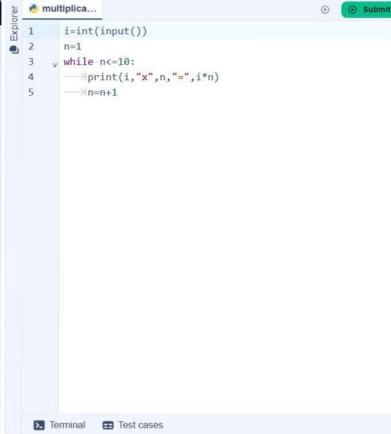
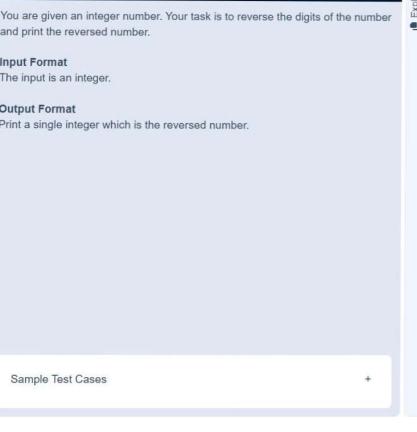


033M A € 🗹 ∂ -

1.1.5. Multiplication Table

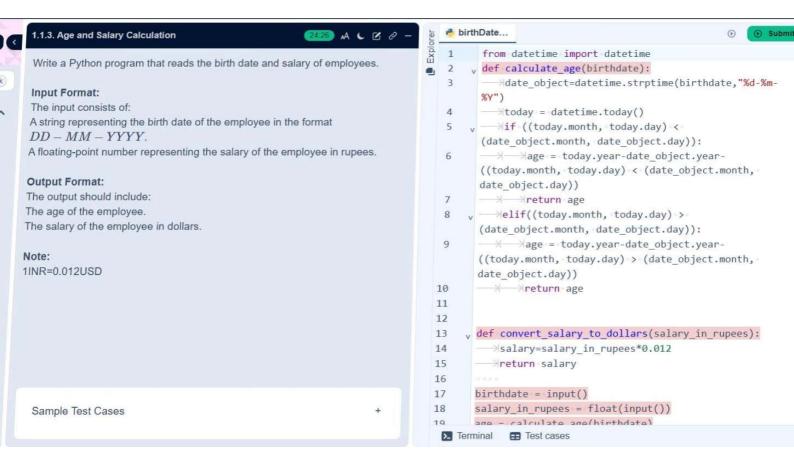


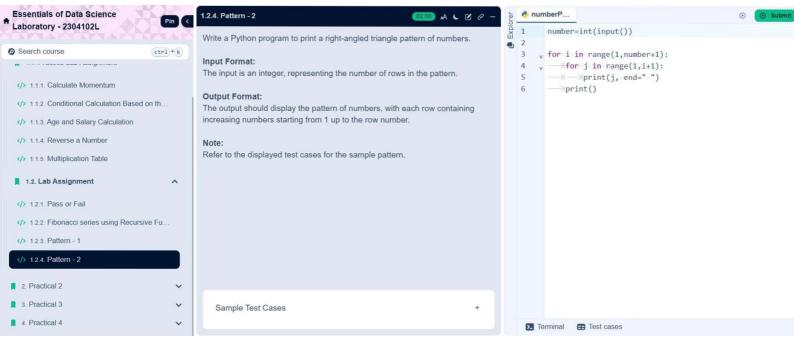


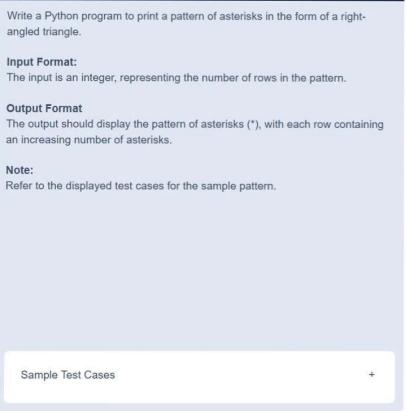
(0201) A L ☑ 8 -

1.1.4. Reverse a Number





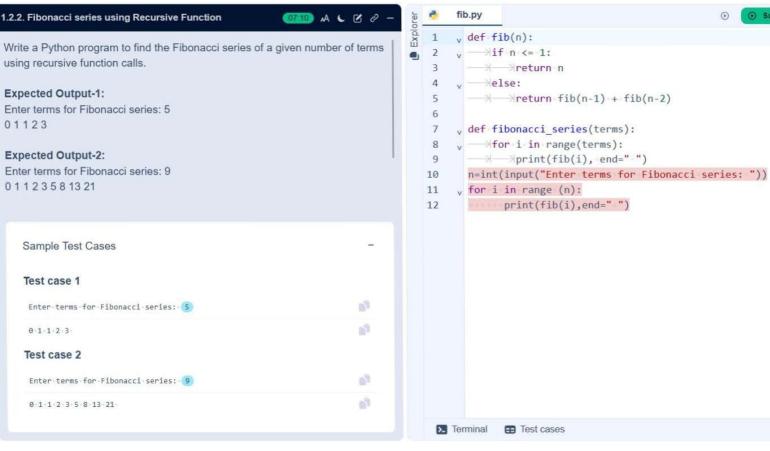


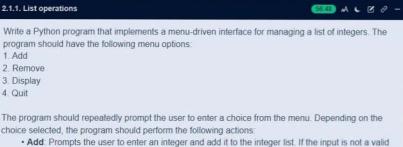


0152 A L & 2 -

1.2.3. Pattern - 1







- 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.

Sample Test Cases

• 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).

