NAME : Nevetha S K
EMAIL ID : nevethasenthilkumar@gmail.com

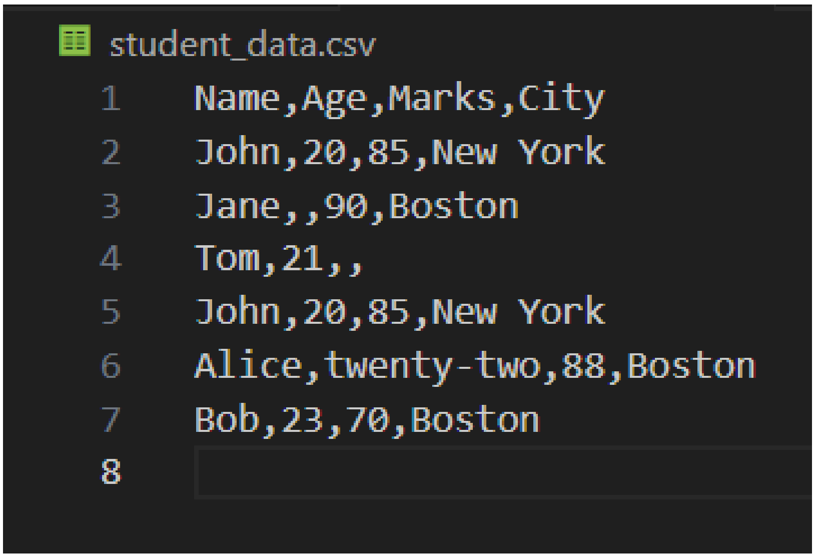
PYTHON CODING ASSESSMENT

DATA CLEANING IN PYTHON USING PANDAS :

Data cleaning is a critical step in the data preprocessing phase. It involves handling:

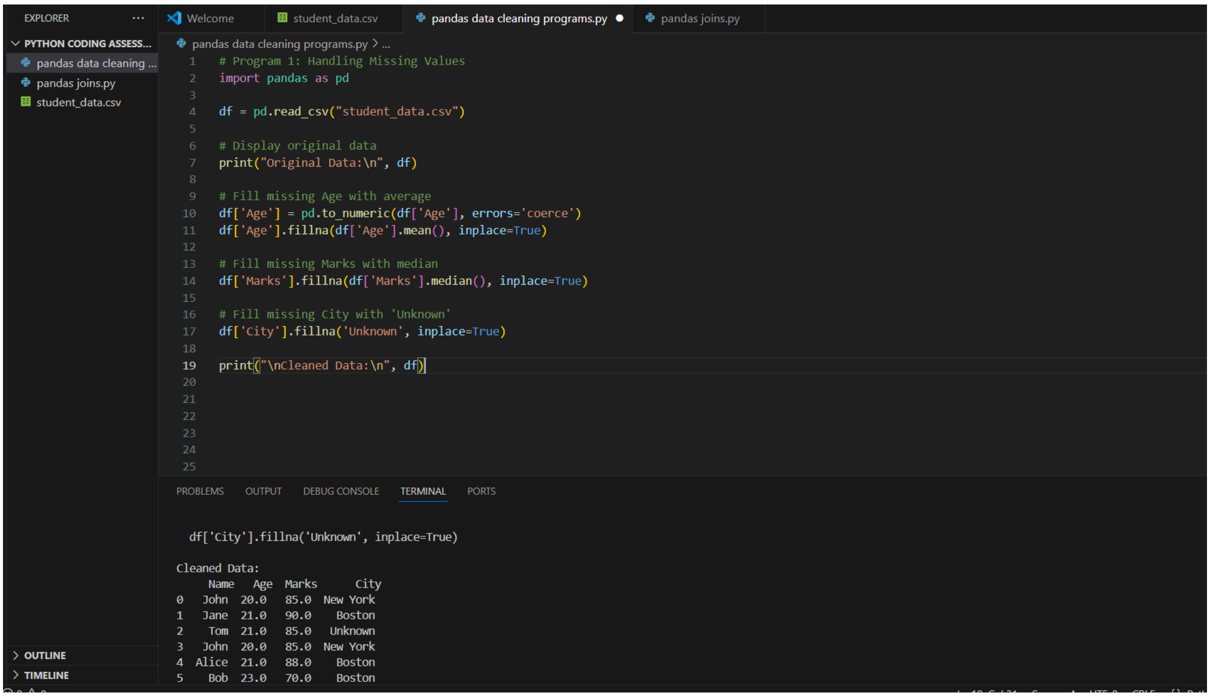
- Missing values
- Duplicates
- Incorrect data types
- Outliers
- Inconsistent formatting

