**SET A**

1. **Write a Python program to create a dataframe containing columns name, age and percentage. Add 10 rows to the dataframe. View the dataframe.**

**import pandas as pd**

**# Step 1: Create a DataFrame with columns: name, age, percentage**

**data = {**

**'name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva', 'Frank', 'Grace', 'Helen', 'Ian', 'Julia'],**

**'age': [21, 22, 20, 23, 21, 24, 22, 21, 23, 20],**

**'percentage': [85.5, 78.0, 92.0, 69.5, 88.0, 75.0, 82.5, 90.0, 73.0, 80.0]**

**}**

**df = pd.DataFrame(data)**

**# Display the DataFrame**

**print("Initial DataFrame:")**

**print(df)**

1. **Write a Python program to print the shape, number of rows-columns, data types, feature names and the description of the data**

**import pandas as pd**

**# Step 1: Create a DataFrame with columns: name, age, percentage**

**data = {**

**'name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva', 'Frank', 'Grace', 'Helen', 'Ian', 'Julia'],**

**'age': [21, 22, 20, 23, 21, 24, 22, 21, 23, 20],**

**'percentage': [85.5, 78.0, 92.0, 69.5, 88.0, 75.0, 82.5, 90.0, 73.0, 80.0]**

**}**

**df = pd.DataFrame(data)**

**print("\nShape of DataFrame (rows, columns):", df.shape)**

**print("\nNumber of rows:", df.shape[0])**

**print("Number of columns:", df.shape[1])**

**# Print data types of each column**

**print("\nData types:\n", df.dtypes)**

**# Print column names**

**print("\nFeature/Column names:", df.columns.tolist())**

**# Description of data (includes count, mean, std, min, max, etc. for numeric columns)**

**print("\nData Description:\n", df.describe())**

1. **Write a Python program to view basic statistical details of the data.**

**import pandas as pd**

**# Step 1: Create a DataFrame with columns: name, age, percentage**

**data = {**

**'name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva', 'Frank', 'Grace', 'Helen', 'Ian', 'Julia'],**

**'age': [21, 22, 20, 23, 21, 24, 22, 21, 23, 20],**

**'percentage': [85.5, 78.0, 92.0, 69.5, 88.0, 75.0, 82.5, 90.0, 73.0, 80.0]**

**}**

**df = pd.DataFrame(data)**

**# Using describe to get statistics**

**print("\nBasic Statistical Details:")**

**print(df.describe(include='all'))**

1. **Write a Python program to Add 5 rows with duplicate values and missing values. Add a column ‘remarks’ with empty values. Display the data**

**import numpy as np**

**# Create 5 new rows with some duplicates and missing values**

**new\_data = pd.DataFrame({**

**'name': ['Alice', 'Zara', 'Bob', np.nan, 'Mira'],**

**'age': [21, 23, np.nan, 22, 25],**

**'percentage': [85.5, np.nan, 78.0, 70.0, np.nan]**

**})**

**# Append the new data to the existing DataFrame**

**df = pd.concat([df, new\_data], ignore\_index=True)**

**# Add a new column 'remarks' with empty values**

**df['remarks'] = np.nan**

**# Display the updated DataFrame**

**print("\nDataFrame with duplicates, missing values and 'remarks' column:")**

**print(df)**

1. **Write a Python program to get the number of observations, missing values and duplicate values.**

**import numpy as np**

**# Create 5 new rows with some duplicates and missing values**

**new\_data = pd.DataFrame({**

**'name': ['Alice', 'Zara', 'Bob', np.nan, 'Mira'],**

**'age': [21, 23, np.nan, 22, 25],**

**'percentage': [85.5, np.nan, 78.0, 70.0, np.nan]**

**})**

**# Append the new data to the existing DataFrame**

**df = pd.concat([df, new\_data], ignore\_index=True)**

**# Add a new column 'remarks' with empty values**

**df['remarks'] = np.nan**

**# Total observations (rows) and total data points**

**print("\nNumber of Observations (rows):", df.shape[0])**

**print("Total Data Points (cells):", df.size)**

**# Count total missing (null) values**

**print("\nTotal Missing Values:")**

**print(df.isnull().sum())**

**# Count duplicate rows**

**duplicate\_count = df.duplicated().sum()**

**print("\nNumber of Duplicate Rows:", duplicate\_count)**

1. **Write a Python program to drop ‘remarks’ column from the dataframe. Also drop all null and empty values. Print the modified data**