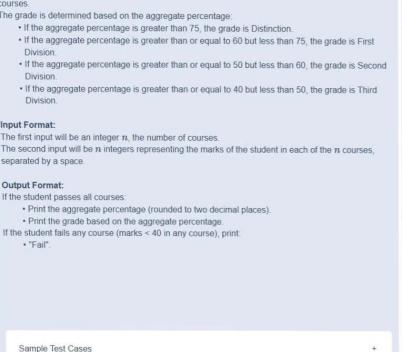
```
print("1. Add")
       print("2. Remove")
 4
       print("3. Display")
      print("4. Quit")
      n=int(input("Enter choice: "))
 8
     _____ if n==1:
 q
     a=int(input("Integer: "))
     → list.append(a)
     print(f"List after adding: {list}")
11
     Helif n==2:
12
13
     if len(list)==0:
14
     → >> >> print("List is empty")
      → welif len(list)!=0:
15
      16
     if remove not in list:
17
      > > > > > print("Element not found")
     —>|-->|else:
19
      > > > > list.remove(remove)
20
     21
22
     elif n==3:
       # Hif len(list)==0:
23
         print("List is empty")
24
     else:
25
26
       27
     elif n==4:
28
      break
     else:
29
     print("Invalid choice")
30
31
```

listOps.py

•

list=[]

v while True :



Write a Python program that accepts the number of courses and the marks of a student in those

2.1. Pass or Fail

```
passorFa...
  1
      v def calculate_grade(num_courses, marks):
•
            if any(mark < 40 for mark in marks):
            return "Fail"
         total_marks = sum(marks)
           aggregate_percentage = (total_marks / (num_courses * 100)) * 100
  10
         if aggregate_percentage > 75:
  11
  12
            grade = "Distinction"
  13
            elif 60 <= aggregate_percentage <= 75:
  14
               grade = "First Division"
  15
           elif 50 <= aggregate_percentage < 60:
  16
            grade = "Second Division"
           elif 40 <= aggregate_percentage < 50:
  17
  18
              grade = "Third Division"
  19
         else:
  20
               grade = "Fail"
  21
  22
          return f"Aggregate Percentage: {aggregate_percentage:.2f}\nGrade:
        {grade}"
  23
  24
  25
        num_courses = int(input())
  26
        marks = list(map(int, input().split()))
  27
        result = calculate_grade(num_courses, marks)
  29
  30
        print(result)
```

```
White a Python program to perform the following dictionary operation:

- Crose an entity dictionary and display is

- Can be sent of dictionary and display is

- Ash the user to update a key you have Prim' Yakes updated if the key root bound*

- Ash the user to update a key you have Prim' Yakes updated if the key root bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value. If the key docted many they not bound*

- Live pay) by rethers a value of the key docted many they not bound*

- Live pay) by rethers a value of the key docted many they not bound*

- Live pay) by rethers a value of the key docted many they not bound*

- Live pay) by rethers a value of the key docted many they not bound*

- Live pay) by rethers a value of the key work they not bound*

- Live pay) by rethers a value of the key work they not bound*

- Live pay (Live pay) by rether the key to update the pay to update they not pay to update the pay to update they not pay to update they not pay to update the new value:

- Live pay to update they not pay to update the pay to update the pay to update they not pay to update they not pay to update the pay to update they not pay to update the pay to update they not pay to update they not pay to update they not pay to update the
```

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
- · 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:

Sample Test Cases

print(result if result != -1 else "Not found") 11

 $\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny M}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny M}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}}{\stackrel{\mbox{\tiny L}}{\stackrel{\mbox{\tiny L}}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}{\stackrel{\mbox{\tiny L}}}}}}}}}}}}}}}}}}}}}}$ 

3 v if element == key: ->->->return index

----return -1

key=int(input())

4

5

7

8

9

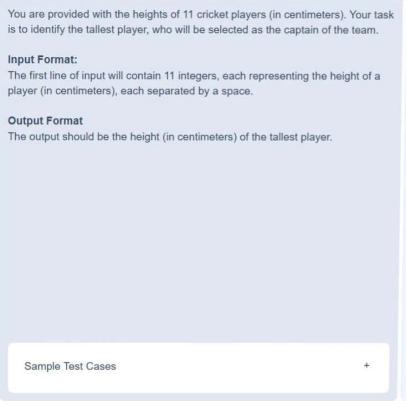
10

> Terminal

2 v —> for index, element in enumerate(arr):

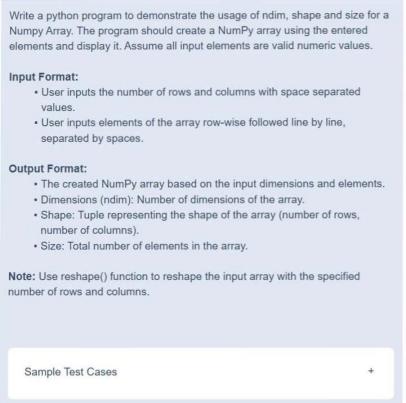
arr=list(map(int,input().split()))

result = linear\_search(arr,key)



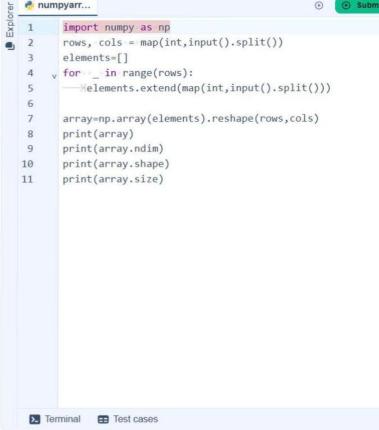
2.2.2. Captain of the Team





) A L B 2 -

3.1.1. Numpy array operations



The given code takes two  $3 \times 3$  matrices,  $matrix\_a$ , and  $matrix\_b$ , as input from the user and converts them into NumPy arrays.

## Task:

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

- 1. Addition (matrix\_a + matrix\_b)
- 2. Subtraction (matrix\_a matrix\_b)
- 3. Element-wise Multiplication (matrix\_a \* matrix\_b)
- 4. Matrix Multiplication (matrix\_a · matrix\_b)
- 5. Transpose of Matrix A

# Input Format:

- The user will input 3 rows for matrix\_a, each containing 3 integers separated by spaces.
- Similarly, the user will input 3 rows for matrix\_b, each containing 3 integers separated by spaces.

