Survived 19:52

plot showing the relationship between the 'Age' and , with points color-coded by survival status. The scatter cifications:

-axis and the **Fare** column for the y-axis.

Survived column: **Red** for passengers who did not for passengers who survived (**Survived = 1**).

e vs. Fare by Survival".

the y-axis as **'Fare'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

AgeFareS...

```
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to nu
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Scatter Pl
    Survived
17  colors = data['Survived'].map({0: 're
18  plt.scatter(data['Age'],data['Fare'],
19  plt.title('Age vs. Fare by Survival')
20  plt.xlabel('Age')
21  plt.ylabel('Fare')
22  plt.show()
23
24
25
```

Terminal Test cases

Science Laboratory - 2304102L - 2304102L

ctrl + k

0 units left

Completed

Pickup from where you left off!

Scatter Plot for Age vs. Fare by Survived

Practical 5 • Lab Assignment

Resume

Practical 1

Unit • 100% completed

Practical 2

Unit • 100% completed

Practical 3

Unit • 100% completed

=6773e3f2f1f9c5320ca6bc85#

12:55

ass of an object (in kilograms) and its velocity (in
and displays the momentum of the object. The
formula:
$= m \times v$

rams).
ers per second).

senting the mass of the object in kilograms.
senting the velocity of the object in meters per second.

omentum with appropriate units (kgm/s) (rounded up to

+

Explorer

calculate...

```
1
2  m=float(input())
3  v=float(input())
4  p=m*v
5  print('%0.2f'%p,end="")
6  print("kgm/s")
```

Terminal Test cases

12:55

ass of an object (in kilograms) and its velocity (in
and displays the momentum of the object. The
formula:
$= m \times v$

rams).
ers per second).

senting the mass of the object in kilograms.
senting the velocity of the object in meters per second.

omentum with appropriate units (kgm/s) (rounded up to

+

Explorer

calculate...

```
1
2  m=float(input())
3  v=float(input())
4  p=m*v
5  print('%0.2f'%p,end="")
6  print("kgm/s")
```

Terminal Test cases

the Number of Digits

19:03

an integer n as input. Depending on the number of

n.

square.
uare root (rounded to two decimal places).
ube root (rounded to two decimal places).

+

Explorer

condition...

```
1  n=int(input())
2  if(0<n<10):
3      print(n*n)
4  elif(9<n<99):
5      print( "%0.2f"%n**0.5)
6  elif(99<n<1000):
7      print("%0.2f"%n**(1/3))
8  else:
9      print("Invalid")
```

Terminal Test cases

the Number of Digits

19:03

an integer n as input. Depending on the number of

n.

square.
uare root (rounded to two decimal places).
ube root (rounded to two decimal places).

+

Explorer

condition...

```
1  n=int(input())
2  if(0<n<10):
3      print(n*n)
4  elif(9<n<99):
5      print( "%0.2f"%n**0.5)
6  elif(99<n<1000):
7      print("%0.2f"%n**(1/3))
8  else:
9      print("Invalid")
```

Terminal Test cases

30:51

h date and salary of employees.

employee in the format $DD - MM - YYYY$.
alary of the employee in rupees.

+

Explorer

birthDate...

```
1  from datetime import datetime
2
3  def calculate_age(birthdate):
4      date_object = datetime.strpt
5      today = datetime.today()
6      if ((today.month, today.day)
   date_object.day)):
7          age = today.year-date_ob
   today.day) < (date_object.month,
8          return age
9      elif((today.month, today.day
   (date_object.month,date_object.
10         age = today.year-date_ob
   today.day) > (date_object.month,
11         return age
12
13
14 def convert_salary_to_dollars(sa
15     salary=salary_in_rupees*0.01
16     return salary
17 birthdate = input()
```

30:51

h date and salary of employees.

employee in the format $DD - MM - YYYY$.
alary of the employee in rupees.

+

Explorer

birthDate...

```
1  from datetime import datetime
2
3  def calculate_age(birthdate):
4      date_object = datetime.strpt
5      today = datetime.today()
6      if ((today.month, today.day)
   date_object.day)):
7          age = today.year-date_ob
   today.day) < (date_object.month,
8          return age
9      elif((today.month, today.day
   (date_object.month,date_object.
10         age = today.year-date_ob
   today.day) > (date_object.month,
11         return age
12
13
14 def convert_salary_to_dollars(sa
15     salary=salary_in_rupees*0.01
16     return salary
17 birthdate = input()
```

15:11

our task is to reverse the digits of the number and print

versed number.

+

Explorer

reverseN...

```
1  n=int(input())
2  i = 0
3  while n != 0:
4      digit = n % 10
5      i = i*10 + digit
6      n = n//10
7  print(i)
```

Terminal Test cases

15:11

our task is to reverse the digits of the number and print

versed number.

+

Explorer

reverseN...

```
1  n=int(input())
2  i = 0
3  while n != 0:
4      digit = n % 10
5      i = i*10 + digit
6      n = n//10
7  print(i)
```

Terminal Test cases

12:09

n integer as input and prints the multiplication table for

eger that represents the number for which the

given number .

+

Explorer

multiplica...

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

Terminal Test cases

12:09

n integer as input and prints the multiplication table for

eger that represents the number for which the

given number .

+

Explorer

multiplica...

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

Terminal Test cases

23:11

s the number of courses and the marks of a student in

he aggregate percentage:
s greater than 75, the grade is Distinction.
s greater than or equal to 60 but less than 75, the

s greater than or equal to 50 but less than 60, the

s greater than or equal to 40 but less than 50, the

ne number of courses.
representing the marks of the student in each of the n

+

Explorer

passorFa...

```
1  n = int(input(""))
2  marks= list(map(int, input().split()))
3  if any(mark<40 for mark in marks):
4      print("Fail")
5  else:
6      totalmarks = sum(marks)
7      Aggregate_percentage=(totalmarks
8      print(f"Aggregate Percentage: {Ag
9      if(Aggregate_percentage>75):
10         print("Grade: Distinction")
11     elif(75>Aggregate_percentage>=60
12         print("Grade: First Division"
13     elif(60>Aggregate_percentage>=50)
14         print("Grade: Second Divisio
15     elif(50>Aggregate_percentage>=40)
16         print("Grade: Third Division"
17     else:
18         print("Fail")
19
```

Terminal Test cases

23:11

s the number of courses and the marks of a student in

he aggregate percentage:
s greater than 75, the grade is Distinction.
s greater than or equal to 60 but less than 75, the

s greater than or equal to 50 but less than 60, the

s greater than or equal to 40 but less than 50, the

ne number of courses.
representing the marks of the student in each of the n

+

Explorer

passorFa...

```
1  n = int(input(""))
2  marks= list(map(int, input().split()))
3  if any(mark<40 for mark in marks):
4      print("Fail")
5  else:
6      totalmarks = sum(marks)
7      Aggregate_percentage=(totalmarks
8      print(f"Aggregate Percentage: {Ag
9      if(Aggregate_percentage>75):
10         print("Grade: Distinction")
11     elif(75>Aggregate_percentage>=60
12         print("Grade: First Division"
13     elif(60>Aggregate_percentage>=50)
14         print("Grade: Second Divisio
15     elif(50>Aggregate_percentage>=40)
16         print("Grade: Third Division"
17     else:
18         print("Fail")
19
```

Terminal Test cases

e Function 02:37

Fibonacci series of a given number of terms using

llow the input and output layout mentioned in the

ass when users' input and output match the expected

+

Explorer

fib.py

```
1  def fib(n):
2      if n <= 0:
3          return 0
4      elif n == 1:
5          return 1
6      else:
7          return fib(n-1) + fib(n-2)
8
9  n=int(input("Enter terms for Fibonacc
10 for i in range (n):
11     print(fib(i),end=" ")
```

Terminal Test cases

e Function 02:37

Fibonacci series of a given number of terms using

llow the input and output layout mentioned in the

ass when users' input and output match the expected

+

Explorer

fib.py

```
1  def fib(n):
2      if n <= 0:
3          return 0
4      elif n == 1:
5          return 1
6      else:
7          return fib(n-1) + fib(n-2)
8
9  n=int(input("Enter terms for Fibonacc
10 for i in range (n):
11     print(fib(i),end=" ")
```

Terminal Test cases

24:19

ttern of asterisks in the form of a right-angled triangle.

the number of rows in the pattern.

n of asterisks (*), with each row containing an

the sample pattern.

