Essentials of Data Science Laboratory - 2304102L Search course 1. Practical 1





Practical 1

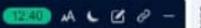
About this unit

Practical 1



Lab Assignment Unit • 100% completed

1.1.1. Calculate Momentum



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:

$$p = m \times v$$

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).

```
Explorer
   calculate...
          m=float(input())
          v=float(input())
          p=m*v
          print('%0.2f'%p,end='')
          print("kgm/s")
    >. Terminal
                 ## Test cases
```





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".

```
@ condition...

    Submit

       n=int(input())
       if(n>=0 and n<=9):
           print(n*n)
       elif(n>=10 and n<=99):
           p=n**0.5
           print( "%0.2f"%p)
       elif(n>=100 and n<=999):
           r=n**(1/3)
10
           print("%.2f"%r)
11
12
13
       else:
14
           print("Invalid")
 > Terminal

    ⊞ Test cases
```



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

```
birthDate...
                                                                         ( Submit
      from datetime import datetime
      def calculate age(birthdate):
           date object = datetime.strptime(birthdate, "%d-%m-%Y")
           today = datetime.today()
           age = today.year - date_object.year
          if (today.month, today.day) < (date_object.month, date_object.day):</pre>
              age -= 1
9
          return age
10
11
12
     , def convert_salary_to_dollars(salary_in_rupees):
13
          return salary in rupees * 0.012
14
15
16
      birthdate = input()
      salary_in_rupees = float(input())
17
      age = calculate age(birthdate)
18
19
       salary_in_dollars = convert_salary_to_dollars(salary_in_rupees)
20
      print(f"Age: {age}")
      print(f"Salary in dollars: {salary in dollars:.2f}")
21
22
            Test cases
 > Terminal
```



1.1.4. Reverse a Number



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.

Sample Test Cases

Output Format

Print a single integer which is the reversed number.

reverseN... number = int(input()) reverse=0 , while number != 0: digit = number % 10 reverse = reverse*10 + digit number = number//10 print(reverse) 10 > Terminal ☐ Test cases Scanned with OKEN Scanner



1.1.5. Multiplication Table



Write a Multiplication Table 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 .







Pass or Fail

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:

```
passorFa...
       n = int(input())
       marks = list(map(int,input().split()))
      if any(mark < 40 for mark in marks):
           print("Fail")
     . else:
           totalmarks = sum(marks)
           aggregate = (totalmarks/(n*100))*100
           print(f"Aggregate Percentage: {aggregate:.2f}")
10
11
           if aggregate > 75:
               print("Grade: Distinction")
12
13
           elif aggregate > 60 and aggregate < 75:
               print("Grade: First Division")
14
           elif aggregate > 50 and aggregate < 60:
15
16
               print("Grade: Second Division")
           elif aggregate > 40 and aggregate < 50:
17
               print("Grade: Third Division")
18
           else:
19
20
               print("Fail")
> Terminal

	☐ Test cases
```





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 01123

Expected Output-2:

Enter terms for Fibonacci series: 9 01123581321

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.

```
fib.py
      def fib(n):
          if n <= 0:
              return 0
           elif n == 1:
          return 1
           else:
              return fib(n - 1) + fib(n - 2)
10
11
12
13
15
16
17
      n=int(input("Enter terms for Fibonacci series: "))
18
     , for i in range (n):
20
             print(fib(i),end=" ")
 > Terminal
            ## Test cases
```



Pattern - 1 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.





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:

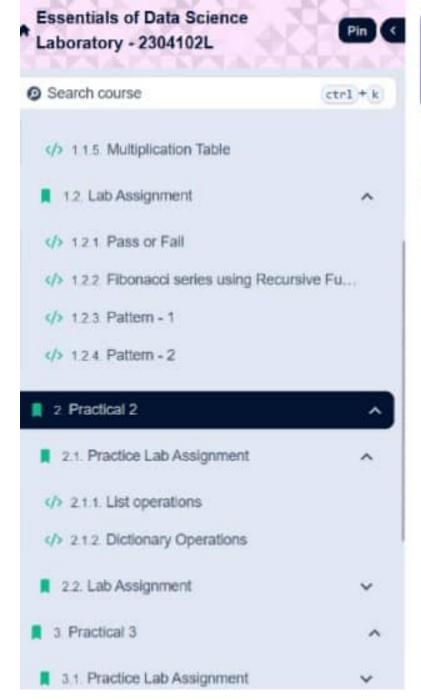
The output should display the pattern of numbers, with each row containing increasing numbers starting from 1 up to the row number.

Note:

Refer to the displayed test cases for the sample pattern.

```
v def print_number_pattern(rows):
       for i in range(1, rows + 1):
            print(" ".join(map(str, range(1, i + 1))) + " ")
     rows = int(input().strip())
     print number pattern(rows)
> Terminal
```

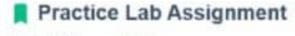




Practical 2

About this unit

Practical 2



Unit • 100% completed

Lab Assignment

Unit • 100% completed





2.1.1. List operations

MED ALBO-

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).

```
Explorer
     listOps.py
  1
        11=[ ]
4
   2
       v while True:
           wprint("1. Add")
   3
           print("2. Remove")
   4
           print("3. Display")
   5
   6
           Mprint("4. Quit")
   7
           m=int(input("Enter choice: "))
           wif n==1:
   8
           # add=int(input("Integer: "))
   9
           | 11.append(add)
  10
          # # print(f"List after adding:
  11
         (11)")
          -melif-n==2:
  12
           # if len(l1)==0:
  13
           memory ("List is empty")
  14
  15
           | elif len(l1)!=0:
  16
         remove=int(input("Integer: "))
           if remove not in 11:
  17
                  print("Element not
  18
         found")
                  else:
  19
                      11.remove(remove)
  20
                       print(f"List after
  21
        removing: {l1}")
           Helif n==3:
  22
           # if len(l1)==0:
  23
           print("List is empty")
  24
           else:
  25
           |---||print(11)
  26
           melif n==4:
  27
           break
  28
           else:
  29
  30
           print("Invalid choice")
  31
  32
```

Sample Test Cases

2.

2.1.2 Dictionary Operations



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 keyvalue 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.

```
dictOperati...
Explorer
                                         Submit
                                                  Debugger
   1
         dict = {}
         print("Empty Dictionary:",dict)
   2
.
   3
   4
         n = int(input("Number of items: "))
   5
       v for __ in range (n):
            key = input("key: ")
   6
            "value = input("value: ")
   7
   8
            dict[key] = value
   9
         print("Dictionary:", dict)
  10
  11
         update_key = input("Enter the key to
         update: ")
         if update_key in dict:
  12
            new_value = input("Enter the new
  13
         value: ")
            ||dict[update_key] = new_value
  14
            print("Value updated")
  15
  16
       v else:
            print("Key not found")
  17
  18
         retrieve_key = input("Enter the key
  19
         to retrieve: ")
       v if retrieve_key in dict:
  20
            mprint(f"Key: {retrieve_key},
  21
         Value: {dict[retrieve_key]}")
       else:
  22
            print("Key not found")
  23
  24
         get_key = input("Enter the key to
  25
         get using the get() method: ")
         value = dict.get(get_key, "Key not
  26
         found")
       v if value != "Key not found":
  27
            print(f"Key: {get_key}, Value:
  28
         {value}")
       v else:
  29
            print(value)
  30
  31
         deleted_key = input("Enter the key
  32
         to delete: ")
       v if deleted_key in dict:
  33
         ----del dict[deleted_key]
  34
            "print("Deleted")
  35
       else:
  36
  37
            print("Key not found")
         print("Updated Dictionary:",dict)
  38
```

)..

===

2.2.1. Linear search Technique



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:

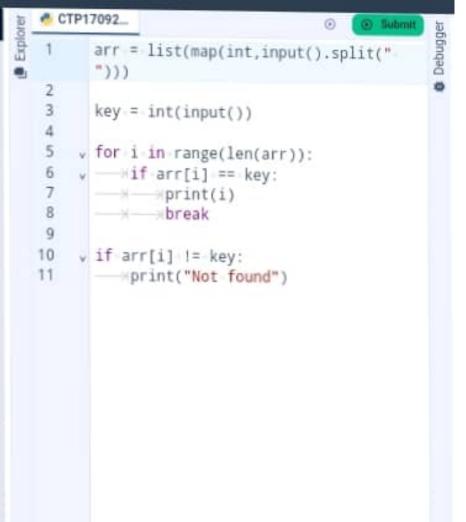
Input:

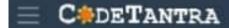
1234356

3

Output:

2









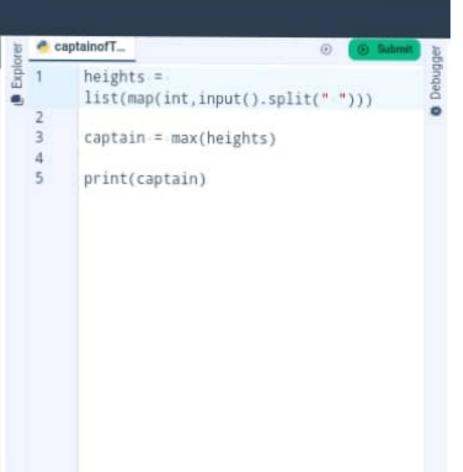
You are provided with the heights of 11 cricket players (in centimeters). Your task is to identify the tallest player, who will be selected as the captain of the team.