# **Output Format:**

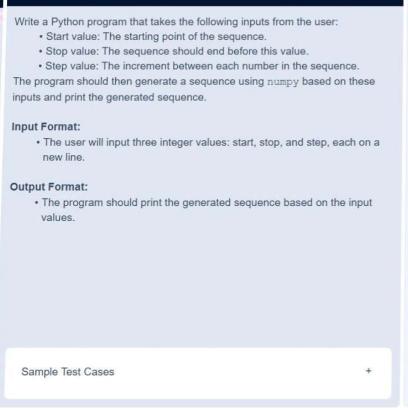
The program should display the results of the operations in the following order:

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

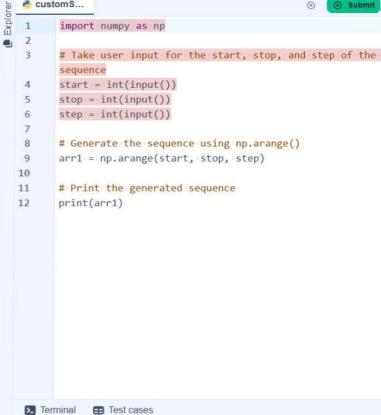
# Input matrices operations on them using NumPy. 3 • Horizontal Stacking: Stack the two matrices horizontally (side by side). print("Enter Array1:") 4 arr1 = np.array([list(map(int, input().split())) for i in range(3)]) • Vertical Stacking: Stack the two matrices vertically (one below the other). 5 print("Enter Array2:") Input Format: arr2 = np.array([list(map(int, input().split())) for i in range(3)]) • The program should first prompt the user to input two 3x3 arrays. 8 • Each array consists of 3 rows, and each row contains 3 space-separated integers. • The user will input the two arrays row by row. 10 # Perform horizontal stacking (hstack) a=np.hstack((arr1,arr2)) 11 print("Horizontal Stack:") Output Format: 12 • The program should display the result of the Horizontal Stack (side-by-side stacking) of the two 13 print(a) 14 • The program should then display the result of the Vertical Stack (one below the other) of the two # Perform vertical stacking (vstack) 15 a=np.vstack((arr1,arr2)) 16 print("Vertical Stack:") 17 18 print(a) Sample Test Cases 

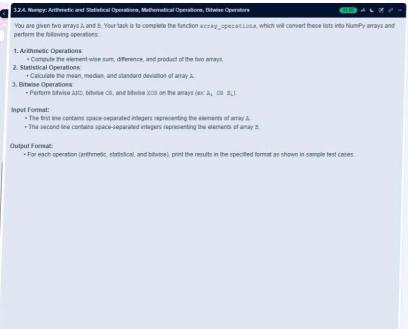
2

You are given two arrays arr1 and arr2. You need to perform horizontal and vertical stacking



3.2.3. Numpy: Custom Sequence Generation





Sample Test Cases

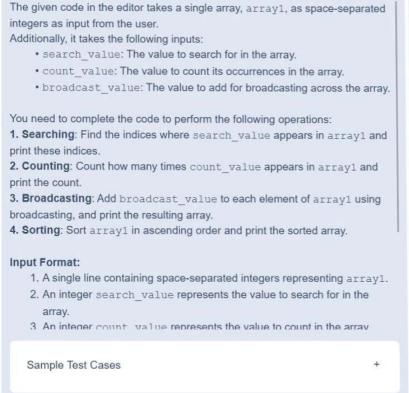
```
different...
                   import numpy as no
                 def array_operations(A, B):
                         # Convert A and B to NumPy arrays
                        # Arithmetic Operations
| sum_result = np.add(A,B)
| diff_result = np.subtract(A,B)
| prod_result = np.multiply(A,B)
  10
11
12
13
14
15
16
17
18
19
20
21
22
                         # Statistical Operations
                         mean_A = np.mean(A)
median_A = np.median(A)
std_dev_A = np.std(A)
                         # Bitwise Operations
                          and_result = np.bitwise_and(A,B)
or_result = np.bitwise_or(A,B)
xor_result = np.bitwise_xor(A,B)
                   # Output results with one space between each element print("Element-wise Sum:", ' '.join(map(str, sum_result))) print("Element-wise Difference:", ' '.join(map(str, diff_result))) print("Element-wise Product:", ' '.join(map(str, prod_result)))
  23
24
25
26
27
                      print(f"Mean of A: {mean_A}")
print(f"Median of A: {median_A}")
print(f"Standard Deviation of A: {std_dev_A}")
 28
29
30
31
32
33
34
35
36
37
                   print("Bitwise AND:", ' '.join(map(str, and_result)))
print("Bitwise OR:", ' '.join(map(str, or_result)))
print("Bitwise XOR:", ' '.join(map(str, xor_result)))
                 A = list(map(int, input().split())) # Elements of array A
B = list(map(int, input().split())) # Elements of array B
array_operations(A, B)
```



DALE?

3.2.5. Numpy: Copying and Viewing Arrays

```
Explorer
   copyAnd...
   1
          import numpy as np
   2
inputlist = list(map(int,input().split(" ")))
   3
   4
          # Original array
   5
   6
         original_array = np.array(inputlist)
   8
          # Create a view
   9
         view_array = original_array.view()
  10
         # Create a copy
  11
         copy_array = original_array.copy()
  12
          # Modify the view
  13
  14
         view_array[0] = 99
  15
         print("Original array after modifying view:",
         original_array)
  16
         print("View array:", view_array)
  17
  18
         # Modify the copy
  19
         copy_array[1] = 88
         print("Original array after modifying copy:",
  20
         original_array)
  21
         print("Copy array:", copy_array)
  22
    > Terminal
```



ALBR

3.2.6. Numpy: Searching, Sorting, Counting, Broadcasting

```
Explorer
   1
          import numpy as np
   2
          # Input array from the user
   3
   4
          array1 = np.array(list(map(int, input().split())))
   5
    6
          # Searching
    7
          search_value = int(input("Value to search: "))
   8
          count_value = int(input("Value to count: "))
   9
          broadcast_value = int(input("Value to add: "))
   10
          # Find indices where value matches in array1
   11
          a=np.where(array1==search_value)[0]
   12
          print(a)
   13
          # Count occurrences in array1
   14
          b=np.count_nonzero(array1==count_value)
   15
          print(b)
   16
          # Broadcasting addition
   17
          c= array1+broadcast_value
   18
          print(c)
   19
          # Sort the first array
          d= np.sort(array1)
   20
   21
          print(d)
                 ■ Test cases
     > Terminal
```

Wittle 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 at list 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 at list student roll numbers: 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 lowest marks in Subject 3.

Print all subject marks: Display the marks of all students for each student.

Find the average marks of subject 1 identify the highest trans in Subject 3.

Find the average marks of each subject. Compute the average marks for each student.

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 2: 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 the students who obtained exactly 24 marks in Subject 2 and print their roll number.

Find the roll number of students who scored 3 marks in Subject 2: Count the number of students who scored less than 40 marks in Subject 1.

Find the count of students who got less than 40 marks in Subject 2: Count the number of students who scored less than 40 marks in Subject 1.

