In [1]:

```
print("hello world")
```

```
hello world
```

In [2]:

```
import ipywidgets as widgets
from IPython.display import display

slider = widgets.IntSlider(value=5, min=0, max=10, step=1)
display(slider)
```

5

In [4]:

```
import pandas as pd
df = pd.read_csv('C:\\Users\\TCS\\Downloads\\annual-enterprise-survey-2024-financial-year-provisional-size-bands (1).csv')
df.head()
```

Out[4]:

	year	industry_code_ANZSIC	industry_name_ANZSIC	rme_size_grp	variable	value	unit	Unnamed: 7	Unnamed: 8	Unnamed: 9	Unnamed: 10
0	2011	A	Agriculture, Forestry and Fishing	a_0	Activity unit	46134	COUNT	NaN	NaN	NaN	NaN
1	2011	A	Agriculture, Forestry and Fishing	a_0	Rolling mean employees	0	COUNT	NaN	NaN	NaN	NaN
2	2011	A	Agriculture, Forestry and Fishing	a_0	Salaries and wages paid	279	DOLLARS(millions)	NaN	NaN	NaN	NaN
3	2011	A	Agriculture, Forestry and Fishing	a_0	Sales, government funding, grants and subsidies	8187	DOLLARS(millions)	NaN	NaN	NaN	NaN
4	2011	A	Agriculture, Forestry and Fishing	a_0	Total income	8866	DOLLARS(millions)	NaN	NaN	NaN	NaN

In [9]:

```
import pandas as pd
df = pd.read_excel(r'C:\Users\TCS\Downloads\FSI-2023-DOWNLOAD (1).xlsx')

df.head()
```

Out[9]:

	Country	Year	Rank	Total	S1: Demographic Pressures	S2: Refugees and IDPs	C3: Group Grievance	E3: Human Flight and Brain Drain	E2: Economic Inequality	E1: Economy	P1: State Legitimacy	P2: Public Services	P3: Human Rights	C1: Security Apparatus	C Factionaliz Elit
0	Somalia	2023	1st	111.9	10.0	9.0	8.7	8.6	9.1	9.5	9.6	9.8	9.0	9.5	10
1	Yemen	2023	2nd	108.9	9.6	9.6	8.8	6.4	7.9	9.9	9.8	9.6	9.6	8.6	9
2	South Sudan	2023	3rd	108.5	9.7	10.0	8.6	6.5	8.6	8.6	9.8	9.7	8.7	9.9	9
3	Congo Democratic Republic	2023	4th	107.2	9.7	9.8	9.4	6.4	8.4	8.1	9.3	9.3	9.3	8.8	9
4	Syria	2023	5th	107.1	7.4	9.1	9.1	8.0	6.5	9.6	10.0	9.0	9.1	9.4	9

In [17]:

```
df.to_excel(r'C:\Users\TCS\Downloads\FSI-2023-DOWNLOAD (1).xlsx', index=False)
```

In [19]:

