the x-axis as **"Age"** and the y-axis as **"Frequency"**.

Bar Plot (Survival Count):

- Create a bar plot to show the count of passengers who survived and those who did not, based on the `Survived`

Sample Test Cases +

Explorer

titanicData... 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..
11 count=df['Pclass'].value_counts().sort_index()
12 count_gen=df['Gender'].value_counts()
13 count_s=df['Survived'].value_counts().sort_index()
14
15 axes[0,0].bar(count.index,count.values,color=['skyblue'])
16 axes[0,0].set_title('Passenger Class Distribution')
17 axes[0,0].set_xlabel('Pclass')
18 axes[0,0].set_ylabel('Count')
19
20 axes[0,1].pie(count_gen.values,labels=['male','female'],colors=['lightblue','lightcoral'],autopct='%1.1f%%')
21 axes[0,1].set_title('Gender Distribution')
22
23 axes[1,0].hist(df['Age'],color=['lightgreen'],edgecolor='black',bins=8)
24 axes[1,0].set_title('Age Distribution')
25 axes[1,0].set_xlabel('Age')
26 axes[1,0].set_ylabel('Frequency')
27
28 axes[1,1].bar(count_s.index,count_s.values,color=['lightblue','lightcoral'])
29 axes[1,1].set_title('Survival Count')
30 axes[1,1].set_xla
```

< Prev Reset Submit Next >

5.2.2. Histogr... 01:40

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,

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

Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

Histogram... 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)
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 Histogram
17 # Write your code here for Histogram
18 plt.hist(data['Age'].dropna(),bins=30,edgecolor='k')
19 plt.xlabel('Age')
20 plt.ylabel('Frequency')
21 plt.title('Age Distribution')
22
23 plt.show()
```

< Prev Reset Submit Next >

5.2.3. Bar plo... 05:02

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,

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

Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris'
2,1,1,"Cumings, Mrs. John Brad
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

BarPlotOfS... Submit

```
'''import pandas as pd
import matplotlib.pyplot as plt

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

# 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

'''
import pandas as pd
import matplotlib.pyplot as plt

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

# 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})
```

Debugger

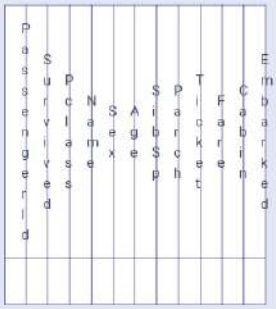
< Prev Reset Submit Next >

5.2.4. Bar Plo...

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Brad
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

BarPlotOfS... 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)
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 # Write your code here for Bar Plot for Survival by Gender
16 grouped = pd.crosstab(data['Sex'],data['Survived'])
17
18 grouped.plot(kind='bar',stacked=True)
19 plt.xlabel('Gender')
20 plt.ylabel('Count')
21 plt.title('Survival by Gender')
22 plt.legend(['Not Survived','Survived'])
23
24 plt.show()
```

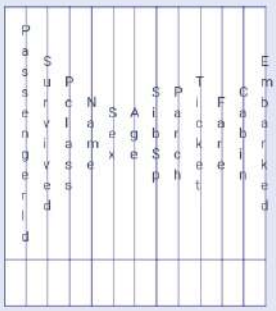
< Prev Reset Submit Next >

5.2.5. Bar Plo... 03:56

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note:

- Refer to the visible test case for better reference.
- Ensure you use the **groupby()** function with **value_counts()** to count the survivors and non-survivors for each **Pclass**.
- Do **not** manually use **size()** or **unstack()** without **value_counts()**. Use the **value_counts()** method for counting survival status directly.

Sample Test Cases +

Explorer

BarPlotOfS... 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)
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 # Write your code here for Bar Plot for Survival by Pclass
16 grouped = pd.crosstab(data['Pclass'],data['Survived'])
17 grouped.plot(kind='bar',stacked=True)
18
19 plt.xlabel('Pclass')
20 plt.ylabel('Count')
21 plt.title('Survival by Pclass')
22 plt.legend(['Not Survived','Survived'])
23
24 plt.show()
```

< Prev Reset Submit Next >

5.2.6. Bar Plo...

02:08

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 Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