# Drop the 'remarks' column

df\_cleaned = df.drop(columns=['remarks'])

# Drop rows with any null values

df\_cleaned = df\_cleaned.dropna()

# Display cleaned DataFrame

print("\nModified DataFrame after removing 'remarks' column and null rows:")

print(df\_cleaned)

7. Write a Python program to generate a line plot of name vs percentage

import matplotlib.pyplot as plt

# To avoid issues with duplicate or missing names, drop nulls first

df\_plot = df[['name', 'percentage']].dropna()

# Line Plot

plt.figure(figsize=(10, 5))

plt.plot(df\_plot['name'], df\_plot['percentage'], marker='o', linestyle='-', linewidth=2)

plt.title('Line Plot: Name vs Percentage')

plt.xlabel('Name')

plt.ylabel('Percentage')

plt.xticks(rotation=45)

plt.grid(True)

plt.tight\_layout()

plt.show()

8. Write a Python program to generate a scatter plot of name vs percentage

# Scatter Plot

plt.figure(figsize=(10, 5))

plt.scatter(df\_plot['name'], df\_plot['percentage'], color='red', s=100)

plt.title('Scatter Plot: Name vs Percentage')

plt.xlabel('Name')

plt.ylabel('Percentage')

plt.xticks(rotation=45)

plt.grid(True)

plt.tight\_layout()

plt.show()

**SET B**

**Download the heights and weights dataset and load the dataset from a given csv file into a dataframe. Print the first, last 10 rows and random 20 rows. (https://www.kaggle.com/burnoutminer/heightsand-weights-dataset)**

**2. Write a Python program to find the shape, size, datatypes of the dataframe object.**

**import pandas as pd**

**# Load the CSV file (adjust filename as needed)**

**df = pd.read\_csv("height\_weight.csv")**

**# Display first 10 rows**

**print("First 10 rows:")**

**print(df.head(10))**

**# Display last 10 rows**

**print("\nLast 10 rows:")**

**print(df.tail(10))**

**# Display 20 random rows**

**print("\nRandom 20 rows:")**

**print(df.sample(20))**

**3. Write a Python program to view basic statistical details of the data.**

**print("\nShape (rows, columns):", df.shape)**

**print("Total size (cells):", df.size)**

**print("\nData types of each column:")**

**print(df.dtypes)**

**4. Write a Python program to get the number of observations, missing values and nan values.**

**# Number of rows (observations)**

**print("\nTotal Observations:", df.shape[0])**

**# Missing values per column**

**print("\nMissing Values in Each Column:")**

**print(df.isnull().sum())**

**# Total NaN values in DataFrame**

**print("\nTotal NaN Values:", df.isna().sum().sum())**

**5. Write a Python program to add a column to the dataframe “BMI” which is calculated as : weight/height2**

**# Assuming height is in centimeters → Convert to meters**

**df['Height\_m'] = df['Height'] / 100**

**# BMI Formula: weight / (height in meters)^2**

**df['BMI'] = df['Weight'] / (df['Height\_m'] \*\* 2)**

**# Display updated DataFrame with BMI**

**print("\nDataFrame with BMI column added:")**

**print(df.head())**

**6. Write a Python program to find the maximum and minimum BMI.**

**max\_bmi = df['BMI'].max()**

**min\_bmi = df['BMI'].min()**

**print("\nMaximum BMI:", max\_bmi)**

**print("Minimum BMI:", min\_bmi)**

**7. Write a Python program to generate a scatter plot of height vs weight.**

**import matplotlib.pyplot as plt**

**plt.figure(figsize=(8, 5))**

**plt.scatter(df['Height'], df['Weight'], color='blue', alpha=0.6)**

**plt.title('Scatter Plot: Height vs Weight')**

**plt.xlabel('Height (cm)')**

**plt.ylabel('Weight (kg)')**

**plt.grid(True)**

**plt.tight\_layout()**

**plt.show()**