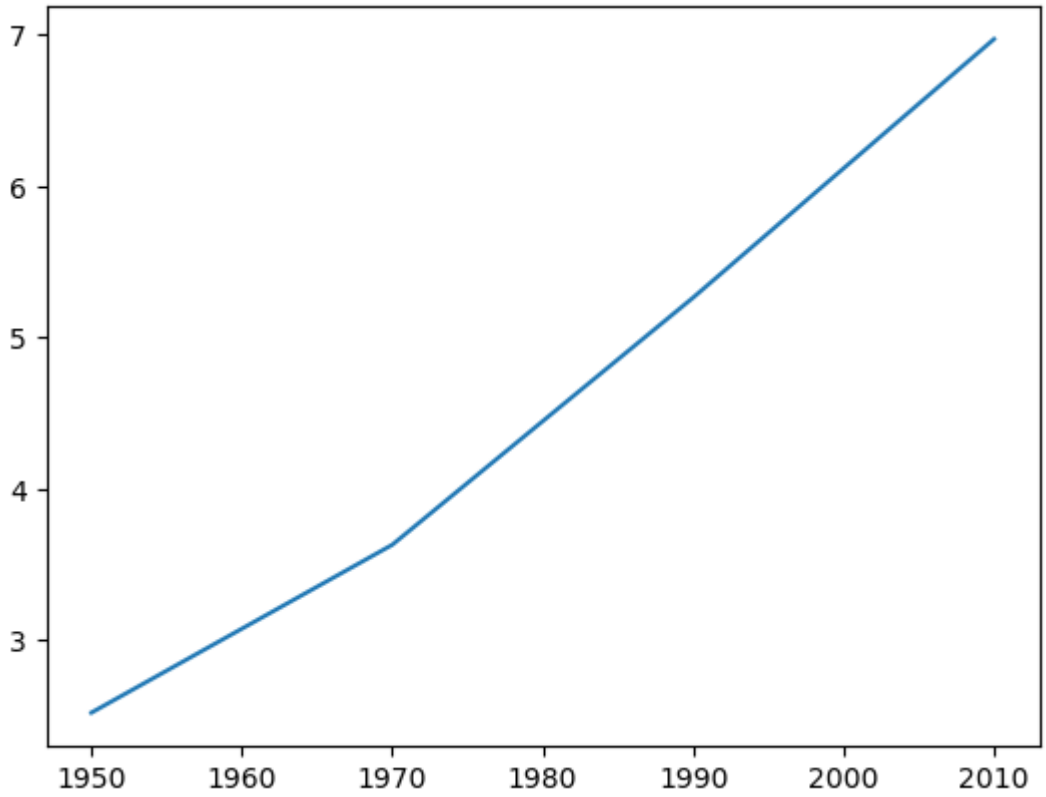
In [1]:

```
import pandas as pd
```

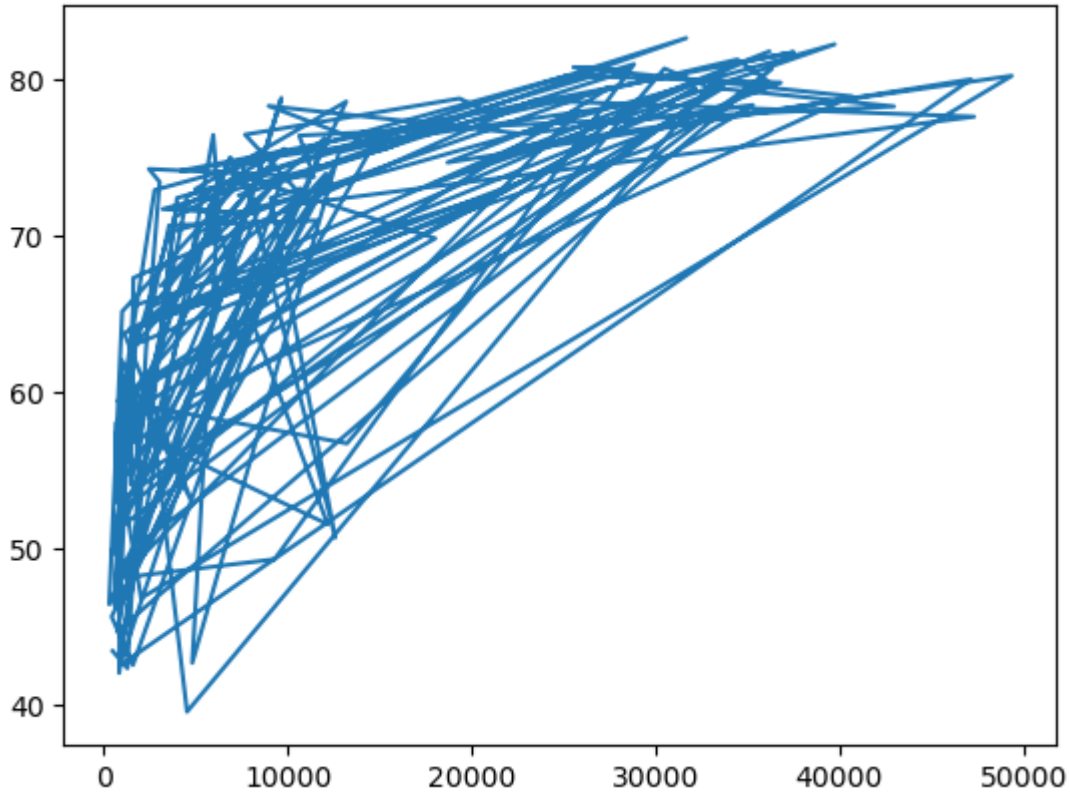
In [3]:

```
import matplotlib.pyplot as plt
%matplotlib inline
year = [1950, 1970, 1990, 2010]
pop = [2.519, 3.629, 5.263, 6.972 ]
plt.plot(year, pop)
plt.show()