BarPlotOfS... 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)
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 Bar Plot for Survival by Embarked
17
18
19 # Write your code here for Bar Plot for Survival by Embarked
20 grouped = pd.crosstab(data['Embarked_Q'],data['Survived'])
21 grouped.plot(kind='bar',stacked=True)
22
23 plt.xlabel('Embarked')
24 plt.ylabel('Count')
25 plt.title('Survival by Embarked')
26 plt.legend(['Not Survived','Survived'])
27
28 plt.show()
```

< Prev Reset Submit Next >

5.2.7. Box plo... 04:18

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,

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

Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris'
2,1,1,"Cumings, Mrs. John Brad
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

BoxPlotFor... 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)
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 Box Plot for Age by Pclass
17
18 plt.figure()
19 data.boxplot(column='Age',by='Pclass')
20 plt.title('Age by Pclass')
21 plt.suptitle('')
22 plt.xlabel("Pclass")
23 plt.ylabel("Age")
24 plt.show()
25
```

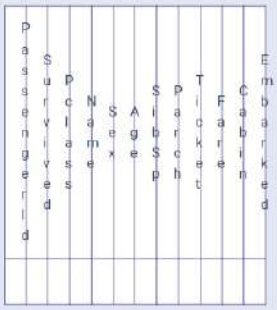
< Prev Reset Submit Next >

5.2.8. Box Pl... 01:38

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

BoxPlotFor... 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)
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 Box Plot for Age by Survived
17
18 plt.figure()
19 data.boxplot(column='Age',by='Survived')
20 plt.title('Age by Survival')
21 plt.suptitle('')
22 plt.xlabel("Survived")
23 plt.ylabel("Age")
24 plt.show()
25
```

< Prev Reset Submit Next >

5.2.9. Box Pl...

02:03

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,

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

Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

BoxPlotFor... Submit

Debugger

```
import pandas as pd
import matplotlib.pyplot as plt

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

# 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 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()
```

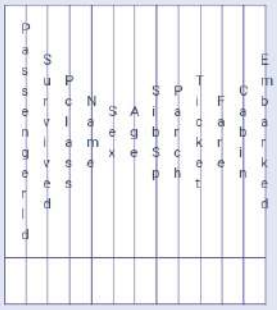
< Prev Reset Submit Next >

5.2.10. Scatt... 01:50

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Brad
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

AgeFareSc... Submit

Explorer Debugger

```
1  import pandas as
   pd
2  import
   matplotlib.pyplot
    as plt
3
4  # Load the
   Titanic dataset
5  data =
   pd.read_csv('Tita
   nic-Dataset.csv')
6
7  # Data Cleaning
8  data['Age'].fillN
   a(data['Age'].med
   ian(),
   inplace=True)
9  data['Embarked'].
   fillna(data['Emba
   rked'].mode()
   [0],
   inplace=True)
10 data.drop('Cabin'
   , axis=1,
   inplace=True)
11
12 # Convert
   categorical
   features to
   numeric
13 data['Sex'] =
   data['Sex'].map({
   'male': 0,
   'female': 1})
14 data =
   pd.get_dummies(da
   ta, columns=
   ['Embarked'],
   drop_first=True)
15
16 # Write your
   code here for
   Box Plot for
   Fare by Pclass
17
18 plt.scatter(data[
   'Age'],data['Fare
   '])
19 plt.title('Age
   vs. Fare')
20 plt.xlabel('Age')
21 plt.ylabel('Fare'
   )
22
23 plt.show()
24
```

CODETANTRA

Essentials of Data Science Laboratory - 2304102L

Pin

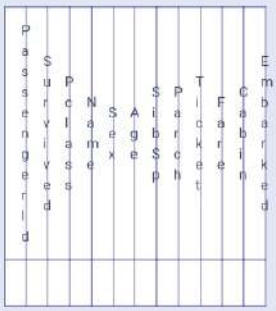
Search course ctrl + k

5.2.11. Scatt... 02:42

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,



Sample Data:

```
PassengerId,Survived,Pclass,Nan
1,0,3,"Braund, Mr. Owen Harris"
2,1,1,"Cumings, Mrs. John Bradl
3,1,3,"Heikkinen, Miss. Laina",
4,1,1,"Futrelle, Mrs. Jacques H
5,0,3,"Allen, Mr. William Henry
6,0,3,"Moran, Mr. James",male,,
7,0,1,"McCarthy, Mr. Timothy J"
8,0,3,"Palsson, Master. Gosta L
9,1,3,"Johnson, Mrs. Oscar W (E
10,1,2,"Nasser, Mrs. Nicholas (
```

Note: Refer to the visible test case for better reference.

Sample Test Cases +

Explorer

AgeFareSc... 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)
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 color_map = {0: 'red', 1: 'blue'}
18 colors = data['Survived'].apply(lambda x: color_map[x])
19
20 # Single scatter plot
21 plt.figure()
22 plt.scatter(data['Age'], data['Fare'], c=colors)
23 plt.title('Age vs. Fare by Survival')
24 plt.xlabel('Age')
25 plt.ylabel('Fare')
26 plt.show()
27
28
```

< Prev Reset Submit