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.

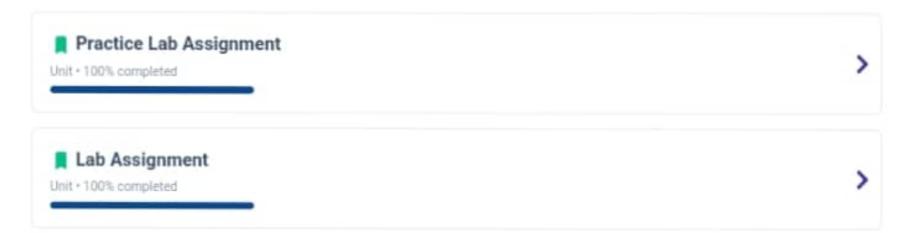


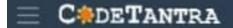


Practical 3

About this unit

Practical 3





3.1.1. Numpy array operations

ALBO

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.

```
Explorer
   numpyarr.py
                                          Submit
         import numpy as np
         rows, cols = map(int,
         input().split())
         elements = []
   5
       for __in range(rows):
   6
            #row_elements = list(map(int,
         input().split()))
             welements.extend(row_elements)
   8
   9
         arr =
         np.array(elements).reshape(rows.
         cols)
  10
  11
         print(arr)
  12
  13
         print(arr.ndim)
  14
  15
         print(-arr.shape)
  16
         print( arr.size)
  17
```



3.2.1. Numpy: Matrix Operations



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

Task:

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

- 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:

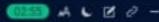
The program should display the results of the operations in the following order:

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

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



3.2.2. Numpy: Horizontal and Vertical Stack... (1335)



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).

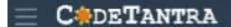
Input Format:

- 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.

```
Explorer
   stacking.py
                                         Submit
         import numpy as np
   3
         # Input matrices
         print("Enter Array1:")
         arr1 = np.array([list(map(int,
         input().split())) for i in range(3)])
   б
   7
         print("Enter Array2:")
         arr2 = np.array([list(map(int,
   8
         input().split())) for i in range(3)])
   9
  10
         # Perform horizontal stacking
         (hstack)
         horizontal stack =
  11
         np.hstack((arr1,arr2))
  12
  13
         # Perform vertical stacking (vstack)
  14
         vertical stack =
  15
         np.vstack((arr1,arr2))
         print("Horizontal Stack:")
  16
         print(horizontal stack)
  17
  18
         print("Vertical Stack:")
  19
         print(vertical_stack)
```



3.2.3. Numpy: Custom Sequence Generation



Write a Python program that takes the following inputs from the user:

- · Start value: The starting point of the sequence.
- · Stop value: The sequence should end before this value.
- · Step value: The increment between each number in the sequence.

The program should then generate a sequence using numpy based on these inputs and print the generated sequence.

Input Format:

. The user will input three integer values: start, stop, and step, each on a new line.

Output Format:

 The program should print the generated sequence based on the input values.

```
customSe...
Explorer
         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()
   9
         sequence = np.arange(start, stop,
         step)
         # Print the generated sequence
  10
         print(sequence)
  11
```



3.2.4. Numpy: Arithmetic and Statistical Op... (VAID) A L Z 2 -

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 аптау А.

3. Bitwise Operations:

· Perform bitwise AND, bitwise OR, and bitwise XOR on the arrays (ex: A, OR B,)

Input Format:

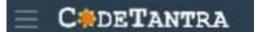
- · The first line contains space-separated integers representing the elements of array A.
- The second line contains space-separated integers representing the elements of array B.

Output Format:

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

different0... Explorer 1 import numpy as np 2 . 3 def array_operations(A, B): 4 5 # Convert A and B to NumPy arrays 6 A = np.array(A) 7 B = np.array(B) 8 # Arithmetic Operations 9 sum result = A + B 10 diff_result = A - B 11 prod_result = A * B 12 13 # Statistical Operations 14 $mean_A = np.mean(A)$ 15 median_A = np.median(A) 16 17 std_dev_A = np.std(A) 18 19 # Bitwise Operations 20 and_result = A & B Hor_result = A | B 21 xor result = A ^ B 22 23 24 # Output results with one space between each element 25 print("Element-wise Sum:", ' '.join(map(str, sum_result))) 26 print("Element-wise Difference:", ' '.join(map(str, diff_result))) print("Element-wise Product:", ' 27 '.join(map(str, prod_result))) 28 29 print(f"Mean of A: {mean_A}") print(f"Median of A: {median_A}") 30 31 print(f"Standard Deviation of A: (std_dev_A)") 32 33 print("Bitwise AND:", ' ',join(map(str, and_result))) 34 print("Bitwise OR:", ' '.join(map(str, or_result))) 35 print("Bitwise XOR:", ' '.join(map(str, xor_result))) 36 A = list(map(int, input().split())) 37 # Elements of array A B = list(map(int, input().split())) 38 # Elements of array B 39 array_operations(A, B) 40 >... =





The given code in the editor takes a single array, array1, as space-separated integers as input from the user.

3.2.6. Numpy: Searching, Sorting, Counting... 🔞 🗀 🖒 🗹 🧷

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:

- Searching: Find the indices where Search_value appears in array1 and print these indices.
- Counting: Count how many times count_value appears in array1 and print the count.
- Broadcasting: Add broadcast_value to each element of array1 using broadcasting, and print the resulting array.
- Sorting: Sort array1 in ascending order and print the sorted array.

Input Format:

- A single line containing space-separated integers representing array1.
- An integer search_value represents the value to search for in the array.
- An integer count_value represents the value to count in the array.
- 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.
- The count of occurrences of count_value in array1.
- The array after adding the broadcast_value to each element.
- The sorted array.

агтауОрега... Explorer import numpy as np 2 # Input array from the user 3 4 array1 = np.array(list(map(int, input().split()))) 5 б # Searching 7 search_value = int(input("Value to search: ")) 8 count_value = int(input("Value to count: ")) 9 broadcast_value = int(input("Value to add: ")) 10 # Find indices where value matches 11 in array1 12 search_indices = np.where(array1 === search_value)[0] print(search_indices) 13 14 15 # Count occurrences in array1 16 count_occurance = np.count_nonzero(array1 == count_value) print(count_occurance) 17 18 19 # Broadcasting addition broadcast result = array1 + 20 broadcast_value print(broadcast_result) 21 22 23 # Sort the first array 24 sorted_array = np.sort(array1) 25 print(sorted_array)

 \blacksquare

).

Scanned with OKEN Scanner

3.2.7. Student Data Analysis and Operations (1361) A 🕻 🗷 🔗 -



Operations...

Debugger

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.
- · Find the count of students who got more than 90 marks in Subject 2: Count the number of students who scored more than 90 marks in Subject 2.
- Find the count of students who scored >=90 in each subject: Count the number of students who scored 90 or more marks in each subject.
- Find the count of subjects in which each student scored >=90: Determine how many subjects each student scored 90 or more marks in.
- · Print Subject 1 marks in ascending order: Sort and print the marks of students in Subject 1 in ascending order.
- · Print students who scored between 50 and 90 in Subject 1: Display students who scored marks between 50 and 90 in Subject 1.
- · Find index positions of students who scored 79 in Subject 1: Identify the index positions of students who scored exactly 79 marks in Subject 1.