Example Dataset : student_data.csv



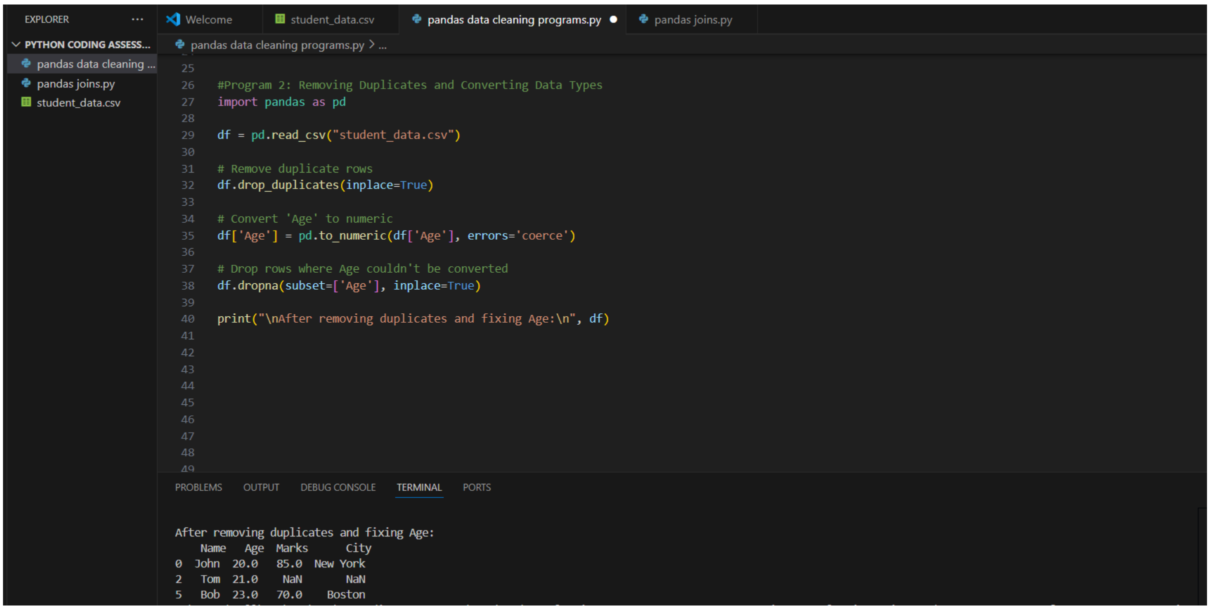
```
Name,Age,Marks,City
John,20,85,New York
Jane,,90,Boston
Tom,21,,
John,20,85,New York
Alice,twenty-two,88,Boston
Bob,23,70,Boston

```

Data Cleaning Programs :