Find the count of students who scored >=90 in each subject: Count the number of students who scored of 90 or more marks in Print Subject 1 marks in ascending order.

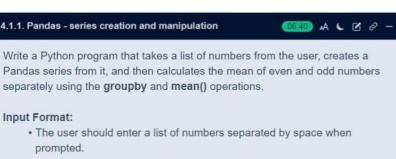
Find the count o

Sample Test Cases

import numpy as np
import numpy as np
import numpy as np

a = np.loadtxt('Sample.csv", delimiter=',', skiprows=1)
print('All student Details:\n",a)
print('Total Students:",r)
print('Notal Students:",r)
print('Mal Students Roll Nos",a[:,0])
print('Subject 1 Marks",a[:,1])
print('Max marks in Subject 2",np.min(a[:,2]))
print('Max marks in Subject 2",np.max(a[:,3]))
print('Max marks in Subject 2",np.max(a[:,3]))
print('Notal Marks",np.sum(a[:,1:],exis=1))
avg=np.mean(a[:,1:],exis=1)
print('Notal Marks',np.sum(a[:,1:],exis=1))
avg=np.mean(a[:,1:],exis=1)
print('Notarage Marks of Si and S2",np.mean(a[:,1:],axis=0))
print('Roll no who got maximum marks in Subject 2",a[m,0])
whren\_p.mer(a[:,1:]=1)
print('Roll no who got marks in Subject 2",a[whr,0])
tchen\_p.umer(a[:,2:]=24)
print('Roll no who got tanks in Subject 2",a[whr,0])
tchen\_p.count\_nonzero(a[:,1:]>90,axis=0))
print('Count of students who got marks in Subject 1 < 40",ct)
print('Count of students who got marks in Subject 2 > 90:",rp.count\_nonzero(a[:,1:]>90,axis=0))
print('Roll no: "a, "i, 0])
print('Count of students who got marks in Subject 2 > 90:",np.count\_nonzero(a[:,1:]>90,axis=0))
print('Roll no: "a, "i, 0])
print('Count of students who got marks in Subject 2",a[whr,0])
print('Count of students who got marks in Subject 2",a[whr,0])
print('Count of students who got marks in Subject 2",a[whr,0])
tchen\_p.count\_nonzero(a[:,1:]>90,axis=0))
print('Roll no wh

Da Te



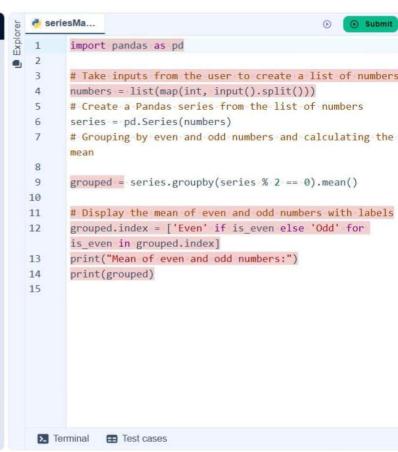
prompted.

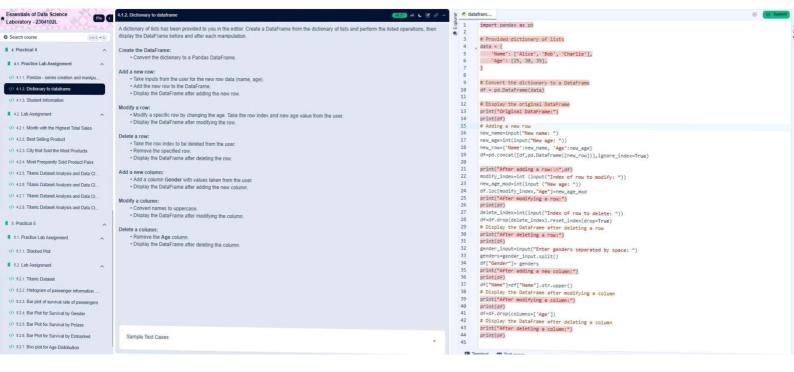
# Output Format:

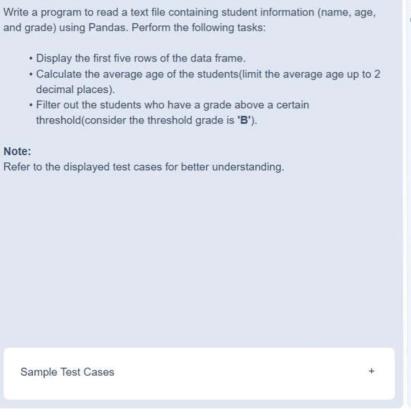
- The program should display the mean of even and odd numbers separately.
- Each mean value should be displayed with a label indicating whether it corresponds to even or odd numbers.

Corresponds to even or odd numbers.

Sample Test Cases +







0149 A C Z 2

4.1.3. Student Information

```
💆 studentin... 🔞 🖹 studentdat... 🔞
Explorer
   1
          import pandas as pd
   2
.
          # Read the text file into a DataFrame
    3
    4
          file = input()
          data = pd.read_csv(file, sep="\s+", header=None, names
    5
          ["Name", "Age", "Grade"])
    6
    7
   8
          # write your code here..
   9
   10
          print("First five rows:")
  11
          print(data.head())
  12
          avg=round(data["Age"].mean(),2)
  13
          print("Average age:",avg)
   14
          # write your code here..
   15
          filter=data[data['Grade']<='B']</pre>
   16
          print("Students with a grade up to B")
   17
   18
          print(filter)
    > Terminal

☐ Test cases
```

```
data, and performs the following operations:
     • The CSV file contains the columns: Date, Product, Quantity,
       Price, and City.
     • Group the data by Month and calculate the total sales for each month.
     • Find the month with the highest total sales and display it.
     · Also, display the total sales for the best month.
Sample Data:
  Date, Product, Quantity, Price, City
  2025-01-01, Product A, 5, 20, New York
  2025-01-01, Product B, 3, 15, Los Angeles
  2025-01-02, Product A, 7, 20, New York
  2025-01-02, Product C, 4, 30, Chicago
  2025-01-03, Product B, 2, 15, Chicago
  2025-01-03, Product A, 8, 20, Los Angeles
  2025-01-04, Product C, 6, 30, New York
  2025-01-04, Product B, 5, 15, Los Angeles
  2025-01-05, Product A, 3, 20, Chicago
  2025-01-05, Product C, 10, 30, Los Angeles
Moto:
    Sample Test Cases
```