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

Sample Test Cases

```
Explorer
   1
         import numpy as np
   2
•
   3
         a = np.loadtxt("Sample.csv",
         delimiter=',', skiprows=1)
   4
         # 1. Print all student details
   5
   6
         print("All student Details:\n",a)
   7
   8
         # 2. print total students
   9
         r,c=a.shape
         print("Total Students:",r)
  10
  11
         # 3. Print all student Roll numbers
  12
         print("All Student Roll Nos",a[:,0])
  13
  14
  15
         # 4. Print subject 1 marks
         print("Subject 1 Marks",a[:,1])
  16
  17
         # 5. print minimum marks of Subject 2
  18
         print("Min marks in Subject 2",
  19
         np.min(a[:,2]))
  20
         # 6. print maximum marks of Subject 3
  21
         print("Max marks in Subject 3",
  22
         np.max(a[:,3]))
  23
         # 7. Print All subject marks
  24
         print("All subject marks:",a[:,1:])
  25
  26
  27
         # 8. print Total marks of students
         total_marks =np.sum(a[:, 1:], axis=1)
  28
         print("Total Marks", total_marks)
  29
  30
         # 9. print average marks of each
  31
         student
  32
         avg=np.mean(a[:,1:],axis=1)
  33
         print(np.round(avg,1))
  34
  35
         # 10. print average marks of each
         subject
         print("Average Marks of each
  36
         subject",np.mean(a[:, 1:],axis=0))
  37
  38
         # 11. print average marks of S1 and
         print("Average Marks of S1 and
  39
         S2",np.mean(a[:, 1:3],axis=0))
  40
         # 12. print average marks of S1 and
  41
         print("Average Marks of S1 and
  42
         $3",np.mean(a[:, [1, 3]],axis=0))
  43
         # 13. print Roll number who got
  44
         maximum marks in Subject 3
         i=np.argmax(a[:,3])
  45
         print("Roll no who got maximum marks
  46
         in Subject 3",a[i,0])
  47
         # 14. print Roll number who got
  48
         minimum marks in Subject 2
         mn=np.argmin(a[:,2])
  49
  50
         print("Roll no who got minimum marks
         in Subject 2",a[mn,0])
  51
  52
         # 15. print Roll number who got 24
         marks in Subject 2
  53
         whr=np.where(a[:, 2] == 24)
         print("Roll no who got 24 marks in
  54
         Subject 2",a[whr,0])
  55
  56
  57
         # 16. print count of students who
         ant marke in Cubinet t . An
   >.
        ==
```

Reset

Next >

3.2.7. Student Data Analysis and Operations (1131) 🗚 🕻 🗷 🔗 -



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.
- Find the count of students who got more than 90 marks in Subject 2: Count the number of students who scored more than 90 marks in Subject 2.
- · Find the count of students who scored >=90 in each subject: Count the number of students who scored 90 or more marks in each subject.
- · Find the count of subjects in which each student scored >=90: Determine how many subjects each student scored 90 or more marks in.
- Print Subject 1 marks in ascending order: Sort and print the marks of students in Subject 1 in ascending order.
- Print students who scored between 50 and 90 in Subject 1: Display students who scored marks between 50 and 90 in Subject 1.
- Find index positions of students who scored 79 in Subject 1: Identify the index positions of students who scored exactly 79 marks in Subject 1.

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

Sample Test Cases

Operations... PITTIE AVEI OF MOINS VI JI ONU S2",np.mean(a[:, 1:3],axis=0)) **4**0 41 # 12. print average marks of S1 and print("Average Marks of 51 and 42 \$3",np.mean(a[:, [1, 3]],axis=0)) 43 44 # 13. print Roll number who got maximum marks in Subject 3 45 i=np.argmax(a[:,3]) print("Roll no who got maximum marks 46 in Subject 3",a[i,0]) 47 48 # 14. print Roll number who got minimum marks in Subject 2 49 mn=np.argmin(a[:,2]) 50 print("Roll no who got minimum marks in Subject 2",a[mn,0]) 51 # 15. print Roll number who got 24 52 marks in Subject 2 53 whr=np.where(a[:, 2] == 24)print("Roll no who got 24 marks in 54 Subject 2",a[whr,0]) 55 56 57 # 16. print count of students who got marks in Subject 1 < 40 58 ct=np.count_nonzero(a[:,1]<40) print("Count of students who got 59 marks in Subject 1 < 40", ct) 60 61 # 17. print count of students who got marks in Subject 2 > 90 62 count_s2_above_90 = np.sum(a[:, 2] > print("Count of students who got 63 marks in Subject 2 > 90:", count_s2_above_90) 64 65. # 18. print count of students in each subject who got marks >= 90 66 print("Count of students in each subject who got marks >= 90:",np.count_nonzero(a[:,1:]>=90,axi s=0)) 67 68 # 19, print count of subjects in 69 which each student got marks >= 90 print("Roll no:", a[:,0]) 70 71 print("Count of subjects in which student got marks >= 90:",np.count_nonzero(a[:,1:]>=90,axi 5=1)) 72 # 20. Print S1 marks in ascending 73 74 srt=np.sort(a[:,1]) 75 print(srt) 76 # 21. Print S1 marks >= 50 and <= 90 77 print(a[(a[:, 1] >= 50) & (a[:, 1] 78 <= 90)]) print(a) 79 80 81 # 22. Print the index position of marks 79 82 ip=np.where(a[:,1]==79) 83 print(ip) 84 85 86 **)**_ ##

Submit

Debugger

Reset

< Prev

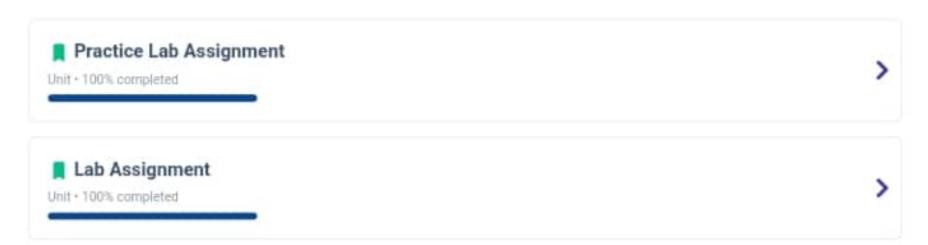
Next >



Practical 4

About this unit

Practical 4





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.

4.1.1. Pandas - series creation and manipul... (155) A C 2 -

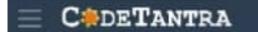
Input Format:

 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.





4.1.2. Dictionary to dataframe



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.

Explorer dataframe... Debugger import pandas as pd 2 -3 # Provided dictionary of lists 4 v data = { 'Name': ['Alice', 'Bob', 5 'Charlie'], 6 'Age': [25, 30, 35], 7 8 9 # Convert the dictionary to a DataFrame 10 df = pd.DataFrame(data) 11 12 # Display the original DataFrame 13 print("Original DataFrame:") 14 print(df) 15 16 # Adding a new row new_name = input("New name: ") 17 18 new_age = int(input("New age: ")) 19 df.loc[len(df)] = [new_name, new_age] 20 21 # Display the DataFrame after adding a new row print("After adding a row:\n",df) 22 23 # Modifying a row 24 mod_index = int(input("Index of row 25 to modify: ")) 26 new_age = int(input("New age: ")) 27 v if 0 <= mod_index < len(df):</pre> 28 df.at[mod_index, 'Age'] = new_age 29 v else: 30 print("Invalid index!:") 31 # Display the DataFrame after 32 modifying a row print("After modifying a row:") 33 34 print(df) 35 36 # Deleting a row 37 del_index = int(input("Index of row to delete: ")) v if 0 <= del_index < len(df):</pre> 38 39 → df = df.drop(del_index).reset_index(drop=T rue) else: 40 41 print("Invalid index!") 42 43 # Display the DataFrame after deleting a row 44 print("After deleting a row:") print(df) 45 46 47 # Adding a new column genders = input("Enter genders 48 separated by space: ").split() 49 df['Gender'] = genders 50 51 # Display the DataFrame after adding a new column print("After adding a new column:") 52 print(df) 53 54 55 # Modifying a column 56 df['Name'] = df['Name'].str.upper() 57 58 # Display the DataFrame after modifying a column print("After modifying a column:") 59 print(df) 60

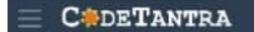
Dalating a calumn

61

63

X.

=



4.1.2. Dictionary to dataframe

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.

ORD ALEBO

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.