+

Explorer

rightangl...

```
1  n=int(input())
2  for i in range(1,n+1):
3      for j in range(0,i):
4          print("*",end=" ")
5      print()
6
```

Terminal Test cases

24:19

ttern of asterisks in the form of a right-angled triangle.

the number of rows in the pattern.

n of asterisks (*), with each row containing an

the sample pattern.

+

Explorer

rightangl...

```
1  n=int(input())
2  for i in range(1,n+1):
3      for j in range(0,i):
4          print("*",end=" ")
5      print()
6
```

Terminal Test cases

10:22

ht-angled triangle pattern of numbers.

the number of rows in the pattern.

n of numbers, with each row containing increasing
w number.

the sample pattern.

+

Explorer

numberP...

```
1  n=int(input())
2  for i in range(1,n+1):
3      for j in range(1,i+1):
4          print(j,end=" ")
5      print()
```

Terminal Test cases

10:22

ht-angled triangle pattern of numbers.

the number of rows in the pattern.

n of numbers, with each row containing increasing
w number.

the sample pattern.

+

Explorer

numberP...

```
1  n=int(input())
2  for i in range(1,n+1):
3      for j in range(1,i+1):
4          print(j,end=" ")
5      print()
```

Terminal Test cases

10:20

ents a menu-driven interface for managing a list of
he following menu options:

npt the user to enter a choice from the menu.
he program should perform the following actions:
er an integer and add it to the integer list. If the input is
valid input".
enter an integer to remove from the list. If the integer
otherwise, display "Element not found". If the list is

list of integers. If the list is empty, display "List is

+

Explorer

listOps.py

```
a=[]
while True:
    print("1. Add")
    print("2. Remove")
    print("3. Display")
    print("4. Quit")
    choice=int(input("Enter choice: "
    if choice==1:
        add=int(input("Integer: "))
        a.append(add)
        print(f"List after adding: {a
    elif choice==2:
        if len(a)==0:
            print("List is empty")
        elif len(a)!=0:
            remove=int(input("Integer
            if remove not in a :
                print("Element not fo
            else:
                a.remove(remove)
                print(f"List after re
```

Terminal Test cases

10:20

ents a menu-driven interface for managing a list of
he following menu options:

npt the user to enter a choice from the menu.
he program should perform the following actions:
er an integer and add it to the integer list. If the input is
valid input".
enter an integer to remove from the list. If the integer
otherwise, display "Element not found". If the list is

list of integers. If the list is empty, display "List is

+

Explorer

listOps.py

```
a=[]
while True:
    print("1. Add")
    print("2. Remove")
    print("3. Display")
    print("4. Quit")
    choice=int(input("Enter choice: "
    if choice==1:
        add=int(input("Integer: "))
        a.append(add)
        print(f"List after adding: {a
    elif choice==2:
        if len(a)==0:
            print("List is empty")
        elif len(a)!=0:
            remove=int(input("Integer
            if remove not in a :
                print("Element not fo
            else:
                a.remove(remove)
                print(f"List after re
```

Terminal Test cases

10:20

ents a menu-driven interface for managing a list of
he following menu options:

npt the user to enter a choice from the menu.
he program should perform the following actions:
er an integer and add it to the integer list. If the input is
valid input".
enter an integer to remove from the list. If the integer
otherwise, display "Element not found". If the list is

list of integers. If the list is empty, display "List is

+

Explorer

listOps.py

```
11          print(f"List after adding: {
12      elif choice==2:
13          if len(a)==0:
14              print("List is empty")
15          elif len(a)!=0:
16              remove=int(input("Intege
17              if remove not in a :
18                  print("Element not f
19              else:
20                  a.remove(remove)
21                  print(f"List after r
22      elif choice==3:
23          if len(a)==0:
24              print("List is empty")
25          else:
26              print(a)
27      elif choice==4:
28          break
29      else:
30          print("Invalid choice")
31
```

Terminal Test cases

10:20

ents a menu-driven interface for managing a list of
he following menu options:

npt the user to enter a choice from the menu.
he program should perform the following actions:
er an integer and add it to the integer list. If the input is
valid input".
enter an integer to remove from the list. If the integer
otherwise, display "Element not found". If the list is

list of integers. If the list is empty, display "List is

+

Explorer

listOps.py

```
11          print(f"List after adding: {
12      elif choice==2:
13          if len(a)==0:
14              print("List is empty")
15          elif len(a)!=0:
16              remove=int(input("Intege
17              if remove not in a :
18                  print("Element not f
19              else:
20                  a.remove(remove)
21                  print(f"List after r
22      elif choice==3:
23          if len(a)==0:
24              print("List is empty")
25          else:
26              print(a)
27      elif choice==4:
28          break
29      else:
30          print("Invalid choice")
31
```

Terminal Test cases

08:55

he following dictionary operations:
nd display it.
s to add, then input key-value pairs.
ing items.
s value. Print "Value updated" if the key exists,
d".
ng a key. If not found, print "Key not found".
If the key doesn't exist, print "Key not found".
key exists, delete and print "Deleted". If not, print "Key

y.

+

Explorer

dictOpera...

```
1   dict = {}
2   print("Empty Dictionary:",dict)
3   n = int(input("Number of items: "))
4   for _ in range (n):
5       key = input("key: ")
6       value = input("value: ")
7       dict[key] = value
8   print("Dictionary:",dict)
9
10  update_key = input("Enter the key to
11  if update_key in dict:
12      new_value = input("Enter the new
13      dict[update_key] = new_value
14      print("Value updated")
15  else:
16      print("Key not found")
17
18  retrieve_key = input("Enter the key
19  if retrieve_key in dict:
20      print(f"Key: {retrieve_key}, Val
21  else:
```

Terminal Test cases

08:55

the following dictionary operations:
nd display it.
s to add, then input key-value pairs.
ling items.
's value. Print "Value updated" if the key exists,
d".
ing a key. If not found, print "Key not found".
If the key doesn't exist, print "Key not found".
e key exists, delete and print "Deleted". If not, print "Key

y.

