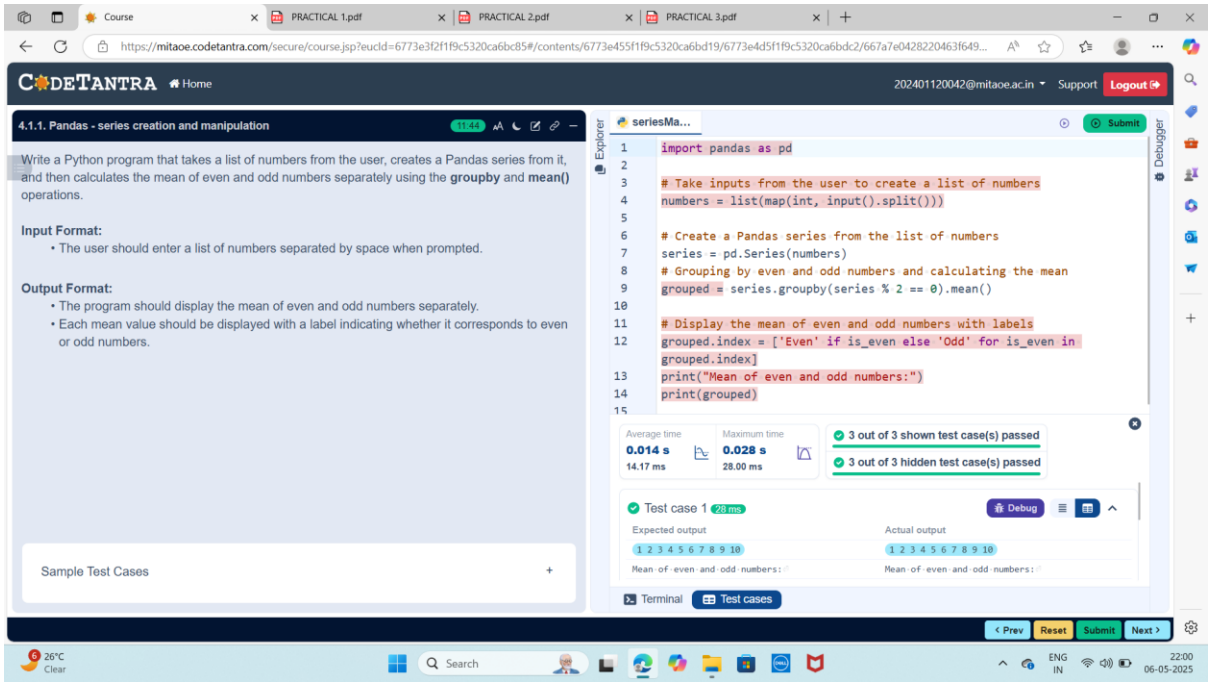
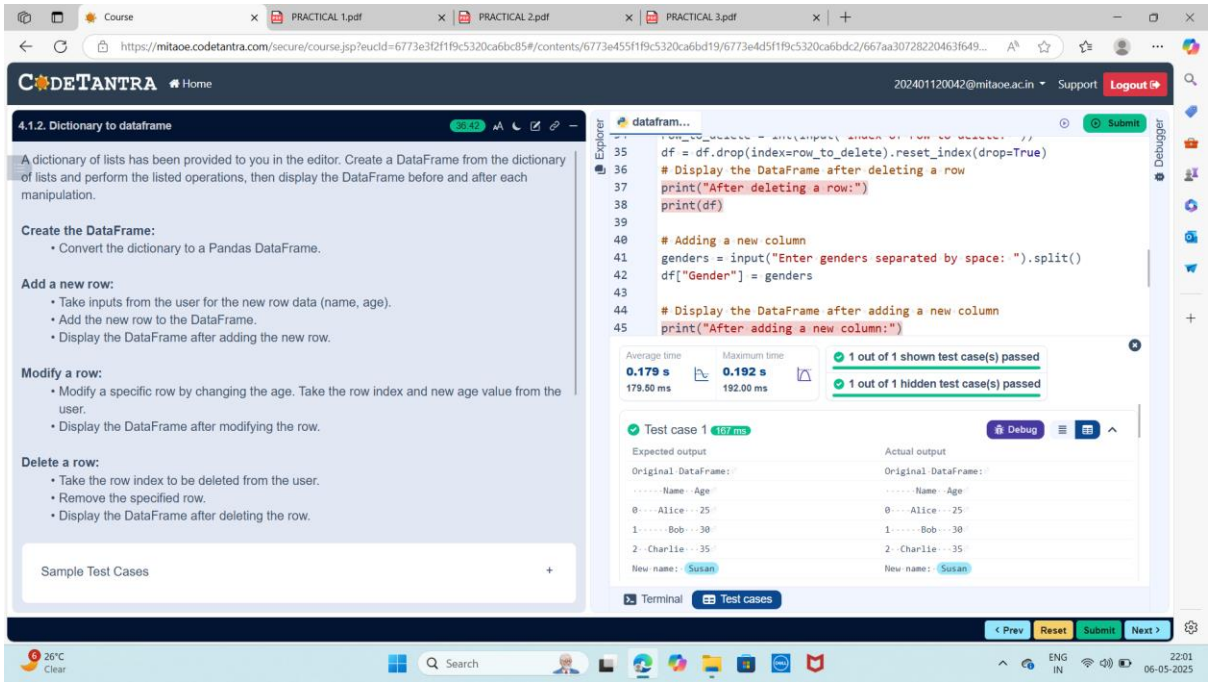
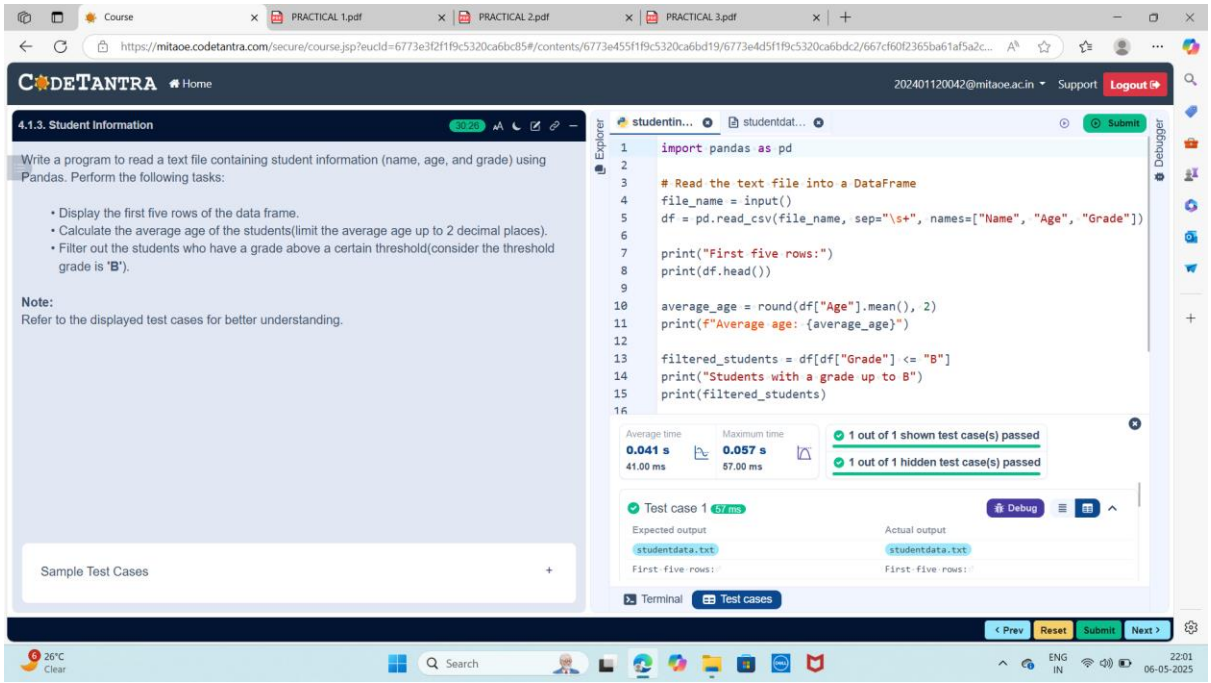
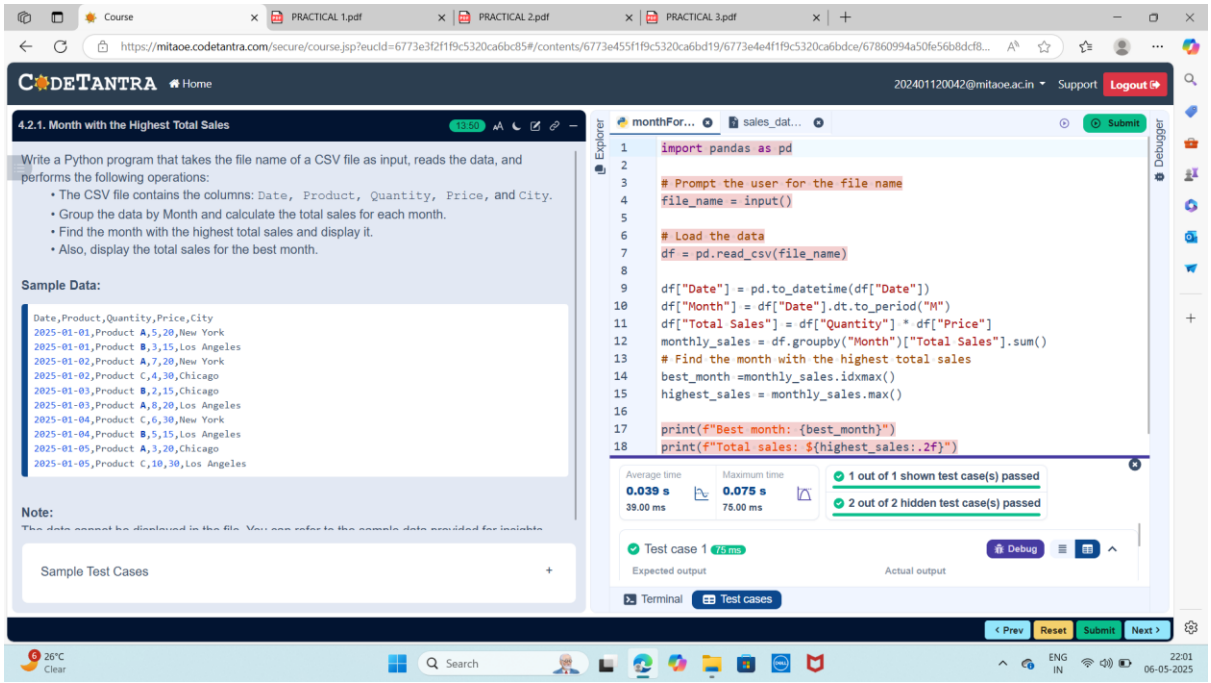
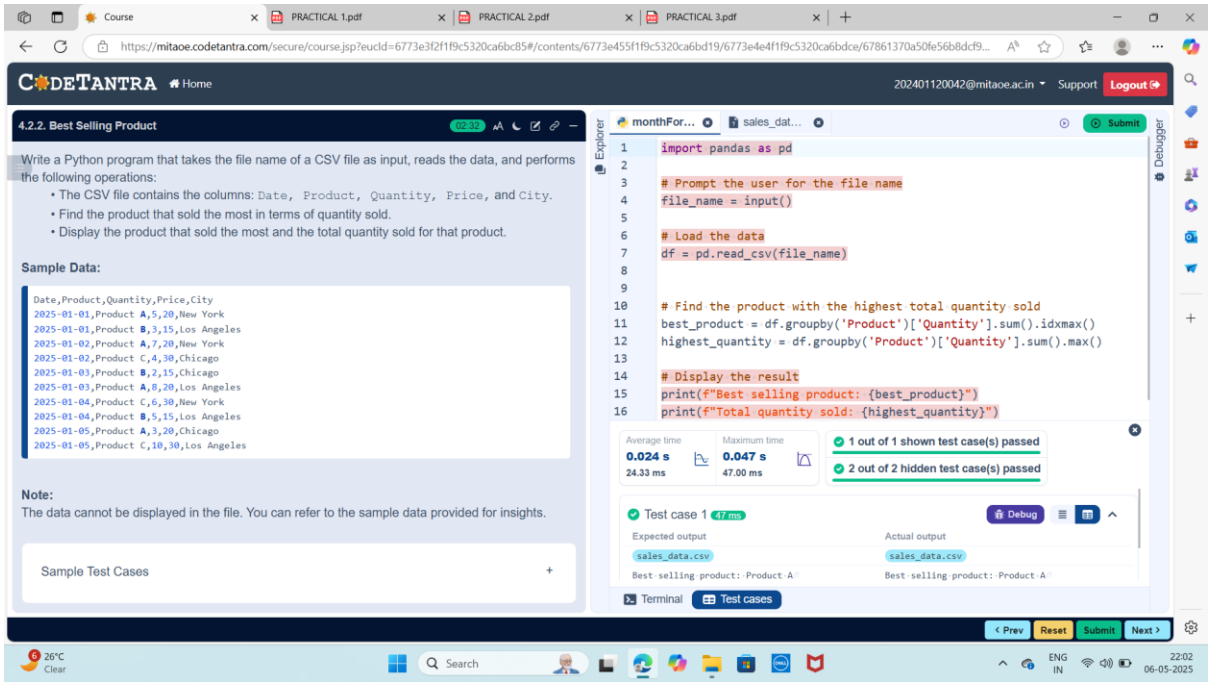
PRACTICAL 4:

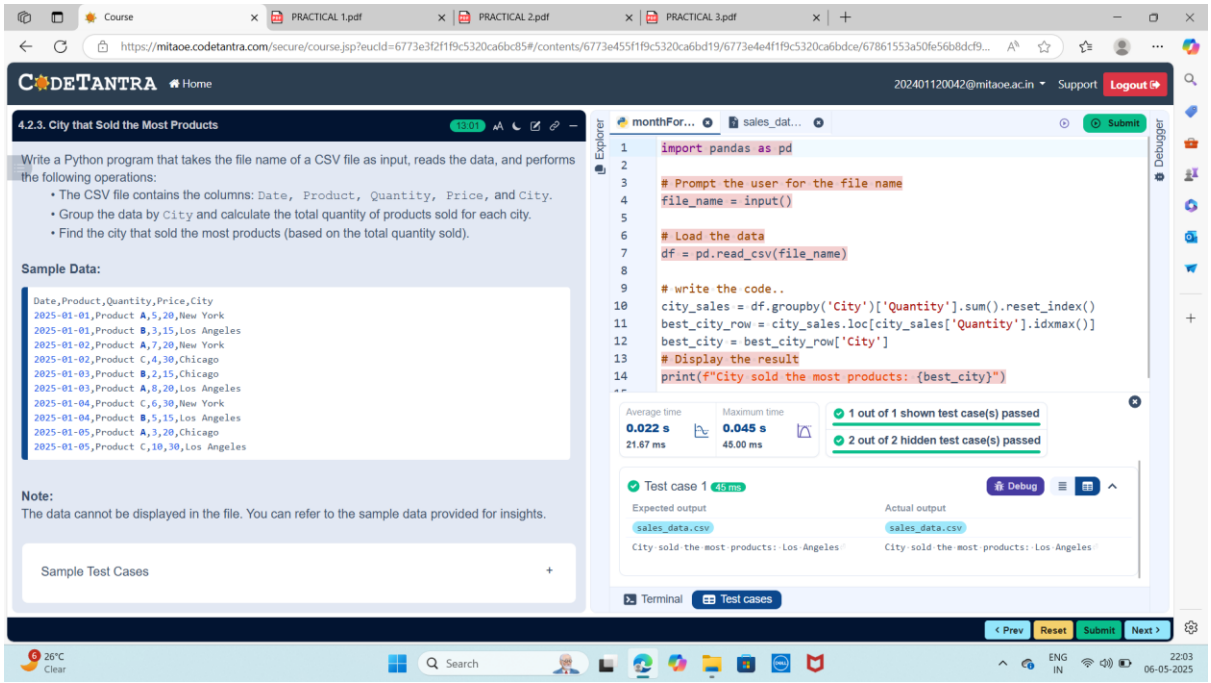

CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.1.1. Pandas - series creation and manipulation
Write a Python program that takes a list of numbers from the user, creates a Pandas series from it, and then calculates the mean of even and odd numbers separately using the groupby and mean() operations.
Input Format:
• The user should enter a list of numbers separated by space when prompted.
Output Format:
• The program should display the mean of even and odd numbers separately.
• Each mean value should be displayed with a label indicating whether it corresponds to even or odd numbers.
Sample Test Cases
seriesMa...