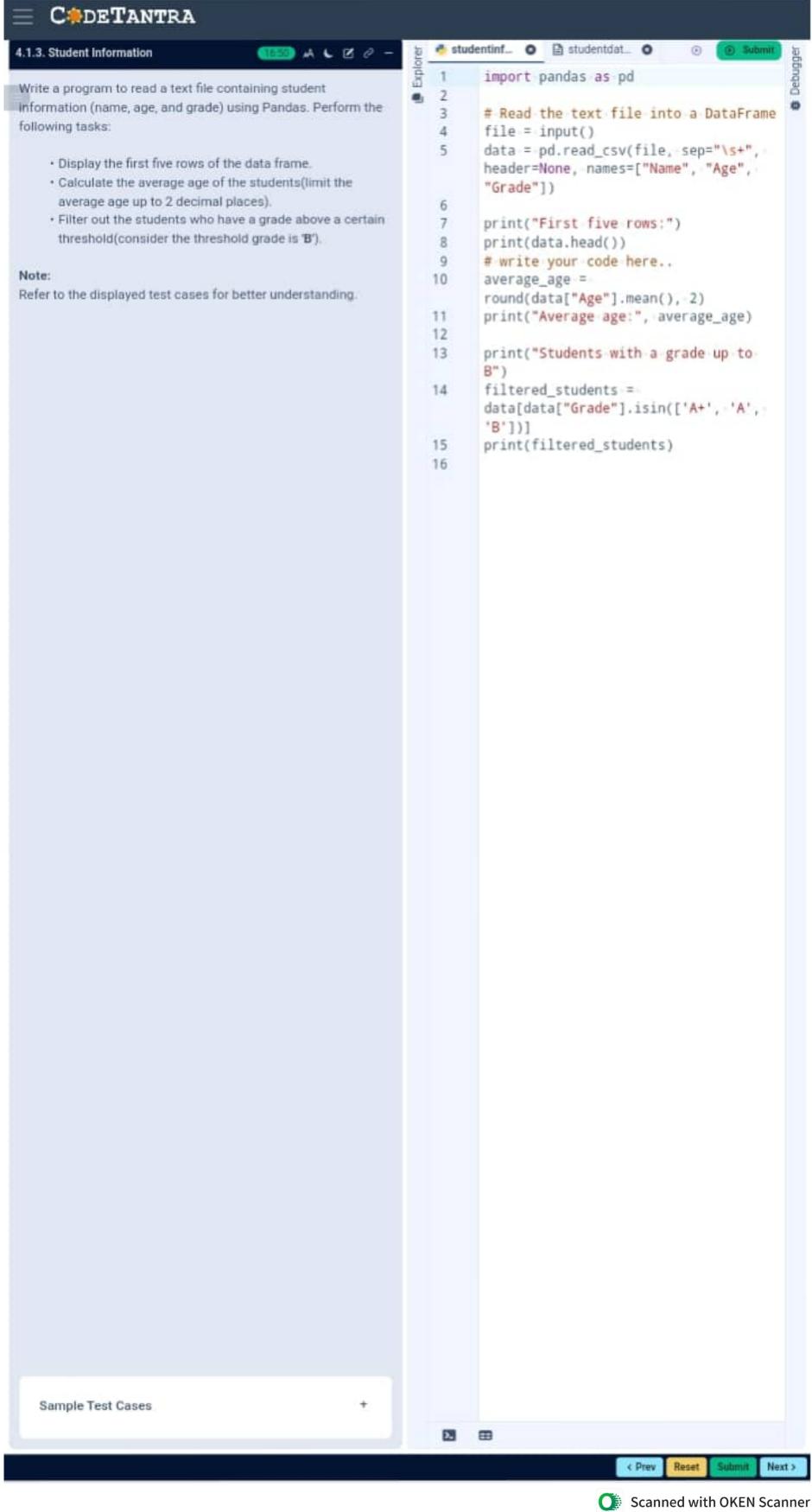
Delete a column:

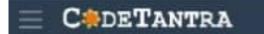
- · Remove the Age column.
- Display the DataFrame after deleting the column.

Explorer dataframe... 8 -9 # Convert the dictionary to a DataFrame 10 df = pd.DataFrame(data) 11 12 # Display the original DataFrame 13 print("Original DataFrame:") 14 print(df) 15 16 # Adding a new row 17 new_name = input("New name: ") new_age = int(input("New age:-")) 18 19 df.loc[len(df)] = [new_name, new_age] 20 21 # Display the DataFrame after adding a new row 22 print("After adding a row:\n",df) 23 24 # Modifying a row 25 mod_index = int(input("Index of row to modify: ")) 26 new_age = int(input("New age: ")) if 0 <= mod_index < len(df):</pre> 27 28 df.at[mod_index, 'Age'] = new_age v else: 29 print("Invalid index!:") 30 31 32 # Display the DataFrame after modifying a row print("After modifying a row:") 33 34 print(df) 35 36 # Deleting a row del_index = int(input("Index of row 37 to delete: ")) v if 0 <= del_index < len(df):</pre> 38 df = 39 df.drop(del_index).reset_index(drop=T rue) else: 40 41 print("Invalid index!") 42 43 # Display the DataFrame after deleting a row 44 print("After deleting a row:") 45 print(df) 46 47 # Adding a new column genders = input("Enter genders 48 separated by space: ").split() 49 df['Gender'] = genders 50 51 # Display the DataFrame after adding a new column print("After adding a new column:") 52 53 print(df) 54 55 # Modifying a column df['Name'] = df['Name'].str.upper() 56 57 58 # Display the DataFrame aftermodifying a column print("After modifying a column:") 59 60 print(df) 61 # Deleting a column 62 df.drop('Age', axis=1, inplace=True) 63 64 # Display the DataFrame after 65 deleting a column print("After deleting a column:") 66 67 print(df) 68

X.

=





4.2.1. Month with the Highest Total Sales

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

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

Sample Data:

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

Note:

Sample Test Cases

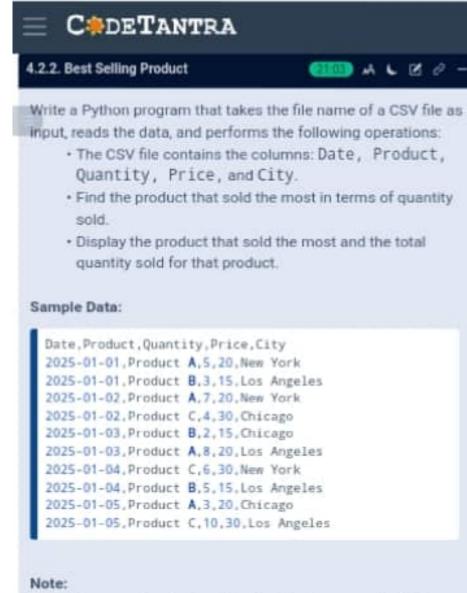
The data cannot be displayed in the file. You can refer to the sample data provided for insights.

Explorer monthForS... O is sales_data... O import pandas as pd 2 # Prompt the user for the file name 3 file name = input() 4 5 6 # Load the data 7 df = pd.read_csv(file_name) 8 df['Date'] = 9 pd.to_datetime(df['Date']) df['Month'] = 10 df['Date'].dt.to_period('M').astype(s df['Total'] = df['Quantity'] * 11 df['Price'] 12 monthly_sales = df.groupby('Month') 13 ['Total'].sum() 14 # Find the month with the highest 15 total sales best_month = monthly_sales.idxmax() 16 highest_sales = monthly_sales.max() 17 18 19 print(f"Best month: {best_month}") print(f"Total sales: 20 \${highest_sales:.2f}") 21







