FIT1043 Introduction to Data Science Assignment 1

Name : Ooi Yu Zhang
Student ID : 32713339
1st April 2022

Introduction

The way I will be approaching the assignment is to first have a thorough look at the data, both in the csv files and in jupyter notebook. I will be cleaning up the data as frequent as possible to make sure that the data is always easily readable, this way I can locate, fix and solve problems more efficiently.

Importing Libraries

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
from IPython.display import display, Markdown, Latex
```

Reading The Files

In [2]:

```
led_df = pd.read_csv('data/LifeExpectancyData-v2.csv')
gdp_df = pd.read_csv('data/2019-GDP.csv')
population_df = pd.read_csv('data/2020-Population.csv')
```

Wrangling The Data

Wrangling data from Life Expectancy Data

Checking what the data looks like

In [3]:

```
led_df
```

Out[3]:

	country	Year	Status	Life expectancy	infant deaths	Adult Mortality	BMI	Alcohol consumption	Hepatitis B	Mea
0	Afghanistan	2015	Developing	65.0	62	263.0	19.1	0.01	65.0	1

	country	Year	Status	Life expectancy	infant deaths	Adult Mortality	BMI	Alcohol consumption	Hepatitis B	Mea
1	Afghanistan	2014	Developing	59.9	64	271.0	18.6	0.01	62.0	
2	Afghanistan	2013	Developing	59.9	66	268.0	18.1	0.01	64.0	
3	Afghanistan	2012	Developing	59.5	69	272.0	17.6	0.01	67.0	2
4	Afghanistan	2011	Developing	59.2	71	275.0	17.2	0.01	68.0	3
...	...	...	...	...	...	...	...	...	...	
2933	Zimbabwe	2004	Developing	44.3	27	723.0	27.1	4.36	68.0	
2934	Zimbabwe	2003	Developing	44.5	26	715.0	26.7	4.06	7.0	
2935	Zimbabwe	2002	Developing	44.8	25	73.0	26.3	4.43	73.0	
2936	Zimbabwe	2001	Developing	45.3	25	686.0	25.9	1.72	76.0	
2937	Zimbabwe	2000	Developing	46.0	24	665.0	25.5	1.68	79.0	1

2938 rows × 15 columns

In [4]:

```
led_df.head()
```

Out[4]:

	country	Year	Status	Life expectancy	infant deaths	Adult Mortality	BMI	Alcohol consumption	Hepatitis B	Measles
0	Afghanistan	2015	Developing	65.0	62	263.0	19.1	0.01	65.0	1154
1	Afghanistan	2014	Developing	59.9	64	271.0	18.6	0.01	62.0	492
2	Afghanistan	2013	Developing	59.9	66	268.0	18.1	0.01	64.0	430
3	Afghanistan	2012	Developing	59.5	69	272.0	17.6	0.01	67.0	2787
4	Afghanistan	2011	Developing	59.2	71	275.0	17.2	0.01	68.0	3013

Checking the dimensions of the data

In [5]:

```
led_df.shape
```

Out[5]:

```
(2938, 15)
```

Checking how the column headers are stored

In [6]:

```
led_df.columns
```

Out[6]:

```
Index(['country', 'Year', 'Status', 'Life expectancy ', 'infant deaths',
       'Adult Mortality', ' BMI ', 'Alcohol consumption', 'Hepatitis B',
       'Measles ', 'Polio', 'Diphtheria ', ' HIV/AIDS',
       'Income composition of resources', 'Schooling'],
      dtype='object')
```

Checking for null values in the data

In [7]:

```
led_df.info
```

Out[7]:

```
<bound method DataFrame.info of           country  Year      Status  Life expectancy   i
nfant deaths  \
0     Afghanistan  2015  Developing             65.0              62
1     Afghanistan  2014  Developing             59.9              64
2     Afghanistan  2013  Developing             59.9              66
3     Afghanistan  2012  Developing             59.5              69
4     Afghanistan  2011  Developing             59.2              71
...           ...   ...         ...              ...             ...
2933     Zimbabwe  2004  Developing             44.3              27
2934     Zimbabwe  2003  Developing             44.5              26
2935     Zimbabwe  2002  Developing             44.8              25
2936     Zimbabwe  2001  Developing             45.3              25
2937     Zimbabwe  2000  Developing             46.0              24

      Adult Mortality   BMI   Alcohol consumption  Hepatitis B  Measles   \
0               263.0  19.1                  0.01         65.0      1154
1               271.0  18.6                  0.01         62.0       492
2               268.0  18.1                  0.01         64.0       430
3               272.0  17.6                  0.01         67.0      2787
4               275.0  17.2                  0.01         68.0      3013
...               ...   ...                   ...          ...       ...
2933            723.0  27.1                  4.36         68.0        31
2934            715.0  26.7                  4.06          7.0       998
2935             73.0  26.3                  4.43         73.0       304
2936            686.0  25.9                  1.72         76.0       529
2937            665.0  25.5                  1.68         79.0      1483

      Polio  Diphtheria    HIV/AIDS  Income composition of resources  \
0       6.0        65.0        0.1                             0.479
1      58.0        62.0        0.1                             0.476
2      62.0        64.0        0.1                             0.470
3      67.0        67.0        0.1                             0.463
4      68.0        68.0        0.1                             0.454
...     ...         ...        ...                               ...
2933   67.0        65.0       33.6                             0.407
2934    7.0        68.0       36.7                             0.418
2935   73.0        71.0       39.8                             0.427
2936   76.0        75.0       42.1                             0.427
2937   78.0        78.0       43.5                             0.434

      Schooling
0          10.1
1          10.0
2           9.9
3           9.8
4           9.5
...         ...
2933        9.2
2934        9.5
2935       10.0
2936        9.8
2937        9.8
```

```
[2938 rows x 15 columns]>
```

Renaming the column headers (Making the column headers look tidier and removing whitespace from some of the column headers)

In [8]:

```
led_df.rename(
    columns={
        'country':'Country',
        'Life expectancy ':'Life Expectancy',
        'infant deaths':'Infant Deaths',
        ' BMI ':'BMI',
        'Alcohol consumption':'Alcohol Consumption',
        'Measles ':'Measles',
        'Diphtheria ':' HIV/AIDS',
        'Income composition of resources':'Income Composition of Resources'
    }, inplace=True
)
```

Verifying the change in column headers

In [9]:

```
led_df.head()
```

Out[9]:

	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	Measles
0	Afghanistan	2015	Developing	65.0	62	263.0	19.1	0.01	65.0	1154
1	Afghanistan	2014	Developing	59.9	64	271.0	18.6	0.01	62.0	492
2	Afghanistan	2013	Developing	59.9	66	268.0	18.1	0.01	64.0	430
3	Afghanistan	2012	Developing	59.5	69	272.0	17.6	0.01	67.0	2787
4	Afghanistan	2011	Developing	59.2	71	275.0	17.2	0.01	68.0	3013

Getting the list of all countries in the data

In [10]:

```
led_df['Country'].unique()
```

Out[10]:

```
array(['Afghanistan', 'Albania', 'Algeria', 'Angola',
       'Antigua and Barbuda', 'Argentina', 'Armenia', 'Australia',
       'Austria', 'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh',
       'Barbados', 'Belarus', 'Belgium', 'Belize', 'Benin', 'Bhutan',
       'Bolivia (Plurinational State of)', 'Bosnia and Herzegovina',
       'Botswana', 'Brazil', 'Brunei Darussalam', 'Bulgaria',
       'Burkina Faso', 'Burundi', "Côte d'Ivoire", 'Cabo Verde',
       'Cambodia', 'Cameroon', 'Canada', 'Central African Republic',
       'Chad', 'Chile', 'China', 'Colombia', 'Comoros', 'Congo',
       'Cook Islands', 'Costa Rica', 'Croatia', 'Cuba', 'Cyprus',
       'Czechia', "Democratic People's Republic of Korea",
       'Democratic Republic of the Congo', 'Denmark', 'Djibouti',
       'Dominica', 'Dominican Republic', 'Ecuador', 'Egypt',
       'El Salvador', 'Equatorial Guinea', 'Eritrea', 'Estonia',
       'Ethiopia', 'Fiji', 'Finland', 'France', 'Gabon', 'Gambia',
       'Georgia', 'Germany', 'Ghana', 'Greece', 'Grenada', 'Guatemala',
       'Guinea', 'Guinea-Bissau', 'Guyana', 'Haiti', 'Honduras',
       'Hungary', 'Iceland', 'India', 'Indonesia',
       'Iran (Islamic Republic of)', 'Iraq', 'Ireland', 'Israel', 'Italy',
       'Jamaica', 'Japan', 'Jordan', 'Kazakhstan', 'Kenya', 'Kiribati',
       'Kuwait', 'Kyrgyzstan', "Lao People's Democratic Republic",
       'Latvia', 'Lebanon', 'Lesotho', 'Liberia', 'Libya', 'Lithuania',
       'Luxembourg', 'Madagascar', 'Malawi', 'Malaysia', 'Maldives',
       'Mali', 'Malta', 'Marshall Islands', 'Mauritania', 'Mauritius',
       'Mexico', 'Micronesia (Federated States of)', 'Monaco', 'Mongolia',
       'Montenegro', 'Morocco', 'Mozambique', 'Myanmar', 'Namibia',
       'Nauru', 'Nepal', 'Netherlands', 'New Zealand', 'Nicaragua',
       'Niger', 'Nigeria', 'Niue', 'Norway', 'Oman', 'Pakistan', 'Palau',
       'Panama', 'Papua New Guinea', 'Paraguay', 'Peru', 'Philippines',
       'Poland', 'Portugal', 'Qatar', 'Republic of Korea',
       'Republic of Moldova', 'Romania', 'Russian Federation', 'Rwanda',
       'Saint Kitts and Nevis', 'Saint Lucia',
       'Saint Vincent and the Grenadines', 'Samoa', 'San Marino',
       'Sao Tome and Principe', 'Saudi Arabia', 'Senegal', 'Serbia',
       'Seychelles', 'Sierra Leone', 'Singapore', 'Slovakia', 'Slovenia',
       'Solomon Islands', 'Somalia', 'South Africa', 'South Sudan',
       'Spain', 'Sri Lanka', 'Sudan', 'Suriname', 'Swaziland', 'Sweden',
       'Switzerland', 'Syrian Arab Republic', 'Tajikistan', 'Thailand',
       'The former Yugoslav republic of Macedonia', 'Timor-Leste', 'Togo',
       'Tonga', 'Trinidad and Tobago', 'Tunisia', 'Turkey',
       'Turkmenistan', 'Tuvalu', 'Uganda', 'Ukraine',
       'United Arab Emirates',
       'United Kingdom of Great Britain and Northern Ireland',
       'United Republic of Tanzania', 'United States of America',
       'Uruguay', 'Uzbekistan', 'Vanuatu',
       'Venezuela (Bolivarian Republic of)', 'Viet Nam', 'Yemen',
       'Zambia', 'Zimbabwe'], dtype=object)
```

Saving all South East Asian countries into a list

Explanation: I have chosen list as the data structure to store the countries into, this is because lists are mutable and easy to manipulate in case I need to modify it in the future.

In [11]:

```
sea_countries = ['Brunei Darussalam', 'Cambodia', 'Indonesia', 'Philippines', 'Lao Peop
                 'Myanmar', 'Singapore', 'Thailand', 'Timor-Leste', 'Viet Nam']
```

Filtering the data to only contain data from South East Asian countries and verifying the change

In [12]:

```
sealed_df = led_df[led_df['Country'].isin(sea_countries)]
sealed_df
```

Out[12]:

	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	Mea
368	Brunei Darussalam	2015	Developing	77.7	0	78.0	41.2	NaN	99.0	
369	Brunei Darussalam	2014	Developing	77.6	0	8.0	4.2	0.01	99.0	

	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	Mea
370	Brunei Darussalam	2013	Developing	77.1	0	84.0	39.2	0.01	98.0	
371	Brunei Darussalam	2012	Developing	78.3	0	79.0	38.2	0.01	99.0	
372	Brunei Darussalam	2011	Developing	77.4	0	79.0	37.2	0.97	93.0	
...	...	...	...	...	...	...	...	...	...	
2885	Viet Nam	2004	Developing	74.2	29	136.0	1.9	2.86	94.0	
2886	Viet Nam	2003	Developing	74.0	30	137.0	1.4	2.19	78.0	2
2887	Viet Nam	2002	Developing	73.8	30	137.0	1.0	2.03	NaN	6
2888	Viet Nam	2001	Developing	73.6	32	138.0	9.6	1.84	NaN	12
2889	Viet Nam	2000	Developing	73.4	33	139.0	9.2	1.60	NaN	16

176 rows × 15 columns

Creating a copy of the 'Life Expectancy' column for a different aggregation function

In [13]:

```
sealed_df = sealed_df.assign(LED=sealed_df['Life Expectancy'])
sealed_df
```

Out[13]:

	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	Mea
368	Brunei Darussalam	2015	Developing	77.7	0	78.0	41.2	NaN	99.0	
369	Brunei Darussalam	2014	Developing	77.6	0	8.0	4.2	0.01	99.0	
370	Brunei Darussalam	2013	Developing	77.1	0	84.0	39.2	0.01	98.0	
371	Brunei Darussalam	2012	Developing	78.3	0	79.0	38.2	0.01	99.0	
372	Brunei Darussalam	2011	Developing	77.4	0	79.0	37.2	0.97	93.0	
...	...	...	...	...	...	...	...	...	...	
2885	Viet Nam	2004	Developing	74.2	29	136.0	1.9	2.86	94.0	
2886	Viet Nam	2003	Developing	74.0	30	137.0	1.4	2.19	78.0	2
2887	Viet Nam	2002	Developing	73.8	30	137.0	1.0	2.03	NaN	6
2888	Viet Nam	2001	Developing	73.6	32	138.0	9.6	1.84	NaN	12

	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	Mea
2889	Viet Nam	2000	Developing	73.4	33	139.0	9.2	1.60	NaN	16

176 rows × 16 columns

Grouping the dataframe by country and status

In [14]:

```
seastatus_df = sealed_df.groupby(['Country','Status'])
```

Using the grouped dataframe to perform aggregation on the required columns

In [15]:

```
seaagg_df = seastatus_df.agg({
    'Life Expectancy':'max',
    'Adult Mortality':'sum',
    'BMI':'mean',
    'Income Composition of Resources':'mean',
    'Schooling':'mean',
    'LED':'mean'
})
seaagg_df.reset_index(inplace=True)
seaagg_df
```

Out[15]:

	Country	Status	Life Expectancy	Adult Mortality	BMI	Income Composition of Resources	Schooling	LED
0	Brunei Darussalam	Developing	78.3	1073.0	29.71875	0.839375	14.10625	76.48750
1	Cambodia	Developing	68.7	3142.0	15.36250	0.491937	9.87500	64.34375
2	Indonesia	Developing	69.1	2665.0	19.95625	0.641437	11.61250	67.55625
3	Lao People's Democratic Republic	Developing	65.7	3155.0	14.36250	0.515625	9.23125	62.38125
4	Malaysia	Developing	75.0	1897.0	29.16875	0.749125	12.56250	73.75625
5	Myanmar	Developing	66.6	2469.0	17.12500	0.488250	8.32500	64.20000
6	Philippines	Developing	68.5	3487.0	19.18750	0.650438	11.54375	67.57500
7	Singapore	Developed	87.0	992.0	25.90625	0.866875	13.98125	81.47500
8	Thailand	Developing	74.9	2566.0	21.59375	0.694688	12.55000	73.08125
9	Timor-Leste	Developing	68.3	2726.0	14.55000	0.517625	10.70000	64.75625
10	Viet Nam	Developing	76.0	2025.0	11.18750	0.627063	11.51250	74.77500

Renaming the column headers for the aggregated dataframe

In [16]:

```
seaagg_df.rename(
    columns = {
        'Life Expectancy':'Max Life Expectancy',
        'BMI':'Mean BMI',
        'Income Composition of Resources':'Mean Income Composition of Resources',
        'Schooling':'Mean Schooling',
        'LED':'Mean Life Expectancy'
    }, inplace=True
)
seaagg_df
```

Out[16]:

	Country	Status	Max Life Expectancy	Adult Mortality	Mean BMI	Mean Income Composition of Resources	Mean Schooling	Mean Life Expectancy
0	Brunei Darussalam	Developing	78.3	1073.0	29.71875	0.839375	14.10625	76.48750
1	Cambodia	Developing	68.7	3142.0	15.36250	0.491937	9.87500	64.34375
2	Indonesia	Developing	69.1	2665.0	19.95625	0.641437	11.61250	67.55625
3	Lao People's Democratic Republic	Developing	65.7	3155.0	14.36250	0.515625	9.23125	62.38125
4	Malaysia	Developing	75.0	1897.0	29.16875	0.749125	12.56250	73.75625
5	Myanmar	Developing	66.6	2469.0	17.12500	0.488250	8.32500	64.20000
6	Philippines	Developing	68.5	3487.0	19.18750	0.650438	11.54375	67.57500
7	Singapore	Developed	87.0	992.0	25.90625	0.866875	13.98125	81.47500
8	Thailand	Developing	74.9	2566.0	21.59375	0.694688	12.55000	73.08125
9	Timor-Leste	Developing	68.3	2726.0	14.55000	0.517625	10.70000	64.75625
10	Viet Nam	Developing	76.0	2025.0	11.18750	0.627063	11.51250	74.77500

Wrangling data from GDP

Checking what the data looks like

In [17]:

```
gdp_df
```

Out[17]:

	Unnamed: 0	Gross domestic product 2019	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5
0	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	(millions of	NaN
2	NaN	Ranking	NaN	Economy	US dollars)	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN
4	USA	1	NaN	United States	21,427,700	NaN

	Unnamed: 0	Gross domestic product 2019	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5
...	...	...	...	...	...	...
239	NaN	.. Not available.	NaN	NaN	NaN	NaN
240	NaN	Note: Rankings include only those economies wi...	NaN	NaN	NaN	NaN
241	NaN	a. Based on data from official statistics of U...	NaN	NaN	NaN	NaN
242	NaN	GDP data source: http://data.worldbank.org/dat...	NaN	NaN	NaN	NaN
243	NaN	GDP projections: http://data.worldbank.org/da...	NaN	NaN	NaN	NaN

244 rows × 6 columns

In [18]:

```
gdp_df.head()
```

Out[18]:

	Unnamed: 0	Gross domestic product 2019	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5
0	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	(millions of	NaN
2	NaN	Ranking	NaN	Economy	US dollars)	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN
4	USA	1	NaN	United States	21,427,700	NaN

Rereading the file for GDP

Explanation: After looking inside the CSV file, 2019-GDP.csv, I have come to the conclusion that the words in row 2 are supposed to be the actual column headers.

In [19]:

```
gdp_df = pd.read_csv('data/2019-GDP.csv', skiprows=3)
gdp_df
```

Out[19]:

	Unnamed: 0	Ranking	Unnamed: 2	Economy	US dollars)	Unnamed: 5
0	NaN	NaN	NaN	NaN	NaN	NaN
1	USA	1	NaN	United States	21,427,700	NaN
2	CHN	2	NaN	China	14,342,903	NaN
3	JPN	3	NaN	Japan	5,081,770	NaN
4	DEU	4	NaN	Germany	3,845,630	NaN
...	...	...	...	...	...	...

	Unnamed: 0	Ranking	Unnamed: 2	Economy	US dollars)	Unnamed: 5
236	NaN	.. Not available.	NaN	NaN	NaN	NaN
237	NaN	Note: Rankings include only those economies wi...	NaN	NaN	NaN	NaN
238	NaN	a. Based on data from official statistics of U...	NaN	NaN	NaN	NaN
239	NaN	GDP data source: http://data.worldbank.org/dat...	NaN	NaN	NaN	NaN
240	NaN	GDP projections: http://data.worldbank.org/da...	NaN	NaN	NaN	NaN

241 rows × 6 columns

Checking the dimensions of the data

In [20]:

```
gdp_df.shape
```

Out[20]:

```
(241, 6)
```

Checking how the column headers are stored

In [21]:

```
gdp_df.columns
```

Out[21]:

```
Index(['Unnamed: 0', 'Ranking', 'Unnamed: 2', 'Economy', 'US dollars)',
       'Unnamed: 5'],
      dtype='object')
```

Checking for null values in the data

In [22]:

```
gdp_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 241 entries, 0 to 240
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Unnamed: 0   229 non-null    object 
 1   Ranking      208 non-null    object 
 2   Unnamed: 2   0 non-null      float64
 3   Economy      229 non-null    object 
 4   US dollars)  229 non-null    object 
 5   Unnamed: 5   8 non-null      object 
dtypes: float64(1), object(5)
memory usage: 11.4+ KB
```

Retaining the useful columns by dropping the unused ones, and removing the None values

Explanation: After thoroughly reading through the assignment specifications, I've decided to only retain columns 3 and 4 as those will only be used for the rest of the assignment.

In [23]:

```
newgdp_df = gdp_df[['Economy','US dollars)']][~gdp_df['Economy'].isna() & ~gdp_df['US d
newgdp_df
```

Out[23]:

	Economy	US dollars)
1	United States	21,427,700
2	China	14,342,903
3	Japan	5,081,770
4	Germany	3,845,630
5	India	2,875,142
...	...	...
230	Sub-Saharan Africa	1,755,011
231	Low income	521,274
232	Lower middle income	6,341,105
233	Upper middle income	25,817,130
234	High income	55,098,717

229 rows × 2 columns

Appropriately renaming the column headers

In [24]:

```
newgdp_df.rename(columns={
    'Economy':'Country',
    'US dollars)':'GDP'
}, inplace=True)
newgdp_df
```

Out[24]:

	Country	GDP
1	United States	21,427,700
2	China	14,342,903
3	Japan	5,081,770
4	Germany	3,845,630
5	India	2,875,142
...	...	...
230	Sub-Saharan Africa	1,755,011
231	Low income	521,274
232	Lower middle income	6,341,105
233	Upper middle income	25,817,130
234	High income	55,098,717

229 rows × 2 columns

Checking the naming conventions for the countries in the column Country

In [25]:

```
newgdp_df['Country'].unique()
```

Out[25]:

```
array(['United States', 'China', 'Japan', 'Germany', 'India',
       'United Kingdom', 'France', 'Italy', 'Brazil', 'Canada',
       'Russian Federation', 'Korea, Rep.', 'Spain', 'Australia',
       'Mexico', 'Indonesia', 'Netherlands', 'Saudi Arabia', 'Turkey',
       'Switzerland', 'Poland', 'Thailand', 'Sweden', 'Belgium',
       'Argentina', 'Nigeria', 'Austria', 'Iran, Islamic Rep.',
       'United Arab Emirates', 'Norway', 'Israel', 'Ireland',
       'Philippines', 'Singapore', 'Hong Kong SAR, China', 'Malaysia',
       'South Africa', 'Denmark', 'Colombia', 'Egypt, Arab Rep.',
       'Bangladesh', 'Chile', 'Pakistan', 'Finland', 'Vietnam', 'Romania',
       'Czech Republic', 'Portugal', 'Iraq', 'Peru', 'Greece',
       'New Zealand', 'Qatar', 'Kazakhstan', 'Algeria', 'Hungary',
       'Ukraine', 'Kuwait', 'Morocco', 'Ecuador', 'Slovak Republic',
       'Puerto Rico', 'Cuba', 'Ethiopia', 'Kenya', 'Angola',
       'Dominican Republic', 'Sri Lanka', 'Oman', 'Guatemala', 'Myanmar',
       'Luxembourg', 'Bulgaria', 'Ghana', 'Panama', 'Tanzania', 'Belarus',
       'Costa Rica', 'Croatia', "Côte d'Ivoire", 'Uzbekistan', 'Uruguay',
       'Lithuania', 'Macao SAR, China', 'Slovenia', 'Lebanon', 'Libya',
       'Serbia', 'Azerbaijan', 'Congo, Dem. Rep.', 'Jordan', 'Bolivia',
       'Turkmenistan', 'Tunisia', 'Cameroon', 'Bahrain', 'Paraguay',
       'Uganda', 'Latvia', 'Estonia', 'Nepal', 'Yemen, Rep.', 'Cambodia',
       'El Salvador', 'Honduras', 'Papua New Guinea', 'Cyprus', 'Iceland',
       'Trinidad and Tobago', 'Senegal', 'Zambia', 'Zimbabwe',
       'Bosnia and Herzegovina', 'Afghanistan', 'Sudan', 'Botswana',
       'Lao PDR', 'Georgia', 'Mali', 'Gabon', 'Jamaica', 'Burkina Faso',
       'Albania', 'Mozambique', 'Malta', 'West Bank and Gaza', 'Benin',
       'Mauritius', 'Madagascar', 'Mongolia', 'Armenia', 'Guinea',
       'Brunei Darussalam', 'Niger', 'Bahamas, The', 'North Macedonia',
       'Nicaragua', 'Namibia', 'Moldova', 'Chad', 'Equatorial Guinea',
       'Congo, Rep.', 'Rwanda', 'Haiti', 'Kyrgyz Republic', 'Tajikistan',
       'Kosovo', 'Malawi', 'Mauritania', 'Monaco', 'Isle of Man',
       'Liechtenstein', 'Guam', 'Maldives', 'Fiji', 'Montenegro',
       'Cayman Islands', 'Togo', 'Barbados', 'Eswatini', 'Guyana',
       'Suriname', 'Sierra Leone', 'Virgin Islands (U.S.)', 'Djibouti',
       'Andorra', 'Curaçao', 'Liberia', 'Aruba', 'Greenland', 'Burundi',
       'Faroe Islands', 'Lesotho', 'Bhutan', 'Central African Republic',
       'St. Lucia', 'Cabo Verde', 'Belize', 'Gambia, The',
       'Antigua and Barbuda', 'Seychelles', 'Timor-Leste', 'San Marino',
       'Solomon Islands', 'Guinea-Bissau', 'Northern Mariana Islands',
       'Grenada', 'Comoros', 'St. Kitts and Nevis',
       'Turks and Caicos Islands', 'Vanuatu', 'Samoa',
       'St. Vincent and the Grenadines', 'American Samoa', 'Dominica',
       'Tonga', 'São Tomé and Principe', 'Micronesia, Fed. Sts.', 'Palau',
       'Marshall Islands', 'Kiribati', 'Nauru', 'Tuvalu', 'Bermuda',
       'British Virgin Islands', 'Channel Islands', 'Eritrea',
       'French Polynesia', 'Gibraltar', "Korea, Dem. People's Rep.",
       'New Caledonia', 'Sint Maarten (Dutch part)', 'South Sudan',
       'St. Martin (French part)', 'Syrian Arab Republic',
       'Venezuela, RB', 'Somalia', 'World', 'East Asia & Pacific',
       'Europe & Central Asia', 'Latin America & Caribbean',
       'Middle East & North Africa', 'North America', 'South Asia',
       'Sub-Saharan Africa', 'Low income', 'Lower middle income',
       'Upper middle income', 'High income'], dtype=object)
```

Changing the names for two countries to match the ones in the Life Expectancy dataframe

Explanation: After looking through the data for all countries, I've found that the naming convention for Vietnam and Lao PDR here is different from the one in the Life Expectancy dataframe (Viet Nam and Lao People's Democratic Republic)

```
In [26]:
newgdp_df.loc[newgdp_df['Country']=='Vietnam','Country'] = 'Viet Nam'
newgdp_df.loc[newgdp_df['Country']=='Lao PDR','Country'] = "Lao People's Democratic Rep
```

Checking if the names were changed properly

