```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

def read_and_transform_worldbank_data_cleaned(filename):
    # Read the data
    df = pd.read_csv(filename, skiprows= 4)

    # Drop unnecessary columns
    df.drop(['Country Code', 'Indicator Name', 'Indicator Code'], axis=1,
inplace=True)

    # Replace NaN with 0 for numeric columns
    df.fillna(0, inplace=True)

    # Transpose the dataframe to have years as columns
    df_years = df.set_index('Country Name').transpose()

    # Remove rows and columns that are entirely NaN
    df_years.dropna(axis=0, how='all', inplace=True)
    df_years.dropna(axis=1, how='all', inplace=True)

    return df_years

# File path
file_path = 'API_19_DS2_en_csv_v2_5998250.csv'

# Read and transform the data with cleaning
df_years_cleaned = read_and_transform_worldbank_data_cleaned(file_path)

# Select a few countries for analysis
selected_countries = ['United States', 'China', 'India', 'Brazil',
'Germany']

# Extract data for the selected countries
selected_data = df_years_cleaned[selected_countries]

# Display summary statistics using .describe()
summary_statistics = selected_data.describe()

# Displaying summary statistics
print(summary_statistics)

# Checking the first few column names in the dataframe to identify the
correct ones for the indicators
column_names = df_years_cleaned.columns.tolist()
column_names[:50]  # Displaying the first 50 column names for brevity

# Checking the first few rows of the dataframe to understand its
structure
print(df_years_cleaned.head())

# Reprocessing the data to preserve indicator names along with country
names

def reprocess_worldbank_data(filename):
    # Read the data
    df = pd.read_csv(filename, skiprows=4)

    # Drop unnecessary columns
    df.drop(['Country Code', 'Indicator Code'], axis=1, inplace=True)

    # Replace NaN with 0 for numeric columns
    df.fillna(0, inplace=True)
```

```
    # Reshaping the dataframe to have 'Country Name', 'Indicator Name'
and 'Year' as columns
    df_melted = df.melt(id_vars=['Country Name', 'Indicator Name'],
var_name='Year', value_name='Value')

    return df_melted

# File path
file_path = 'API_19_DS2_en_csv_v2_5998250.csv'

# Reprocess the data
df_melted = reprocess_worldbank_data(file_path)

# Checking the first few rows of the reprocessed dataframe
print(df_melted.head())

# Selecting indicators and countries
selected_indicators = ['GDP growth (annual %)', 'Population growth
(annual %)', 'Energy use (kg of oil equivalent per capita)']
selected_countries = ['United States', 'China', 'India', 'Brazil',
'Germany']

# Filtering the data
df_filtered = df_melted[(df_melted['Country
Name'].isin(selected_countries)) & (df_melted['Indicator
Name'].isin(selected_indicators))]

# Creating line plots
plt.figure(figsize=(15, 10))

for i, indicator in enumerate(selected_indicators, 1):
    plt.subplot(len(selected_indicators), 1, i)

    # Check if there is data available for the indicator and countries
    if df_filtered[(df_filtered['Indicator Name'] == indicator)]['Country
Name'].nunique() > 0:
        sns.lineplot(data=df_filtered[df_filtered['Indicator Name'] ==
indicator], x='Year', y='Value', hue='Country Name')
        plt.title(f'{indicator} (2000-2020)')
        plt.ylabel('Value')
        plt.xlabel('Year')
    else:
        print(f"No data available for {indicator} in the selected
countries.")

plt.tight_layout()
plt.show()


# Selecting additional indicators
additional_indicator = 'CO2 emissions (metric tons per capita)'

# Adding the additional indicator to the list of selected indicators
selected_indicators.append(additional_indicator)

# Filtering the data again with the updated list of selected indicators
df_filtered_additional = df_melted[(df_melted['Country
Name'].isin(selected_countries)) & (df_melted['Indicator
Name'].isin(selected_indicators))]

# Creating line plots with the additional indicator
plt.figure(figsize=(15, 12))  # Adjusting the figure size to accommodate
the additional plot
```

```
for i, indicator in enumerate(selected_indicators, 1):
    plt.subplot(len(selected_indicators), 1, i)

    # Check if there is data available for the indicator and countries
    if df_filtered_additional[(df_filtered_additional['Indicator Name']
== indicator)]['Country Name'].nunique() > 0:

sns.lineplot(data=df_filtered_additional[df_filtered_additional['Indicato
r Name'] == indicator], x='Year', y='Value', hue='Country Name')
        plt.title(f'{indicator} (2000-2020)')
        plt.ylabel('Value')
        plt.xlabel('Year')
    else:
        print(f"No data available for {indicator} in the selected
countries.")

plt.tight_layout()
plt.show()



# Creating a heatmap for the additional indicator
plt.figure(figsize=(15, 5))

# Check if there is data available for the additional indicator and
countries
if df_filtered_additional[(df_filtered_additional['Indicator Name'] ==
additional_indicator)]['Country Name'].nunique() > 0:

sns.heatmap(data=df_filtered_additional[df_filtered_additional['Indicator
Name'] == additional_indicator].pivot(index='Country Name',
columns='Year', values='Value'), cmap='YlGnBu')
    plt.title(f'{additional_indicator} Heatmap (2000-2020)')
    plt.ylabel('Country')
    plt.xlabel('Year')
else:
    print(f"No data available for {additional_indicator} in the selected
countries.")

plt.tight_layout()
plt.show()
```