Submit
import pandas as pd
# Take inputs from the user to create a list of numbers
numbers = list(map(int, input().split()))
# Create a Pandas series from the list of numbers
series = pd.Series(numbers)
# Grouping by even and odd numbers and calculating the mean
grouped = series.groupby(series % 2 == 0).mean()
# Display the mean of even and odd numbers with labels
grouped.index = ['Even' if is_even else 'Odd' for is_even in grouped.index]
print("Mean of even and odd numbers:")
print(grouped)
Average time
0.014 s
14.17 ms
Maximum time
0.028 s
28.00 ms
3 out of 3 shown test case(s) passed
3 out of 3 hidden test case(s) passed
Test case 1
28 ms
Debug
Expected output
1 2 3 4 5 6 7 8 9 10
Mean of even and odd numbers:
Actual output
1 2 3 4 5 6 7 8 9 10
Mean of even and odd numbers:
Terminal
Test cases
Prev
Reset
Submit
Next

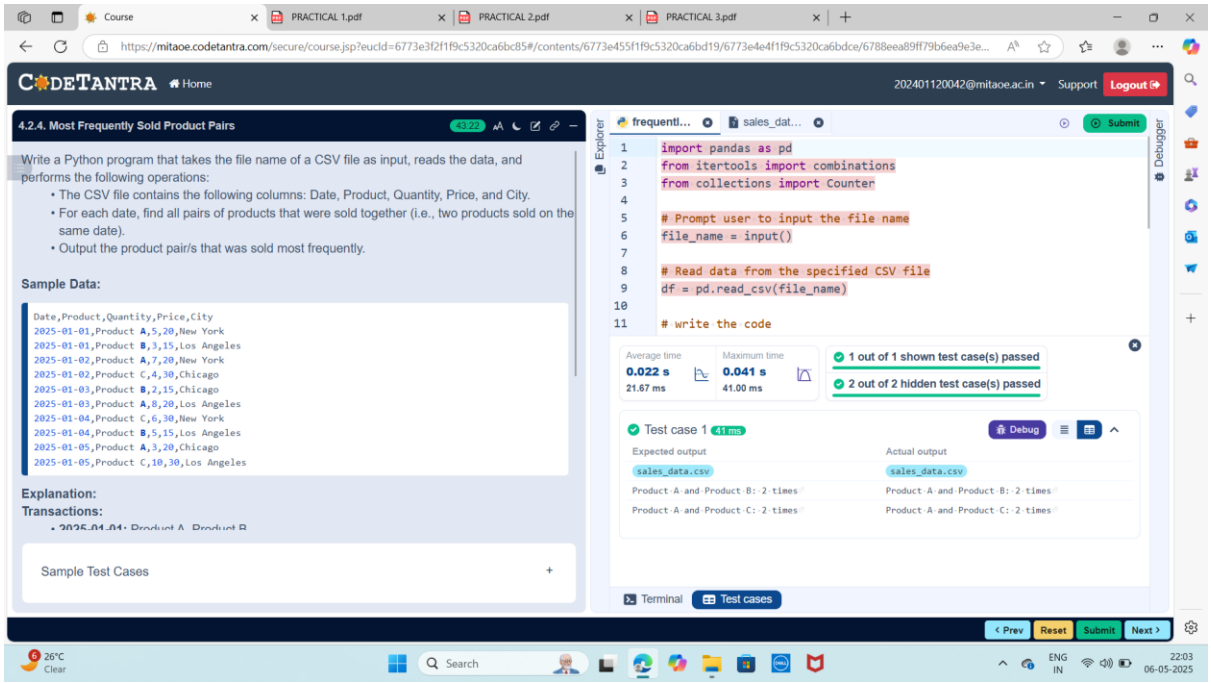

CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.1.2. Dictionary to dataframe
A dictionary of lists has been provided to you in the editor. Create a DataFrame from the dictionary of lists and perform the listed operations, then display the DataFrame before and after each manipulation.
Create 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.
Sample Test Cases
datafram...
Submit
df = df.drop(index=row_to_delete).reset_index(drop=True)
# Display the DataFrame after deleting a row
print("After deleting a row:")
print(df)
# Adding a new column
genders = input("Enter genders separated by space: ").split()
df["Gender"] = genders
# Display the DataFrame after adding a new column
print("After adding a new column:")
Average time
0.179 s
179.50 ms
Maximum time
0.192 s
192.00 ms
1 out of 1 shown test case(s) passed
1 out of 1 hidden test case(s) passed