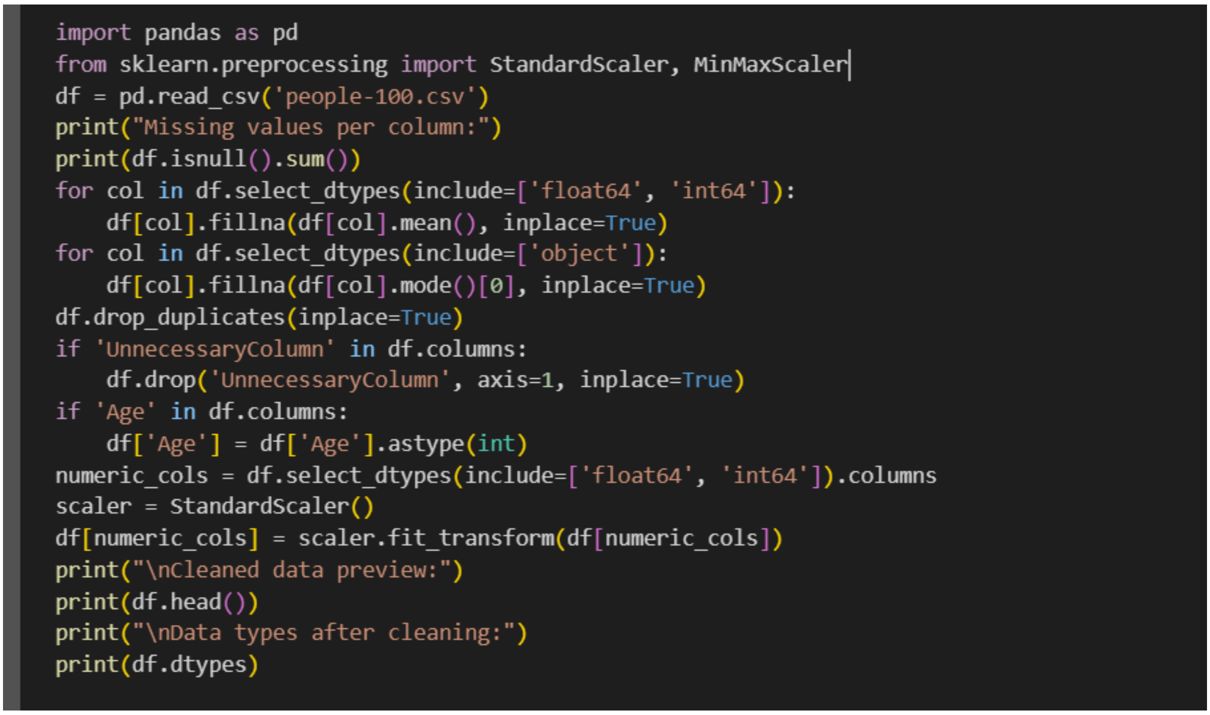
```
import pandas as pd

df = pd.DataFrame({
    'Name': ['Alice', 'Bob'],
    'Department': ['HR', 'IT'],
    'Salary': [50000, 60000]
})

df.to_excel(r'C:\Users\TCS\Downloads\FSI-2023-DOWNLOAD (1).xlsx', index=False)
```

In [ ]:

In [ ]:

