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()
```