Question 3 + 4

720017170

Question 3 - See QMD for Code

(a) Import each dataset into memory as a separate data frame, keeping all countries as your sample.

```
plt.rcParams.update({'font.size': 14})

# Loading in the Data
Health_Data = pd.read_csv('Data/Health.csv', index_col=None)
Infant_Data = pd.read_csv("Data/Infant.csv",index_col=None)

# Replace .. with NA
Health_Data.replace("..", pd.NA, inplace=True)
Infant_Data.replace("..", pd.NA, inplace=True)

# Removing unnecessary columns
Health_Data = Health_Data.drop(columns=['Series Name', 'Series Code'])
Infant_Data = Infant_Data.drop(columns=['Series Name', 'Series Code'])

# Remove names in []
Health_Data.columns = Health_Data.columns.str.replace(r'\[.*\]', '', regex=True)
Infant_Data.columns = Infant_Data.columns.str.replace(r'\[.*\]', '', regex=True)

print(Infant_Data.head())
print(Health_Data.head())
```

```
     Country Name Country Code 2000  2001  2002  2003  2004  2005  2006   \
0     Afghanistan          AFG    92  89.3  86.6  83.7  80.9    78  75.1
1         Albania          ALB    24  22.9  21.6  20.4  19.1  17.8  16.5
2         Algeria          DZA  35.6  34.3    33  31.6  30.3    29  27.8
3  American Samoa          ASM  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
```

```
4         Andorra          AND   6.5   6.3     6   5.8   5.6   5.3   5.1

  2007   ... 2014  2015  2016  2017  2018  2019  2020  2021  2022  2023
0  72.3  ...  56.2  54.6    53  51.5  50.1  48.8  47.4  46.1  44.8  <NA>
1  15.3  ...   8.8   8.5   8.4   8.3   8.3   8.3   8.4   8.4   8.4  <NA>
2  26.6  ...    22  21.7  21.4    21  20.6  20.1  19.7  19.2  18.7  <NA>
3  <NA>  ...  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
4   4.9  ...   3.5   3.4   3.2   3.1     3   2.9   2.8   2.7   2.6  <NA>

[5 rows x 26 columns]
     Country Name Country Code          2000           2001         2002   \
0     Afghanistan          AFG          <NA>           <NA>  17.00758553
1         Albania          ALB     65.1501236    73.78884125  78.99478149
2         Algeria          DZA    62.11769485    67.33850098  66.94760132
3  American Samoa          ASM          <NA>           <NA>         <NA>
4         Andorra          AND  1287.00280762  1336.21142578  1486.171875

          2003          2004          2005          2006          2007   \
0    17.81492424   21.42946434   25.10707283   28.91982269   32.71720505
1   106.29218292  138.11340332  152.12762451  166.81382751  212.61096191
2    76.23547363   93.02433014  101.30373383  117.43313599  151.77920532
3          <NA>          <NA>          <NA>          <NA>          <NA>
4  1772.71337891  1990.0748291  2214.64697266  2139.27539063  2489.43115234

   ...          2014          2015          2016          2017   \
0  ...    60.18957901    60.05854034    61.48645782    66.90921783
1  ...   295.12359619   255.35635376   277.04321289    297.4619751
2  ...   361.15942383     292.275177   261.40023804   265.83843994
3  ...          <NA>          <NA>          <NA>          <NA>
4  ...  3089.84301758  2688.20629883  2755.44848633  2873.29614258

          2018          2019          2020          2021  2022  2023
0    71.33430481    74.23410797    80.28805542    81.31976318  <NA>  <NA>
1    351.3012085   367.75839233   396.88024902   464.74285889  <NA>  <NA>
2   266.46469116   235.99041748   206.03512573   204.56661987  <NA>  <NA>
3          <NA>          <NA>          <NA>          <NA>  <NA>  <NA>
4  3164.38842773  3026.59741211  3269.29736328  3505.99145508  <NA>  <NA>

[5 rows x 26 columns]
```