Test case 1
167 ms
Debug
Expected output
Original DataFrame:
Name Age
0 Alice 25
1 Bob 30
2 Charlie 35
New name: Susan
Actual output
Original DataFrame:
Name Age
0 Alice 25
1 Bob 30
2 Charlie 35
New name: Susan
Terminal
Test cases
Prev
Reset
Submit
Next

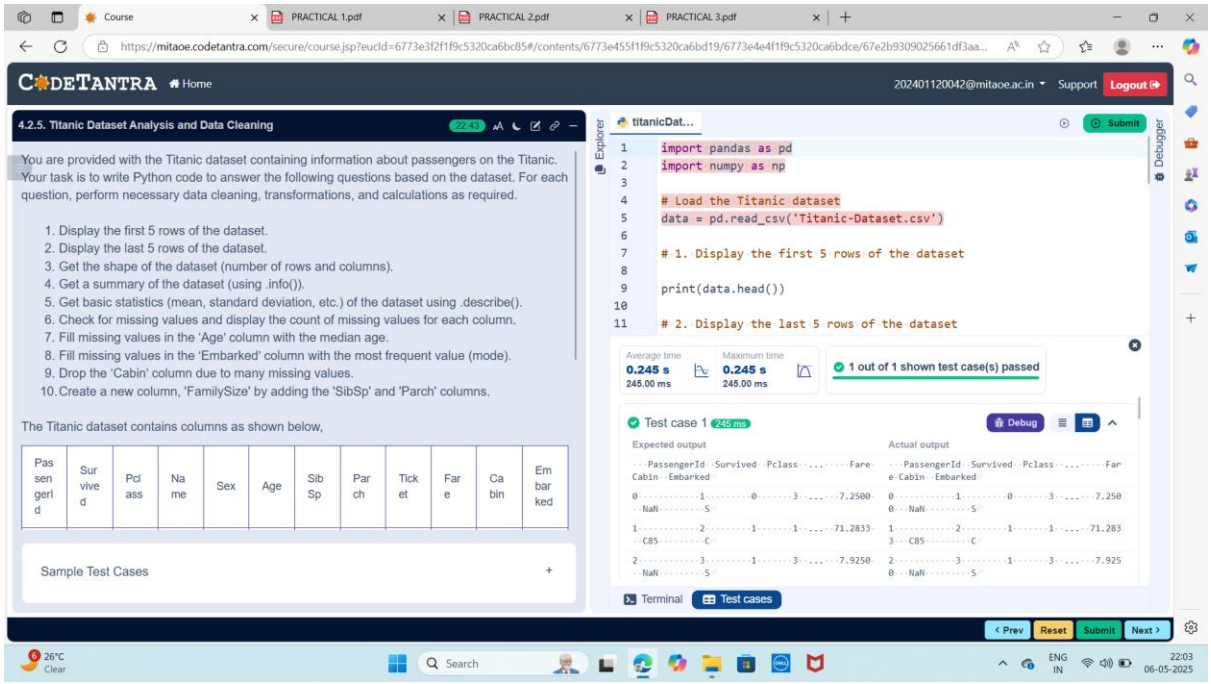

Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.1.3. Student Information
Write a program to read a text file containing student information (name, age, and grade) using Pandas. Perform the following tasks:
• Display the first five rows of the data frame.
• Calculate the average age of the students(limit the average age up to 2 decimal places).
• Filter out the students who have a grade above a certain threshold(consider the threshold grade is 'B').
Note:
Refer to the displayed test cases for better understanding.
Sample Test Cases
import pandas as pd
# Read the text file into a DataFrame
file_name = input()
df = pd.read_csv(file_name, sep="\s+", names=["Name", "Age", "Grade"])
print("First five rows:")
print(df.head())
average_age = round(df["Age"].mean(), 2)
print(f"Average age: {average_age}")
filtered_students = df[df["Grade"] <= "B"]
print("Students with a grade up to B")
print(filtered_students)
Average time
0.041 s
41.00 ms
Maximum time
0.057 s
57.00 ms
1 out of 1 shown test case(s) passed
1 out of 1 hidden test case(s) passed
Test case 1
Debug
Expected output
Actual output
studentdata.txt
First five rows:
Terminal
Test cases
Prev
Reset
Submit
Next

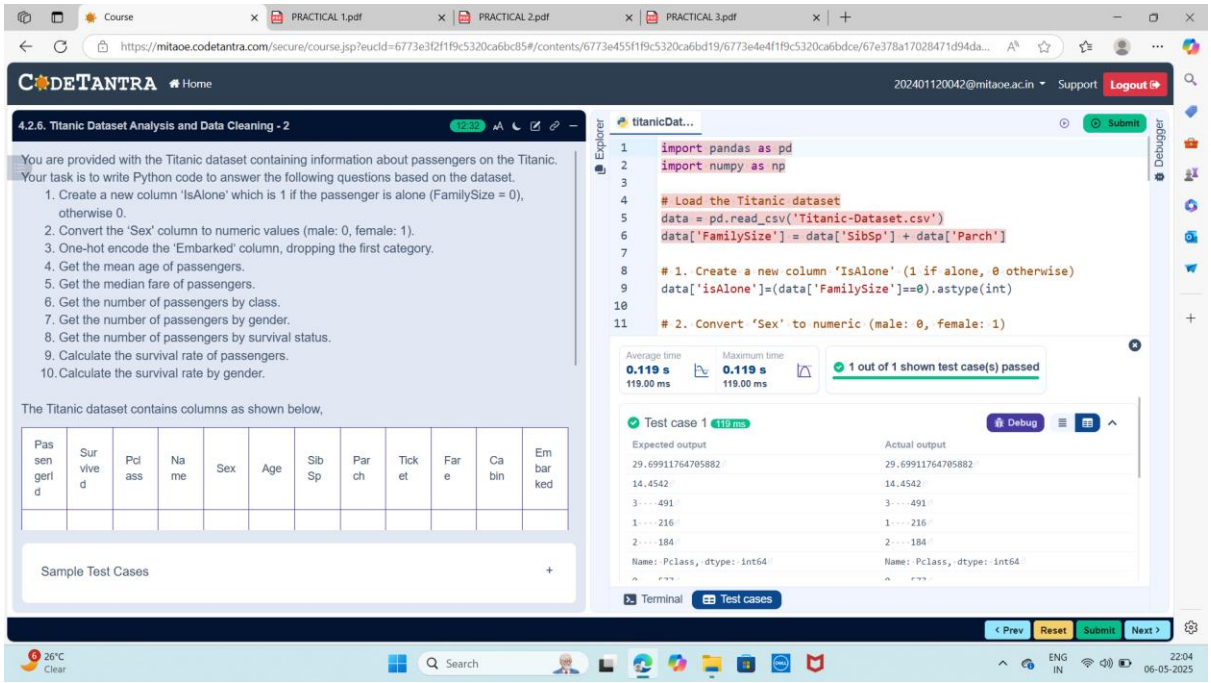

Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.1. Month with the Highest Total Sales
Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:
• The CSV file contains 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,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