```



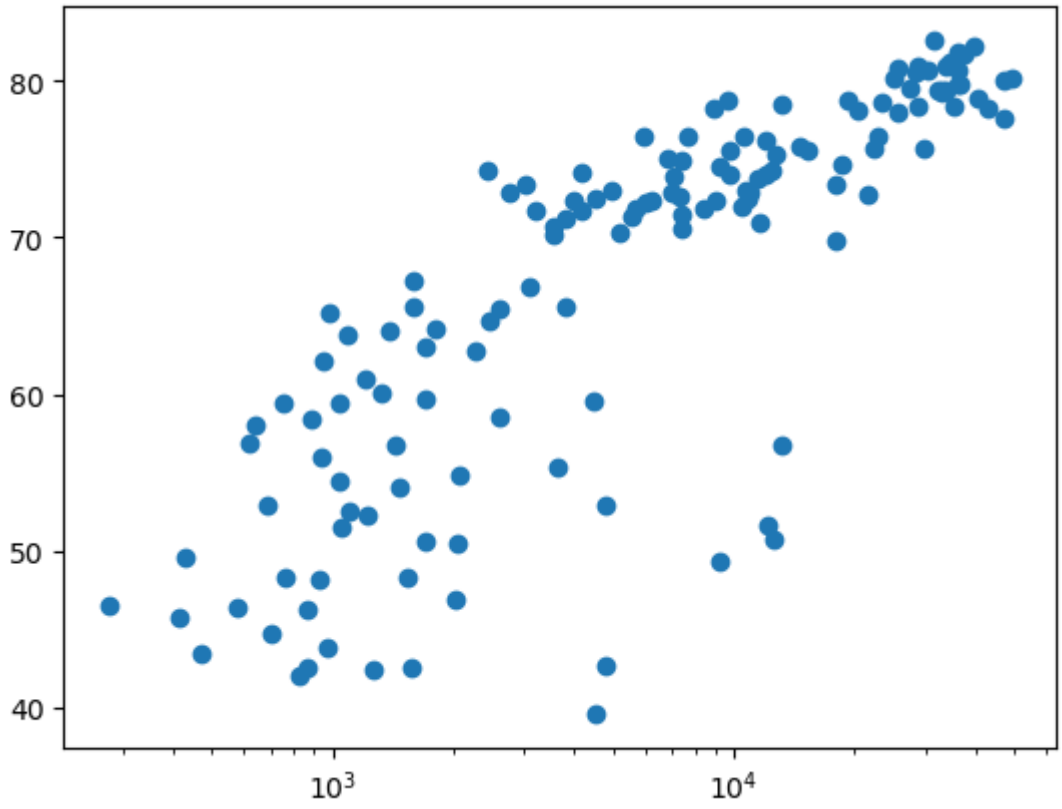
In [14]:

```
gdp_cap =  [974.5803384, 5937.029525999999, 6223.367465, 4797.231267, 12779.
life_exp = [43.828, 76.423, 72.301, 42.731, 75.32, 81.235, 79.829, 75.635, 6
plt.plot(gdp_cap, life_exp)
plt.show()
```



In [12]:

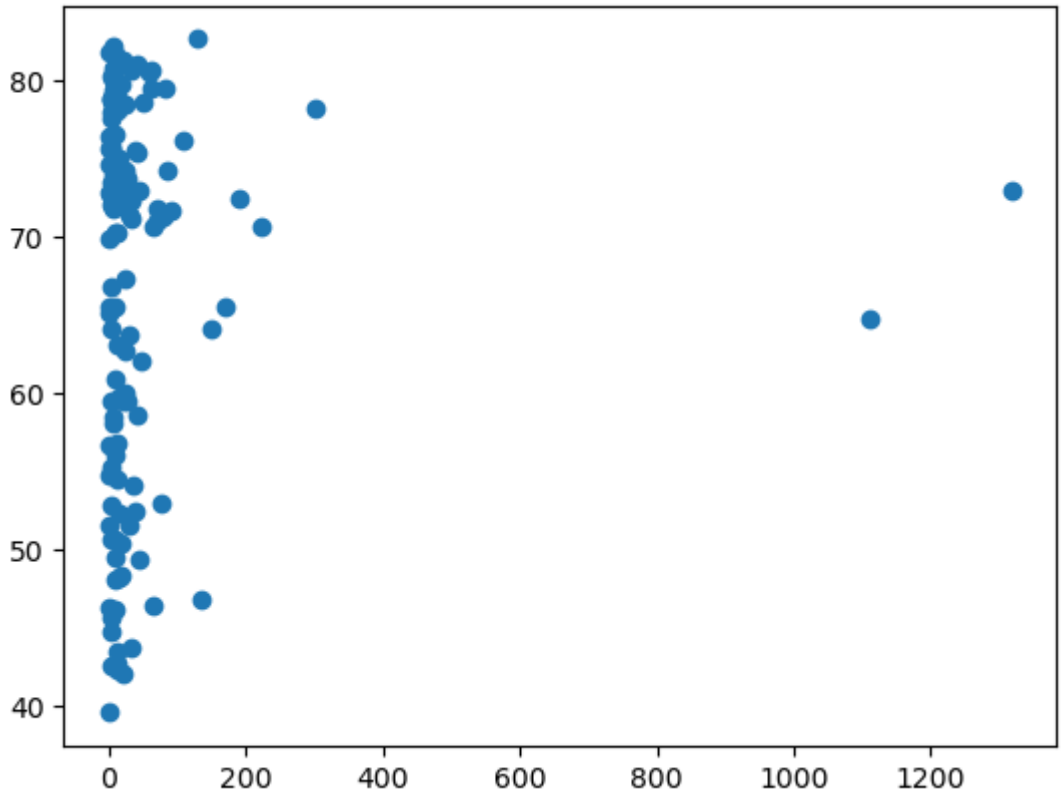
```
gdp_cap =  [974.5803384, 5937.029525999999, 6223.367465, 4797.231267, 12779.
life_exp = [43.828, 76.423, 72.301, 42.731, 75.32, 81.235, 79.829, 75.635, 6
plt.scatter(gdp_cap, life_exp)
plt.xscale('log') #log scale
plt.show()
```