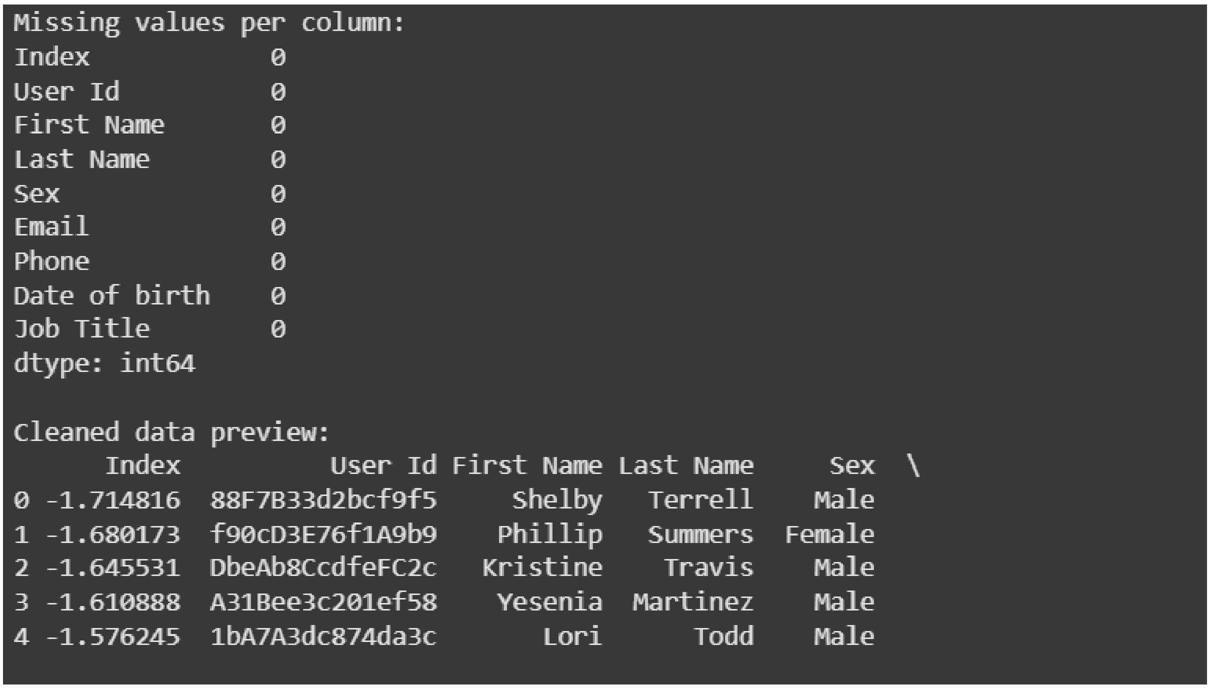
EXP 3

CODE:



```
import pandas as pd
from sklearn.preprocessing import StandardScaler, MinMaxScaler
df = pd.read_csv('people-100.csv')
print("Missing values per column:")
print(df.isnull().sum())
for col in df.select_dtypes(include=['float64', 'int64']):
    df[col].fillna(df[col].mean(), inplace=True)
for col in df.select_dtypes(include=['object']):
    df[col].fillna(df[col].mode()[0], inplace=True)
df.drop_duplicates(inplace=True)
if 'UnnecessaryColumn' in df.columns:
    df.drop('UnnecessaryColumn', axis=1, inplace=True)
if 'Age' in df.columns:
    df['Age'] = df['Age'].astype(int)
numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns
scaler = StandardScaler()
df[numeric_cols] = scaler.fit_transform(df[numeric_cols])
print("\nCleaned data preview:")
print(df.head())
print("\nData types after cleaning:")
print(df.dtypes)
```