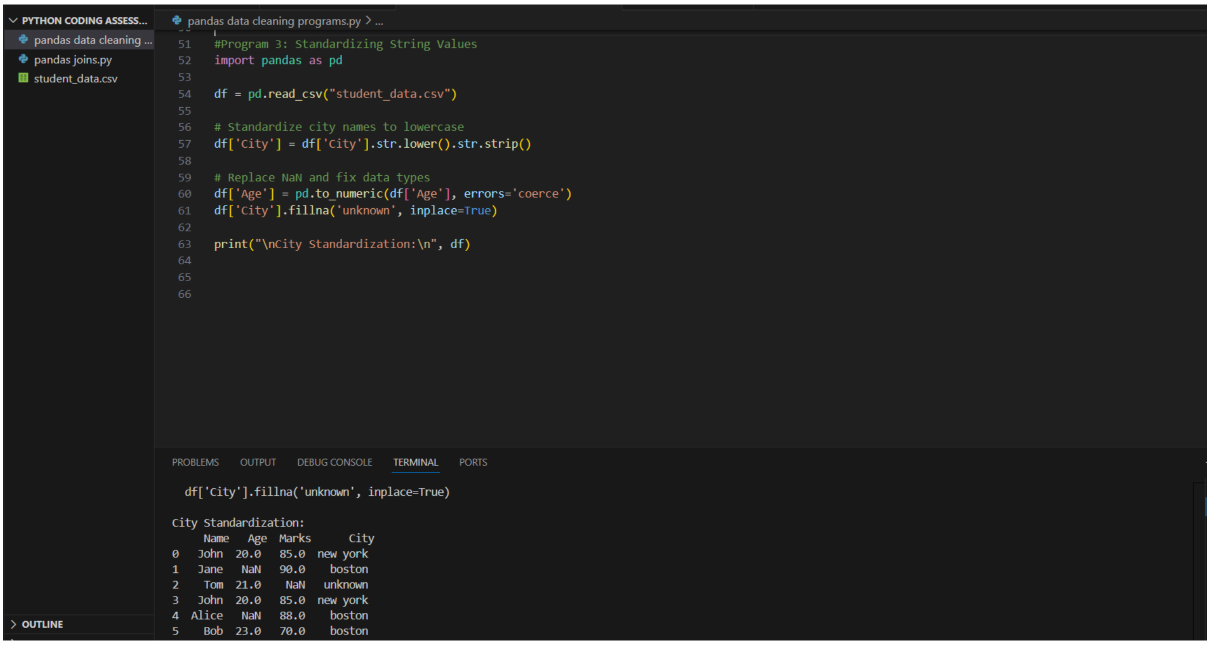
```python
# Program 1: Handling Missing Values
import pandas as pd

df = pd.read_csv("student_data.csv")

# Display original data
print("Original Data:\n", df)

# Fill missing Age with average
df['Age'] = pd.to_numeric(df['Age'], errors='coerce')
df['Age'].fillna(df['Age'].mean(), inplace=True)

# Fill missing Marks with median
df['Marks'].fillna(df['Marks'].median(), inplace=True)

# Fill missing City with 'Unknown'
df['City'].fillna('Unknown', inplace=True)

print("\nCleaned Data:\n", df)
```

```
  df['City'].fillna('Unknown', inplace=True)

Cleaned Data:
     Name   Age  Marks      City
0   John  20.0   85.0  New York
1   Jane  21.0   90.0    Boston
2    Tom  21.0   85.0   Unknown
3   John  20.0   85.0  New York
4  Alice  21.0   88.0    Boston
5    Bob  23.0   70.0    Boston
```



```python
#Program 2: Removing Duplicates and Converting Data Types
import pandas as pd

df = pd.read_csv("student_data.csv")

# Remove duplicate rows
df.drop_duplicates(inplace=True)

# Convert 'Age' to numeric
df['Age'] = pd.to_numeric(df['Age'], errors='coerce')

# Drop rows where Age couldn't be converted
df.dropna(subset=['Age'], inplace=True)

print("\nAfter removing duplicates and fixing Age:\n", df)
```

```
After removing duplicates and fixing Age:
   Name   Age  Marks      City
0  John  20.0   85.0  New York
2   Tom  21.0    NaN       NaN
5   Bob  23.0   70.0    Boston
```