Exercises 1

In [36]:

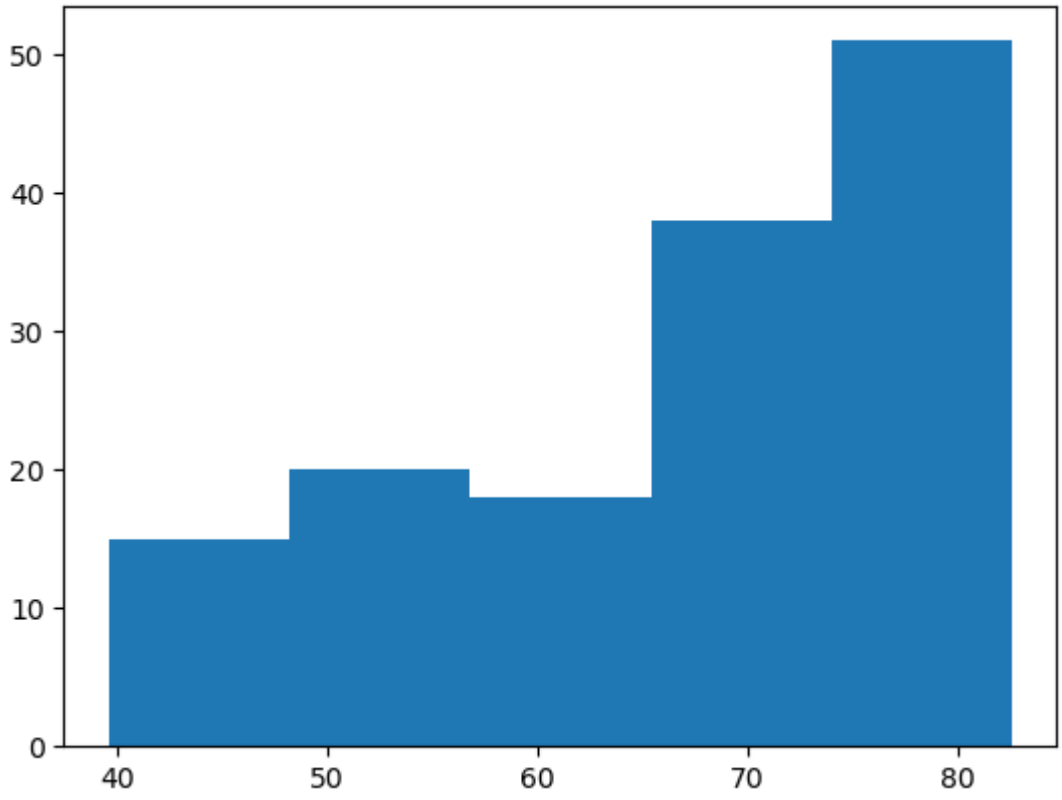
```
#Exercise 1
pop = [31.889923, 3.600523, 33.333216, 12.420476, 40.301927, 20.434176, 8.19
life_exp = [43.828, 76.423, 72.301, 42.731, 75.32, 81.235, 79.829, 75.635, 6
plt.scatter(pop, life_exp)
plt.show()
```