+

Explorer

dictOpera...

```
19  if retrieve_key in dict:
20      print(f"Key: {retrieve_key}, Valu
21  else:
22      print("Key not found")
23
24  get_key = input("Enter the key to get
    ")
25  value = dict.get(get_key,"Key not fou
26  if value != "Key not found":
27      print(f"Key: {get_key}, Value: {v
28  else:
29      print("value")
30
31  deleted_key = input("Enter the key to
32  if deleted_key in dict:
33      del dict[deleted_key]
34      print("Deleted")
35  else:
36      print("Key not found")
37
38  print("Updated Dictionary:",dict)
```

Terminal Test cases

16:50

e given element is present or not in the array of

the array of integers which are separated by space
the key element to be searched

he index.
int **Not found**.

+

Explorer

CTP1709...

```
1  arr = list(map(int,input().split(" "
2
3  key = int(input())
4  for i in range(len(arr)):
5      if arr[i] == key:
6          print(i)
7          break
8
9  if arr[i] != key:
10     print("Not found")
```

Terminal Test cases

03:37

11 cricket players (in centimeters). Your task is to
e selected as the captain of the team.

ntegers, each representing the height of a player (in
pace.

centimeters) of the tallest player.

+

Explorer

captainof...

```
1  heights = list(map(int,input().split(
2  captain = max(heights)
3  print(captain)
4
```

Terminal Test cases

16:05

ate the usage of ndim, shape and size for a Numpy
NumPy array using the entered elements and display
id numeric values.

ws and columns with space separated values.
rray row-wise followed line by line, separated by

sed on the input dimensions and elements.
of dimensions of the array.
e shape of the array (number of rows, number of

ts in the array.

hape the input array with the specified number of rows

+

Explorer

numpyarr...

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

Terminal Test cases

04:04

trices, **matrix_a**, and **matrix_b**, as input from the arrays.

splay the results of the following matrix operations:
trix_b)
matrix_b)
(matrix_a * matrix_b)
ix_a · matrix_b)

matrix_a, each containing 3 integers separated by

rows for **matrix_b**, each containing 3 integers

+

Explorer

matrixOp...

```
1  import numpy as np
2
3  # Input matrices
4  print("Enter Matrix A:")
5  matrix_a = np.array([list(map(int, in
   range(3))])
6
7  print("Enter Matrix B:")
8  matrix_b = np.array([list(map(int, in
   range(3))])
9
10
11 # Addition
12 print("Addition (A + B):")
13 print(matrix_a + matrix_b)
14 # Subtraction
15 print("Subtraction (A - B):")
16 print(matrix_a - matrix_b)
17 # Multiplication (element-wise)
18 print("Element-wise Multiplication (A
19 print(matrix_a * matrix_b)
```

Terminal Test cases

tacking of Arrays 05:19

arr2. You need to perform horizontal and vertical
NumPy.
he two matrices horizontally (side by side).
two matrices vertically (one below the other).

npt the user to input two 3x3 arrays.
, and each row contains 3 space-separated integers.
rays row by row.

he result of the Horizontal Stack (side-by-side stacking)

olay the result of the Vertical Stack (one below the

+

Explorer

stacking.py

```python
import numpy as np

# Input matrices
print("Enter Array1:")
arr1 = np.array([list(map(int, input(
range(3)])

print("Enter Array2:")
arr2 = np.array([list(map(int, input(
range(3)])

# Perform horizontal stacking (hstack
horizontal_stack = np.hstack((arr1,ar


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

Terminal Test cases

ration 08:31

he following inputs from the user:
of the sequence.
ould end before this value.
tween each number in the sequence.
a sequence using `numpy` based on these inputs and

er values: start, stop, and step, each on a new line.

generated sequence based on the input values.

+

Explorer

customS...

```python
import numpy as np

# Take user input for the start, stop
start = int(input())
stop = int(input())
step = int(input())

# Generate the sequence using np.aran
sequence = np.arange(start, stop, ste
# Print the generated sequence
print(sequence)
```

Terminal Test cases

l Operations, Mathematical Op... 04:21

our task is to complete the function
nvert these lists into NumPy arrays and perform the

um, difference, and product of the two arrays.

and standard deviation of array A.

OR, and bitwise XOR on the arrays (ex: A_i OR B_i).

separated integers representing the elements of array

ace-separated integers representing the elements of

+

Explorer

different...

```python
import numpy as np

def array_operations(A, B):

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

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

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

Terminal Test cases

Operations, Mathematical Op... 04:21

ur task is to complete the function
vert these lists into NumPy arrays and perform the

m, difference, and product of the two arrays.

nd standard deviation of array A.

R, and bitwise XOR on the arrays (ex: A_i OR B_i).

eparated integers representing the elements of array

e-separated integers representing the elements of

+

Explorer

different...

```
19      and_result = A & B
20      or_result = A | B
21      xor_result = A ^ B
22
23      # Output results with one space
24      print("Element-wise Sum:", ' '.j
25      print("Element-wise Difference:"
        diff_result)))
26      print("Element-wise Product:", '
        prod_result)))
27
28      print(f"Mean of A: {mean_A}")
29      print(f"Median of A: {median_A}"
30      print(f"Standard Deviation of A:
31
32      print("Bitwise AND:", ' '.join(m
33      print("Bitwise OR:", ' '.join(ma
34      print("Bitwise XOR:", ' '.join(m
35
36  A = list(map(int, input().split()))
37  B = list(map(int, input().split()))
38  array_operations(A, B)
```

Terminal Test cases

rays 04:45

ers as input and converts it into a NumPy array. Your

inal_array and assigning it to view_array.
inal_array and assigning it to copy_array.

ve how modifying the view affects the
the copy does not.

ted integers.

ew: <original_array>

+

Explorer

copyAnd...

```
1  import numpy as np
2
3  inputlist = list(map(int,input().spli
4
5  # Original array
6  original_array = np.array(inputlist)
7
8  # Create a view
9  view_array =original_array.view()
10
11 # Create a copy
12 copy_array =original_array.copy()
13
14 # Modify the view
15 view_array[0] = 99
16 print("Original array after modifying
17 print("View array:", view_array)
18
19 # Modify the copy
20 copy_array[1] = 88
21 print("Original array after modifying
```

Terminal Test cases

nting, Broadcasting 07:58

single array, `array1`, as space-separated integers as

outs:
o search for in the array.
count its occurrences in the array.
ue to add for broadcasting across the array.

erform the following operations:
e `search_value` appears in `array1` and print these

s `count_value` appears in `array1` and print the

`value` to each element of `array1` using
array.
ng order and print the sorted array.

+

Explorer

arrayOpe...

```python
import numpy as np

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

# Searching
search_value = int(input("Value to se
count_value = int(input("Value to cou
broadcast_value = int(input("Value to

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

Terminal Test cases

tions 28:25

ne file name of a CSV file containing student details, ks in three subjects as input, reads the data, and

splay the complete details of all students, including roll bjects.
ine the total number of students in the dataset.
rs: Extract and print the roll numbers of all students.
act and print the marks of all students in Subject 1.
bject 2: Identify the lowest marks in Subject 2.
bject 3: Identify the highest marks in Subject 3.
play the marks of all students for each subject.
s: Compute the total marks for each student across all

each student: Compute the average marks for each

n subject: Compute the average marks for all students

+

Explorer

Operatio...