Write a Python program that takes the file name of a CSV file as input, reads the

A L & 2 -

4.2.1. Month with the Highest Total Sales

```
Explorer
   monthFor... S sales_dat... S
                                                         Submit
   1
          import pandas as pd
   2
.
          # Prompt the user for the file name
    3
    4
          file_name = input()
    5
          # Load the data
          df = pd.read csv(file name)
          df['Date'] = pd.to datetime(df['Date'])
    9
          # Extract the month from the Date column
   10
          df['Month'] = df['Date'].dt.to_period('M')
   11
   12
          # Calculate the total sales for each row
   13
          df['Total_Sales'] = df['Quantity'] * df['Price']
   14
          # Group the data by Month and calculate the total
   15
          sales for each month
   16
          monthly_sales = df.groupby('Month')
          ['Total_Sales'].sum()
   17
          # Find the month with the highest total sales
   18
   19
          best_month = monthly sales.idxmax()
   20
          highest sales = monthly sales.max()
           print(f"Best month: {best month}")
   21
           print(f"Total sales: ${highest_sales:.2f}")
    22
     > Terminal
                 Test cases
```



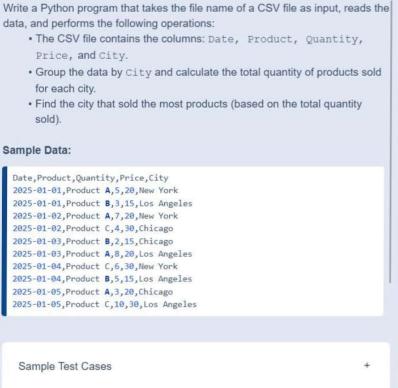
4.2.2. Best Selling Product

```
Explorer
   1
          import pandas as pd
   2
    3
          # Prompt the user for the file name
    4
          file_name = input()
    5
    6
          # Load the data
    7
          df = pd.read_csv(file_name)
   8
   9
          # Find the product with the highest total quantity sole
          product_sales = df.groupby("Product")["Quantity"].sum()
   10
   11
          best_product = product_sales.idxmax()
  12
          highest_quantity = product_sales.max()
  13
          # Display the result
  14
          print(f"Best selling product: {best_product}")
  15
  16
          print(f"Total quantity sold: {highest_quantity}")
   17
    > Terminal

    ⊞ Test cases
```

monthFor... 
sales\_dat...

ALB8-



01112 A L B 8 -

4.2.3. City that Sold the Most Products

```
Explorer
   monthFor... 
sales_dat...
                                                           Submit
   1
          import pandas as pd
    2
.
          # Prompt the user for the file name
    3
    4
          file_name = input()
    5
          # Load the data
          df = pd.read_csv(file_name)
    7
          # write the code..
    8
    9
          city_sales = df.groupby("City")["Quantity"].sum()
  10
  11
          # Find the city with the highest total quantity sold
  12
          best_city = city_sales.idxmax()
  13
          # Display the result
  14
          print(f"City sold the most products: {best_city}")
  15
  16
    > Terminal

    ⊞ Test cases
```

# 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.

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

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

The Titanic dataset contains columns as shown below

Pas sen gerl d	Sur vive d	PcI ass	Na me	Sex	Age	Sib Sp	Par ch	Tick et	Far e	Cab in	Em bar ked

#### Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked
1 0 3 "Braund Mc Dwan Harris" male 22 1 0 4/5 21171 7 25 5

Sample Test Cases

1 import pandas as pd 2 🛕 import numpy as np . 4 # Load the Titanic dataset 5 data = pd.read\_csv('Titanic-Dataset.csv') 6 # 1. Display the first 5 rows of the dataset 8 print(data.head()) 9 # 2. Display the last 5 rows of the dataset 10 print(data.tail()) 11 # 3. Get the shape of the dataset (number of rows and columns) 12 print(data.shape) # 4. Get a summary of the dataset 13 14 print(data.info()) 15 # 5. Get basic statistics of the dataset 16 print(data.describe()) 17 # 6. Check for missing values and display the count 18 print(data.isnull().sum()) 19 # 7. Fill missing values in the 'Age' column with the median age 20 data['Age'].fillna(data['Age'].median(), inplace=True) 21 # 8. Fill missing values in the 'Embarked' column with the most frequent value (mode) 22 data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True) 23 24 # 9. Drop the 'Cabin' column due to many missing values 25 data.drop(columns=['Cabin'], inplace=True) 26 # 10. Create a new column 'FamilySize' by adding 'SibSp' and 'Parch' 27 28 data['FamilySize'] = data['SibSp'] + data['Parch'] 29 30 31

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

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

The Titanic dataset contains columns as shown below.

Pas sen geri d	Sur vive d	Pcl ass	Na me	Sex	Age	Sib Sp	Par ch	Tick et	Far e	Cab	Em bar ked

#### Sample Data:

PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked 1,0,3, "Braund, Mr. Owen Harris", male,22,1,0,A/S 21171,7.25,,S
2,1,1,"Cumings, Mrs. John Bradley (Florence Briggs Thayer)", female,38,1,0,PC 17599,71.2833,C85,C

```
Exp
         import pandas as pd
         import numpy as np
         # Load the Titanic dataset
   4
         data = pd.read_csv('Titanic-Dataset.csv')
         data['FamilySize'] = data['SibSp'] + data['Parch']
         # 1. Create a new column 'IsAlone' (1 if alone, 0 otherwise)
         data['IsAlone'] = np.where(data['FamilySize'] == 0, 1, 0)
   8
         # 2. Convert 'Sex' to numeric (male: 0, female: 1)
   9
         data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
  10
  11
         # 3. One-hot encode the 'Embarked' column
  12
         data = pd.get_dummies(data, columns=['Embarked'])
         # 4. Get the mean age of passengers
  13
  14
         mean_age = data['Age'].mean()
  15
         print( mean_age)
  16
         # 5. Get the median fare of passengers
  17
         median_fare = data['Fare'].median()
  18
         print( median_fare)
         # 6. Get the number of passengers by class
  19
         print( data['Pclass'].value_counts())
  20
  21
         # 7. Get the number of passengers by gender
  22
         print( data['Sex'].value_counts())
         # 8. Get the number of passengers by survival status
  23
  24
         print( data['Survived'].value_counts())
  25
         # 9. Calculate the overall survival rate
         survival_rate = data['Survived'].mean()
  26
  27
         print(format(survival_rate))
  28
         # 10. Calculate the survival rate by gender
         survival_by_gender = data.groupby('Sex')['Survived'].mean()
  29
  30
         print( survival_by_gender)
   31
  32
```