Exercise 2

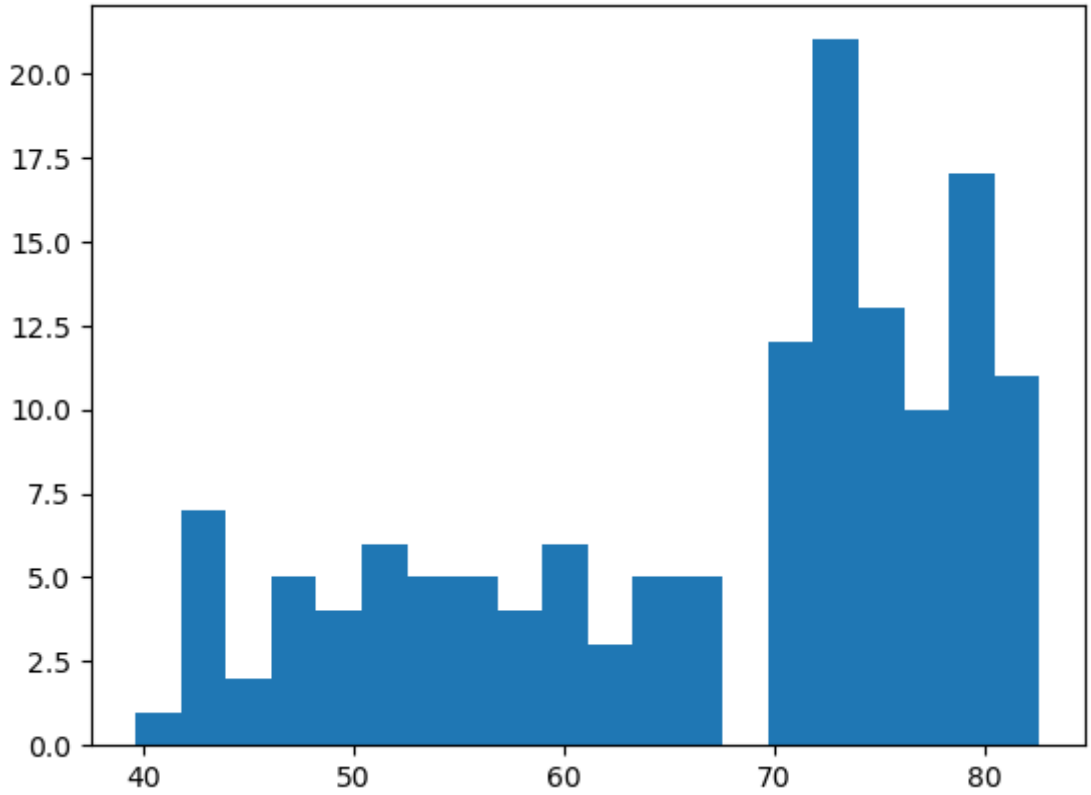
In [21]:

```
life_exp = [43.828, 76.423, 72.301, 42.731, 75.32, 81.235, 79.829, 75.635, 64.0
plt.hist(life_exp,bins=5)
plt.show()
```



In [25]:

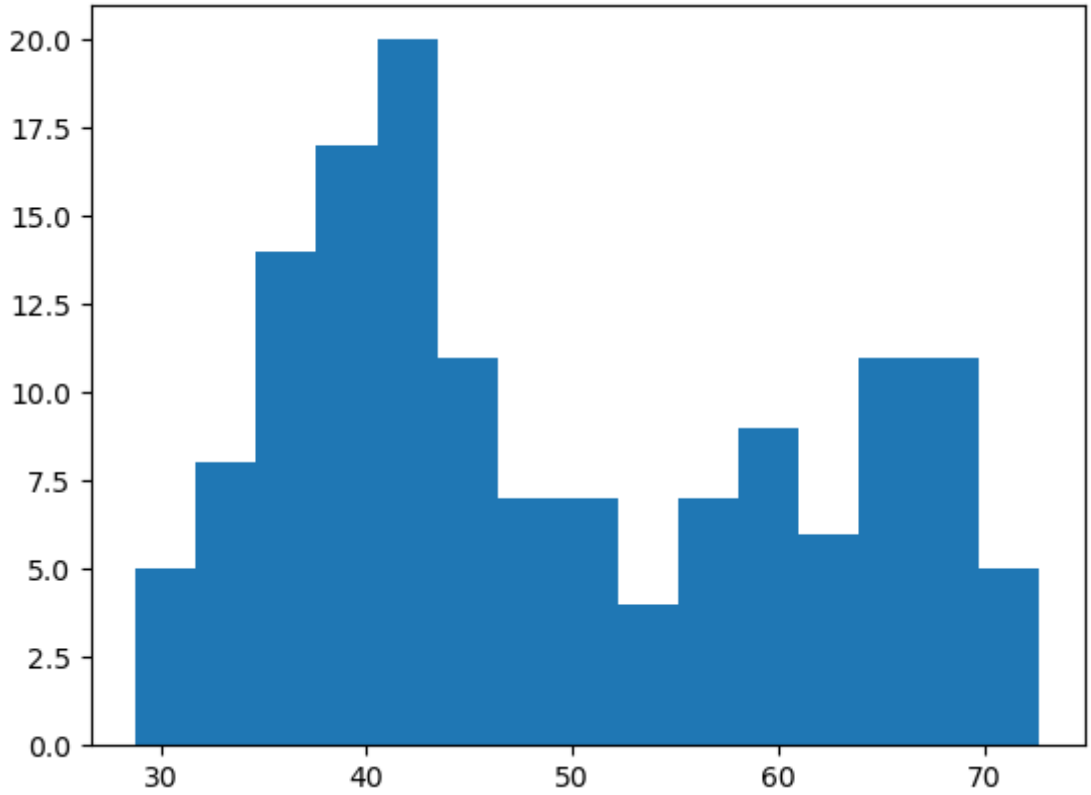
```
1
2 life_exp = [43.828, 76.423, 72.301, 42.731, 75.32, 81.235, 79.829, 75.635, 6
3 plt.hist(life_exp,bins=20)
4 plt.show()
```



Exercise 3 Ans

In [29]:

```
#Exercise 3
life_exp1950 = [28.8, 55.23, 43.08, 30.02, 62.48, 69.12, 66.8, 50.94, 37.48,
plt.hist(life_exp1950,bins=15)
plt.show()
```



The histogram shows that the life expectancy in 2007 is much higher than in 1950. The bin with the highest frequency in the histogram for life_exp1950 is the bin from 40 to 45 years old, with a frequency of 35. The bin with the highest frequency in the histogram for life_exp is the bin from 70 to 75 years old, with a frequency of 43. This shows that life expectancy has increased significantly since 1950.