```
1   import numpy as np
2
3   a = np.loadtxt("Sample.csv", delimite
4
5   # 1. Print all student details
6   print("All student Details:\n",a)
7
8   # 2. print total students
9   r,c=a.shape
10  print("Total Students:",r)
11
12  # 3. Print all student Roll numbers
13  print("All Student Roll Nos",a[:,0])
14
15  # 4. Print subject 1 marks
16  print("Subject 1 Marks", a[:,1])
17
18  # 5. print minimum marks of Subject
19  print("Min marks in Subject 2",np.mi
20
21  # 6. print maximum marks of Subject
```

Terminal Test cases

tions 28:25

ne file name of a CSV file containing student details, ks in three subjects as input, reads the data, and

play the complete details of all students, including roll bjects.
ne the total number of students in the dataset.
rs: Extract and print the roll numbers of all students.
act and print the marks of all students in Subject 1.
bject 2: Identify the lowest marks in Subject 2.
bject 3: Identify the highest marks in Subject 3.
play the marks of all students for each subject.
s: Compute the total marks for each student across all

each student: Compute the average marks for each

subject: Compute the average marks for all students

+

Explorer

Operatio...

```
19  print( Min marks in Subject 2 ,np.min
20
21  # 6. print maximum marks of Subject 3
22  print("Max marks in Subject 3",np.max
23
24  # 7. Print All subject marks
25  print("All subject marks:", a[:,1:])
26
27  # 8. print Total marks of students
28  print("Total Marks",np.sum(a[:,1:],ax
29
30  # 9. print average marks of each stud
31  avg=np.mean(a[:,1:],axis=1)
32  print(np.round(avg,1))
33  # 10. print average marks of each sub
34  print("Average Marks of each subject"
35
36  # 11. print average marks of S1 and S
37  print("Average Marks of S1 and S2",np
38
39  # 12. print average marks of S1 and S
40  print("Average Marks of S1 and S3" np
```

Terminal Test cases

tions 28:25

he file name of a CSV file containing student details,
rks in three subjects as input, reads the data, and

splay the complete details of all students, including roll
bjects.
ine the total number of students in the dataset.
ers: Extract and print the roll numbers of all students.
act and print the marks of all students in Subject 1.
bject 2: Identify the lowest marks in Subject 2.
ubject 3: Identify the highest marks in Subject 3.
play the marks of all students for each subject.
s: Compute the total marks for each student across all

each student: Compute the average marks for each

n subject: Compute the average marks for all students

+

Explorer

Operatio...

```
38
39  # 12. print average marks of S1 and S
40  print("Average Marks of S1 and S3",np
41
42  # 13. print Roll number who got maxim
43  i=np.argmax(a[:,3])
44  print("Roll no who got maximum marks
45
46  # 14. print Roll number who got minim
47  mn=np.argmin(a[:,2])
48  print("Roll no who got minimum marks
49
50  # 15. print Roll number who got 24 ma
51  whr=np.where(a[:,2]==24)
52  print("Roll no who got 24 marks in Su
53
54  # 16. print count of students who got
55  ct=np.count_nonzero(a[:,1]<40)
56  print("Count of students who got mark
57
58  # 17. print count of students who got
```

Terminal Test cases

tions 28:25

he file name of a CSV file containing student details,
rks in three subjects as input, reads the data, and

splay the complete details of all students, including roll
bjects.
ine the total number of students in the dataset.
ers: Extract and print the roll numbers of all students.
act and print the marks of all students in Subject 1.
bject 2: Identify the lowest marks in Subject 2.
ubject 3: Identify the highest marks in Subject 3.
play the marks of all students for each subject.
ts: Compute the total marks for each student across all

each student: Compute the average marks for each

h subject: Compute the average marks for all students

+

Explorer

Operatio...

```
64
65  # 19. print count of subjects in whic
    >= 90
66  print("Roll no:",a[:,0])
67  print("Count of subjects in which stu
    90:",np.count_nonzero(a[:,1:]>=90,axi
68
69  # 20. Print S1 marks in ascending ord
70  srt=np.sort(a[:,1])
71  print(srt)
72
73  # 21. Print S1 marks >= 50 and <= 90
74  print(a[(a[:,1]>=50)&(a[:,1]<90)])
75  print(a)
76  ip=np.where(a[:,1]==79)
77  print(ip)
78  # 22. Print the index position of mar
79
80
81
82
```

Terminal Test cases

nipulation 33:02

a list of numbers from the user, creates a Pandas series
an of even and odd numbers separately using the

of numbers separated by space when prompted.

the mean of even and odd numbers separately.
displayed with a label indicating whether it corresponds

+

Explorer

seriesMa...

```python
import pandas as pd

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

# Create a Pandas series from the lis
data_list=numbers
series_from_list=pd.Series(data_list)
# Grouping by even and odd numbers an
grouped =series_from_list.groupby(ser
0).mean()

# Display the mean of even and odd nu
grouped.index = ['Even' if is_even el
grouped.index]
print("Mean of even and odd numbers:"
print(grouped)

```

Terminal Test cases

15:18

d to you in the editor. Create a DataFrame from the
ed operations, then display the DataFrame before and

ndas DataFrame.

ne new row data (name, age).
rame.
dding the new row.

ging the age. Take the row index and new age value

nodifying the row.

+

Explorer

datafram...

```python
import pandas as pd

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

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

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

# Adding a new row
new_name=input("New name: ")
new_age=int(input("New age: "))
new_row={'Name': new_name,'Age':new_
df=pd.concat([df,pd.DataFrame([new_r
# Display the DataFrame after adding
```

Terminal Test cases

15:18

ed to you in the editor. Create a DataFrame from the
ted operations, then display the DataFrame before and

ndas DataFrame.

the new row data (name, age).
Frame.
adding the new row.

ging the age. Take the row index and new age value

modifying the row.

+

Explorer

datafram...

```
38
39  # Adding a new column
40  gender_input=input("Enter genders sep
41  genders=gender_input.split()
42  df["Gender"]=genders
43  # Display the DataFrame after adding
44  print("After adding a new column:")
45  print(df)
46
47  # Modifying a column
48  df["Name"]=df["Name"].str.upper()
49  # Display the DataFrame after modifyi
50  print("After modifying a column:")
51  print(df)
52
53  # Deleting a column
54  df=df.drop(columns=['Age'])
55  # Display the DataFrame after deletin
56  print("After deleting a column:")
57  print(df)
58
```

Terminal Test cases

09:19

ontaining student information (name, age, and grade)
g tasks:

he data frame.
the students(limit the average age up to 2 decimal

ave a grade above a certain threshold(consider the

r better understanding.

+

Explorer

studentin...

studentdat...

```
1  import pandas as pd
2
3  # Read the text file into a DataFrame
4  file = input()
5  data = pd.read_csv(file, sep="\s+", h
   ["Name", "Age", "Grade"])
6  print("First five rows:")
7  print(data.head(5))
8  # write your code here..
9  age=round(data['Age'].mean(),2)
10 print("Average age:",age)
11 print("Students with a grade up to B"
12 df=pd.DataFrame(data)
13 a=df[df['Grade']<='B']
14 print(a)
```

Terminal

Test cases

s 17:03

e file name of a CSV file as input, reads the data, and

umns: `Date, Product, Quantity, Price,` and

calculate the total sales for each month.
st total sales and display it.
r the best month.

les

les

les

+

Explorer

monthFor... sales_dat...

```python
import pandas as pd

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

# Load the data
df = pd.read_csv(file_name)
df['sales']=df['Quantity'].multiply(
df['month']=pd.to_datetime(df['Date'
# Find the month with the highest to
best_month =df.groupby('month')['sal
highest_sales = df['sales'].sum()

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

```

Terminal Test cases

03:14

he file name of a CSV file as input, reads the data, and

lumns: `Date, Product, Quantity, Price,` and

e most in terms of quantity sold.
the most and the total quantity sold for that product.

k
eles
k

eles
k
eles

+

Explorer

monthFor...

sales_dat...