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

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

The Titanic dataset contains columns as shown below,

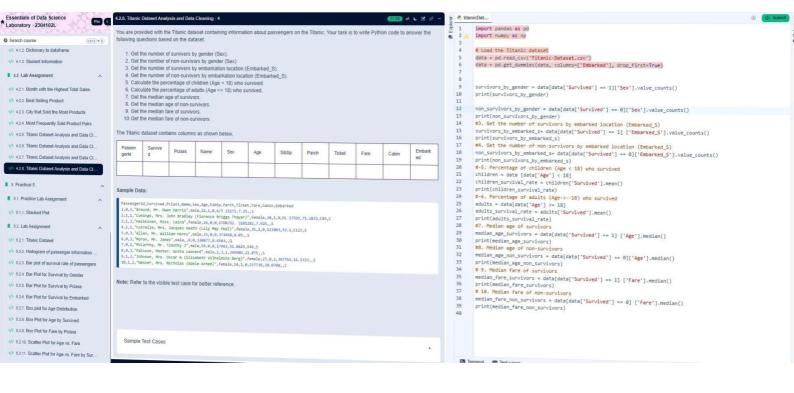
Pas sen gerl d	Sur vive d	Pcl ass	Na me	Sex	Age	Sib Sp	Par ch	Tick et	Far e	Cab	Em bar ked

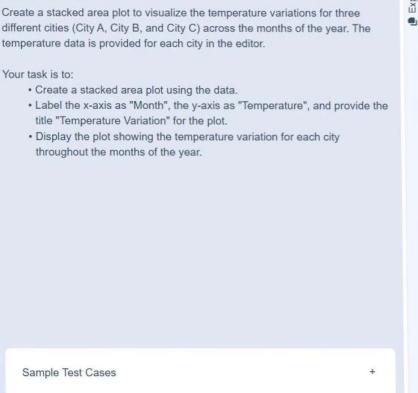
#### Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked 1,0,3, Braund, Mr. Owen Harris\*, male,22,1,0,A/5 21171,7.25,,5
2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thayer)", female,38,1,0,PC 17599,71.2833,C85,C

Sample Test Cases

1 import pandas as pd import numpy as np # Load the Titanic dataset 4 data = pd.read\_csv('Titanic-Dataset.csv') 5 6 data['FamilySize'] = data['SibSp'] + data['Parch'] 7 data['IsAlone'] = np.where(data['FamilySize'] > 0, 0, 1) data = pd.get\_dummies(data, columns=['Embarked'], drop\_first=True) 8 9 10 # 1. Calculate the survival rate by class 11 12 print(data.groupby('Pclass') ['Survived'].mean()) 13 #2. Calculate the survival rate by embarked location (Embarked S) print(data.groupby('Embarked\_S') ['Survived'].mean()) 14 15 #3. Calculate the survival rate by family size 16 print(data.groupby('FamilySize') ['Survived'].mean()) 17 #4. Calculate the survival rate by being alone 18 print(data.groupby('IsAlone') ['Survived'].mean()) 19 #5. Get the average fare by class print(data.groupby('Pclass') ['Fare'].mean()) 20 21 #6. Get the average age by class print(data.groupby('Pclass') ['Age'].mean()) 22 23 #7. Get the average age by survival status 24 print(data.groupby('Survived')['Age'].mean()) 25 #8. Get the average fare by survival status 26 print(data.groupby('Survived') ['Fare'].mean()) 27 #9. Get the number of survivors by class (sort by values descending) print(data[data['Survived'] == 1] ['Pclass'].value\_counts()) 29 #10. Get the number of non-survivors by class (sort by values descending) 30 print(data[data['Survived'] == 0] ['Pclass'].value\_counts()) 31





0102 A L Z 2

5.1.1. Stacked Plot

```
stackedpl...
Explorer
                                                         Submit
   1
          import matplotlib.pyplot as plt
   2
          import pandas as pd
   3
   4
          # Data for Months and Temperature for three cities
   5
   6
              'Month': ['January', 'February', 'March', 'April',
          'May', 'June', 'July', 'August', 'September',
          'October', 'November', 'December'],
             'City_A_Temperature': [5, 7, 10, 13, 17, 20, 22,
    7
          21, 18, 12, 8, 6],
             'City_B_Temperature': [2, 3, 5, 6, 10, 14, 16, 17,
    8
          12, 9, 5, 3],
          'City_C_Temperature': [3, 4, 6, 8, 9, 12, 15, 14,
    9
          10, 7, 4, 2]
   10
   11
   12
          # Write your code...
          plt.stackplot(data['Month'],data['City_A_Temperature']
   13
          data['City_B_Temperature'],data['City_C_Temperature'])
          plt.xlabel('Month')
   14
          plt.ylabel('Temperature')
   15
          plt.title('Temperature Variation')
   16
   17
          plt.show()
     > Terminal

	☐ Test cases
```

Write a Python program to analyze and visualize data from the Titanic dataset based on the following instructions import pandas as pd import matplotlib.pyplot as plt Dataset Information Dataset Information:
The dataset is stored in a CSV file named titanic.csv and has been loaded using the pandas library. It contains the following columns:

\* Colass: Passenger class (1 = First, 2 = Second, 3 = Third).

\* Gender: Gender of the passenger (male/female).