Note:
Sample Test Cases
import pandas as pd
# Prompt the user for the file name
file_name = input()
# Load the data
df = pd.read_csv(file_name)
df["Date"] = pd.to_datetime(df["Date"])
df["Month"] = df["Date"].dt.to_period("M")
df["Total Sales"] = df["Quantity"] * df["Price"]
monthly_sales = df.groupby("Month")["Total Sales"].sum()
# Find the month with the highest total sales
best_month =monthly_sales.idxmax()
highest_sales = monthly_sales.max()
print(f"Best month: {best_month}")
print(f"Total sales: ${highest_sales:.2f}")
Average time
0.039 s
39.00 ms
Maximum time
0.075 s
75.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
Debug
Expected output
Actual output
Terminal
Test cases
Prev
Reset
Submit
Next

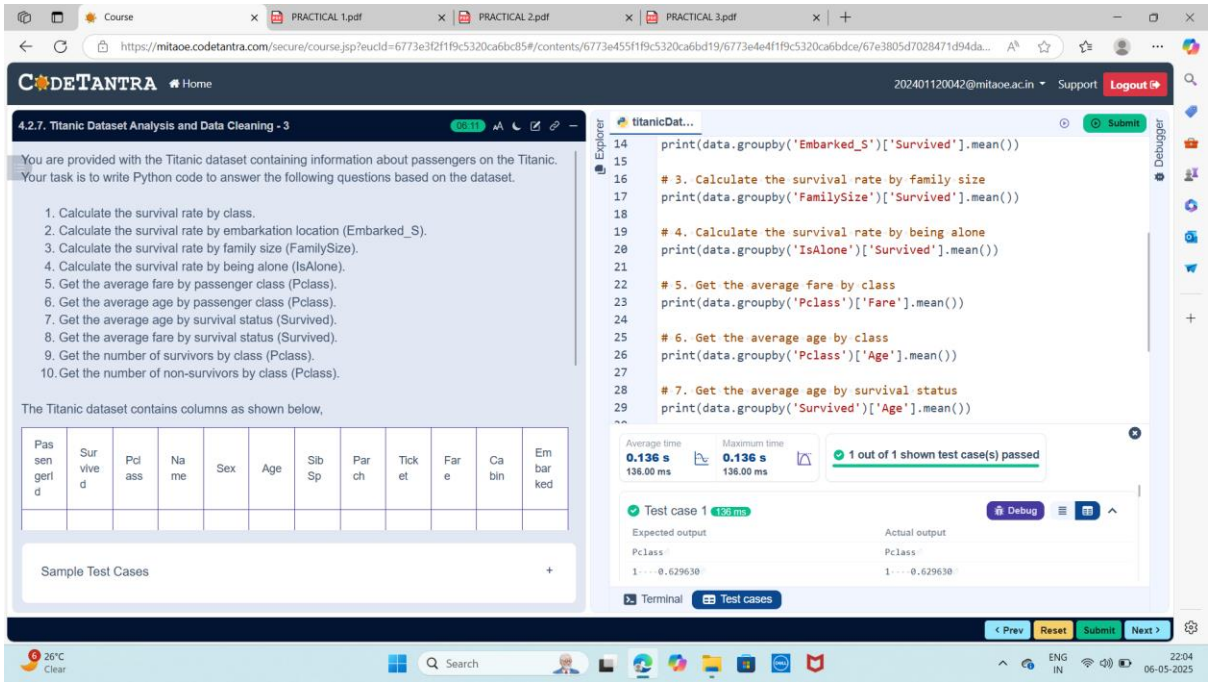

Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.2. Best Selling Product
02:32
Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:
• The CSV file contains the columns: Date, Product, Quantity, Price, and City.
• Find the product that sold the most in terms of quantity sold.
• Display the product that sold the most and the total quantity sold for that product.
Sample Data:
Date,Product,Quantity,Price,City
2025-01-01,Product A,5,20,New York
2025-01-01,Product B,3,15,Los Angeles
2025-01-02,Product A,7,20,New York
2025-01-02,Product C,4,30,Chicago
2025-01-03,Product B,2,15,Chicago
2025-01-03,Product A,8,20,Los Angeles
2025-01-04,Product C,6,30,New York
2025-01-04,Product B,5,15,Los Angeles
2025-01-05,Product A,3,20,Chicago
2025-01-05,Product C,10,30,Los Angeles
Note:
The data cannot be displayed in the file. You can refer to the sample data provided for insights.
Sample Test Cases
monthFor...
sales_dat...