```
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 t
11 product_sales = df.groupby("Product")
12 best_product = product_sales.idxmax()
13 highest_quantity = product_sales.max(
14
15 # Display the result
16 print(f"Best selling product: {best_p
17 print(f"Total quantity sold: {highest
18
```

Terminal

Test cases

03:34

ne file name of a CSV file as input, reads the data, and

umns: `Date, Product, Quantity, Price`, and

calculate the total quantity of products sold for each

st products (based on the total quantity sold).

Explorer

monthFor... | sales_dat...

```python
import pandas as pd

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

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

# write the code..
city_sales = df.groupby("City")["Quan
best_city = city_sales.idxmax()
# Display the result
print(f"City sold the most products:
```

Terminal | Test cases

irs 15:25

he file name of a CSV file as input, reads the data, and

owing columns: Date, Product, Quantity, Price, and

f products that were sold together (i.e., two products

t was sold most frequently.

k
eles
k

eles
k
eles

+

Explorer

frequentI... sales_dat...

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

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

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

# write the code
date_products = {}

for date, group in df.groupby('Date')
    products = group['Product'].uniqu
    if len(products) > 1:
        date_products[date] = product

pair_counter = Counter()

for products in date_products.values(
```

Terminal Test cases

irs 15:25

he file name of a CSV file as input, reads the data, and

owing columns: Date, Product, Quantity, Price, and

f products that were sold together (i.e., two products

t was sold most frequently.

k
eles
k

eles
k
eles

+

Explorer

frequentl... sales_dat...

```
15      products = group['Product'].uniqu
16  v   if len(products) > 1:
17          date_products[date] = product
18
19  pair_counter = Counter()
20
21  v for products in date_products.values(
22      pairs = combinations(sorted(produ
23      pair_counter.update(pairs)
24
25  v if pair_counter:
26      max_count = max(pair_counter.valu
27
28  v   for pair, count in pair_counter.i
29  v       if count == max_count:
30              print(f"{pair[0]} and {pa
31  v else:
32      print("No product pairs found.")
33
34
35  # Output the most frequent product pa
```

Terminal Test cases

a Cleaning 10:26

ataset containing information about passengers on the
code to answer the following questions based on the
n necessary data cleaning, transformations, and

e dataset.
e dataset.
(number of rows and columns).
et (using .info()).
tandard deviation, etc.) of the dataset using .describe().
d display the count of missing values for each column.
e' column with the median age.
nbarked' column with the most frequent value (mode).
to many missing values.
ySize' by adding the 'SibSp' and 'Parch' columns.

ns as shown below,

+

Explorer

titanicDat...

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

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

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

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

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

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

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

```

Terminal Test cases

Cleaning 10:26

taset containing information about passengers on the
code to answer the following questions based on the
necessary data cleaning, transformations, and

dataset.
dataset.
(number of rows and columns).
t (using .info()).
andard deviation, etc.) of the dataset using .describe().
display the count of missing values for each column.
' column with the median age.
parked' column with the most frequent value (mode).
to many missing values.
Size' by adding the 'SibSp' and 'Parch' columns.

s as shown below,

+

Explorer

titanicDat...

```
18
19  # 5. Get basic statistics of the dat
20  print(data.describe())
21
22  # 6. Check for missing values
23  print(data.isnull().sum())
24
25  # 7. Fill missing values in the 'Age
    age
26  median_age = data['Age'].median()
27  data['Age'].fillna(median_age, inpla
28  # 8. Fill missing values in the 'Emb
29  mode_embarked = data['Embarked'].mod
30  data['Embarked'].fillna(mode_embarke
31  # 9. Drop the 'Cabin' column due to
32  data.drop('Cabin', axis=1,inplace=Tr
33
34  # 10. Create a new column 'FamilySiz
    'Parch'
35  data['FamilySize'] = data['SibSp'] +
36
```

Terminal Test cases

a Cleaning - 2 30:28

taset containing information about passengers on the
code to answer the following questions based on the

e' which is 1 if the passenger is alone (FamilySize = 0),

numeric values (male: 0, female: 1).
ed' column, dropping the first category.
gers.
engers.
rs by class.
rs by gender.
rs by survival status.
passengers.
gender.

ns as shown below,

Explorer

titanicDat...

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

# Load the Titanic dataset
data = pd.read_csv('Titanic-Dataset.c
data['FamilySize'] = data['SibSp'] +

# 1. Create a new column 'IsAlone' (1
data['IsAlone'] = np.where(data['Fami

# 2. Convert 'Sex' to numeric (male:
data['Sex'] = data['Sex'].map({'male'

# 3. One-hot encode the 'Embarked' co
data = pd.get_dummies(data, columns=[
drop_first=True)

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

Terminal Test cases

a Cleaning - 2 30:28

taset containing information about passengers on the
code to answer the following questions based on the

e' which is 1 if the passenger is alone (FamilySize = 0),

numeric values (male: 0, female: 1).
ed' column, dropping the first category.
gers.
engers.
rs by class.
rs by gender.
rs by survival status.
passengers.
gender.

ns as shown below,

+

Explorer

titanicDat...

```
26  passengers_by_class = data['Pclass'].
27  print(passengers_by_class)
28
29  # 7. Get the number of passengers by
30  passengers_by_gender= data['Sex'].val
31  print(passengers_by_gender)
32
33  # 8. Get the number of passengers by
34  passengers_by_survival =
    data['Survived'].value_counts().sort_
35  print(passengers_by_survival)
36
37  # 9. Calculate the survival rate
38  survival_rate = data['Survived'].mean
39  print(survival_rate)
40
41  # 10. Calculate the survival rate by
42  survival_rate_by_gender = data.groupb
    ['Survived'].mean()
43  print(survival_rate_by_gender)
44
```

Terminal Test cases

Data Cleaning - 3 50:02

dataset containing information about passengers on the
on code to answer the following questions based on the

by class.
by embarkation location (Embarked_S).
by family size (FamilySize).
by being alone (IsAlone).
assenger class (Pclass).
assenger class (Pclass).
rvival status (Survived).
rvival status (Survived).
rs by class (Pclass).
vivors by class (Pclass).

mns as shown below,

Explorer

titanicDat...

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

# Load the Titanic dataset
data = pd.read_csv('Titanic-Dataset.csv
data['FamilySize'] = data['SibSp'] + da
data['IsAlone'] = np.where(data['Family
data = pd.get_dummies(data, columns=['E
drop_first=True)

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

# 2. Calculate the survival rate by emb
print(data.groupby('Embarked_S')['Survi

# 3. Calculate the survival rate by fam
print(data.groupby('FamilySize')['Survi

# 4. Calculate the survival rate by bei
print(data.groupby('IsAlone')['Survived
```

Terminal Test cases

a Cleaning - 3 50:02

taset containing information about passengers on the
code to answer the following questions based on the

class.
embarkation location (Embarked_S).
family size (FamilySize).
being alone (IsAlone).
enger class (Pclass).
enger class (Pclass).
val status (Survived).
val status (Survived).
by class (Pclass).
ors by class (Pclass).