Exercise 4 Ans

To visually assess if the grades on your exam follow a particular distribution, you would typically use a histogram plot. A histogram provides a visual representation of the distribution of a dataset by dividing it into bins and displaying the frequency or count of values within each bin.

Exercise 5

To visually assess if longer answers on exam questions lead to higher grades, you would typically use a scatter plot. A scatter plot is suitable for comparing two continuous variables and allows you to examine the relationship or correlation between them.

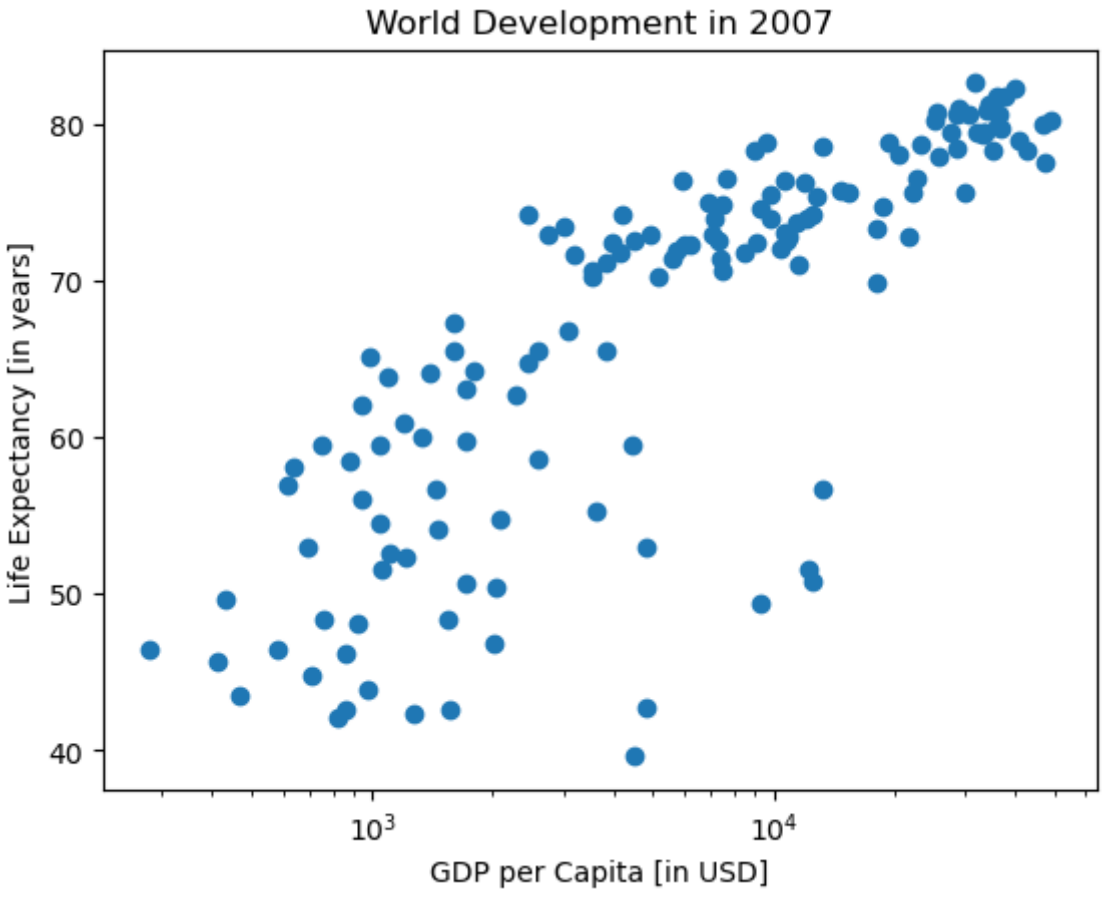
Exercise 6

In [31]:

```
# Basic scatter plot, log scale
plt.scatter(gdp_cap, life_exp)
plt.xscale('log')
# Strings
xlab = 'GDP per Capita [in USD]'
ylab = 'Life Expectancy [in years]'
title = 'World Development in 2007'
# Add axis labels
plt.xlabel(xlab)
plt.ylabel(ylab)
plt.title(title)
```

Out[31]:

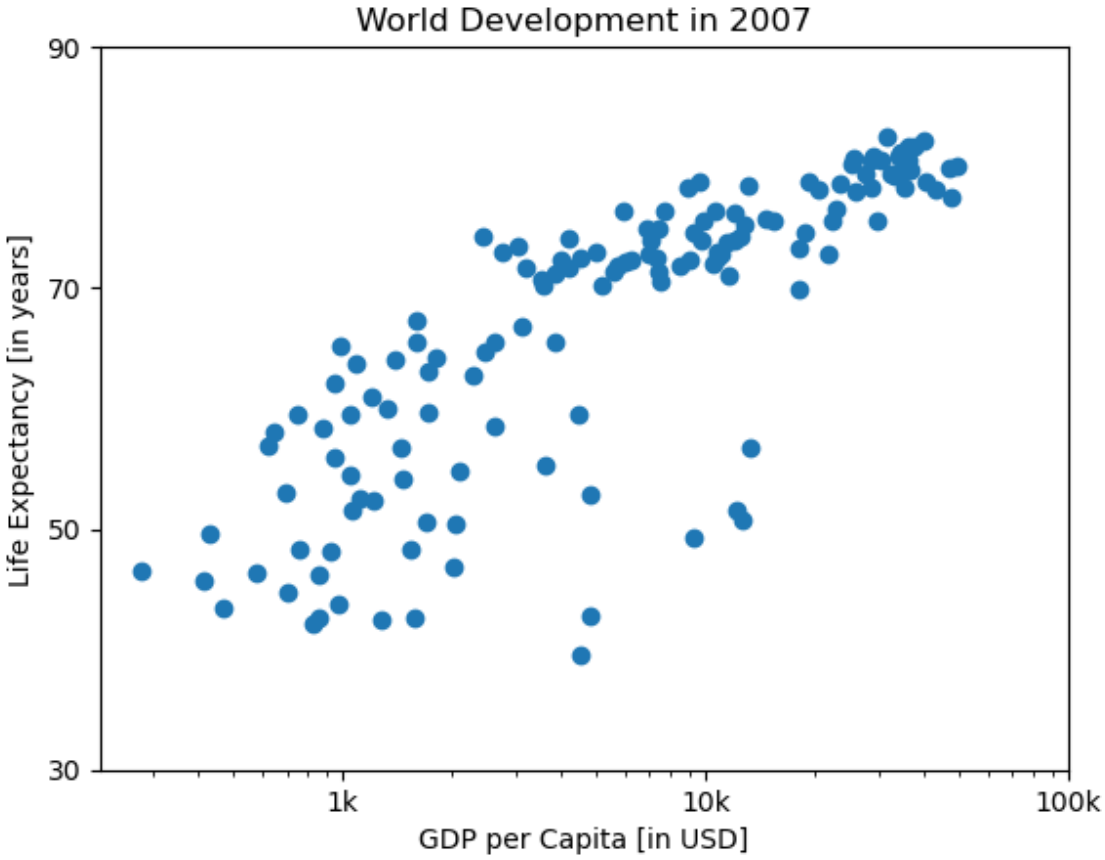
```
Text(0.5, 1.0, 'World Development in 2007')
```



Exercise 7

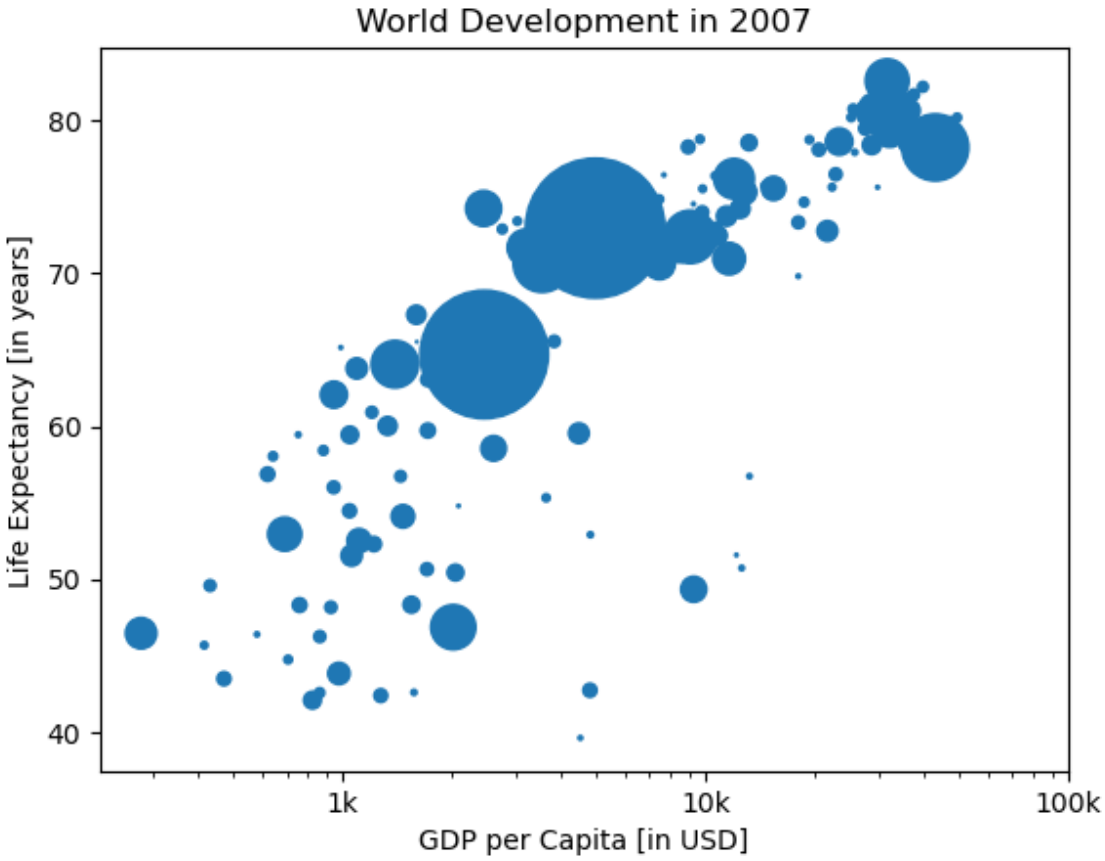
In [37]:

```
# Scatter plot
plt.scatter(gdp_cap, life_exp)
# Previous customizations
plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in years]')
plt.title('World Development in 2007')
# Definition of tick_val and tick_lab
tick_val = [1000,10000,100000]
tick_lab = ['1k','10k','100k']
# Adapt the ticks on the x-axis
plt.xticks(tick_val,tick_lab)
plt.yticks([30, 50, 70, 90]) #setting tick
# After customizing, display the plot
plt.show()