Submit
Explorer
Debugger
1 import pandas as pd
2
3 # Prompt the user for the file name
4 file_name = input()
5
6 # Load the data
7 df = pd.read_csv(file_name)
8
9
10 # Find the product with the highest total quantity sold
11 best_product = df.groupby('Product')['Quantity'].sum().idxmax()
12 highest_quantity = df.groupby('Product')['Quantity'].sum().max()
13
14 # Display the result
15 print(f"Best selling product: {best_product}")
16 print(f"Total quantity sold: {highest_quantity}")
Average time
0.024 s
24.33 ms
Maximum time
0.047 s
47.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
47 ms
Debug
Expected output
Actual output
sales_data.csv
sales_data.csv
Best selling product: Product A
Best selling product: Product A
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
22:02
06-05-2025

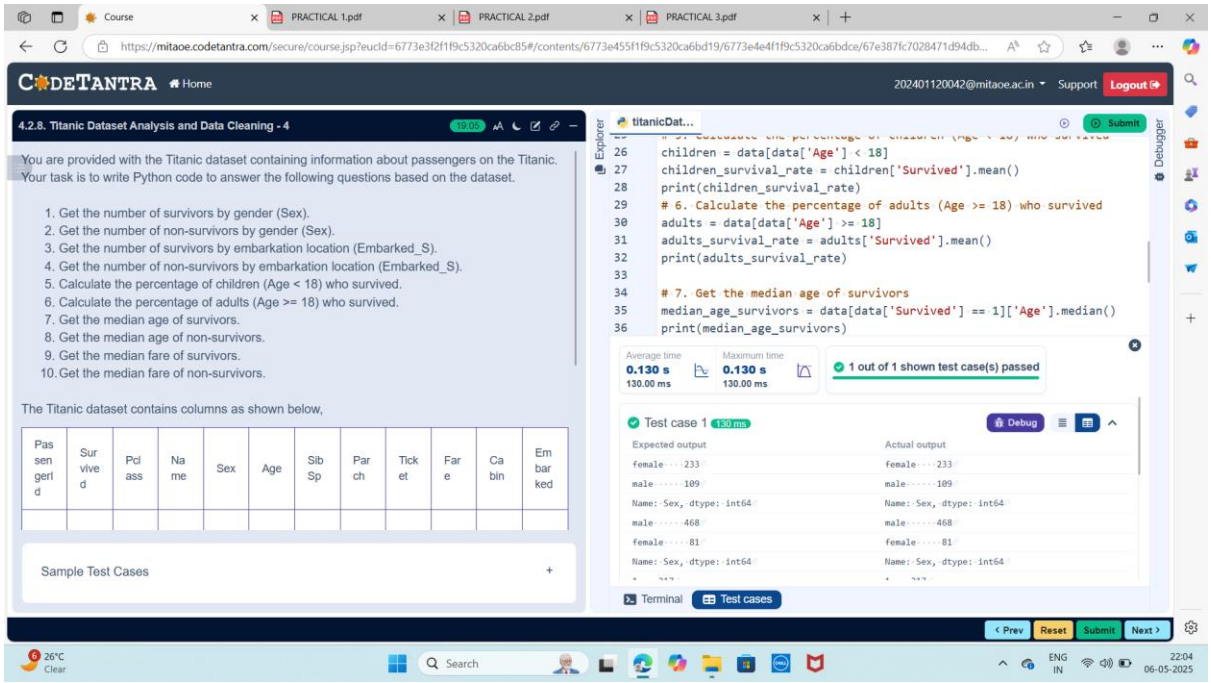

Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.3. City that Sold the Most Products
13:01
Write a Python program that takes the file name of a CSV file as input, reads the data, and performs the following operations:
• The CSV file contains the columns: Date, Product, Quantity, Price, and City.
• Group the data by City and calculate the total quantity of products sold for each city.
• Find the city that sold the most products (based on the total quantity sold).
Sample Data:
Date,Product,Quantity,Price,City
2025-01-01,Product A,5,20,New York
2025-01-01,Product B,3,15,Los Angeles
2025-01-02,Product A,7,20,New York
2025-01-02,Product C,4,30,Chicago
2025-01-03,Product B,2,15,Chicago
2025-01-03,Product A,8,20,Los Angeles
2025-01-04,Product C,6,30,New York
2025-01-04,Product B,5,15,Los Angeles
2025-01-05,Product A,3,20,Chicago
2025-01-05,Product C,10,30,Los Angeles
Note:
The data cannot be displayed in the file. You can refer to the sample data provided for insights.
Sample Test Cases
monthFor...
sales_dat...
Submit
Explorer
Debugger
1 import pandas as pd
2
3 # Prompt the user for the file name
4 file_name = input()
5
6 # Load the data
7 df = pd.read_csv(file_name)
8
9 # write the code..
10 city_sales = df.groupby('City')['Quantity'].sum().reset_index()
11 best_city_row = city_sales.loc[city_sales['Quantity'].idxmax()]
12 best_city = best_city_row['City']
13 # Display the result
14 print(f"City sold the most products: {best_city}")
Average time
0.022 s
21.67 ms
Maximum time
0.045 s
45.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
45 ms
Debug
Expected output
Actual output
sales_data.csv
sales_data.csv
City sold the most products: Los Angeles
City sold the most products: Los Angeles
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
22:03
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
https://mitaoe.codetantra.com/secure/course.jsp?eucId=6773e3f2f1f9c5320ca6bc85#/contents/6773e455f1f9c5320ca6bd19/6773e4e4f1f9c5320ca6bdce/6788eea89ff79b6ea9e3e...
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.4. Most Frequently Sold Product Pairs
43:22
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:
Sample Test Cases
frequentI...
sales_dat...
Submit
Explorer
Debugger
import pandas as pd
from itertools import combinations
from collections import Counter
# Prompt user to input the file name
file_name = input()
# Read data from the specified CSV file
df = pd.read_csv(file_name)
# write the code
Average time
0.022 s
21.67 ms
Maximum time
0.041 s
41.00 ms
1 out of 1 shown test case(s) passed
2 out of 2 hidden test case(s) passed
Test case 1
41 ms
Debug
Expected output
Actual output
sales_data.csv
Product A and Product B: 2 times
Product A and Product C: 2 times
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
22:03
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
https://mitaoe.codetantra.com/secure/course.jsp?eucId=6773e3f2f1f9c5320ca6bc85#/contents/6773e455f1f9c5320ca6bd19/6773e4e4f1f9c5320ca6bdce/67e2b9309025661df3aa...
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.5. Titanic Dataset Analysis and Data Cleaning
22:43
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,
PassengerId | Survived | Pclass | Name | Sex | Age | SibSp | Parch | Ticket | Fare | Cabin | Embarked
Sample Test Cases
titanicDat...