\* Age: Age of the passenger. # Load the Titanic dataset from the CSV file df = pd.read\_csv('titanic.csv') # Set up the figure for 5 subplots fig, axes = plt.subplots(3, 2, figsize=(12, 12)) # write the code.. Survived: Survival status (0 = Did not survive, 1 = Survived).
 Fare: Ticket fare paid by the passenger. write the code.
import pandas as pd
import matplotlib.pyplot as plt
# Load the Titanic dataset from the CSV file
df = pd.read\_csv('titanic.csv')
# Set up the figure for S subplots
fig, axes = plt.subplots(3, 2, figsize=(12, 12))
# Plot 1: Count of passengers by class
axes[0, 0].bar(df['Pclass'].value\_counts().index, df['Pclass'].value\_counts(), color='skyblue')
axes[0, 0].set\_vitle("Passenger class Distribution")
axes[0, 0].set\_vitle("Pclass")
axes[0, 0].set\_vitle("Count")
# Plot 2: Gender distribution
axes[0, 1].pic(dff'(Gender'].value\_counts(), labels=dff'Gender'].value\_counts() index\_value\_counts(). 12 To represent these trends, you will create 5 visualizations using Matplotlib. The visualizations should be arranged in a 3x2 grid (3 rows and 2 18 19 20 21 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). avxs[0, 1].pie(df['Gender'].value\_counts(), labels=df['Gender'].value\_counts().index, autopct='%1.1f%6',
colors=['lightblue', 'lightcoral'])
axes[0, 1].set\_title("Gender Distribution") Use the color akyblue for the bars.
Title the plot as "Passenger Class Distribution".
Label the x-axis as "Pclass" and the y-axis as "Count". 22 23 24 25 axes[0, 1].set\_[tite("Gender Diskribution")
# Plot 3: Age distribution
axes[1, 0].hist(df['Age'].dropna(), bins=8, color='lightgreen', edgecolor='black')
axes[1, 0].set\_title("Age Distribution")
axes[1, 0].set\_vlabel("Age")
axes[1, 0].set\_vlabel("Frequency") Pie Chart (Gender Distribution):

Create a pie chart to display the distribution of male and female passengers.

Use lightblue for males and lightboral for females. 26 27 \* Use lightblue for males and lightbocal for remails.

Include percentages on the silices (use autopot='\$1.1f\u00e8').

\*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').

\*Cet the number of bine to \$0 for the histogram to a f # Plot 4: Survival count 
# Plot 4: Survival count 
axes[1, 1].bar(df['Survived'].value\_counts().index, df['Survived'].value\_counts(), color=['lightblue', 'lightcoral'])
axes[1, 1].set\_title("Survival Count") 31 32 33 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): axes[1, 1].set\_title("Survival Count")
axes[1, 1].set\_vlabel("Survival (0 = No, 1 = Yes)")
axes[1, 1].set\_vlabel("Count")
# Plot 5: Fare vs Age
axes[2, 0].set\_title("fare"), df['Fare'], color='orange', edgecolors='black')
axes[2, 0].set\_title("Fare vs Age") Create a bar plot to show the count of passengers who survived and those who did to use the colors hightblue for survivors (1) and hightcoral for non-survivors (0). ngers who survived and those who did not, based on the Survived colu axes[2, 0].set\_xlabel("Age")
axes[2, 0].set\_ylabel("Fare") Title the plot as "Survival Count".

Label the x-axis as "Survived (0 = No, 1 = Yes)" and the y-axis as "Count". plt.tight\_layout()
plt.show() Sample Test Cases

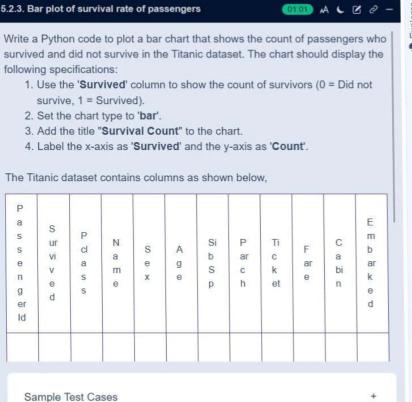
#### Write a Python code to plot a histogram for the distribution of the 'Age' column from the Titanic dataset. The histogram should display the frequency of different age ranges with the following specifications: 1. Use 30 bins for the histogram. 2. Set the edge color of the bars to black (k). 3. Label the x-axis as 'Age' and the y-axis as 'Frequency'. 4. Add the title "Age Distribution" to the histogram. The Titanic dataset contains columns as shown below, P E a S S P m N P Ti C ur Si F cl S A h S vi ar е a ar q S bi n S k et е h е g d ld

0218 A L Z & -

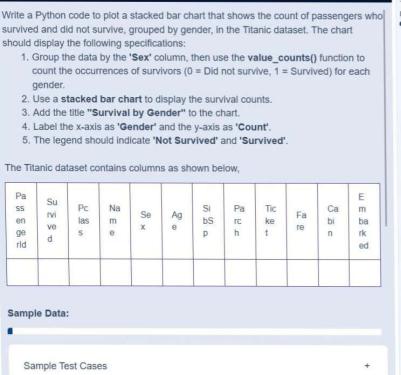
5.2.2. Histogram of passenger information of Titanic