```



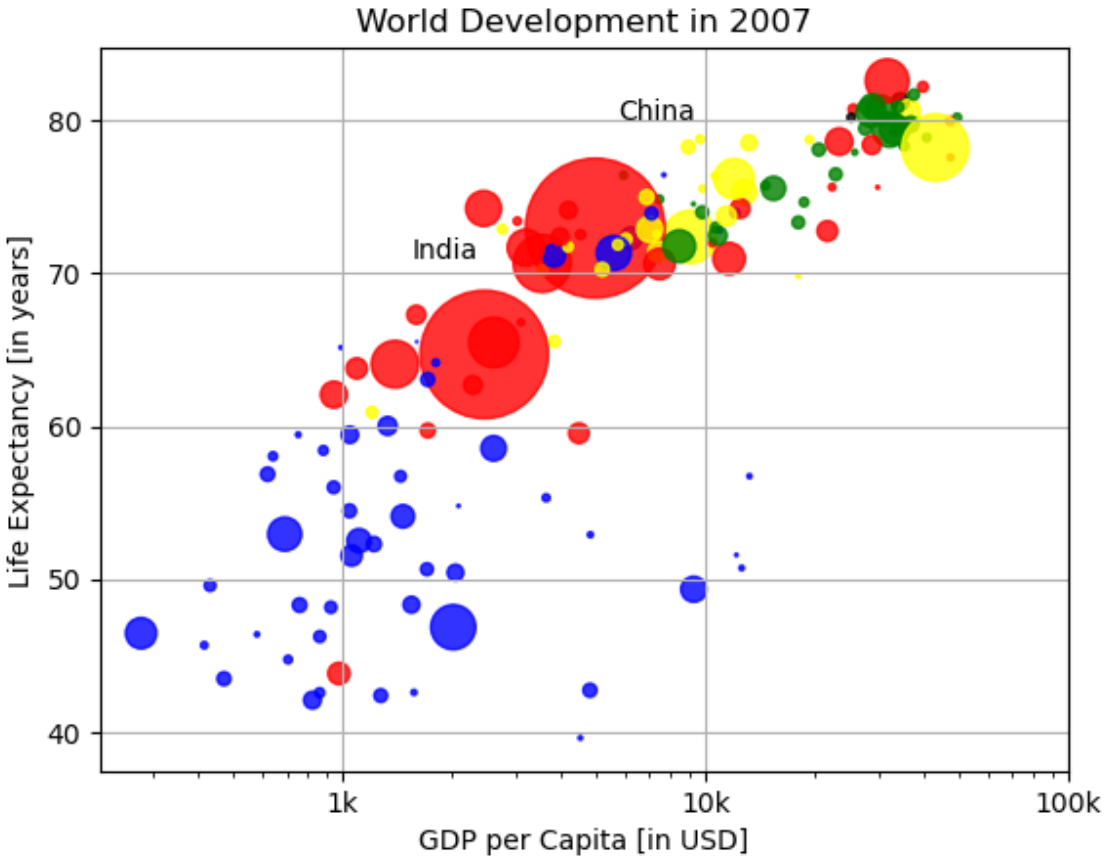
In [38]:

```
# Import numpy as np
import numpy as np
# Store pop as a numpy array: np_pop
np_pop = np.array(pop)
# Double np_pop
np_pop = np_pop * 2
# Update: set s argument to np_pop
plt.scatter(gdp_cap, life_exp, s = np_pop)
# Previous customizations
plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in years]')
plt.title('World Development in 2007')
plt.xticks([1000, 10000, 100000],['1k', '10k', '100k'])
# Display the plot
plt.show()

```



In [40]:

```
col = ['red', 'green', 'blue', 'blue', 'yellow', 'black', 'green', 'red', 're
# Scatter plot
plt.scatter(x = gdp_cap, y = life_exp, s = np.array(pop) * 2, c = col, alpha
# Previous customizations
plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in years]')
plt.title('World Development in 2007')
plt.xticks([1000,10000,100000], ['1k','10k','100k'])
# Additional customizations
plt.text(1550, 71, 'India')
plt.text(5700, 80, 'China')
# Add grid() call
plt.grid(1)
# Show the plot
plt.show()

```



Exercise 8

8) What can you say about the plot? Which one is True?