```
In [27]:
newgdp_df['Country'].unique()
```

```
Out[27]:
array(['United States', 'China', 'Japan', 'Germany', 'India',
       'United Kingdom', 'France', 'Italy', 'Brazil', 'Canada',
       'Russian Federation', 'Korea, Rep.', 'Spain', 'Australia',
       'Mexico', 'Indonesia', 'Netherlands', 'Saudi Arabia', 'Turkey',
       'Switzerland', 'Poland', 'Thailand', 'Sweden', 'Belgium',
       'Argentina', 'Nigeria', 'Austria', 'Iran, Islamic Rep.',
       'United Arab Emirates', 'Norway', 'Israel', 'Ireland',
       'Philippines', 'Singapore', 'Hong Kong SAR, China', 'Malaysia',
       'South Africa', 'Denmark', 'Colombia', 'Egypt, Arab Rep.',
       'Bangladesh', 'Chile', 'Pakistan', 'Finland', 'Viet Nam',
       'Romania', 'Czech Republic', 'Portugal', 'Iraq', 'Peru', 'Greece',
       'New Zealand', 'Qatar', 'Kazakhstan', 'Algeria', 'Hungary',
       'Ukraine', 'Kuwait', 'Morocco', 'Ecuador', 'Slovak Republic',
       'Puerto Rico', 'Cuba', 'Ethiopia', 'Kenya', 'Angola',
       'Dominican Republic', 'Sri Lanka', 'Oman', 'Guatemala', 'Myanmar',
       'Luxembourg', 'Bulgaria', 'Ghana', 'Panama', 'Tanzania', 'Belarus',
       'Costa Rica', 'Croatia', "Côte d'Ivoire", 'Uzbekistan', 'Uruguay',
       'Lithuania', 'Macao SAR, China', 'Slovenia', 'Lebanon', 'Libya',
       'Serbia', 'Azerbaijan', 'Congo, Dem. Rep.', 'Jordan', 'Bolivia',
       'Turkmenistan', 'Tunisia', 'Cameroon', 'Bahrain', 'Paraguay',
       'Uganda', 'Latvia', 'Estonia', 'Nepal', 'Yemen, Rep.', 'Cambodia',
       'El Salvador', 'Honduras', 'Papua New Guinea', 'Cyprus', 'Iceland',
       'Trinidad and Tobago', 'Senegal', 'Zambia', 'Zimbabwe',
       'Bosnia and Herzegovina', 'Afghanistan', 'Sudan', 'Botswana',
       "Lao People's Democratic Republic", 'Georgia', 'Mali', 'Gabon',
       'Jamaica', 'Burkina Faso', 'Albania', 'Mozambique', 'Malta',
       'West Bank and Gaza', 'Benin', 'Mauritius', 'Madagascar',
       'Mongolia', 'Armenia', 'Guinea', 'Brunei Darussalam', 'Niger',
       'Bahamas, The', 'North Macedonia', 'Nicaragua', 'Namibia',
       'Moldova', 'Chad', 'Equatorial Guinea', 'Congo, Rep.', 'Rwanda',
       'Haiti', 'Kyrgyz Republic', 'Tajikistan', 'Kosovo', 'Malawi',
       'Mauritania', 'Monaco', 'Isle of Man', 'Liechtenstein', 'Guam',
       'Maldives', 'Fiji', 'Montenegro', 'Cayman Islands', 'Togo',
       'Barbados', 'Eswatini', 'Guyana', 'Suriname', 'Sierra Leone',
       'Virgin Islands (U.S.)', 'Djibouti', 'Andorra', 'Curaçao',
       'Liberia', 'Aruba', 'Greenland', 'Burundi', 'Faroe Islands',
       'Lesotho', 'Bhutan', 'Central African Republic', 'St. Lucia',
       'Cabo Verde', 'Belize', 'Gambia, The', 'Antigua and Barbuda',
       'Seychelles', 'Timor-Leste', 'San Marino', 'Solomon Islands',
       'Guinea-Bissau', 'Northern Mariana Islands', 'Grenada', 'Comoros',
       'St. Kitts and Nevis', 'Turks and Caicos Islands', 'Vanuatu',
       'Samoa', 'St. Vincent and the Grenadines', 'American Samoa',
       'Dominica', 'Tonga', 'São Tomé and Principe',
       'Micronesia, Fed. Sts.', 'Palau', 'Marshall Islands', 'Kiribati',
       'Nauru', 'Tuvalu', 'Bermuda', 'British Virgin Islands',
       'Channel Islands', 'Eritrea', 'French Polynesia', 'Gibraltar',
       "Korea, Dem. People's Rep.", 'New Caledonia',
       'Sint Maarten (Dutch part)', 'South Sudan',
       'St. Martin (French part)', 'Syrian Arab Republic',
       'Venezuela, RB', 'Somalia', 'World', 'East Asia & Pacific',
       'Europe & Central Asia', 'Latin America & Caribbean',
       'Middle East & North Africa', 'North America', 'South Asia',
       'Sub-Saharan Africa', 'Low income', 'Lower middle income',
       'Upper middle income', 'High income'], dtype=object)
```

Filtering the data to only contain data from South East Asian countries and verifying the change

In [28]:

```
seagdp_df = newgdp_df[newgdp_df['Country'].isin(sea_countries)]
seagdp_df.reset_index(inplace=True)
seagdp_df
```

Out[28]:

	index	Country	GDP
0	16	Indonesia	1,119,191
1	22	Thailand	543,650
2	33	Philippines	376,796
3	34	Singapore	372,063
4	36	Malaysia	364,702
5	45	Viet Nam	261,921
6	71	Myanmar	76,086
7	103	Cambodia	27,089
8	117	Lao People's Democratic Republic	18,174
9	133	Brunei Darussalam	13,469
10	182	Timor-Leste	1,674

Wrangling data from Population

Checking what the data looks like

In [29]:

```
population_df
```

Out[29]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unn
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unn
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	United Nations	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	Population Division	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	
300	285	Estimates	Bermuda	14	60	Country/Area	918	37	
301	286	Estimates	Canada	NaN	124	Country/Area	918	13 733	
302	287	Estimates	Greenland	26	304	Country/Area	918	23	
303	288	Estimates	Saint Pierre and Miquelon	2	666	Country/Area	918	5	
304	289	Estimates	United States of America	35	840	Country/Area	918	158 804	1

305 rows × 78 columns

In [30]:

```python
population_df.head()
```

Out[30]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unname
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
3	United Nations	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
4	Population Division	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

5 rows × 78 columns

Rereading the file for Population

Explanation: After looking through the CSV file 2020-Population.csv, I found that the actual data that would be used in the assignment begins from row 17

In [31]:

```python
popu_df = pd.read_csv('data/2020-Population.csv', skiprows=16)
popu_df
```

Out[31]:

	Index	Variant	Region, subregion, country or area *	Notes	Country code	Type	Parent code	1950	1951	1952	...	20
0	1	Estimates	WORLD	NaN	900	World	0	2 536 431	2 584 034	2 630 862	...	0 1
1	2	Estimates	UN development groups	a	1803	Label/Separator	900	...	...	...	...	
2	3	Estimates	More developed regions	b	901	Development Group	1803	814 819	824 004	833 720	...	2 5
3	4	Estimates	Less developed regions	c	902	Development Group	1803	1 721 612	1 760 031	1 797 142	...	8 6
4	5	Estimates	Least developed countries	d	941	Development Group	902	195 428	199 180	203 015	...	8 4
...	...	...	...	...	...	...	...	...	...	...	...	
284	285	Estimates	Bermuda	14	60	Country/Area	918	37	38	38	...	
285	286	Estimates	Canada	NaN	124	Country/Area	918	13 733	14 078	14 445	...	5
286	287	Estimates	Greenland	26	304	Country/Area	918	23	23	24	...	
287	288	Estimates	Saint Pierre and Miquelon	2	666	Country/Area	918	5	5	5	...	
288	289	Estimates	United States of America	35	840	Country/Area	918	158 804	160 872	163 266	...	3 5

289 rows × 78 columns

In [32]:

```
popu_df.shape
```

Out[32]:

```
(289, 78)
```

In [33]:

```
popu_df.columns
```

Out[33]:

```
Index(['Index', 'Variant', 'Region, subregion, country or area *', 'Notes',
       'Country code', 'Type', 'Parent code', '1950', '1951', '1952', '1953',
       '1954', '1955', '1956', '1957', '1958', '1959', '1960', '1961', '1962',
       '1963', '1964', '1965', '1966', '1967', '1968', '1969', '1970', '1971',
       '1972', '1973', '1974', '1975', '1976', '1977', '1978', '1979', '1980',
       '1981', '1982', '1983', '1984', '1985', '1986', '1987', '1988', '1989',
       '1990', '1991', '1992', '1993', '1994', '1995', '1996', '1997', '1998',
       '1999', '2000', '2001', '2002', '2003', '2004', '2005', '2006', '2007',
       '2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015', '2016',
       '2017', '2018', '2019', '2020'],
      dtype='object')
```

In [34]:

```
popu_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 289 entries, 0 to 288
Data columns (total 78 columns):
 #   Column                                Non-Null Count  Dtype 
---  ------                                --------------  ----- 
 0   Index                                 289 non-null    int64 
 1   Variant                               289 non-null    object
 2   Region, subregion, country or area *  289 non-null    object
 3   Notes                                 82 non-null     object
 4   Country code                          289 non-null    int64 
 5   Type                                  289 non-null    object
 6   Parent code                           289 non-null    int64 
 7   1950                                  289 non-null    object
 8   1951                                  289 non-null    object
 9   1952                                  289 non-null    object
 10  1953                                  289 non-null    object
 11  1954                                  289 non-null    object
 12  1955                                  289 non-null    object
 13  1956                                  289 non-null    object
 14  1957                                  289 non-null    object
 15  1958                                  289 non-null    object
 16  1959                                  289 non-null    object
 17  1960                                  289 non-null    object
 18  1961                                  289 non-null    object
 19  1962                                  289 non-null    object
 20  1963                                  289 non-null    object
 21  1964                                  289 non-null    object
 22  1965                                  289 non-null    object
 23  1966                                  289 non-null    object
 24  1967                                  289 non-null    object
 25  1968                                  289 non-null    object
 26  1969                                  289 non-null    object
 27  1970                                  289 non-null    object
 28  1971                                  289 non-null    object
 29  1972                                  289 non-null    object
 30  1973                                  289 non-null    object
 31  1974                                  289 non-null    object
 32  1975                                  289 non-null    object
 33  1976                                  289 non-null    object
 34  1977                                  289 non-null    object
 35  1978                                  289 non-null    object
 36  1979                                  289 non-null    object
 37  1980                                  289 non-null    object
 38  1981                                  289 non-null    object
 39  1982                                  289 non-null    object
 40  1983                                  289 non-null    object
 41  1984                                  289 non-null    object
 42  1985                                  289 non-null    object
 43  1986                                  289 non-null    object
 44  1987                                  289 non-null    object
 45  1988                                  289 non-null    object
 46  1989                                  289 non-null    object
 47  1990                                  289 non-null    object
 48  1991                                  289 non-null    object
 49  1992                                  289 non-null    object
 50  1993                                  289 non-null    object
 51  1994                                  289 non-null    object
 52  1995                                  289 non-null    object
 53  1996                                  289 non-null    object
 54  1997                                  289 non-null    object
 55  1998                                  289 non-null    object
 56  1999                                  289 non-null    object
 57  2000                                  289 non-null    object
 58  2001                                  289 non-null    object
 59  2002                                  289 non-null    object
 60  2003                                  289 non-null    object
 61  2004                                  289 non-null    object
 62  2005                                  289 non-null    object
 63  2006                                  289 non-null    object
 64  2007                                  289 non-null    object
 65  2008                                  289 non-null    object
 66  2009                                  289 non-null    object
 67  2010                                  289 non-null    object
 68  2011                                  289 non-null    object
 69  2012                                  289 non-null    object
 70  2013                                  289 non-null    object
 71  2014                                  289 non-null    object
 72  2015                                  289 non-null    object
 73  2016                                  289 non-null    object
 74  2017                                  289 non-null    object
 75  2018                                  289 non-null    object
 76  2019                                  289 non-null    object
 77  2020                                  289 non-null    object
dtypes: int64(3), object(75)
memory usage: 176.2+ KB
```

Renaming the column header 'Region, subregion, country or area * '

Explanation: This is because the column mostly contains countries

In [35]:

```
popu_df.rename(columns={
    'Region, subregion, country or area *':'Country'
}, inplace=True)
popu_df
```

Out[35]:

	Index	Variant	Country	Notes	Country code	Type	Parent code	1950	1951	1952	...	20
0	1	Estimates	WORLD	NaN	900	World	0	2 536 431	2 584 034	2 630 862	...	0 1
1	2	Estimates	UN development groups	a	1803	Label/Separator	900	...	...	...	...	
2	3	Estimates	More developed regions	b	901	Development Group	1803	814 819	824 004	833 720	...	2 5

	Index	Variant	Country	Notes	Country code	Type	Parent code	1950	1951	1952	...	20
3	4	Estimates	Less developed regions	c	902	Development Group	1803	1 721 612	1 760 031	1 797 142	...	8 6
4	5	Estimates	Least developed countries	d	941	Development Group	902	195 428	199 180	203 015	...	8 4
...	...	...	...	...	...	...	...	...	...	...	...	
284	285	Estimates	Bermuda	14	60	Country/Area	918	37	38	38	...	
285	286	Estimates	Canada	NaN	124	Country/Area	918	13 733	14 078	14 445	...	5
286	287	Estimates	Greenland	26	304	Country/Area	918	23	23	24	...	
287	288	Estimates	Saint Pierre and Miquelon	2	666	Country/Area	918	5	5	5	...	
288	289	Estimates	United States of America	35	840	Country/Area	918	158 804	160 872	163 266	...	3 5

289 rows × 78 columns

Checking the naming conventions for the countries in the column Country

In [36]:

```
popu_df['Country'].unique()
```

Out[36]:

```
array(['WORLD', 'UN development groups', 'More developed regions',
       'Less developed regions', 'Least developed countries',
       'Less developed regions, excluding least developed countries',
       'Less developed regions, excluding China',
       'Land-locked Developing Countries (LLDC)',
       'Small Island Developing States (SIDS)',
       'World Bank income groups', 'High-income countries',
       'Middle-income countries', 'Upper-middle-income countries',
       'Lower-middle-income countries', 'Low-income countries',
       'No income group available', 'Geographic regions', 'Africa',
       'Asia', 'Europe', 'Latin America and the Caribbean',
       'Northern America', 'Oceania',
       'Sustainable Development Goal (SDG) regions', 'SUB-SAHARAN AFRICA',
       'Eastern Africa', 'Burundi', 'Comoros', 'Djibouti', 'Eritrea',
       'Ethiopia', 'Kenya', 'Madagascar', 'Malawi', 'Mauritius',
       'Mayotte', 'Mozambique', 'Réunion', 'Rwanda', 'Seychelles',
       'Somalia', 'South Sudan', 'Uganda', 'United Republic of Tanzania',
       'Zambia', 'Zimbabwe', 'Middle Africa', 'Angola', 'Cameroon',
       'Central African Republic', 'Chad', 'Congo',
       'Democratic Republic of the Congo', 'Equatorial Guinea', 'Gabon',
       'Sao Tome and Principe', 'Southern Africa', 'Botswana', 'Eswatini',
       'Lesotho', 'Namibia', 'South Africa', 'Western Africa', 'Benin',
       'Burkina Faso', 'Cabo Verde', "Côte d'Ivoire", 'Gambia', 'Ghana',
       'Guinea', 'Guinea-Bissau', 'Liberia', 'Mali', 'Mauritania',
       'Niger', 'Nigeria', 'Saint Helena', 'Senegal', 'Sierra Leone',
       'Togo', 'NORTHERN AFRICA AND WESTERN ASIA', 'Northern Africa',
       'Algeria', 'Egypt', 'Libya', 'Morocco', 'Sudan', 'Tunisia',
       'Western Sahara', 'Western Asia', 'Armenia', 'Azerbaijan',
       'Bahrain', 'Cyprus', 'Georgia', 'Iraq', 'Israel', 'Jordan',
       'Kuwait', 'Lebanon', 'Oman', 'Qatar', 'Saudi Arabia',
       'State of Palestine', 'Syrian Arab Republic', 'Turkey',
       'United Arab Emirates', 'Yemen', 'CENTRAL AND SOUTHERN ASIA',
       'Central Asia', 'Kazakhstan', 'Kyrgyzstan', 'Tajikistan',
       'Turkmenistan', 'Uzbekistan', 'Southern Asia', 'Afghanistan',
       'Bangladesh', 'Bhutan', 'India', 'Iran (Islamic Republic of)',
       'Maldives', 'Nepal', 'Pakistan', 'Sri Lanka',
       'EASTERN AND SOUTH-EASTERN ASIA', 'Eastern Asia', 'China',
       'China, Hong Kong SAR', 'China, Macao SAR',
       'China, Taiwan Province of China',
       "Dem. People's Republic of Korea", 'Japan', 'Mongolia',
       'Republic of Korea', 'South-Eastern Asia', 'Brunei Darussalam',
       'Cambodia', 'Indonesia', "Lao People's Democratic Republic",
       'Malaysia', 'Myanmar', 'Philippines', 'Singapore', 'Thailand',
       'Timor-Leste', 'Viet Nam', 'LATIN AMERICA AND THE CARIBBEAN',
       'Caribbean', 'Anguilla', 'Antigua and Barbuda', 'Aruba', 'Bahamas',
       'Barbados', 'Bonaire, Sint Eustatius and Saba',
       'British Virgin Islands', 'Cayman Islands', 'Cuba', 'Curaçao',
       'Dominica', 'Dominican Republic', 'Grenada', 'Guadeloupe', 'Haiti',
       'Jamaica', 'Martinique', 'Montserrat', 'Puerto Rico',
       'Saint Barthélemy', 'Saint Kitts and Nevis', 'Saint Lucia',
       'Saint Martin (French part)', 'Saint Vincent and the Grenadines',
       'Sint Maarten (Dutch part)', 'Trinidad and Tobago',
       'Turks and Caicos Islands', 'United States Virgin Islands',
       'Central America', 'Belize', 'Costa Rica', 'El Salvador',
       'Guatemala', 'Honduras', 'Mexico', 'Nicaragua', 'Panama',
       'South America', 'Argentina', 'Bolivia (Plurinational State of)',
       'Brazil', 'Chile', 'Colombia', 'Ecuador',
       'Falkland Islands (Malvinas)', 'French Guiana', 'Guyana',
       'Paraguay', 'Peru', 'Suriname', 'Uruguay',
       'Venezuela (Bolivarian Republic of)', 'AUSTRALIA/NEW ZEALAND',
       'Australia', 'New Zealand',
       'OCEANIA (EXCLUDING AUSTRALIA AND NEW ZEALAND)', 'Melanesia',
       'Fiji', 'New Caledonia', 'Papua New Guinea', 'Solomon Islands',
       'Vanuatu', 'Micronesia', 'Guam', 'Kiribati', 'Marshall Islands',
       'Micronesia (Fed. States of)', 'Nauru', 'Northern Mariana Islands',
       'Palau', 'Polynesia', 'American Samoa', 'Cook Islands',
       'French Polynesia', 'Niue', 'Samoa', 'Tokelau', 'Tonga', 'Tuvalu',
       'Wallis and Futuna Islands', 'EUROPE AND NORTHERN AMERICA',
       'EUROPE', 'Eastern Europe', 'Belarus', 'Bulgaria', 'Czechia',
       'Hungary', 'Poland', 'Republic of Moldova', 'Romania',
       'Russian Federation', 'Slovakia', 'Ukraine', 'Northern Europe',
       'Channel Islands', 'Denmark', 'Estonia', 'Faroe Islands',
       'Finland', 'Iceland', 'Ireland', 'Isle of Man', 'Latvia',
       'Lithuania', 'Norway', 'Sweden', 'United Kingdom',
       'Southern Europe', 'Albania', 'Andorra', 'Bosnia and Herzegovina',
       'Croatia', 'Gibraltar', 'Greece', 'Holy See', 'Italy', 'Malta',
       'Montenegro', 'North Macedonia', 'Portugal', 'San Marino',
       'Serbia', 'Slovenia', 'Spain', 'Western Europe', 'Austria',
       'Belgium', 'France', 'Germany', 'Liechtenstein', 'Luxembourg',
       'Monaco', 'Netherlands', 'Switzerland', 'NORTHERN AMERICA',
       'Bermuda', 'Canada', 'Greenland', 'Saint Pierre and Miquelon',
       'United States of America'], dtype=object)
```

Filtering the data to only contain data from South East Asian countries and

verifying the change

In [37]:

```
seapopu_df = popu_df[popu_df['Country'].isin(sea_countries)]
seapopu_df.reset_index(inplace=True)
seapopu_df
```

Out[37]:

	index	Index	Variant	Country	Notes	Country code	Type	Parent code	1950	1951	...	2011
0	136	137	Estimates	Brunei Darussalam	NaN	96	Country/Area	920	48	51	...	394
1	137	138	Estimates	Cambodia	NaN	116	Country/Area	920	4 433	4 538	...	14 541
2	138	139	Estimates	Indonesia	NaN	360	Country/Area	920	69 543	70 849	...	245 116
3	139	140	Estimates	Lao People's Democratic Republic	NaN	418	Country/Area	920	1 683	1 723	...	6 348
4	140	141	Estimates	Malaysia	13	458	Country/Area	920	6 110	6 271	...	28 651
5	141	142	Estimates	Myanmar	NaN	104	Country/Area	920	17 780	18 104	...	50 991
6	142	143	Estimates	Philippines	NaN	608	Country/Area	920	18 580	19 247	...	95 570
7	143	144	Estimates	Singapore	NaN	702	Country/Area	920	1 022	1 068	...	5 264
8	144	145	Estimates	Thailand	NaN	764	Country/Area	920	20 710	21 263	...	67 518
9	145	146	Estimates	Timor-Leste	NaN	626	Country/Area	920	415	419	...	1 113
10	146	147	Estimates	Viet Nam	NaN	704	Country/Area	920	24 810	25 365	...	88 871

11 rows × 79 columns

In [38]:

```
gdppopu_df = seagdp_df.merge(seapopu_df,on='Country')
gdppopu_df
```

Out[38]:

	index_x	Country	GDP	index_y	Index	Variant	Notes	Country code	Type	Parent code	...
0	16	Indonesia	1,119,191	138	139	Estimates	NaN	360	Country/Area	920	...
1	22	Thailand	543,650	144	145	Estimates	NaN	764	Country/Area	920	...

	index_x	Country	GDP	index_y	Index	Variant	Notes	Country code	Type	Parent code	...
2	33	Philippines	376,796	142	143	Estimates	NaN	608	Country/Area	920	...
3	34	Singapore	372,063	143	144	Estimates	NaN	702	Country/Area	920	...
4	36	Malaysia	364,702	140	141	Estimates	13	458	Country/Area	920	...
5	45	Viet Nam	261,921	146	147	Estimates	NaN	704	Country/Area	920	...
6	71	Myanmar	76,086	141	142	Estimates	NaN	104	Country/Area	920	...
7	103	Cambodia	27,089	137	138	Estimates	NaN	116	Country/Area	920	...
8	117	Lao People's Democratic Republic	18,174	139	140	Estimates	NaN	418	Country/Area	920	...
9	133	Brunei Darussalam	13,469	136	137	Estimates	NaN	96	Country/Area	920	...
10	182	Timor-Leste	1,674	145	146	Estimates	NaN	626	Country/Area	920	...

11 rows × 81 columns

Retaining the columns that will only be used later on in the assignment

Explanation: After thoroughly reading through the assignment specification and looking through the csv files, I've decided that only the following columns below will be needed.

In [39]:

```
gdppopu_df = gdppopu_df[['Country', 'GDP', '2019']]
gdppopu_df
```

Out[39]:

	Country	GDP	2019
0	Indonesia	1,119,191	270 626
1	Thailand	543,650	69 626
2	Philippines	376,796	108 117
3	Singapore	372,063	5 804
4	Malaysia	364,702	31 950
5	Viet Nam	261,921	96 462
6	Myanmar	76,086	54 045
7	Cambodia	27,089	16 487

	Country	GDP	2019
8	Lao People's Democratic Republic	18,174	7 169
9	Brunei Darussalam	13,469	433
10	Timor-Leste	1,674	1 293

Converting GDP to numerical form for usage in calculations

Removing the commas and whitespaces from the numbers

In [40]:

```
gdppopu_df = gdppopu_df.assign(**{
    'GDP':gdppopu_df['GDP'].apply(lambda x : x.replace(',','')),
    '2019':gdppopu_df['2019'].apply(lambda x : x.replace(' ',''))
})
gdppopu_df
```

Out[40]:

	Country	GDP	2019
0	Indonesia	1119191	270626
1	Thailand	543650	69626
2	Philippines	376796	108117
3	Singapore	372063	5804
4	Malaysia	364702	31950
5	Viet Nam	261921	96462
6	Myanmar	76086	54045
7	Cambodia	27089	16487
8	Lao People's Democratic Republic	18174	7169
9	Brunei Darussalam	13469	433
10	Timor-Leste	1674	1293

Converting the numbers from String into int

In [41]:

```
gdppopu_df = gdppopu_df.assign(**{
    'GDP':gdppopu_df['GDP'].astype(int),
    '2019':gdppopu_df['2019'].astype(int)
})
gdppopu_df
```

Out[41]:

	Country	GDP	2019
0	Indonesia	1119191	270626
1	Thailand	543650	69626
2	Philippines	376796	108117
3	Singapore	372063	5804

	Country	GDP	2019
4	Malaysia	364702	31950
5	Viet Nam	261921	96462
6	Myanmar	76086	54045
7	Cambodia	27089	16487
8	Lao People's Democratic Republic	18174	7169
9	Brunei Darussalam	13469	433
10	Timor-Leste	1674	1293

Renaming the column '2019' to 'Population'

In [42]:

```
gdppopu_df.rename(columns={'2019':'Population'}, inplace=True)
gdppopu_df
```

Out[42]:

	Country	GDP	Population
0	Indonesia	1119191	270626
1	Thailand	543650	69626
2	Philippines	376796	108117
3	Singapore	372063	5804
4	Malaysia	364702	31950
5	Viet Nam	261921	96462
6	Myanmar	76086	54045
7	Cambodia	27089	16487
8	Lao People's Democratic Republic	18174	7169
9	Brunei Darussalam	13469	433
10	Timor-Leste	1674	1293

Calculation for Per Capita GDP, by multiplying the population (in thousands) by 1000 for it to be equal to GDP (in millions)

In [43]:

```
gdppopu_df['PerCapitaGDP'] = gdppopu_df['GDP'] / gdppopu_df['Population'] * 1000
gdppopu_df
```

Out[43]:

	Country	GDP	Population	PerCapitaGDP
0	Indonesia	1119191	270626	4135.563471
1	Thailand	543650	69626	7808.146382
2	Philippines	376796	108117	3485.076352
3	Singapore	372063	5804	64104.583046

	Country	GDP	Population	PerCapitaGDP
4	Malaysia	364702	31950	11414.773083
5	Viet Nam	261921	96462	2715.276482
6	Myanmar	76086	54045	1407.826811
7	Cambodia	27089	16487	1643.052102
8	Lao People's Democratic Republic	18174	7169	2535.081601
9	Brunei Darussalam	13469	433	31106.235566
10	Timor-Leste	1674	1293	1294.663573

Merging all the dataframes together to form one complete dataframe

In [44]:

```
merged_df = seaagg_df.merge(gdppopu_df, on=['Country'])
merged_df
```

Out[44]:

	Country	Status	Max Life Expectancy	Adult Mortality	Mean BMI	Mean Income Composition of Resources	Mean Schooling	Mean Life Expectancy	C
0	Brunei Darussalam	Developing	78.3	1073.0	29.71875	0.839375	14.10625	76.48750	13
1	Cambodia	Developing	68.7	3142.0	15.36250	0.491937	9.87500	64.34375	27
2	Indonesia	Developing	69.1	2665.0	19.95625	0.641437	11.61250	67.55625	1119
3	Lao People's Democratic Republic	Developing	65.7	3155.0	14.36250	0.515625	9.23125	62.38125	18
4	Malaysia	Developing	75.0	1897.0	29.16875	0.749125	12.56250	73.75625	364
5	Myanmar	Developing	66.6	2469.0	17.12500	0.488250	8.32500	64.20000	76
6	Philippines	Developing	68.5	3487.0	19.18750	0.650438	11.54375	67.57500	376
7	Singapore	Developed	87.0	992.0	25.90625	0.866875	13.98125	81.47500	372
8	Thailand	Developing	74.9	2566.0	21.59375	0.694688	12.55000	73.08125	543
9	Timor-Leste	Developing	68.3	2726.0	14.55000	0.517625	10.70000	64.75625	1
10	Viet Nam	Developing	76.0	2025.0	11.18750	0.627063	11.51250	74.77500	261

Generate descriptive statistics for the merged dataframe

In [45]:

```
merged_df.describe()
```

Out[45]:

	Max Life Expectancy	Adult Mortality	Mean BMI	Mean Income Composition of Resources	Mean Schooling	Mean Life Expectancy	GDP	Popu
count	11.000000	11.000000	11.000000	11.000000	11.000000	11.000000	1.100000e+01	11.0
mean	72.554545	2381.545455	19.828977	0.643858	11.454545	70.035227	2.886195e+05	60182.9
std	6.393804	815.123103	6.206451	0.134771	1.835239	6.190168	3.347960e+05	79711.3
min	65.700000	992.000000	11.187500	0.488250	8.325000	62.381250	1.674000e+03	433.0
25%	68.400000	1961.000000	14.956250	0.516625	10.287500	64.550000	2.263150e+04	6486.5
50%	69.100000	2566.000000	19.187500	0.641437	11.543750	67.575000	2.619210e+05	31950.0
75%	75.500000	2934.000000	23.750000	0.721906	12.556250	74.265625	3.744295e+05	83044.0
max	87.000000	3487.000000	29.718750	0.866875	14.106250	81.475000	1.119191e+06	270626.0

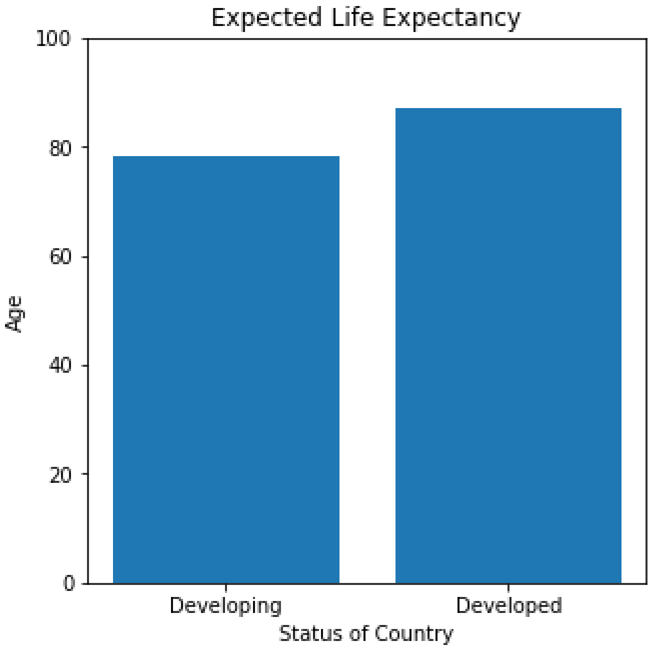
Solving The Questions

Question 1

Approach: Since we want to compare numerical data (Age) to categorical data (Status of Country), I have decided to use a vertical bar chart to visualise the data

In [46]:

```
plt.figure(figsize=(5,5))
plt.bar(merged_df['Status'], merged_df['Max Life Expectancy'])
plt.title('Expected Life Expectancy')
plt.xlabel('Status of Country')
plt.ylabel('Age')
plt.ylim(0, 100)
plt.show()
```



From the bar chart, it can be concluded that the expected life expectancy in developed countries is greater than in developing countries.

This could be due to the health care provided by the government in the different countries

Question 2

Approach for first problem:

Due to the big disparities in the numbers, I have decided to divide the data in the Adult Mortality column by 100 and divide the data in the Population column by 1000 so that the numbers are closer to the range of the data in the Mean Life Expectancy column

Approach for second problem:

In this case, since we will be comparing 3 instances of numerical data (Adult Mortality, Population and Mean Life Expectancy) to categorical data (Country), I've decided to use a stacked horizontal bar chart as I have found that the visualisation is most easy to understand this way as compared to using a basic bar graph. This is because attempting to visualise this data using a basic bar graph will result in a very messy and overly large bar graph

In [47]:

```
merged_df['Adult Mortality per 100'] = merged_df['Adult Mortality'] / 100
merged_df['Population per 1000'] = merged_df['Population'] / 1000
merged_df
```

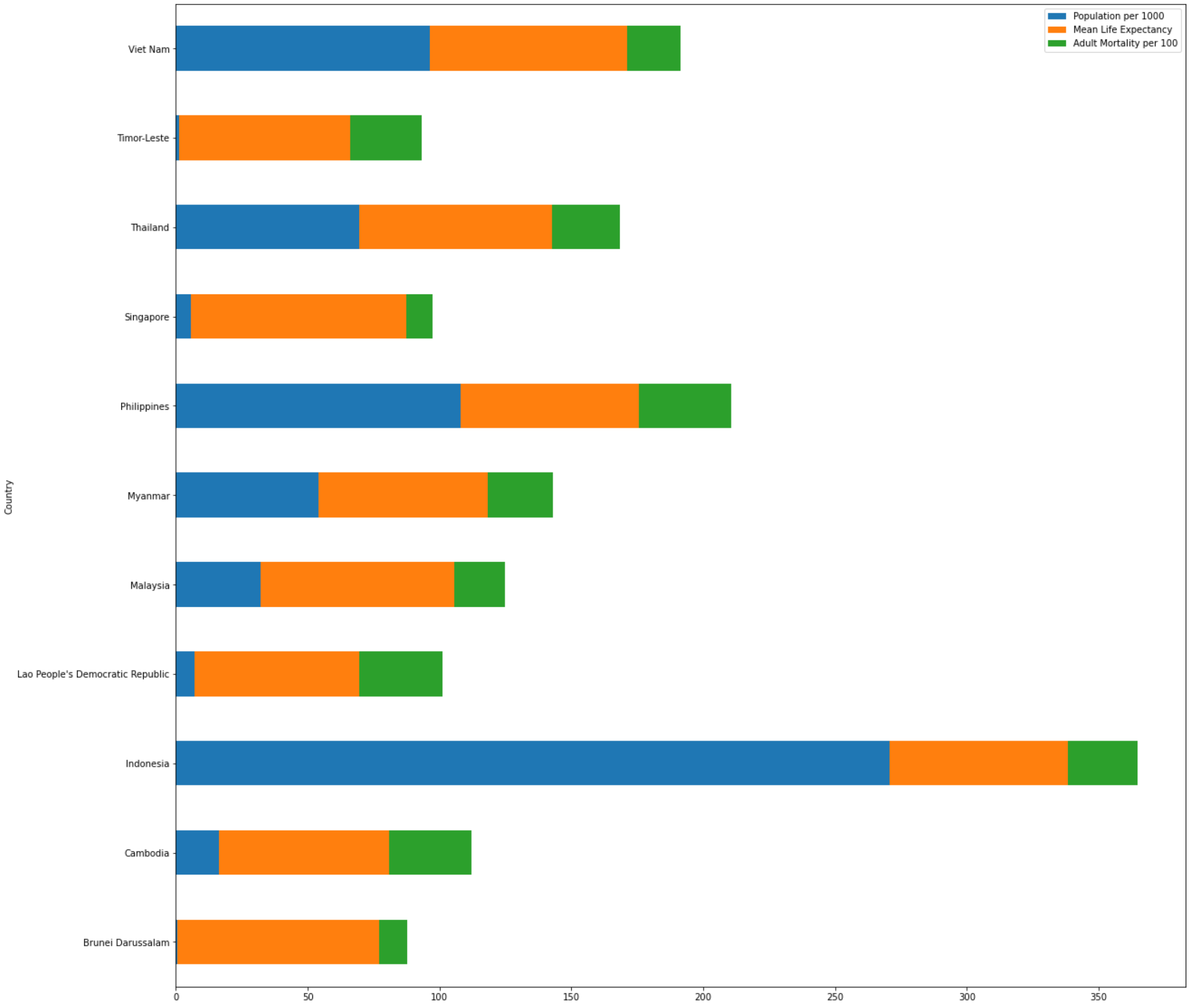
Out[47]:

	Country	Status	Max Life Expectancy	Adult Mortality	Mean BMI	Mean Income Composition of Resources	Mean Schooling	Mean Life Expectancy	(
0	Brunei Darussalam	Developing	78.3	1073.0	29.71875	0.839375	14.10625	76.48750	13

	Country	Status	Max Life Expectancy	Adult Mortality	Mean BMI	Mean Income Composition of Resources	Mean Schooling	Mean Life Expectancy	C
1	Cambodia	Developing	68.7	3142.0	15.36250	0.491937	9.87500	64.34375	27
2	Indonesia	Developing	69.1	2665.0	19.95625	0.641437	11.61250	67.55625	1119
3	Lao People's Democratic Republic	Developing	65.7	3155.0	14.36250	0.515625	9.23125	62.38125	18
4	Malaysia	Developing	75.0	1897.0	29.16875	0.749125	12.56250	73.75625	364
5	Myanmar	Developing	66.6	2469.0	17.12500	0.488250	8.32500	64.20000	76
6	Philippines	Developing	68.5	3487.0	19.18750	0.650438	11.54375	67.57500	376
7	Singapore	Developed	87.0	992.0	25.90625	0.866875	13.98125	81.47500	372
8	Thailand	Developing	74.9	2566.0	21.59375	0.694688	12.55000	73.08125	543
9	Timor-Leste	Developing	68.3	2726.0	14.55000	0.517625	10.70000	64.75625	1
10	Viet Nam	Developing	76.0	2025.0	11.18750	0.627063	11.51250	74.77500	261

In [48]:

```
merged_df.plot.barh(
    x='Country',
    y=['Population per 1000', 'Mean Life Expectancy', 'Adult Mortality per 100'],
    figsize=(20,20),
    stacked=True
)
plt.show()
```



The data used for the graph may be misleading because there's a large disparity in the data for Population between the countries.

Some countries with a lower population have similar adult mortality to those with a much high population.

Question 3

Approach: I will first extract the data regarding Singapore then use it to plot the line graphs. For the second line graph, I will need to create a legend for improved visualisation

Recall non-aggregated data from "LifeExpectancyData-v2.csv"

In [49]:

```
led_df
```

Out[49]:

	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	Mea
0	Afghanistan	2015	Developing	65.0	62	263.0	19.1	0.01	65.0	
1	Afghanistan	2014	Developing	59.9	64	271.0	18.6	0.01	62.0	

	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	Mea
2	Afghanistan	2013	Developing	59.9	66	268.0	18.1	0.01	64.0	
3	Afghanistan	2012	Developing	59.5	69	272.0	17.6	0.01	67.0	
4	Afghanistan	2011	Developing	59.2	71	275.0	17.2	0.01	68.0	
...	...	...	...	...	...	...	...	...	...	
2933	Zimbabwe	2004	Developing	44.3	27	723.0	27.1	4.36	68.0	
2934	Zimbabwe	2003	Developing	44.5	26	715.0	26.7	4.06	7.0	
2935	Zimbabwe	2002	Developing	44.8	25	73.0	26.3	4.43	73.0	
2936	Zimbabwe	2001	Developing	45.3	25	686.0	25.9	1.72	76.0	
2937	Zimbabwe	2000	Developing	46.0	24	665.0	25.5	1.68	79.0	

2938 rows × 15 columns

Extract data related to Singapore

In [50]:

```
sing_df = led_df[led_df['Country']=='Singapore'].reset_index()
sing_df
```

Out[50]:

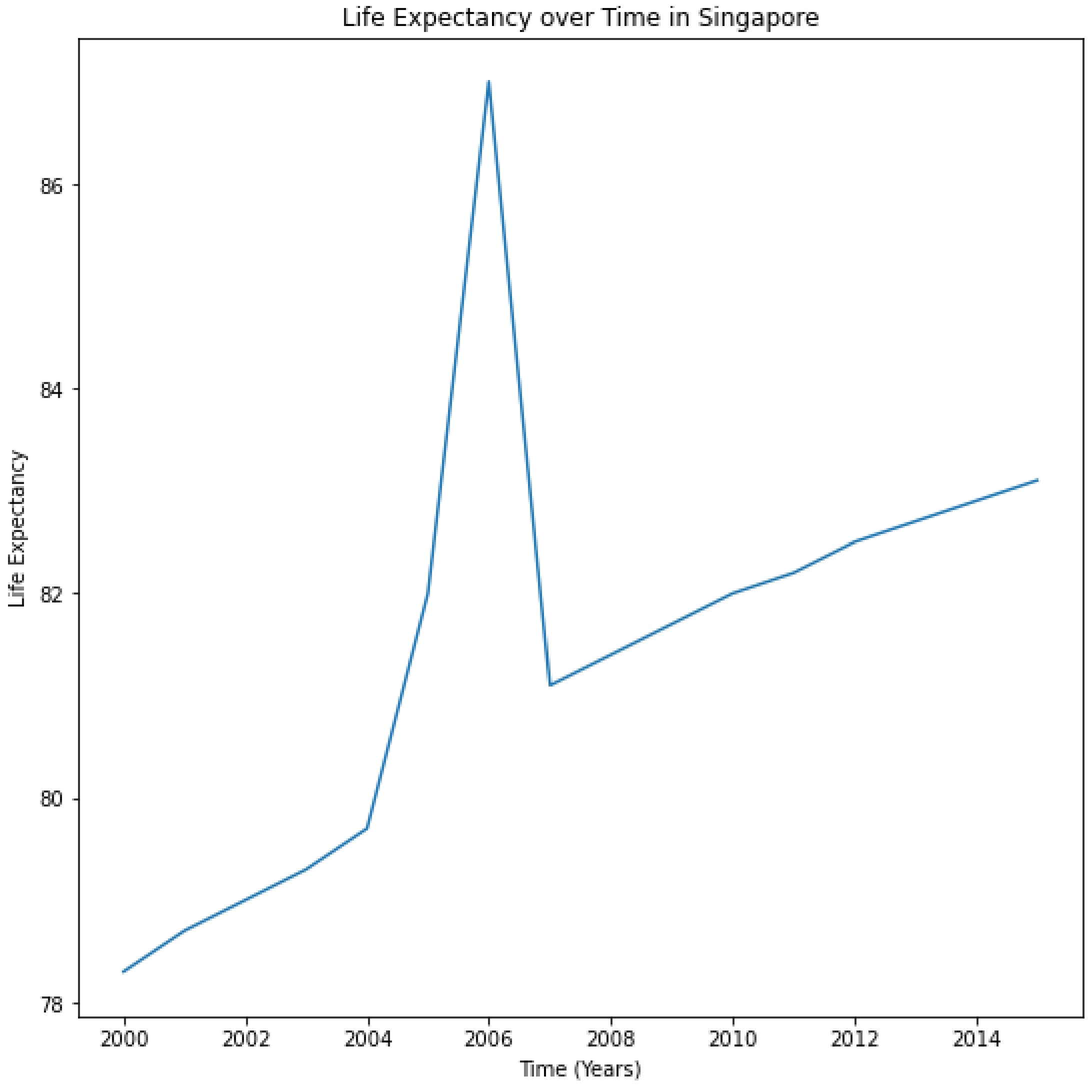
	index	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B	M
0	2313	Singapore	2015	Developed	83.1	0	55.0	33.2	1.79	96.0	
1	2314	Singapore	2014	Developed	82.9	0	56.0	32.9	1.83	96.0	
2	2315	Singapore	2013	Developed	82.7	0	57.0	32.7	1.83	97.0	
3	2316	Singapore	2012	Developed	82.5	0	59.0	32.4	1.89	97.0	
4	2317	Singapore	2011	Developed	82.2	0	6.0	32.1	1.80	96.0	
5	2318	Singapore	2010	Developed	82.0	0	61.0	31.8	1.84	96.0	
6	2319	Singapore	2009	Developed	81.7	0	62.0	31.5	1.73	96.0	
7	2320	Singapore	2008	Developed	81.4	0	64.0	31.2	1.70	97.0	
8	2321	Singapore	2007	Developed	81.1	0	65.0	3.9	1.60	96.0	
9	2322	Singapore	2006	Developed	87.0	0	66.0	3.5	1.55	95.0	
10	2323	Singapore	2005	Developed	82.0	0	69.0	3.2	1.49	96.0	
11	2324	Singapore	2004	Developed	79.7	0	71.0	29.9	1.45	94.0	
12	2325	Singapore	2003	Developed	79.3	0	73.0	29.6	1.43	95.0	
13	2326	Singapore	2002	Developed	79.0	0	74.0	29.2	2.16	95.0	
14	2327	Singapore	2001	Developed	78.7	0	76.0	28.9	2.08	95.0	

	index	Country	Year	Status	Life Expectancy	Infant Deaths	Adult Mortality	BMI	Alcohol Consumption	Hepatitis B
15	2328	Singapore	2000	Developed	78.3	0	78.0	28.5	2.03	97.0

Plot a line graph on Life Expectancy over Time

In [51]:

```
plt.figure(figsize=(9,9))
plt.plot(sing_df['Year'], sing_df['Life Expectancy'])
plt.title('Life Expectancy over Time in Singapore')
plt.xlabel('Time (Years)')
plt.ylabel('Life Expectancy')
plt.show()
```

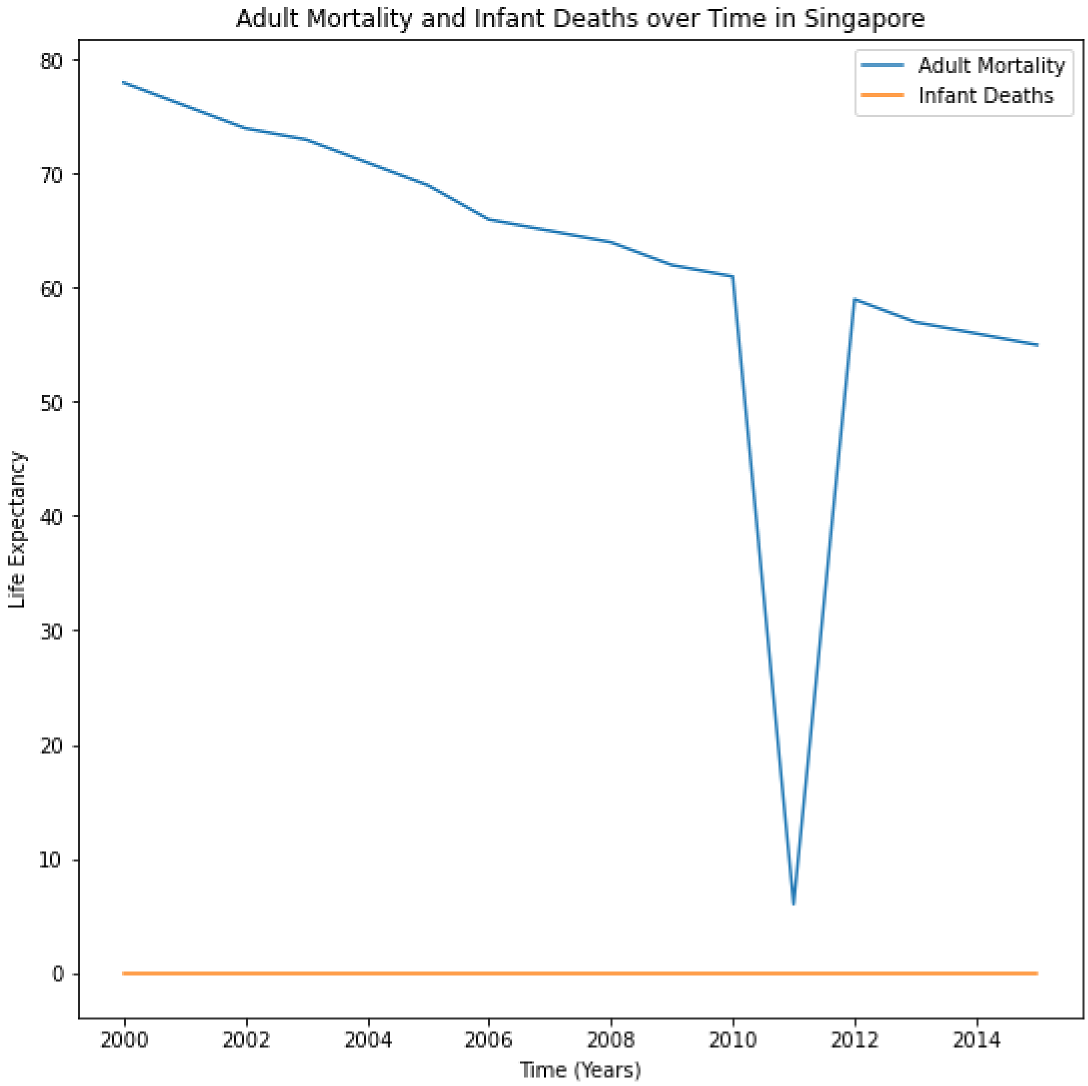


From the result of the graph plotted, I can conclude that the graph would be quite useful in calculating the rate of increase in life expectancy over time as long as the large outlier in the year 2006 is ignored.

Plot a line graph of Adult Mortality and Infant Deaths over Time

In [52]:

```
plt.figure(figsize=(9,9))
plt.plot(sing_df['Year'], sing_df['Adult Mortality'], label='Adult Mortality')
plt.plot(sing_df['Year'], sing_df['Infant Deaths'], label='Infant Deaths')
plt.legend(loc='upper right')
plt.title('Adult Mortality and Infant Deaths over Time in Singapore')
plt.xlabel('Time (Years)')
plt.ylabel('Life Expectancy')
plt.show()
```



From the result of the graph plotted, I can conclude that the graph would be useful to calculate the decreasing rate of adult mortality over time as long as the outlier in the year 2011 is ignored. This graph would also be useful in the circumstance that the average infant deaths per year needs to be calculated.

The greater the number of infant mortality, the lower the life expectancy. Life expectancy will decrease or increase in accordance with whether the adult passess away at an age lower or greater than the average life expectancy

---

Conclusion

In conclusion, what I can draw from what I've done in this assignment is that there are generally many different ways to approach data and many different ways to visualise it. Data is quite difficult to handle due to possible discrepancies in the data and the various possible data types present that a programmer would have to work with. However, the availability of the many different types of graphs makes presenting data a much more fun and interactive experience.