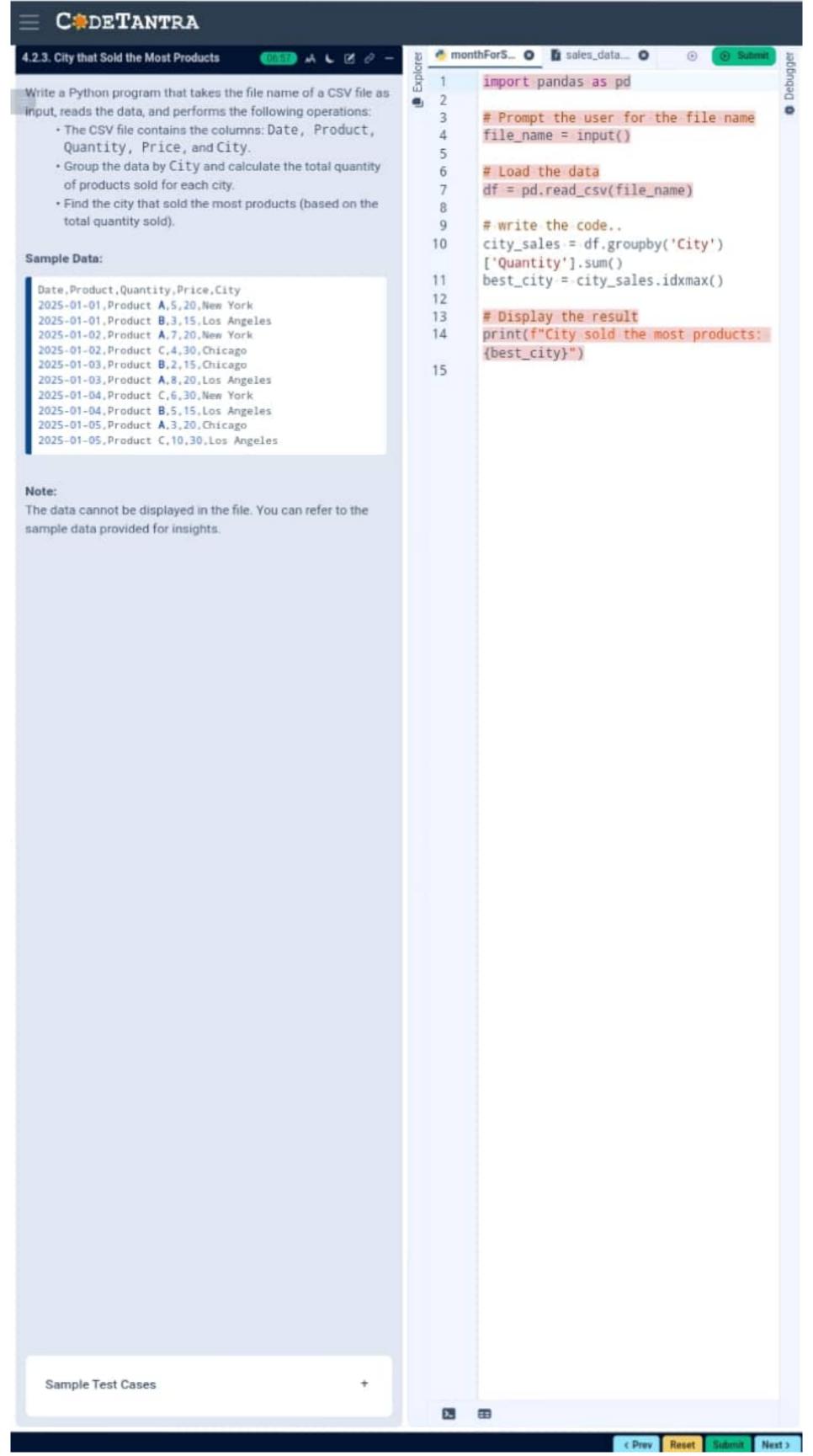


Explorer monthForS... O is sales_data... O import pandas as pd 2 # Prompt the user for the file name 3 4 file_name = input() 5 6 # Load the data df = pd.read_csv(file_name) 7 8 product_sales = df.groupby('Product') ['Quantity'].sum() # Find the product with the highest 9 total quantity sold best_product = product_sales.idxmax() 10 highest_quantity = 11 product_sales.max() 12 13 # Display the result print(f"Best selling product: 14

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

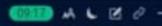
{best_product}") print(f"Total quantity sold: 15 {highest_quantity}") 16 A === C Prev Reset Submit Next >







4.2.4. Most Frequently Sold Product Pairs



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

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

Sample Data:

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

Explanation:

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)

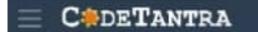
Note:

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

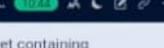
Explorer frequently... O is sales_data... O import pandas as pd 2 from itertools import combinations 3 from collections import Counter 4 5 # Prompt user to input the file name 6 file_name = input() 7 8 # Read data from the specified CSV file df = pd.read csv(file name) 9 10 # write the code 11 date_products = {} 12 13 v for date, group in 14 df.groupby('Date'): products = 15 group['Product'].unique() if len(products) > 1: 16 date_products[date] = 17 products 18 19 pair_counter = Counter() 20 for products in 21 date_products.values(): 22 pairs = combinations(sorted(products), 2) pair_counter.update(pairs) 23 24 25 v if pair_counter: 26 max_count = max(pair_counter.values()) 27 28 ofor pair, count in pair_counter.items(): 29 if count == max count: # # print(f*{pair[0]} and 30 {pair[1]}: {count} times") 31 else: print("No product pairs found.") 32 33 # Output the most frequent product 34 pairs

X.

===



4.2.5. Titanic Dataset Analysis and Data Cle... (IIIII) 🗚 📞 🗷 🔗



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.

- Display the first 5 rows of the dataset.
- Display the last 5 rows of the dataset.
- 3. Get the shape of the dataset (number of rows and columns).
- 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).
- 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.

Passuper laserd	N S a e g	S P B F C P h	T F C a k r e e t	C b a a r i k d
-----------------	-----------	---------------	-------------------	-----------------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3,"Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cunings, Mrs. John Bradley (Florence Briggs Thay 3,1,3, "Heikkinen, Miss. Laina", female, 26,0,0,5TON/02, 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",fe 5,0,3, "Allen, Mr, William Henry", male, 35,0,0,373450,8.0 6,0,3, "Moran, Mr. James", male, .0,0,330877,8.4583, .Q 7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8,0,3, "Palsson, Master. Gosta Leonard", male, 2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

Explorer titanicData... import pandas as pd 2 import numpy as np 3 4 # Load the Titanic dataset data = pd.read_csv('Titanic-5 Dataset.csv') 6 # 1. Display the first 5 rows of the 7 dataset 8 9 print(data.head()) 10 # 2. Display the last 5 rows of the 11 dataset 12 13 print(data.tail()) 14 # 3. Get the shape of the dataset 15 16 17 print(data.shape) 18 19 # 4. Get a summary of the dataset (info) 20 print(data.info()) 21 22 # 5. Get basic statistics of the 23 dataset 24 25 print(data.describe()) 26 # 6. Check for missing values 27 28 29 print(data.isnull().sum()) 30 31 # 7. Fill missing values in the 'Age' column with the median age median_age = data['Age'].median() 32 33 data['Age'].fillna(median_age, inplace=True) 34 35 # 8. Fill missing values in the 'Embarked' column with the mode 36 mode_embarked = data['Embarked'].mode()[0] data['Embarked'].fillna(mode embarked 37 , inplace=True) 38 # 9. Drop the 'Cabin' column due to 39 many missing values 40 data.drop('Cabin', axis=1, inplace=True) 41 42 # 10. Create a new column 'FamilySize' by adding 'SibSp' and 'Parch' data['FamilySize'] = data['SibSp'] +-43 data['Parch'] 44

A ===



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.

4.2.6. Titanic Dataset Analysis and Data Cle... (1211) 🗚 🕻 🗷 🗷

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

The Titanic dataset contains columns as shown below,

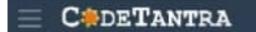
Passengerid

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",fe 5,0,3, "Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3,"Moran, Mr. James", male,,0,0,330877,8.4583,,Q 7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8,0,3,"Palsson, Master. Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

```
Explorer
   titanicData...
         import pandas as pd
   2
         import numpy as np
   3
   4
         # Load the Titanic dataset
         data = pd.read_csv('Titanic-
   5
         Dataset.csv')
   6
         data['FamilySize'] = data['SibSp'] +
         data['Parch']
   8
         data['Alone'] =
         np.where(data['FamilySize'] == 0, 1,
   9
  10
         # 3. Convert 'Sex' to numeric (male:
         0, female: 1)
         data['Sex'] =
  11
         data['Sex'].map({'male': 0,
         'female': 1})
  12
         # 4. One-hot encode the 'Embarked'
  13
         column, dropping the first category
         data = pd.get_dummies(data, columns=
  14
         ['Embarked'], drop_first=True)
  15
  16
         # 6. Get the mean age of passengers
  17
         mean_age = data['Age'].mean()
         print(mean age)
  18
  19
         # 7. Get the median fare of
  20
         passengers
         median fare = data['Fare'].median()
  21
         print(median_fare)
  22
  23
         # 8. Get the number of passengers by
  24
         class
         passengers_by_class =
  25
         data['Pclass'].value_counts()
         print(passengers_by_class)
  26
  27
         # 9. Get the number of passengers by
  28
         gender
  29
         passengers_by_gender =
         data['Sex'].value_counts().sort_index
         print(passengers_by_gender)
  30
  31
  32
         # 10. Get the number of passengers
         by survival status
         passengers_by_survival =
  33
         data['Survived'].value_counts().sort_
         index()
  34
         print(passengers_by_survival)
  35
  36
         # 11. Calculate the survival rate
  37
         survival rate =
         data['Survived'].mean()
         print(survival_rate)
  38
  39
  40
         # 12. Calculate the survival rate by
         gender
         survival_rate_by_gender =
  41
         data.groupby('Sex')
         ['Survived'].mean()
         print(survival_rate_by_gender)
  42
  43
  44
  45
  46
  47
  48
    X.
       =
```



4.2.7. Titanic Dataset Analysis and Data Cle... (IIIII) 🗚 📞 🗹 🖉



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.

- Calculate the survival rate by class.
- Calculate the survival rate by embarkation location. (Embarked_S).
- Calculate the survival rate by family size (FamilySize).
- Calculate the survival rate by being alone (IsAlone).
- Get the average fare by passenger class (Pclass).
- Get the average age by passenger class (Pclass).
- Get the average age by survival status (Survived).
- Get the average fare by survival status (Survived).
- Get the number of survivors by class (Pclass).
- Get the number of non-survivors by class (Pclass).