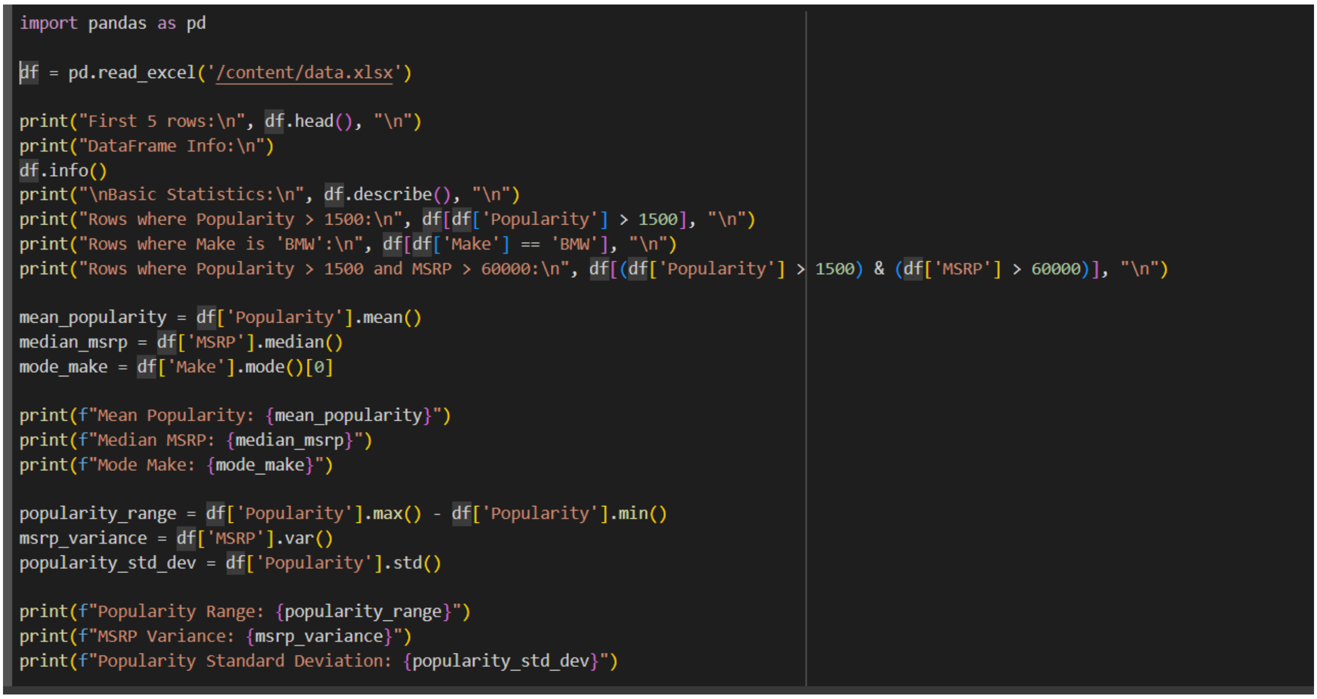
OUTPUT:



```
Missing values per column:
Index            0
User Id          0
First Name       0
Last Name        0
Sex              0
Email            0
Phone            0
Date of birth    0
Job Title        0
dtype: int64

Cleaned data preview:
      Index          User Id First Name Last Name     Sex  \
0 -1.714816  88F7B33d2bcf9f5     Shelby   Terrell    Male
1 -1.680173  f90cD3E76f1A9b9    Phillip   Summers  Female
2 -1.645531  DbeAb8CcdfeFC2c   Kristine    Travis    Male
3 -1.610888  A31Bee3c201ef58    Yesenia  Martinez    Male
4 -1.576245  1bA7A3dc874da3c       Lori      Todd    Male
```

