**PYTHON PART B PROGRAMS (1-3)**

**SOLUTIONS**

1. import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from scipy import stats

# Function to read data from CSV or Excel file

def read\_data(file\_path):

if file\_path.endswith('.csv'):

return pd.read\_csv(file\_path)

elif file\_path.endswith('.xlsx'):

return pd.read\_excel(file\_path)

else:

raise ValueError("Unsupported file format. Use CSV or Excel.")

# Function to scatter plot all points

def scatter\_plot(data, x\_label, y\_label):

plt.scatter(data[x\_label], data[y\_label])

plt.xlabel(x\_label)

plt.ylabel(y\_label)

plt.title(f'Scatter Plot of {x\_label} vs {y\_label}')

plt.show()

# Function to calculate mean

def calculate\_mean(data):

return np.mean(data)

# Function to calculate median

def calculate\_median(data):

return np.median(data)

# Function to calculate standard deviation

def calculate\_std\_dev(data):

return np.std(data)

# Function to calculate variance

def calculate\_variance(data):

return np.var(data)

# Function to calculate slope and intercept for regression line

def calculate\_regression\_line(data, x\_label, y\_label):

slope, intercept, r\_value, p\_value, std\_err = stats.linregress(data[x\_label], data[y\_label])

return slope, intercept

# Function to draw regression line

def draw\_regression\_line(data, x\_label, y\_label):

slope, intercept = calculate\_regression\_line(data, x\_label, y\_label)

plt.scatter(data[x\_label], data[y\_label])

plt.plot(data[x\_label], slope \* data[x\_label] + intercept, color='red', label='Regression Line')

plt.xlabel(x\_label)

plt.ylabel(y\_label)

plt.title(f'Regression Line of {x\_label} vs {y\_label}')

plt.legend()

plt.show()

# Example usage

file\_path = 'your\_data\_file.csv' # Replace with your actual file path

x\_label = 'Label\_X'

y\_label = 'Label\_Y'

data = read\_data(file\_path)

scatter\_plot(data, x\_label, y\_label)

mean\_value = calculate\_mean(data[y\_label])

median\_value = calculate\_median(data[y\_label])

std\_dev\_value = calculate\_std\_dev(data[y\_label])

variance\_value = calculate\_variance(data[y\_label])

print(f'Mean: {mean\_value}')

print(f'Median: {median\_value}')

print(f'Standard Deviation: {std\_dev\_value}')

print(f'Variance: {variance\_value}')

draw\_regression\_line(data, x\_label, y\_label)

2. import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Reading the database and displaying the top 10 rows

tips\_data = pd.read\_csv('tips.csv') # Replace with your actual file path

print("Top 10 rows of the dataset:")

print(tips\_data.head(10))

# Scatter Plot (day vs tip)

sns.scatterplot(x='day', y='tip', data=tips\_data)

plt.title('Scatter Plot of Day vs Tip')

plt.show()

# Line Chart (day against tip)

sns.lineplot(x='day', y='tip', data=tips\_data, ci=None)

plt.title('Line Chart of Day vs Tip')

plt.show()

# Bar chart with day against tip

sns.barplot(x='day', y='tip', data=tips\_data, ci=None)

plt.title('Bar Chart of Day vs Tip')

plt.show()

# Histogram of total\_bills

plt.hist(tips\_data['total\_bill'], bins=20, color='skyblue', edgecolor='black')

plt.title('Histogram of Total Bills')

plt.xlabel('Total Bill')

plt.ylabel('Frequency')

plt.show()

3. import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Read data from the result file (replace 'result.csv' with your actual file path)

result\_data = pd.read\_csv('result.csv') # or pd.read\_excel('result.xlsx')

# Display the top 10 rows of the dataset

print("Top 10 rows of the dataset:")

print(result\_data.head(10))

# Count the number of pass and fail for each subject and overall result

subjects = result\_data.columns[1:-2] # Assuming columns from 2nd to second-to-last are subjects

pass\_fail\_counts = {'Overall': {'Pass': 0, 'Fail': 0}}

for subject in subjects:

pass\_fail\_counts[subject] = {'Pass': result\_data[result\_data[subject] >= 40][subject].count(),

'Fail': result\_data[result\_data[subject] < 40][subject].count()}

# Update overall pass/fail counts

pass\_fail\_counts['Overall']['Pass'] += pass\_fail\_counts[subject]['Pass']

pass\_fail\_counts['Overall']['Fail'] += pass\_fail\_counts[subject]['Fail']

# Display pass/fail counts

print("\nPass/Fail Counts:")

print(pd.DataFrame(pass\_fail\_counts))

# Visualize the data

# Scatter Plot of Subject1 vs Subject2

sns.scatterplot(x='Subject1', y='Subject2', hue='Result', data=result\_data)

plt.title('Scatter Plot of Subject1 vs Subject2')

plt.show()

# Line Chart of Subject-wise Pass/Fail Counts

pass\_fail\_df = pd.DataFrame(pass\_fail\_counts).T

pass\_fail\_df.plot(kind='bar', stacked=True)

plt.title('Pass/Fail Counts by Subject')

plt.xlabel('Subject')

plt.ylabel('Count')

plt.show()

# Bar Chart of Overall Pass/Fail Counts

overall\_pass\_fail = pd.DataFrame(pass\_fail\_counts['Overall'], index=['Overall'])

overall\_pass\_fail.plot(kind='bar', stacked=True)

plt.title('Overall Pass/Fail Counts')

plt.xlabel('Overall')

plt.ylabel('Count')

plt.show()

# Histogram of Total Marks

plt.hist(result\_data['Total'], bins=20, color='skyblue', edgecolor='black')

plt.title('Histogram of Total Marks')

plt.xlabel('Total Marks')

plt.ylabel('Frequency')

plt.show()