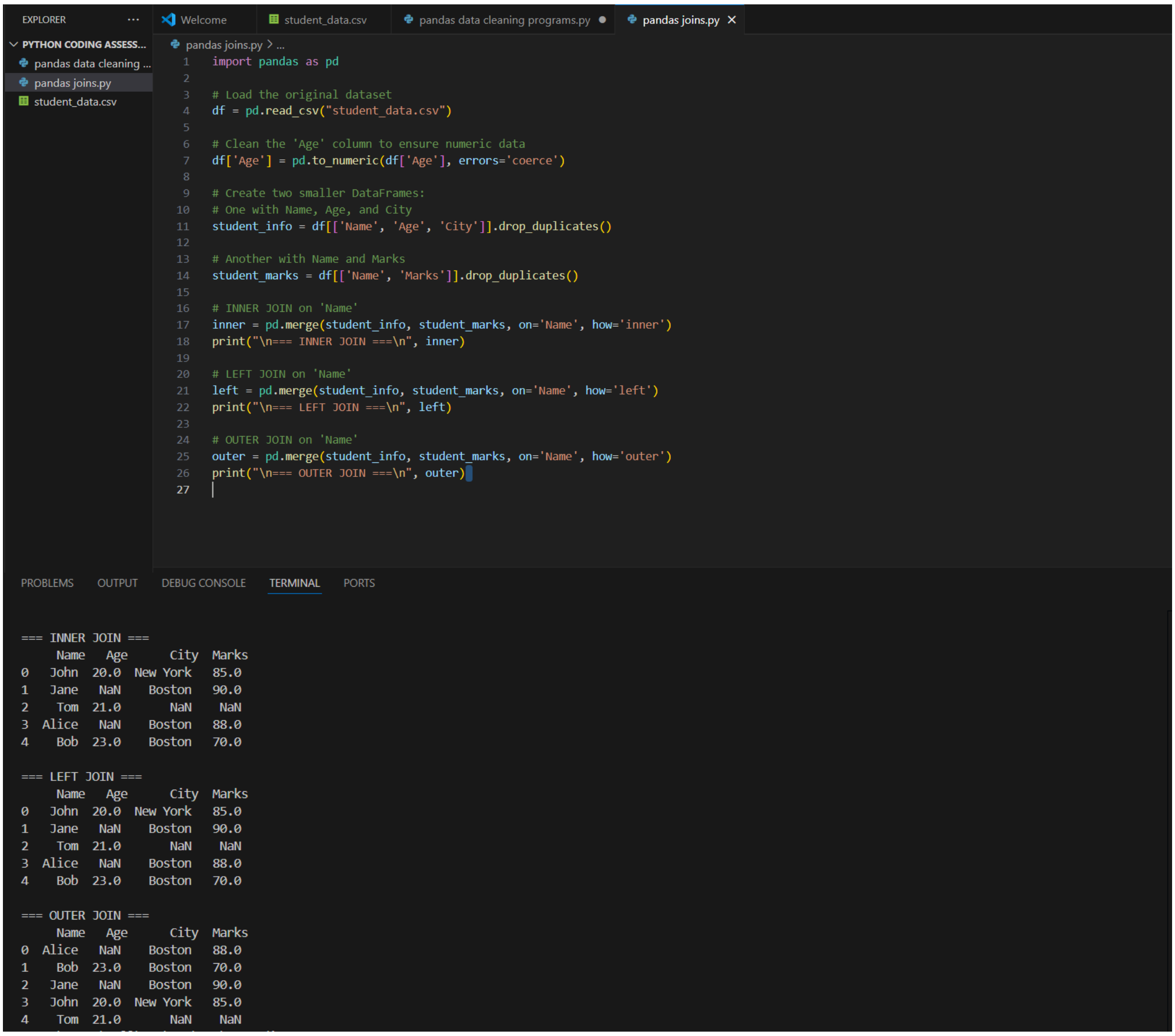
```python
#Program 3: Standardizing String Values
import pandas as pd

df = pd.read_csv("student_data.csv")

# Standardize city names to lowercase
df['City'] = df['City'].str.lower().str.strip()

# Replace NaN and fix data types
df['Age'] = pd.to_numeric(df['Age'], errors='coerce')
df['City'].fillna('unknown', inplace=True)

print("\nCity Standardization:\n", df)
```

```
  df['City'].fillna('unknown', inplace=True)

City Standardization:
     Name   Age  Marks      City
0   John  20.0   85.0  new york
1   Jane   NaN   90.0    boston
2    Tom  21.0    NaN   unknown
3   John  20.0   85.0  new york
4  Alice   NaN   88.0    boston
5    Bob  23.0   70.0    boston
```

PANDAS JOINS IN PYTHON :

Pandas supports the following joins similar to SQL:

- Inner Join
- Left Join
- Right Join
- Outer Join



```python
import pandas as pd

# Load the original dataset
df = pd.read_csv("student_data.csv")

# Clean the 'Age' column to ensure numeric data
df['Age'] = pd.to_numeric(df['Age'], errors='coerce')

# Create two smaller DataFrames:
# One with Name, Age, and City
student_info = df[['Name', 'Age', 'City']].drop_duplicates()

# Another with Name and Marks
student_marks = df[['Name', 'Marks']].drop_duplicates()

# INNER JOIN on 'Name'
inner = pd.merge(student_info, student_marks, on='Name', how='inner')
print("\n=== INNER JOIN ===\n", inner)

# LEFT JOIN on 'Name'
left = pd.merge(student_info, student_marks, on='Name', how='left')
print("\n=== LEFT JOIN ===\n", left)

# OUTER JOIN on 'Name'
outer = pd.merge(student_info, student_marks, on='Name', how='outer')
print("\n=== OUTER JOIN ===\n", outer)

```

```
=== INNER JOIN ===
     Name   Age      City  Marks
0   John  20.0  New York   85.0
1   Jane   NaN    Boston   90.0
2    Tom  21.0       NaN    NaN
3  Alice   NaN    Boston   88.0
4    Bob  23.0    Boston   70.0

=== LEFT JOIN ===
     Name   Age      City  Marks
0   John  20.0  New York   85.0
1   Jane   NaN    Boston   90.0
2    Tom  21.0       NaN    NaN
3  Alice   NaN    Boston   88.0
4    Bob  23.0    Boston   70.0

=== OUTER JOIN ===
     Name   Age      City  Marks
0  Alice   NaN    Boston   88.0
1    Bob  23.0    Boston   70.0
2   Jane   NaN    Boston   90.0
3   John  20.0  New York   85.0
4    Tom  21.0       NaN    NaN
```