(b) If data are not already stored in this way, please reshape data so that they consist of a single line of data for each country and year.

```
# Pivoting the Data into a long format
Health_Data_long = pd.melt(Health_Data,
        id_vars=['Country Name', 'Country Code'],
        var_name='Year',
        value_name='Heathcare Expenditure (USD)')
Infant_Data_long = pd.melt(Infant_Data,
        id_vars=['Country Name', 'Country Code'],
        var_name='Year',
        value_name='Infant Mortality Rates (per 1,000 live births)')
```

	Country Name	Country Code	Year	Heathcare Expenditure (USD)
0	Afghanistan	AFG	2000	<NA>
1	Albania	ALB	2000	65.1501236
2	Algeria	DZA	2000	62.11769485
3	American Samoa	ASM	2000	<NA>
4	Andorra	AND	2000	1287.00280762

	Country Name	Country Code	Year	Infant Mortality Rates (per 1,000 live births)
0	Afghanistan	AFG	2000	92
1	Albania	ALB	2000	24
2	Algeria	DZA	2000	35.6
3	American Samoa	ASM	2000	<NA>
4	Andorra	AND	2000	6.5

(c) Calculate the total number of countries observed in each data frame Calculate the total number of years observed in each data frame.

```
# Counts the number of unique contries
num_countries_FDI = Health_Data_long["Country Name"].nunique()

# Outputs the number of unique countries using an f string
print(f"Total number of unique countries in Health_Data: {num_countries_FDI}")

num_years_FDI = Health_Data_long["Year"].nunique() # Counts the number of unique years
print(f"Total number of unique years observed in Health_Data: {num_years_FDI}")
```

```
num_countries_GDP = Infant_Data_long["Country Name"].nunique()
print(f"Total number of unique countries in Infant_Data: {num_countries_GDP}")

num_years_GDP = Infant_Data_long["Year"].nunique()
print(f"Total number of unique years observed in Infant_Data: {num_years_GDP}")
```

```
Total number of unique countries in Health_Data: 217
Total number of unique years observed in Health_Data: 24
Total number of unique countries in Infant_Data: 217
Total number of unique years observed in Infant_Data: 24
```

(d) Calculate the number of observations for which data is missing

```
# Sums the number of missing values in each dataset
missing_values_Health = Health_Data_long.isna().sum().sum()
print(f"Total missing observations in Health_Data: {missing_values_Health}")

missing_values_Infant = Infant_Data_long.isna().sum().sum()
print(f"Total missing observations in Infant_Data: {missing_values_Infant}")


# Create a dataframe showing the number of missing values for each dataset
missing_values_table = pd.DataFrame({
    'Dataset': ['Health Expenditure Dataset', 'Infant Mortality Rate Dataset'],
    'Total Missing Observations': [missing_values_Health, missing_values_Infant],
    'Missing Observations (%)': [((missing_values_Health / len(Health_Data_long)) *100).roun
})
```

```
Total missing observations in Health_Data: 1099
Total missing observations in Infant_Data: 700
```

(e) Join the two files by country and year so that you have single dataframe containing both variables. Explain clearly what type of join this is, and carefully check that the number of observations resulting from the join makes sense.

```
# Merge the data on Country Name, Country code and Year
merged_data = pd.merge(Health_Data_long, Infant_Data_long, on=['Country Name',
'Country Code', 'Year'])
print(merged_data.head())
```

```
# Print the number of rows in the DataFrame
num_rows = merged_data.shape[0]
print(f"Number of rows in the DataFrame: {num_rows}")
```

```
     Country Name Country Code   Year Heathcare Expenditure (USD)  \
0     Afghanistan          AFG  2000                         <NA>
1         Albania          ALB  2000                   65.1501236
2         Algeria          DZA  2000                  62.11769485
3  American Samoa          ASM  2000                         <NA>
4         Andorra          AND  2000                1287.00280762

  Infant Mortality Rates (per 1,000 live births)
0                                             92
1                                             24
2                                           35.6
3                                           <NA>
4                                            6.5
Number of rows in the DataFrame: 5208
```

The join completed in the above code chunk is an inner join and only keeps rows that exist in both Health_Data_long and Infant_Data_long. If a country-year exists in one dataset but not the other, it will be dropped.