Advanced Python – Practical Programs (Pandas)

Course: MCA Semester-III
Subject: Advanced Python (3CS2010308T)
Topic: Pandas

Program 1: Create and Access Series with Labels

Objective:
Create a Pandas Series from a list of integers with custom labels.
Access elements using both position and labels.

Program Code:

```
import pandas as pd
data = [100, 200, 300, 400]
labels = ['a', 'b', 'c', 'd']
series = pd.Series(data, index=labels)
print("Series:\n", series)
print("Element at label 'b':", series['b'])
print("Element at index 2:", series[2])
```

Expected Output:

```
Series:
a   100
b   200
c   300
d   400
dtype: int64
Element at label 'b': 200
Element at index 2: 300
```

Program 2: Create and Analyze a DataFrame

Objective:
Create a Pandas DataFrame using a dictionary.
Display column names, shape, and data types.

Program Code:

```
import pandas as pd
data = {
   'Name': ['Alice', 'Bob', 'Charlie'],
   'Age': [25, 30, 35],
   'City': ['New York', 'Los Angeles', 'Chicago']
}
df = pd.DataFrame(data)
print("DataFrame:\n", df)
print("Column Names:", df.columns)
print("Shape:", df.shape)
print("Data Types:\n", df.dtypes)
```

Expected Output:

```
DataFrame:
     Name  Age        City
0   Alice   25     New York
1     Bob   30  Los Angeles
2 Charlie   35     Chicago
Column Names: Index(['Name', 'Age', 'City'], dtype='object')
Shape: (3, 3)
Data Types:
Name    object
Age      int64
City    object
dtype: object
```

Program 3: Read CSV and Perform Basic Analysis

Objective:
Read a CSV file into a Pandas DataFrame and perform analysis:
- Display first 5 rows
- Display summary statistics
- Check for missing values

Program Code:

```
import pandas as pd
# Assume 'students.csv' exists with Name, Age, Score columns
df = pd.read_csv('students.csv')
print("First 5 Rows:\n", df.head())
```

```
print("Summary:\n", df.describe())
print("Missing Values:\n", df.isnull().sum())
```

Expected Output:

```
First 5 Rows:
   Name  Age  Score
0  John   20     85
1  Emma   21     90
2  Alex   19     78
3  Mia    22     88
4  Ryan   20     92
Summary:
             Age      Score
count   5.000000   5.000000
mean   20.400000  86.600000
std     1.140175   5.128352
min    19.000000  78.000000
max    22.000000  92.000000
Missing Values:
Name     0
Age      0
Score    0
dtype: int64
```

Program 4: Read JSON Data into DataFrame

Objective:
Read JSON data from a string and convert it into a Pandas DataFrame.

Program Code:

```
import pandas as pd
import json

json_data = '[{"Name": "Alice", "Age": 25}, {"Name": "Bob", "Age": 30}]'
df = pd.read_json(json_data)
print("DataFrame from JSON:\n", df)
```

Expected Output:

```
DataFrame from JSON:
    Name  Age
```

```
0  Alice   25
1    Bob   30
```

Program 5: DataFrame Filtering and Grouping

Objective:
Filter rows based on condition and perform group-wise operation.
Group by a column and calculate average age.

Program Code:

```
import pandas as pd
data = {
   'Name': ['Anna', 'Ben', 'Cara', 'David', 'Eva'],
   'Department': ['HR', 'IT', 'HR', 'IT', 'HR'],
   'Age': [29, 35, 26, 30, 28]
}
df = pd.DataFrame(data)
filtered = df[df['Age'] > 28]
grouped = df.groupby('Department')['Age'].mean()
print("Filtered Rows (Age > 28):\n", filtered)
print("Average Age by Department:\n", grouped)
```

Expected Output:

```
Filtered Rows (Age > 28):
   Name Department  Age
0  Anna        HR   29
1   Ben        IT   35
3 David        IT   30
Average Age by Department:
Department
HR    27.666667
IT    32.5
Name: Age, dtype: float64
```