Submit
Explorer
Debugger
import pandas as pd
import numpy as np
# Load the Titanic dataset
data = pd.read_csv('Titanic-Dataset.csv')
# 1. Display the first 5 rows of the dataset
print(data.head())
# 2. Display the last 5 rows of the dataset
Average time
0.245 s
245.00 ms
Maximum time
0.245 s
245.00 ms
1 out of 1 shown test case(s) passed
Test case 1
245 ms
Debug
Expected output
Actual output
PassengerId Survived Pclass ... Fare Cabin Embarked
0 1 0 3 ... 7.2500 NaN S
1 2 1 1 ... 71.2833 C85 C
2 3 1 3 ... 7.9250 NaN S
PassengerId Survived Pclass ... Fare Cabin Embarked
0 1 0 3 ... 7.250 NaN S
1 2 1 1 ... 71.283 C85 C
2 3 1 3 ... 7.925 NaN S
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
22:03
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.6. Titanic Dataset Analysis and Data Cleaning - 2
You are provided with the Titanic dataset containing information about passengers on the Titanic. Your task is to write Python code to answer the following questions based on the dataset.
1. Create a new column 'IsAlone' which is 1 if the passenger is alone (FamilySize = 0), otherwise 0.
2. Convert the 'Sex' column to numeric values (male: 0, female: 1).
3. One-hot encode the 'Embarked' column, dropping the first category.
4. Get the mean age of passengers.
5. Get the median fare of passengers.
6. Get the number of passengers by class.
7. Get the number of passengers by gender.
8. Get the number of passengers by survival status.
9. Calculate the survival rate of passengers.
10. Calculate the survival rate by gender.
The Titanic dataset contains columns as shown below,
PassengerId
Survived
Pclass
Name
Sex
Age
SibSp
Parch
Ticket
Fare
Cabin
Embarked
Sample Test Cases
titanicDat...
Submit
Explorer
Debugger
import pandas as pd
import numpy as np
# Load the Titanic dataset
data = pd.read_csv('Titanic-Dataset.csv')
data['FamilySize'] = data['SibSp'] + data['Parch']
1. Create a new column 'IsAlone' (1 if alone, 0 otherwise)
data['isAlone']=(data['FamilySize']==0).astype(int)
2. Convert 'Sex' to numeric (male: 0, female: 1)
Average time
0.119 s
119.00 ms
Maximum time
0.119 s
119.00 ms
1 out of 1 shown test case(s) passed
Test case 1
119 ms
Debug
Expected output
Actual output
29.69911764705882
14.4542
3 491
1 216
2 184
Name: Pclass, dtype: int64
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
22:04
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.7. Titanic Dataset Analysis and Data Cleaning - 3
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,
PassengerId
Survived
Pclass
Name
Sex
Age
SibSp
Parch
Ticket
Fare
Cabin
Embarked
Sample Test Cases
titanicDat...
Submit
Explorer
Debugger
print(data.groupby('Embarked_S')['Survived'].mean())
# 3. Calculate the survival rate by family size
print(data.groupby('FamilySize')['Survived'].mean())
# 4. Calculate the survival rate by being alone
print(data.groupby('IsAlone')['Survived'].mean())
# 5. Get the average fare by class
print(data.groupby('Pclass')['Fare'].mean())
# 6. Get the average age by class
print(data.groupby('Pclass')['Age'].mean())
# 7. Get the average age by survival status
print(data.groupby('Survived')['Age'].mean())
Average time
0.136 s
136.00 ms
Maximum time
0.136 s
136.00 ms
1 out of 1 shown test case(s) passed
Test case 1
136 ms
Debug
Expected output
Actual output
Pclass
1 0.629630
Terminal
Test cases
Prev
Reset
Submit
Next
26°C
Clear
Search
ENG
IN
22:04
06-05-2025


Course
PRACTICAL 1.pdf
PRACTICAL 2.pdf
PRACTICAL 3.pdf
https://mitaoe.codetantra.com/secure/course.jsp?eucId=6773e3f2f1f9c5320ca6bc85#/contents/6773e455f1f9c5320ca6bd19/6773e4e4f1f9c5320ca6bdce/67e387fc7028471d94db...
CODETANTRA
Home
202401120042@mitaoe.ac.in
Support
Logout
4.2.8. Titanic Dataset Analysis and Data Cleaning - 4
19:05
You are provided with the Titanic dataset containing information about passengers on the Titanic. Your task is to write Python code to answer the following questions based on the dataset.
1. Get the number of survivors by gender (Sex).
2. Get the number of non-survivors by gender (Sex).
3. Get the number of survivors by embarkation location (Embarked_S).
4. Get the number of non-survivors by embarkation location (Embarked_S).
5. Calculate the percentage of children (Age < 18) who survived.
6. Calculate the percentage of adults (Age >= 18) who survived.
7. Get the median age of survivors.
8. Get the median age of non-survivors.
9. Get the median fare of survivors.
10. Get the median fare of non-survivors.
The Titanic dataset contains columns as shown below,
Passengerld
Survived
Pclass
Name
Sex
Age
SibSp
Parch
Ticket
Fare
Cabin
Embarked
Sample Test Cases
titanicDat...
Submit
Explorer
Debugger
children = data[data['Age'] < 18]
children_survival_rate = children['Survived'].mean()
print(children_survival_rate)
# 6. Calculate the percentage of adults (Age >= 18) who survived
adults = data[data['Age'] >= 18]
adults_survival_rate = adults['Survived'].mean()
print(adults_survival_rate)
# 7. Get the median age of survivors
median_age_survivors = data[data['Survived'] == 1]['Age'].median()
print(median_age_survivors)
Average time
0.130 s
130.00 ms
Maximum time
0.130 s
130.00 ms
1 out of 1 shown test case(s) passed
Test case 1
130 ms
Debug
Expected output
Actual output
female 233
male 109
Name: Sex, dtype: int64
male 468
female 81
Terminal
Test cases
‹ Prev
Reset
Submit
Next ›
26°C
Clear
Search
ENG
IN
22:04
06-05-2025