The Titanic dataset contains columns as shown below,

Passenger I d	a	N a m e	S e x	A g e	SIBSP	p a r c h	T i c k e t	F a r e	C a b i n	E m b a r k e d
---------------	---	------------------	-------	-------	-------	-----------------------	-------------	------------------	-----------	-----------------------------------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3,"Braund, Mr. Owen Harris", male, 22,1,0,A/S 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)", fee 5,0,3,"Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3,"Moran, Mr. James",male,,0,0,330877,8.4583,,Q 7,0,1,"McCarthy, Mr. Timothy J",male,54,0,0,17463,51.86 8,0,3, "Palsson, Master. Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2, "Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

```
Explorer
   titanicData...
         import pandas as pd
   2
         import numpy as np
   3
   4
         # Load the Titanic dataset
         data = pd.read_csv('Titanic-
   5
         Dataset.csv')
   6
         data['FamilySize'] = data['SibSp'] +
         data['Parch']
         data['IsAlone'] =
   7
         np.where(data['FamilySize'] > 0, 0,
         data = pd.get_dummies(data, columns=
   8
         ['Embarked'], drop_first=True)
   9
         print(data.groupby('Pclass')
  10
         ['Survived'].mean())
  11
         #2. Calculate the survival rate by
  12
         embarked location (Embarked_S)
         print(data.groupby('Embarked_S')
  13
         ['Survived'].mean())
  14
         #3. Calculate the survival rate by
  15
         family size
         print(data.groupby('FamilySize')
  16
         ['Survived'].mean())
  17
         #4. Calculate the survival rate by
  18
         being alone
         print(data.groupby('IsAlone')
  19
         ['Survived'].mean())
  20
  21
         #5. Get the average fare by class
         print(data.groupby('Pclass')
  22
         ['Fare'].mean())
  23
  24
         #6. Get the average age by class
  25
         print(data.groupby('Pclass')
         ['Age'].mean())
  26
  27
         #7. Get the average age by survival
         print(data.groupby('Survived')
  28
         ['Age'].mean())
  29
  30
         #8. Get the average fare by survival
         print(data.groupby('Survived')
  31
         ['Fare'].mean())
  32
  33
         #9. Get the number of survivors by
         class (sort by values descending)
         print(data[data['Survived'] == 1]
  34
         ['Pclass'].value_counts())
  35
         #10. Get the number of non-survivors
  36
         by class (sort by values descending)
         print(data[data['Survived'] == 0]
  37
         ['Pclass'].value_counts())
  38
  39
  40
    A ===
```



the dataset.

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

4.2.8. Titanic Dataset Analysis and Data Cle... (EDD) 🗚 📞 🗷 🔗

Explorer

2

3

4

5

6

7

8

9

titanicData...

import pandas as pd

Load the Titanic dataset

survivors_by_gender =

['Sex'].value counts()

data[data['Survived'] == 1]

data = pd.read csv('Titanic-

['Embarked'], drop_first=True)

data = pd.get_dummies(data, columns=

0

import numpy as np

Dataset.csv')

- Get the number of survivors by gender (Sex).
- 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
- Calculate the percentage of adults (Age >= 18) who survived.
- Get the median age of survivors.
- 8. Get the median age of non-survivors.
- 9. Get the median fare of survivors.
- Get the median fare of non-survivors.

The Titanic dataset contains columns as shown below,

PassuPSPNSAiacFabingvsexepht	E m b a r k e d
------------------------------	-----------------------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",fe 5,0,3, "Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3,"Moran, Mr. James", male, ,0,0,330877,8.4583, ,Q 7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8,0,3, "Palsson, Master, Gosta Leonard", male,2,3,1,34990 9,1.3. "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2, "Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

print(survivors_by_gender) 10 11 # 2. Get the number of non-survivors-12 by-gender non_survivors_by_gender = 13 data[data['Survived'] == 0] ['Sex'].value_counts() 14 print(non_survivors_by_gender) 15 16 # 3. Get the number of survivors by embarked location (Embarked_S) survivors by embarked s = 17 data[data['Survived'] == 1] ['Embarked_S'].value_counts() print(survivors by embarked s) 18 19 # 4. Get the number of non-survivors 20 by embarked location (Embarked_S) non_survivors_by_embarked_s = 21 data[data['Survived'] == 0] ['Embarked_S'].value_counts() print(non_survivors_by_embarked_s) 22 23 24 #5. Percentage of children (Age < 18) who survived children=data [data['Age'] < 18] 25 children_survival_rate= 26 children['Survived'].mean() print(children_survival_rate) 27 28 29 #6. Percentage of adults (Age->--18) who survived adults= data[data['Age'] >=18] 30 adults_survival_rate= 31 adults['Survived'].mean() print(adults_survival_rate) 32 33 #7. Median age of survivors 34 median_age_survivors = 35 data[data['Survived'] == 1] ['Age'].median() print(median_age_survivors) 36 37 38 #8. Median age of non-survivors 39 median_age_non_survivors = data[data['Survived'] == 0] ['Age'].median() print(median_age_non_survivors) 40 41 #9. Median fare of survivors 42 median_fare_survivors = 43 data[data['Survived'] == 1] ['Fare'].median() print(median_fare_survivors) 44 45 # 10. Median fare of non-survivors 46 47 median_fare_non_survivors = data[data['Survived'] == 0] ['Fare'].median() print(median_fare_non_survivors) 48 Att

X.

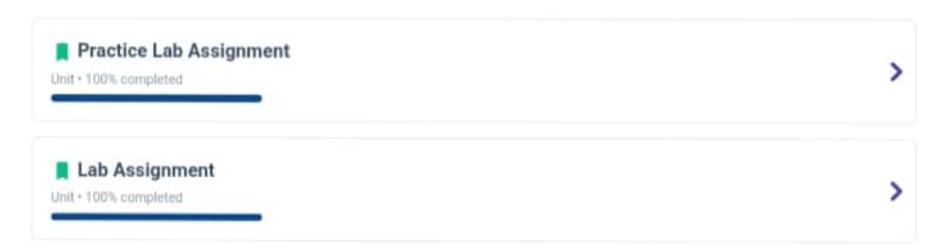
■



Practical 5

About this unit

Practical 5





A ==

Sample Test Cases

c Prev Reset Submit Next >



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

(IIII) A L Z 2 -

Dataset Information:

5.2.1. Titanic Dataset

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
- · Use the colors lightblue for survivors (1) and lightcoral for non-survivors (0).
- . Title the plot as "Survival Count".
- . Label the x-axis as "Survived (0 = No, 1 = Yes)" and the yaxis as "Count".

Scatter Plot (Fare vs Age):

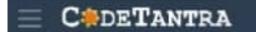
- . Create a scatter plot to visualize the relationship between the Fare and Age of passengers.
- · Use orange for the data points.
- . Title the plot as "Fare vs Age".
- Label the x-axis as "Age" and the y-axis as "Fare".

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

Sample Test Cases

```
titanicData...
Explorer
         import pandas as pd
         import matplotlib.pyplot as plt
   2
   3
   4
         # Load the Titanic dataset from the
         CSV file
         df = pd.read_csv('titanic.csv')
   5
   6
   7
         # Set up the figure for 5 subplots
   8
         fig, axes = plt.subplots(3, 2,
         figsize=(12, 12))
   9
         # write the code...
  10
         # Plot 1: Count of passengers by
  11
         class
  12
         axes[0,
         0].bar(df['Pclass'].value_counts().in
         dex, df['Pclass'].value_counts(),
         color='skyblue')
         axes[0, 0].set_title("Passenger
  13
         Class Distribution")
  14
         axes[0, 0].set_xlabel("Pclass")
  15
         axes[0, 0].set_ylabel("Count")
  16
         # plot 2: Gender distribution
  17
         axes[0,
  18
         1].pie(df['Gender'].value_counts(),
         labels=df['Gender'].value_counts().in
         dex, autopct='%1.1f%%', colors=
         ['lightblue', 'lightcoral'])
         axes[0, 1].set_title("Gender
  19
         Distribution")
  20
  21
         # plot 3: Age distribution
  22
         axes[1, 0].hist(df['Age'].dropna().
         bins=8, color='lightgreen',
         edgecolor='black')
  23
         axes[1, 0].set_title("Age
         Distribution")
         axes[1, 0].set_xlabel("Age")
  24
  25
         axes[1, 0].set_ylabel("Frequency")
  26
  27
         # plot 4: Survival count
  28
         axes[1,
         1].bar(df['Survived'].value_counts().
         df['Survived'].value_counts(), color=
         ['lightblue', 'lightcoral'])
  29
         axes[1, 1].set_title("Survival
         Count")
         axes[1, 1].set_xlabel("Survived (0 =
  30
         No, 1 = Yes)")
  31
         axes[1, 1].set_ylabel("Count")
  32
  33
         # plot 5: Fare vs Age
  34
         axes[2, 0].scatter(df['Age'],
         df['Fare'], color='orange',
         edgecolors='black')
  35
         axes[2, 0].set_title("Fare vs Age")
         axes[2, 0].set_xlabel("Age")
  36
  37
         axes[2, 0].set_ylabel("Fare")
  38
  39
         plt.tight_layout()
         plt.show()
  40
    X.
```

C Prev Reset Submit Next >



5.2.2. Histogram of passenger information ... (IIIII) 🧀 📞 🗷 🔗

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.
- 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,

a S u s e n g e e d d	P c l a s s	N a m e	Sex	A g e	Sbsp	P a r c h	T c k e t	Fare	C a b i n	E m b a r k e d
-----------------------	-------------	------------------	-----	-------	------	-----------------------	-----------------------	------	-----------------------	-----------------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,STON/02. 3 4,1,1, "Futrelle, Mrs. Jacques Heath (Lily May Peel)", fe 5,0,3, "Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3, "Moran, Mr. James", male, ,0,0,330877,8.4583, ,Q 7,0,1,"McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8.0.3, "Palsson, Master. Gosta Leonard", male.2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

Explorer Histogram... import pandas as pd import matplotlib.pyplot as plt 2 3 # Load the Titanic dataset 4 data = pd.read_csv('Titanic-5 Dataset.csv') 6 7 # Data Cleaning 8 data['Age'].fillna(data['Age'].median (), inplace=True) 9 data['Embarked'].fillna(data['Embarke d'].mode()[0], inplace=True) data.drop('Cabin', axis=1, 10 inplace=True) 11 12 # Convert categorical features to numeric 13 data['Sex'] = data['Sex'].map({'male': 0, 'female': 1}) data = pd.get_dummies(data, columns= 14 ['Embarked'], drop_first=True) 15 # Write your code here for Histogram 16 plt.hist(data['Age'], bins=30, 17 edgecolor='k') 18 plt.xlabel('Age') 19 plt.ylabel('Frequency') plt.title('Age Distribution') 20 21 plt.show()

A ===



5.2.3. Bar plot of survival rate of passengers (1989) A 📞 🗷 🔗



BarPiotOf...