s as shown below,

+

Explorer titanicDat...

```
18
19  # 4. Calculate the survival rate by b
20  print(data.groupby('IsAlone')['Survi
21
22  # 5. Get the average fare by class
23  print(data.groupby('Pclass')['Fare'].
24
25  # 6. Get the average age by class
26  print(data.groupby('Pclass')['Age'].m
27
28  # 7. Get the average age by survival
29  print(data.groupby('Survived')['Age']
30
31  # 8. Get the average fare by survival
32  print(data.groupby('Survived')['Fare'
33
34  # 9. Get the number of survivors by c
35  print(data[data['Survived'] == 1]['Pc
36
37  # 10. Get the number of non-survivors
38  print(data[data['Survived'] == 0]['Pc
```

Terminal Test cases

a Cleaning - 4 34:06

taset containing information about passengers on the
code to answer the following questions based on the

by gender (Sex).
vors by gender (Sex).
by embarkation location (Embarked_S).
vors by embarkation location (Embarked_S).
children (Age < 18) who survived.
adults (Age >= 18) who survived.
ors.
urvivors.
ors.
urvivors.

s as shown below,

+

Explorer

titanicDat...

```
1  import pandas as pd
2  import numpy as np
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.c
6  data = pd.get_dummies(data, columns=[
   drop_first=True)
7
8
9  survivors_by_gender = data[data['Surv
   ['Sex'].value_counts()
10 print(survivors_by_gender)
11
12 # 2. Get the number of non-survivors
13 non_survivors_by_gender = data[data['
   ['Sex'].value_counts()
14 print(non_survivors_by_gender)
15
16 # 3. Get the number of survivors by e
17 survivors_by_embarked_s = data[data['
   ['Embarked_S'].value_counts()
```

Terminal Test cases

a Cleaning - 4 34:06

taset containing information about passengers on the
code to answer the following questions based on the

oy gender (Sex).
ors by gender (Sex).
oy embarkation location (Embarked_S).
ors by embarkation location (Embarked_S).
hildren (Age < 18) who survived.
dults (Age >= 18) who survived.
ors.
urvivors.
ors.
urvivors.

s as shown below,

Explorer titanicDat...

```
16  # 3. Get the number of survivors by e
17  survivors_by_embarked_s = data[data['
    ['Embarked_S'].value_counts()
18  print(survivors_by_embarked_s)
19
20  # 4. Get the number of non-survivors
21  non_survivors_by_embarked_s= data[dat
    ['Embarked_S'].value_counts()
22  print(non_survivors_by_embarked_s)
23
24  # 5. Calculate the percentage of chil
    survived
25  children = data[data['Age'] < 18]
26  children_survival_rate = children['Su
27  print(children_survival_rate)
28
29  # 6. Calculate the percentage of adul
    survived
30  adults = data[data['Age'] >= 18]
31  adults_survival_rate = adults['Surviv
32  print(adults_survival_rate)
33
```

Terminal Test cases

a Cleaning - 4 34:06

taset containing information about passengers on the
code to answer the following questions based on the

by gender (Sex).
vors by gender (Sex).
by embarkation location (Embarked_S).
vors by embarkation location (Embarked_S).
children (Age < 18) who survived.
adults (Age >= 18) who survived.
ors.
urvivors.
ors.
urvivors.

ns as shown below,

+

Explorer titanicDat...

```
31  adults_survival_rate = adults['Surviv
32  print(adults_survival_rate)
33
34  # 7. Get the median age of survivors
35  median_age_survivors = data[data['Sur
    ['Age'].median()
36  print(median_age_survivors)
37
38  # 8. Get the median age of non-surviv
39  median_age_non_survivors = data[data[
    ['Age'].median()
40  print(median_age_non_survivors)
41
42  # 9. Get the median fare of survivors
43  median_fare_survivors = data[data['Su
    ['Fare'].median()
44  print(median_fare_survivors)
45
46  # 10. Get the median fare of non-surv
47  median_fare_non_survivors = data[data
    ['Fare'].median()
```

Terminal Test cases

08:30

ze the temperature variations for three different cities
e months of the year. The temperature data is provided

ing the data.
he y-axis as "Temperature", and provide the title
e plot.
emperature variation for each city throughout the

+

Explorer

stackedpl...

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

# Data for Months and Temperature fo
data = {
    'Month': ['January', 'February',
'June', 'July', 'August', 'September
'December'],
    'City_A_Temperature': [5, 7, 10,
12, 8, 6],
    'City_B_Temperature': [2, 3, 5,
5, 3],
    'City_C_Temperature': [3, 4, 6,
4, 2]
}

# Write your code...
df = pd.DataFrame(data)
plt.stackplot(df['Month'],df['City_A
emperature'],df['City_C_Temperature'
plt.title('Temperature Variation')
```

Terminal Test cases

08:30

ze the temperature variations for three different cities
ne months of the year. The temperature data is provided

sing the data.
the y-axis as "Temperature", and provide the title
ne plot.
temperature variation for each city throughout the

+

Explorer

stackedpl...

```
4   # Data for Months and Temperature for
5   data = {
6       'Month': ['January', 'February',
    'June', 'July', 'August', 'September'
    'December'],
7       'City_A_Temperature': [5, 7, 10,
    12, 8, 6],
8       'City_B_Temperature': [2, 3, 5, 6
    5, 3],
9       'City_C_Temperature': [3, 4, 6, 8
    4, 2]
10  }
11
12  # Write your code...
13  df = pd.DataFrame(data)
14  plt.stackplot(df['Month'],df['City_A_
    emperature'],df['City_C_Temperature']
15  plt.title('Temperature Variation')
16  plt.xlabel('Month')
17  plt.ylabel('Temperature')
18  plt.show()
```

Terminal Test cases

39:05

and visualize data from the Titanic dataset based on

amed `titanic.csv` and has been loaded using the
ving columns:
= First, 2 = Second, 3 = Third).
enger (male/female).

) = Did not survive, 1 = Survived).
passenger.

reate 5 visualizations using Matplotlib. The
a 3x2 grid (3 rows and 2 columns).

+

Explorer

titanicDat...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset from the C
5   df = pd.read_csv('titanic.csv')
6
7   # Set up the figure for 5 subplots
8   fig, axes = plt.subplots(3, 2, figsiz
9
10  # write the code..
11  df = pd.read_csv('titanic.csv')
12
13  fig,axes = plt.subplots(3, 2, figsize
14
15  axes[0, 0].bar(df['Pclass'].value_cou
    df['Pclass'].value_counts(), color='s
16  axes[0, 0].set_title("Passenger Class
17  axes[0, 0].set_xlabel('Pclass')
18  axes[0, 0].set_ylabel('Count')
19
20  axes[0, 1].pie(df['Gender'].value_cou
```

Terminal Test cases

39:05

and visualize data from the Titanic dataset based on

named `titanic.csv` and has been loaded using the
wing columns:
= First, 2 = Second, 3 = Third).
enger (male/female).

0 = Did not survive, 1 = Survived).
e passenger.

create 5 visualizations using Matplotlib. The
n a 3x2 grid (3 rows and 2 columns).