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

    ⊞ Test cases
```



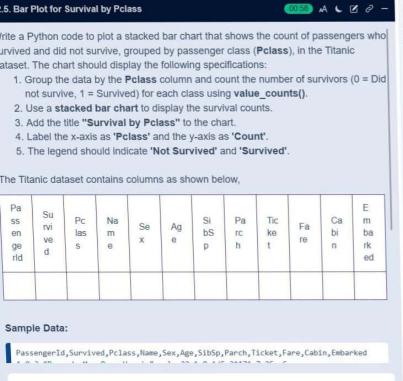
```
Explorer
  BarPlotOf...
                                                        Submit
   1
         import pandas as pd
   2
         import matplotlib.pyplot as plt
4
   3
         # Load the Titanic dataset
   4
   5
         data = pd.read_csv('Titanic-Dataset.csv')
   6
    7
          # Data Cleaning
   8
          data['Age'].fillna(data['Age'].median(), inplace=True)
   9
          data['Embarked'].fillna(data['Embarked'].mode()[0],
          inplace=True)
          data.drop('Cabin', axis=1, inplace=True)
   10
   11
   12
          # Convert categorical features to numeric
   13
          data['Sex'] = data['Sex'].map({'male': 0, 'female': 1}
          data = pd.get_dummies(data, columns=['Embarked'],
   14
          drop_first=True)
   15
          # Write your code here for Bar Plot for Survival Rate
   16
   17
   18
          survival counts = data['Survived'].value counts()
   19
          survival counts.plot(kind='bar')
          plt.title('Survival Count')
   20
   21
          plt.xlabel('Survived')
   22
          plt.ylabel('Count')
   23
          plt.show()
```



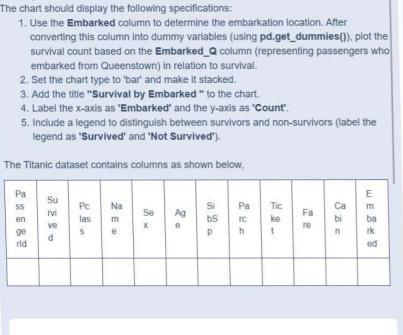
A L B 8

5.2.4. Bar Plot for Survival by Gender

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



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



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.

00249 A L Z 2 -

.2.6. Bar Plot for Survival by Embarked

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

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

The Titanic dataset contains columns as shown below,

Pa ss en ge rld	Su rvi ve d	Pc las s	Na m e	Se x	Ag e	Si bS p	Pa rc h	Tic ke t	Fa re	Ca bi n	E m ba rk ed

## Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked 1,0,3, "Braund, Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,5 2,1,1,"Cumings, Mrs. John Bradley (Florence Briggs Thayer)",female,38,1,0,PC 17599,7 3,1,3,"Heikkinen, Miss. Laina",female,26,0,0,STON/O2. 3101282,7.925,,S

Sample Test Cases

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

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

The Titanic dataset contains columns as shown below,

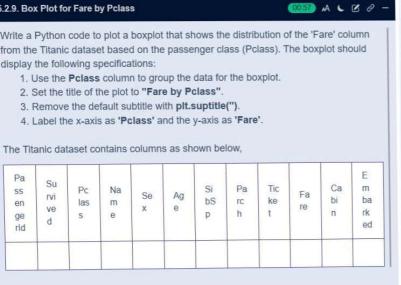
Pa ss en ge rld	Su rvi ve d	Pc las s	Na m e	Se x	Ag e	Si bS p	Pa rc h	Tic ke t	Fa re	Ca bi n	m ba rk ed
nu											Cu

#### Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked 1,0,3," Fraund, Mr. Owen Harris", male, 22,1,0,A/5 21171,7.25,,5
2 1 1 "Cumings Mrs John Bradley (Florence Briggs Thayer)" female 38 1 0 PC 17599 7

Sample Test Cases

BoxPlotF... Explorer import pandas as pd 1 import matplotlib.pyplot as plt 2 • 3 # Load the Titanic dataset 1 5 data = pd.read\_csv('Titanic-Dataset.csv') 6 # Data Cleaning data['Age'].fillna(data['Age'].median(), inplace=True) 8 9 data['Embarked'].fillna(data['Embarked'].mode()[0], inplace=True) data.drop('Cabin', axis=1, inplace=True) 10 11 # Convert categorical features to numeric 12 data['Sex'] = data['Sex'].map({'male': 0, 'female': 1}) 13 data = pd.get\_dummies(data, columns=['Embarked'], 14 drop\_first=True) 15 16 # Write your code here for Box Plot for Age by Survived 17 plt.figure(figsize=(8, 6)) 18 19 data.boxplot(column='Age', by='Survived') plt.suptitle('') 20 21 plt.title('Age by Survival') plt.xlabel('Survived') plt.ylabel('Age') 23 24 plt.show() 25 > Terminal ☐ Test cases



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

2,1,1, "Cumings, Mrs. John Bradley (Florence Briggs Thayer)",female,38,1,0,PC 17599,7

1,0,3,"Braund, Mr. Owen Harris", male,22,1,0,A/5 21171,7.25,,S

3,1,3, "Heikkinen, Miss. Laina", female, 26,0,0, STON/02. 3101282,7.925,,S

Sample Data:

```
BoxPlotF
Explorer
         import pandas as pd
   1
         import matplotlib.pyplot as plt
   2
.
                                                                       4
   3
         # Load the Titanic dataset
   4
         data = pd.read_csv('Titanic-Dataset.csv')
   6
         # Data Cleaning
   7
         data['Age'].fillna(data['Age'].median(), inplace=True)
   8
         data['Embarked'].fillna(data['Embarked'].mode()[0],
   9
         inplace=True)
         data.drop('Cabin', axis=1, inplace=True)
  10
  11
          # Convert categorical features to numeric
  12
          data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
   13
          data = pd.get_dummies(data, columns=['Embarked'],
   14
          drop_first=True)
   15
          # Write your code here for Box Plot for Fare by Pclass
   16
   17
   18
          plt.figure(figsize=(8, 6))
   19
          data.boxplot(column='Fare', by='Pclass')
   20
          plt.suptitle('')
   21
          plt.title('Fare by Pclass')
   22
          plt.xlabel('Pclass')
   23
          plt.ylabel('Fare')
   24
          plt.show()
   25
   26
```



Write a Python code to plot a scatter plot showing the relationship between the 'Age'

(0317) A L Z 2 -

.2.10. Scatter Plot for Age vs. Fare

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

#### 5.2.11. Scatter Plot for Age vs. Fare by Survived



AgeFareS...

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

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

The Titanic dataset contains columns as shown below,

Pa ss en ge rld	Su rvi ve d	Pc las s	Na m e	Se x	Ag e	Si bS p	Pa rc h	Tic ke t	Fa re	Ca bi n	m ba rk ed
-----------------------------	----------------------	----------------	--------------	---------	---------	---------------	---------------	----------------	----------	---------------	---------------------

### Sample Data:

PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked 1,0,3,"Braund, Mr. Owen Harris",male,22,1,0,4/5 21171,7.25,,5 2 1 1 "Cumings Mrs John Bradley (Florence Briggs Thaver)" female 38 1 0 PC 17599 7

```
Explore
         import pandas as pd
   1
         import matplotlib.pyplot as plt
.
         # Load the Titanic dataset
   4
         data = pd.read_csv('Titanic-Dataset.csv')
   5
   6
         # Data Cleaning
         data['Age'].fillna(data['Age'].median(), inplace=True)
   8
         data['Embarked'].fillna(data['Embarked'].mode()[0],
   9
          inplace=True)
   10
         data.drop('Cabin', axis=1, inplace=True)
   11
          # Convert categorical features to numeric
   12
          data['Sex'] = data['Sex'].map({'male': 0, 'female': 1})
   13
          data = pd.get_dummies(data, columns=['Embarked'],
   14
          drop_first=True)
   15
          # Write your code here for Scatter Plot for Age vs. Fare by
          Survived
          plt.figure()
   16
   17
          colors = {0: 'red', 1: 'blue'}
   18
          plt.scatter(data['Age'], data['Fare'],
          c=data['Survived'].apply(lambda x: colors[x]))
          plt.title('Age vs. Fare by Survival')
   19
          plt.xlabel('Age')
   21
          plt.ylabel('Fare')
          plt.show()
   22
   23
   24
```