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'.
- Add the title "Survival Count" to the chart.
- Label the x-axis as 'Survived' and the y-axis as 'Count'.

The Titanic dataset contains columns as shown below,

S S P T T E M B S P T T A B B B B B B B B B B B B B B B B B

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,STON/02. 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",fe 5,0,3, "Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3, "Moran, Mr. James", male, ,0,0,330877,8.4583, ,Q 7,0,1,"McCarthy, Mr. Timothy J",male,54,0,0,17463,51.86 8.0,3,"Palsson, Master. Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

Explorer import pandas as pd 2 import matplotlib.pyplot as plt 3 4 # Load the Titanic dataset data = pd.read csv('Titanic-5 Dataset.csv') 6 7 # Data Cleaning 8 data['Age'].fillna(data['Age'].median (), inplace=True) 9 data['Embarked'].fillna(data['Embarke d'].mode()[0], inplace=True) data.drop('Cabin', axis=1, 10 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 16 for Survival Rate 17 18 survival counts = data['Survived'].value_counts() survival counts.plot(kind='bar') 19 plt.title('Survival Count') 20 plt.xlabel('Survived') 21 plt.ylabel('Count') 22 23 plt.show() 24

X.

===



5.2.4. Bar Plot for Survival by Gender



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.
- Use a stacked bar chart to display the survival counts.
- Add the title "Survival by Gender" to the chart.
- 4. Label the x-axis as 'Gender' and the y-axis as 'Count'.
- The legend should indicate 'Not Survived' and 'Survived'.

The Titanic dataset contains columns as shown below,

a s s s s s s s s s s s s s s s s s s s	p c	N a m e	S e x	A g e	8 - b 8 p	P a r c h	T i c k e t	F a r e	C a b i n	E m b a r k e d
-----------------------------------------	-----	------------------	-------	-------	-----------	-----------	-------------	------------------	-----------	--------------------------------

Sample Data:

Passengerld, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4,1,1, "Futrelle, Mrs. Jacques Heath (Lily May Peel)", fee 5,0,3, "Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3, "Moran, Mr. James", male, .0,0,330877,8.4583, .Q 7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8,0,3,"Palsson, Master, Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

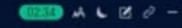
```
Explorer
   BarPiotOf...
         import pandas as pd
         import matplotlib.pyplot as plt
   2
   3
   4
         # Load the Titanic dataset
         data = pd.read csv('Titanic-
   5
         Dataset.csv')
   6
   7
         # Data Cleaning
         data['Age'].fillna(data['Age'].median
   8
         (), inplace=True)
   9
         data['Embarked'].fillna(data['Embarke
         d'].mode()[0], inplace=True)
         data.drop('Cabin', axis=1,
  10
         inplace=True)
  11
  12
         # Convert categorical features to
         numeric
  13
         data['Sex'] =
         data['Sex'].map({'male': 0,
         'female': 1})
         data = pd.get_dummies(data, columns=
  14
         ['Embarked'], drop_first=True)
  15
         # Write your code here for Bar Plot
  16
         for Survival by Gender
  17
         survival by gender =
  18
         data.groupby('Sex')
         ['Survived'].value_counts().unstack()
         .fillna(0)
  19
         survival_by_gender.columns = ['Not
         Survived', 'Survived']
         survival_by_gender.index = ['0', '1']
  20
         survival_by_gender.plot(kind='bar',
  21
         stacked=True)
         plt.title('Survival by Gender')
  22
         plt.xlabel('Gender')
  23
  24
         plt.ylabel('Count')
  25
         plt.legend(title=None)
  26
         plt.show()
```

X.

==



5.2.5. Bar Plot for Survival by Pclass



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().
- Use a stacked bar chart to display the survival counts.
- 3. Add the title "Survival by Pclass" to the chart.
- 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.

a Suser ve d	p c l a s s	N a m e	S e x	A g e	Sibs	b a t	T c k e t	Fare	C a b i n	E m b a r k e d
--------------	-------------	------------------	-------------	-------	------	-------	-----------------------	------	-----------------------	-----------------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3,"Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",fe 5,0,3,"Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3, "Moran, Mr. James", male,,0,0,330877,8,4583,,Q 7,0,1,"McCarthy, Mr. Timothy J",male,54,0,0,17463,51.86 8,0,3,"Palsson, Master. Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

Note:

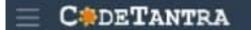
- Refer to the visible test case for better reference.
- . Ensure you use the groupby() function with value_counts() to count the survivors and non-survivors for each Polass.
- Do not manually use size() or unstack() without value_counts(). Use the value_counts() method for counting survival status directly.

Explorer BarPiotOf... import pandas as pd import matplotlib.pyplot as plt 2 3 4 # Load the Titanic dataset 5 data = pd.read_csv('Titanic-Dataset.csv') 6 7 # Data Cleaning data['Age'].fillna(data['Age'].median 8 (), inplace=True) data['Embarked'].fillna(data['Embarke 9 d'].mode()[0], inplace=True) data.drop('Cabin', axis=1, 10 inplace=True) 11 12 # Convert categorical features to numeric data['Sex'] = 13 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 16 for Survival by Pclass 17 survival by class = 18 data.groupby('Pclass') ['Survived'].value_counts().unstack() .fillna(0) 19 survival_by_class.columns = ['Not Survived', 'Survived'] survival_by_class.plot(kind='bar', 20 stacked=True) 21 plt.title('Survival by Pclass') 22 plt.xlabel('Pclass') plt.ylabel('Count') 23 24 plt.legend(title=None) 25 plt.show() 26

Sample Test Cases

X.

==



5.2.6. Bar Plot for Survival by Embarked



BarPiotOf...

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:

- 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.
- Set the chart type to 'bar' and make it stacked.
- 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.

S u P r u a s e d	N S e m x	A D B S P	p T i c k c e t	F a b i n	E m b a r k e d
-------------------	-----------	-----------	-----------------	-----------	-----------------------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SihSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)", fel 5,0,3,"Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3, "Moran, Mr. James", male, ,0,0,330877,8.4583, ,Q 7,0,1,"McCarthy, Mr. Timothy J",male,54,0,0,17463,51.86 8,0,3, "Palsson, Master. Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2, "Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

```
Explorer
         import pandas as pd
         import matplotlib.pyplot as plt
   2
   3
   4
         # Load the Titanic dataset
         data = pd.read_csv('Titanic-
   5
         Dataset.csv')
   6
   7
         # Data Cleaning
   8
         data['Age'].fillna(data['Age'].median
         (), inplace=True)
         data['Embarked'].fillna(data['Embarke
   9
         d'].mode()[0], inplace=True)
         data.drop('Cabin', axis=1,
  10
         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
  16
         for Survival by Embarked
  17
  18
         grouped = data.groupby('Embarked Q')
         ['Survived'].value_counts().unstack()
         .fillna(0)
  19
         grouped.columns = ['Not Survived',
         'Survived'1
         grouped.plot(kind='bar',
  20
         stacked=True)
         plt.title('Survival by Embarked')
  21
  22
         plt.xlabel('Embarked')
         plt.ylabel('Count')
  23
         plt.legend(title=None)
  24
  25
         plt.show()
  26
```



5.2.7. Box plot for Age Distribution



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(").
- Label the x-axis as 'Pclass' and the y-axis as 'Age'.