+

Explorer

titanicDat...

```
18  axes[0, 0].set_ylabel('Count')
19
20  axes[0, 1].pie(df['Gender'].value_cou
    labels=df['Gender'].value_counts().in
    colors=['lightblue', 'lightcoral'])
21  axes[0, 1].set_title("Gender Distribu
22
23  axes[1, 0].hist(df["Age"].dropna(), b
    color="lightgreen", edgecolor='black'
24  axes[1, 0].set_title("Age Distributio
25  axes[1, 0].set_xlabel("Age")
26  axes[1, 0].set_ylabel("Frequency")
27
28  axes[1, 1].bar(df["Survived"].value_c
    df['Survived'].value_counts(), color=
    'lightcoral'])
29  axes[1, 1].set_title("Survival Count"
30  axes[1, 1].set_xlabel("Survived (0 =
31  axes[1, 1].set_ylabel("Count")
32
33  axes[2, 0].scatter(df['Age'], df['Far
    edgecolors='black')
```

Terminal Test cases

39:05

and visualize data from the Titanic dataset based on

named `titanic.csv` and has been loaded using the
wing columns:
= First, 2 = Second, 3 = Third).
enger (male/female).

) = Did not survive, 1 = Survived).
passenger.

create 5 visualizations using Matplotlib. The
a 3x2 grid (3 rows and 2 columns).

+

Explorer

titanicDat...

```
23  axes[1, 0].hist(df["Age"].dropna(), b
    color="lightgreen", edgecolor='black'
24  axes[1, 0].set_title("Age Distributio
25  axes[1, 0].set_xlabel("Age")
26  axes[1, 0].set_ylabel("Frequency")
27
28  axes[1, 1].bar(df["Survived"].value_c
    df['Survived'].value_counts(), color=
    'lightcoral'])
29  axes[1, 1].set_title("Survival Count"
30  axes[1, 1].set_xlabel("Survived (0 =
31  axes[1, 1].set_ylabel("Count")
32
33  axes[2, 0].scatter(df['Age'], df['Far
    edgecolors='black')
34  axes[2, 0].set_title("Fare vs Age")
35  axes[2, 0].set_xlabel("Age")
36  axes[2, 0].set_ylabel("Fare")
37
38  plt.tight_layout()
39  plt.show()
```

Terminal Test cases

ion of Titanic 02:25

am for the distribution of the 'Age' column from the
d display the frequency of different age ranges with

n.
s to **black** (k).
the y-axis as **'Frequency'**.
n" to the histogram.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

Histogra...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].media
9   data['Embarked'].fillna(data['Embark
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=T
11
12  # Convert categorical features to nu
13  data['Sex'] = data['Sex'].map({'male
14  data = pd.get_dummies(data, columns=
    drop_first=True)
15
16  # Write your code here for Histogram
17
18  plt.hist(data['Age'], bins=30, edgec
19  plt.xlabel('Age')
```

Terminal Test cases

tion of Titanic 02:25

ram for the distribution of the 'Age' column from the
ld display the frequency of different age ranges with

n.
rs to **black** (k).
the y-axis as **'Frequency'**.
on" to the histogram.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

Histogra...

```
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Histogram
17
18  plt.hist(data['Age'], bins=30, edgeco
19  plt.xlabel('Age')
20  plt.ylabel('Frequency')
21  plt.title('Age Distribution')
22  plt.show()
```

Terminal Test cases

ngers 03:51

hart that shows the count of passengers who survived
taset. The chart should display the following

o show the count of survivors (0 = Did not survive, 1 =

t" to the chart.
d' and the y-axis as '**Count**'.

ns as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BarPlotOf...

```
1  import pandas as pd
2  import matplotlib.pyplot as plt
3
4  # Load the Titanic dataset
5  data = pd.read_csv('Titanic-Dataset.c
6
7  # Data Cleaning
8  data['Age'].fillna(data['Age'].median
9  data['Embarked'].fillna(data['Embarke
   inplace=True)
10 data.drop('Cabin', axis=1, inplace=Tr
11
12 # Convert categorical features to num
13 data['Sex'] = data['Sex'].map({'male'
14 data = pd.get_dummies(data, columns=[
   drop_first=True)
15
16 # Write your code here for Bar Plot f
17
18 survival_counts = data['Survived'].va
19 survival_counts.plot(kind='bar')
```

Terminal Test cases

ngers 03:51

art that shows the count of passengers who survived
aset. The chart should display the following

show the count of survivors (0 = Did not survive, 1 =

" to the chart.
' and the y-axis as '**Count**'.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BarPlotOf...

```
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Bar Plot f
17
18  survival_counts = data['Survived'].va
19  survival_counts.plot(kind='bar')
20  plt.title('Survival Count')
21  plt.xlabel('Survived')
22  plt.ylabel('Count')
23  plt.show()
```

Terminal Test cases

08:23

d bar chart that shows the count of passengers who
d by gender, in the Titanic dataset. The chart should

olumn, then use the **value_counts()** function to count
(0 = Did not survive, 1 = Survived) for each gender.
lisplay the survival counts.
nder" to the chart.
and the y-axis as **'Count'**.
lot Survived' and **'Survived'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BarPlotOf...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Bar Plot f
17
18
19  survival_by_gender = data.groupby('Se
```

Terminal Test cases

08:23

ed bar chart that shows the count of passengers who
ed by gender, in the Titanic dataset. The chart should

column, then use the **value_counts()** function to count
(0 = Did not survive, 1 = Survived) for each gender.
display the survival counts.
nder" to the chart.
and the y-axis as **'Count'**.
Not Survived' and **'Survived'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BarPlotOf...

```
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Bar Plot f
17
18
19  survival_by_gender = data.groupby('Se
    ['Survived'].value_counts().unstack()
20  survival_by_gender.columns = ["Not Su
21  survival_by_gender.index = ['0', '1']
22  survival_by_gender.plot(kind='bar', s
23  plt.title('Survival by Gender')
24  plt.xlabel('Gender')
25  plt.ylabel("Count")
26  plt.legend(title=None)
27  plt.show()
```

Terminal Test cases

06:18

ed bar chart that shows the count of passengers who
ed by passenger class (**Pclass**), in the Titanic dataset.
ng specifications:
s column and count the number of survivors (0 = Did
each class using **value_counts()**.
display the survival counts.
lass" to the chart.
and the y-axis as **'Count'**.
Not Survived' and **'Survived'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BarPlotOf...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Bar Plot f
17
18  survival_by_class = data.groupby('Pcl
    ['Survived'].value_counts().unstack()
```

Terminal Test cases

06:18

ed bar chart that shows the count of passengers who
ed by passenger class (**Pclass**), in the Titanic dataset.
ng specifications:
s column and count the number of survivors (0 = Did
each class using **value_counts()**.
display the survival counts.
lass" to the chart.
and the y-axis as **'Count'**.
Not Survived' and **'Survived'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BarPlotOf...