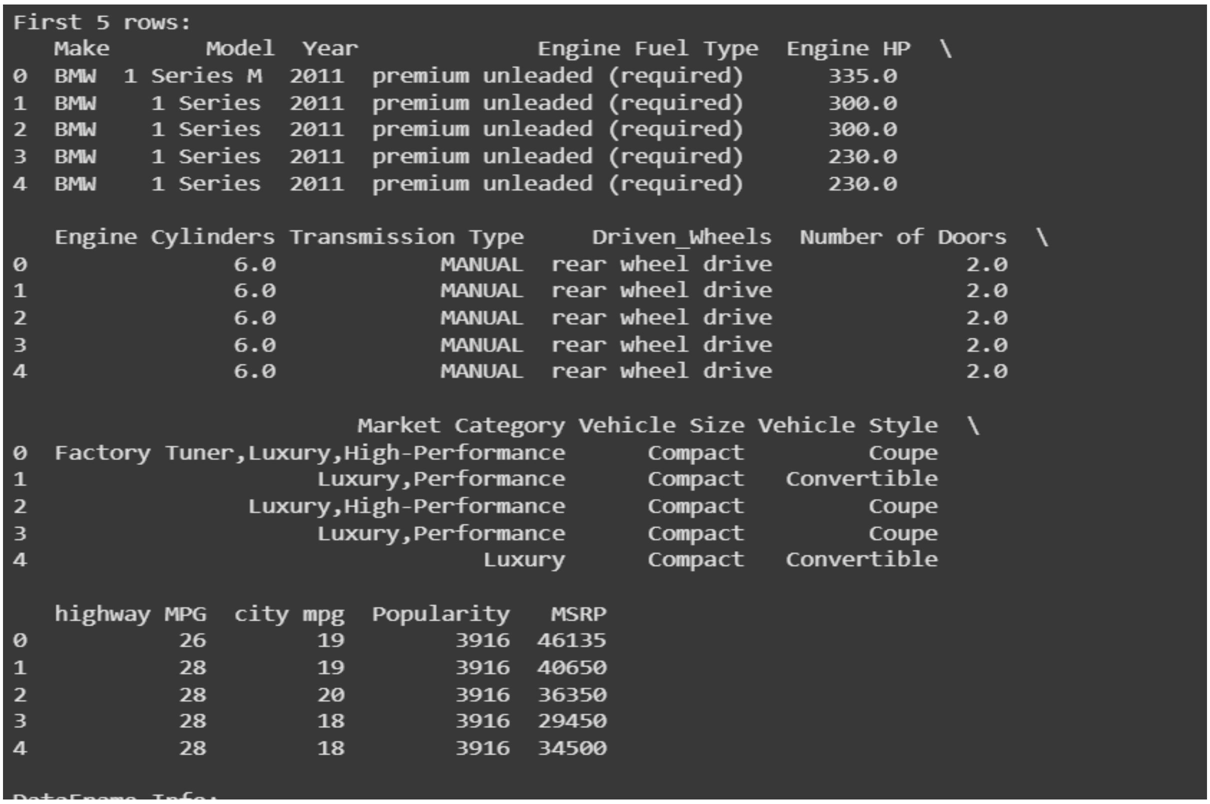
EXP 4

CODE:



```
import pandas as pd

df = pd.read_excel('/content/data.xlsx')

print("First 5 rows:\n", df.head(), "\n")
print("DataFrame Info:\n")
df.info()
print("\nBasic Statistics:\n", df.describe(), "\n")
print("Rows where Popularity > 1500:\n", df[df['Popularity'] > 1500], "\n")
print("Rows where Make is 'BMW':\n", df[df['Make'] == 'BMW'], "\n")
print("Rows where Popularity > 1500 and MSRP > 60000:\n", df[(df['Popularity'] > 1500) & (df['MSRP'] > 60000)], "\n")

mean_popularity = df['Popularity'].mean()
median_msrp = df['MSRP'].median()
mode_make = df['Make'].mode()[0]

print(f"Mean Popularity: {mean_popularity}")
print(f"Median MSRP: {median_msrp}")
print(f"Mode Make: {mode_make}")

popularity_range = df['Popularity'].max() - df['Popularity'].min()
msrp_variance = df['MSRP'].var()
popularity_std_dev = df['Popularity'].std()

print(f"Popularity Range: {popularity_range}")
print(f"MSRP Variance: {msrp_variance}")
print(f"Popularity Standard Deviation: {popularity_std_dev}")
```

OUTPUT:



```
First 5 rows:
   Make       Model  Year             Engine Fuel Type  Engine HP  \
0  BMW  1 Series M  2011  premium unleaded (required)      335.0   
1  BMW    1 Series  2011  premium unleaded (required)      300.0   
2  BMW    1 Series  2011  premium unleaded (required)      300.0   
3  BMW    1 Series  2011  premium unleaded (required)      230.0   
4  BMW    1 Series  2011  premium unleaded (required)      230.0   

   Engine Cylinders Transmission Type     Driven_Wheels  Number of Doors  \
0               6.0            MANUAL  rear wheel drive              2.0   
1               6.0            MANUAL  rear wheel drive              2.0   
2               6.0            MANUAL  rear wheel drive              2.0   
3               6.0            MANUAL  rear wheel drive              2.0   
4               6.0            MANUAL  rear wheel drive              2.0   

                       Market Category Vehicle Size Vehicle Style  \
0  Factory Tuner,Luxury,High-Performance      Compact         Coupe   
1                     Luxury,Performance      Compact   Convertible   
2                Luxury,High-Performance      Compact         Coupe   
3                     Luxury,Performance      Compact         Coupe   
4                                 Luxury      Compact   Convertible   

   highway MPG  city mpg  Popularity   MSRP  
0           26        19        3916  46135  
1           28        19        3916  40650  
2           28        20        3916  36350  
3           28        18        3916  29450  
4           28        18        3916  34500  

DataFrame Info:
```