The Titanic dataset contains columns as shown below,

a s u r v i v e d d	Pclass	N a m e	S e x	A g e	SIBSP	Parch	T c k e t	F a r e	C a b i	E m b a r k e d
---------------------	--------	------------------	-------	-------	-------	-------	-----------------------	------------------	------------------	-----------------

Sample Data:

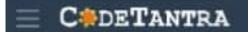
PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3, "Heikkinen, Miss. Laina", female, 26,0,0,STON/O2. 3 4,1,1, "Futrelle, Mrs. Jacques Heath (Lily May Peel)", fe 5,0,3, "Allen, Mr. William Henry", male, 35,0,0,373450.8.0 6,0,3, "Moran, Mr. James", male,0,0,330877,8.4583.,Q 7,0,1, "McCarthy, Mr. Timothy J", male,54,0,0,17463,51.86 8,0,3, "Palsson, Master. Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2, "Nasser, Mrs. Nicholas (Adele Achem)", female,14,

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

Explorer BoxPlotFo... import pandas as pd import matplotlib.pyplot as plt 2 3 4 # Load the Titanic dataset data = pd.read csv('Titanic-5 Dataset.csv') 6 7 # Data Cleaning 8 data['Age'].fillna(data['Age'].median (), inplace=True) 9 data['Embarked'].fillna(data['Embarke d'].mode()[0], inplace=True) data.drop('Cabin', axis=1, 10 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 Box Plot 16 for Age by Pclass 17 plt.figure(figsize=(8, 6)) 18 19 data.boxplot(column='Age', by='Pclass') plt.suptitle('') 20 21 plt.title('Age by Pclass') plt.xlabel('Pclass') 22 23 plt.ylabel('Age') plt.show() 24 25

X.

===



5.2.8. Box Plot for Age by Survived



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(").
- Label the x-axis as 'Survived' and the y-axis as 'Age'.

The Titanic dataset contains columns as shown below.

as Suppose of Span Cab a

Sample Data:

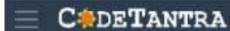
PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3, "Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4.1.1, "Futrelle, Mrs. Jacques Heath (Lily May Peel)", fee 5.0,3, "Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3,"Moran, Mr. James", male,,0,0,330877,8.4583,,Q 7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8,0,3,"Palsson, Master, Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

Explorer BoxPlotFo... import pandas as pd import matplotlib.pyplot as plt 2 3 # Load the Titanic dataset 4 data = pd.read csv('Titanic-5 Dataset.csv') 6 7 # Data Cleaning data['Age'].fillna(data['Age'].median 8 (), inplace=True) 9 data['Embarked'].fillna(data['Embarke d'].mode()[0], inplace=True) data.drop('Cabin', axis=1, 10 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 Box Plot 16 for Age by Survived 17 18 plt.figure(figsize=(8, 6)) data.boxplot(column='Age', 19 by='Survived') plt.suptitle('') 20 21 plt.title('Age by Survival') plt.xlabel('Survived') 22 23 plt.ylabel('Age') plt.show() 24 25

X.

===



5.2.9. Box Plot for Fare by Pclass



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:

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

The Titanic dataset contains columns as shown below,

a S u r v e n g e r d d		N a m e	S e x	A g e	S i b S p	p a r c h	T i c k e t	F a r e	Cabin	Embarked
-------------------------	--	---------	-------------	-------	-----------	-----------------------	-------------	------------------	-------	----------

Sample Data:

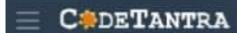
```
PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti
1,0,3, "Braund, Mr. Onen Harris", male, 22,1,0,A/5 21171,7
2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thay
3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3
4.1.1, "Futrelle, Mrs. Jacques Heath (Lily May Peel)", fee
5,0,3, "Allen, Mr. William Henry", male, 35,0,0,373450,8.0
6,0,3, "Moran, Mr. James", male,,0,0,330877,8.4583,,Q
7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86
8,0,3,"Palsson, Master. Gosta Leonard", male,2,3,1,34990
9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg
10,1,2,"Nasser, Mrs. Nicholas (Adele Achem)", female, 14,
```

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

```
BoxPlotFo...
Explorer
         import pandas as pd
         import matplotlib.pyplot as plt
   2
   3
   4
         # Load the Titanic dataset
         data = pd.read_csv('Titanic-
   5
         Dataset.csv')
   6
   7
         # Data Cleaning
   8
         data['Age'].fillna(data['Age'].median
         (), inplace=True)
         data['Embarked'].fillna(data['Embarke
   9
         d'].mode()[0], inplace=True)
         data.drop('Cabin', axis=1,
  10
         inplace=True)
  11
         # Convert categorical features to
  12
         numeric
  13
         data['Sex'] =
         data['Sex'].map({'male': 0,
         'female': 1})
         data = pd.get_dummies(data, columns=
  14
         ['Embarked'], drop_first=True)
  15
  16
         # Write your code here for Box Plot
         for Fare by Pclass
  17
         plt.figure(figsize=(8, 6))
  18
         data.boxplot(column='Fare',
  19
         by='Pclass')
         plt.suptitle('')
  20
  21
         plt.title('Fare by Pclass')
         plt.xlabel('Pclass')
  22
  23
         plt.ylabel('Fare')
         plt.show()
  24
  25
```

)..

 \blacksquare



5.2.10. Scatter Plot for Age vs. Fare



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.
- 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,

ssengerid	S u r v e d	р с _ a я я	N a m e	S e x	A g e	S	Parch	T i c k e t	F a r e	C a b i n	Embarked
-----------	-------------	-------------	------------------	-------	-------	---	-------	-------------	------------------	-----------	----------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3, "Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, "Cumings, Wrs. John Bradley (Florence Briggs Thay 3,1,3,"Heikkinen, Miss. Laina", female, 26,0,0,5TON/02. 3 4,1,1, "Futrelle, Mrs. Jacques Heath (Lily May Peel)", fe 5,0,3,"Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3,"Moran, Mr. James", male,,0,0,330877,8.4583,,0 7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8,0,3, "Palsson, Master, Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2,"Masser, Mrs. Nicholas (Adele Achem)", female, 14,

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

Explorer AgeFareSc... import pandas as pd import matplotlib.pyplot as plt 2 • 3 # Load the Titanic dataset 4 data = pd.read_csv('Titanic-5 Dataset.csv') 6 7 # Data Cleaning 8 data['Age'].fillna(data['Age'].median (), inplace=True) data['Embarked'].fillna(data['Embarke 9 d'].mode()[0], inplace=True) data.drop('Cabin', axis=1, 10 inplace=True) 11 12 # Convert categorical features to numeric 13 data['Sex'] = data['Sex'].map({'male': 0, 'female': 1}) data = pd.get_dummies(data, columns= 14 ['Embarked'], drop first=True) 15 # Write your code here for Box Plot 16 for Fare by Pclass 17 plt.figure(figsize=(6.4,4.8)) 18 plt.scatter(data['Age'],data['Fare']) 19 plt.title('Age vs. Fare') 20 plt.xlabel('Age') 21 plt.ylabel('Fare') 22 23 24 plt.show() 25

=

C*DETANTRA

5.2.11. Scatter Plot for Age vs. Fare by Surv... (1999) 🗚 📞 🗹 🕖

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".
- Label the x-axis as 'Age' and the y-axis as 'Fare'.

The Titanic dataset contains columns as shown below.

Surved	P C I a s s	N a m e	S e x	A g e	SbSp	b a t c p	T c k e t	F a r e	Cab	E m b a r k e d
--------	-------------	------------------	-------	-------------	------	-----------	-----------	------------------	-----	--------------------------

Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ti 1,0,3,"Braund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7 2,1,1, *Cumings, Mrs. John Bradley (Florence Briggs Thay 3,1,3, "Heikkinen, Miss. Laina", female, 26,0,0,5TON/02, 3 4,1,1,"Futrelle, Mrs. Jacques Heath (Lily May Peel)",fer 5,0,3,"Allen, Mr. William Henry", male, 35,0,0,373450,8.0 6,0,3,"Moran, Mr. James", male,,0,0,330877,8.4583,,Q 7,0,1, "McCarthy, Mr. Timothy J", male, 54,0,0,17463,51.86 8,0,3,"Palsson, Master. Gosta Leonard", male,2,3,1,34990 9,1,3, "Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg 10,1,2, "Nasser, Mrs. Nicholas (Adele Achem)", female, 14,

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

Explorer AgeFareSc... import pandas as pd 2 import matplotlib.pyplot as plt 3 4 # Load the Titanic dataset data = pd.read_csv('Titanic-5 Dataset.csv') 6 7 # Data Cleaning 8 data['Age'].fillna(data['Age'].median (), inplace=True) data['Embarked'].fillna(data['Embarke 9 d'].mode()[0], inplace=True) data.drop('Cabin', axis=1, 10 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 Scatter 16 Plot for Age vs. Fare by Survived 17 colors = data['Survived'].map({0: 'red', 1: 'blue'}) plt.scatter(data['Age'], 18 data['Fare'], c=colors) 19 # Set labels and title 20 plt.xlabel("Age") 21 plt.ylabel("Fare") 22 plt.title("Age vs. Fare by Survival") 23 24 # Show the plot 25 plt.show() 26 27 28 29 30

3.