```
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Bar Plot f
17
18  survival_by_class = data.groupby('Pcl
    ['Survived'].value_counts().unstack()
19  survival_by_class.columns = ['Not Sur
20  survival_by_class.plot(kind='bar', st
21  plt.title('Survival by Pclass')
22  plt.xlabel('Pclass')
23  plt.ylabel('Count')
24  plt.legend(title=None)
25  plt.show()
```

Terminal Test cases

07:58

d bar chart showing the survival count for passengers
in the Titanic dataset.
g specifications:
o determine the embarkation location. After converting
bles (using **pd.get_dummies()**), plot the survival count
olumn (representing passengers who embarked from
rvival.
make it stacked.
barked " to the chart.
d' and the y-axis as **'Count'**.
h between survivors and non-survivors (label the
t Survived').

s as shown below,

Ag	Si	Pa	Tic	Fa	Ca	E mb

+

Explorer

BarPlotOf...

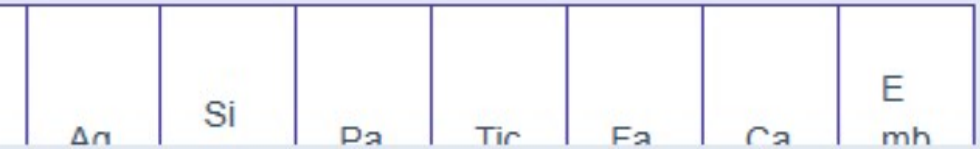
```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Bar Plot f
17
18  grouped = data.groupby('Embarked_Q')
    ['Survived'].value_counts().unstack()
```

Terminal Test cases

d 07:58

d bar chart showing the survival count for passengers
in the Titanic dataset.
ng specifications:
o determine the embarkation location. After converting
bles (using **pd.get_dummies()**), plot the survival count
column (representing passengers who embarked from
rvival.
make it stacked.
nbarked " to the chart.
ed' and the y-axis as **'Count'**.
h between survivors and non-survivors (label the
ot Survived').

s as shown below,



+

Explorer

BarPlotOf...

```
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Bar Plot f
17
18  grouped = data.groupby('Embarked_Q')
    ['Survived'].value_counts().unstack()
19  grouped.columns = ['Not Survived', 'S
20  grouped.plot(kind='bar', stacked=True
21  plt.title('Survival by Embarked')
22  plt.xlabel('Embarked')
23  plt.ylabel('Count')
24  plt.legend(title=None)
25  plt.show()
```

Terminal Test cases

02:34

t that shows the distribution of the 'Age' column from
assenger classes. The boxplot should display the

oup the data for the boxplot.
e by Pclass".
ith **plt.suptitle('')**.
nd the y-axis as **'Age'**.

s as shown below,

	Ag e	Si bS p	Pa rch	Tic ket	Fa re	Ca bin	E mb ark ed

+

Explorer

BoxPlotF...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Box Plot f
17
18  plt.figure(figsize=(8, 6))
19  data.boxplot(column='Age', by='Pclass
```

Terminal Test cases

02:34

t that shows the distribution of the 'Age' column from
assenger classes. The boxplot should display the

up the data for the boxplot.
e by Pclass".
ith **plt.suptitle('')**.
nd the y-axis as **'Age'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BoxPlotF...

```
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].media
9   data['Embarked'].fillna(data['Embark
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=T
11
12  # Convert categorical features to nu
13  data['Sex'] = data['Sex'].map({'male
14  data = pd.get_dummies(data, columns=
    drop_first=True)
15
16  # Write your code here for Box Plot
17
18  plt.figure(figsize=(8, 6))
19  data.boxplot(column='Age', by='Pclas
20  plt.suptitle('')
21  plt.title('Age by Pclass')
22  plt.xlabel('Pclass')
23  plt.ylabel('Age')
24  plt.show()
```

Terminal Test cases

03:04

t that shows the distribution of the 'Age' column from
r passengers survived or not. The boxplot should

group the data for the boxplot (0 = Did not survive, 1 =

e by Survival".
ith **plt.suptitle('')**.
' and the y-axis as **'Age'**.

s as shown below,

Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BoxPlotF...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Box Plot t
17
18  plt.figure(figsize=(8, 6))
19  data.boxplot(column='Age', by='Surviv
```

Terminal Test cases

03:04

ot that shows the distribution of the 'Age' column from
er passengers survived or not. The boxplot should

group the data for the boxplot (0 = Did not survive, 1 =

ge by Survival".
with **plt.suptitle('')**.
d' and the y-axis as **'Age'**.

ns as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BoxPlotF...

```
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Box Plot f
17
18  plt.figure(figsize=(8, 6))
19  data.boxplot(column='Age', by='Surviv
20  plt.suptitle('')
21  plt.title('Age by Survival')
22  plt.xlabel('Survived')
23  plt.ylabel('Age')
24  plt.show()
```

Terminal Test cases

02:42

ot that shows the distribution of the 'Fare' column from
ssenger class (Pclass). The boxplot should display the

oup the data for the boxplot.
re by Pclass".
with **plt.suptitle('')**.
and the y-axis as **'Fare'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BoxPlotF...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].media
9   data['Embarked'].fillna(data['Embark
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=T
11
12  # Convert categorical features to nu
13  data['Sex'] = data['Sex'].map({'male
14  data = pd.get_dummies(data, columns=
    drop_first=True)
15
16  # Write your code here for Box Plot
17
18  plt.figure(figsize=(8,6))
19  data.boxplot(column='Fare', by='Pcla
```

Terminal Test cases

02:42

ot that shows the distribution of the 'Fare' column from
ssenger class (Pclass). The boxplot should display the

oup the data for the boxplot.
re by Pclass".
vith **plt.suptitle('')**.
and the y-axis as **'Fare'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

BoxPlotF...

```
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].media
9   data['Embarked'].fillna(data['Embark
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=T
11
12  # Convert categorical features to nu
13  data['Sex'] = data['Sex'].map({'male
14  data = pd.get_dummies(data, columns=
    drop_first=True)
15
16  # Write your code here for Box Plot
17
18  plt.figure(figsize=(8,6))
19  data.boxplot(column='Fare', by='Pcla
20  plt.suptitle('')
21  plt.title('Fare by Pclass')
22  plt.xlabel('Pclass')
23  plt.ylabel('Fare')
24  plt.show()
```

Terminal Test cases

03:04

plot showing the relationship between the 'Age' and
The scatter plot should display the following

axis and the **Fare** column for the y-axis.
vs. Fare".
the y-axis as **'Fare'**.

as shown below,

Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

AgeFareS...

```
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to nu
13  data['Sex'] = data['Sex'].map({'male
14  data = pd.get_dummies(data, columns=
    drop_first=True)
15
16  # Write your code here for Box Plot
17
18  plt.figure(figsize=(6.4,4.8))
19  plt.scatter(data['Age'],data['Fare']
20  plt.title('Age vs. Fare')
21  plt.xlabel("Age")
22  plt.ylabel('Fare')
23  plt.show()
```

Terminal Test cases

Survived 19:52

plot showing the relationship between the 'Age' and , with points color-coded by survival status. The scatter cifications:

-axis and the **Fare** column for the y-axis.

Survived column: **Red** for passengers who did not for passengers who survived (**Survived = 1**).

e vs. Fare by Survival".

the y-axis as **'Fare'**.

s as shown below,

	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

+

Explorer

AgeFareS...

```
1   import pandas as pd
2   import matplotlib.pyplot as plt
3
4   # Load the Titanic dataset
5   data = pd.read_csv('Titanic-Dataset.c
6
7   # Data Cleaning
8   data['Age'].fillna(data['Age'].median
9   data['Embarked'].fillna(data['Embarke
    inplace=True)
10  data.drop('Cabin', axis=1, inplace=Tr
11
12  # Convert categorical features to num
13  data['Sex'] = data['Sex'].map({'male'
14  data = pd.get_dummies(data, columns=[
    drop_first=True)
15
16  # Write your code here for Scatter Pl
    Survived
17  colors = data['Survived'].map({0: 're
18  plt.scatter(data['Age'],data['Fare'],
```

Terminal Test cases