- The countries in blue, corresponding to Africa, have both low life expectancy and a low GDP per capita.

In [12]:

```
import pandas as pd
brics = pd.read_csv('brics.csv', index_col = 0)
brics.head()
```

Out[12]:

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.40
RU	Russia	Moscow	17.100	143.50
IN	India	New Delhi	3.286	1252.00
CH	China	Beijing	9.597	1357.00
SA	South Africa	Pretoria	1.221	52.98

In [46]:

```
area = brics["area"]
greater_than_8_and_10 = (area > 8) & (area < 10)
brics[greater_than_8_and_10]
```

Out[46]:

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.4
CH	China	Beijing	9.597	1357.0

In [47]:

```
area = brics.loc[:, "area"]

greater_than_8 = area > 8
brics[greater_than_8]
```

Out[47]:

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.4
RU	Russia	Moscow	17.100	143.5
CH	China	Beijing	9.597	1357.0

In [48]:

```
area = brics.iloc[:,2]

greater_than_8 = area > 8
brics[greater_than_8]
```

Out[48]:

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.4
RU	Russia	Moscow	17.100	143.5
CH	China	Beijing	9.597	1357.0

In [25]:

```
import numpy as np
area810 = np.logical_and(brics['area'] > 8, brics['area'] < 10)
brics[area810]
```

Out[25]:

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.4
CH	China	Beijing	9.597	1357.0

Exercise 9

In [33]:

```
population = brics.loc[:,"population"]
greater_than_200 = population >= 200
brics[greater_than_200]
```

Out[33]:

	country	capital	area	population
BR	Brazil	Brasilia	8.516	200.4
IN	India	New Delhi	3.286	1252.0
CH	China	Beijing	9.597	1357.0

Exercise 10

In [49]:

```
population_over_1000 = np.greater(brics["population"], 1000)
area_less_than_8 = np.less(brics["area"], 8)
combined_array = np.logical_or(population_over_1000, area_less_than_8)
capitals = brics[combined_array]["capital"]
brics[combined_array]
```

Out[49]:

	country	capital	area	population
IN	India	New Delhi	3.286	1252.00
CH	China	Beijing	9.597	1357.00
SA	South Africa	Pretoria	1.221	52.98

Exercise 11

In [39]:

```
import pandas as pd
cars = pd.read_csv('cars.csv', index_col = 0)
cars.head(8)
```

Out[39]:

	cars_per_cap	country	drives_right
US	809	United States	True
AUS	731	Australia	False
JAP	588	Japan	False
IN	18	India	False
RU	200	Russia	True
MOR	70	Morocco	True
EG	45	Egypt	True

In [45]:

```
cpc = cars["cars_per_cap"]
many_cars = cpc >= 500
car_maniac = cars[many_cars]
print(car_maniac)
```

```
     cars_per_cap        country  drives_right
US            809  United States          True
AUS           731      Australia         False
JAP           588          Japan         False
```

Exercise 12

In [43]:

```
 cpc100500 = cars["cars_per_cap"]
cpc100_500 = np.logical_and(cpc100500 >= 100, cpc100500 <= 500)
cars[cpc100_500]
```

Out[43]:

	cars_per_cap	country	drives_right
RU	200	Russia	True

Loop over DataFrame.

In [55]:

```
# Import cars data
import pandas as pd
cars = pd.read_csv('cars.csv', index_col = 0)
# Iterate over rows of cars
for key in cars:
    print(key)

```

```
cars_per_cap
country
drives_right
```

In [56]:

```
# Import cars data
import pandas as pd
cars = pd.read_csv('cars.csv', index_col = 0)
# Iterate over rows of cars
for key,value in cars.iterrows():
    print(key)
    print(value)

```

```
US
cars_per_cap              809
country         United States
drives_right             True
Name: US, dtype: object
AUS
cars_per_cap          731
country         Australia
drives_right        False
Name: AUS, dtype: object
JAP
cars_per_cap      588
country         Japan
drives_right    False
Name: JAP, dtype: object
IN
cars_per_cap       18
country         India
drives_right    False
Name: IN, dtype: object
RU
cars_per_cap       200
country         Russia
drives_right      True
Name: RU, dtype: object
MOR
cars_per_cap         70
country         Morocco
drives_right       True
Name: MOR, dtype: object
EG
cars_per_cap       45
country         Egypt
drives_right     True
Name: EG, dtype: object
```

In [57]:

```
# Import cars data
import pandas as pd
cars = pd.read_csv('cars.csv', index_col = 0)
# Adapt for loop
for lab,row in cars.iterrows():
    print(lab +": "+ str(row['cars_per_cap']))

```

```
US: 809
AUS: 731
JAP: 588
IN: 18
RU: 200
MOR: 70
EG: 45
```

Add Column to DataFrame

In [58]:

```
# Import cars data
import pandas as pd
cars = pd.read_csv('cars.csv', index_col = 0)
# Code for loop that adds COUNTRY column
for lab,row in cars.iterrows():
    cars.loc[lab, "COUNTRY"] = row["country"].upper()
# Print cars
print(cars)

```

```
     cars_per_cap        country  drives_right        COUNTRY
US            809  United States          True  UNITED STATES
AUS           731      Australia         False      AUSTRALIA
JAP           588          Japan         False          JAPAN
IN             18          India         False          INDIA
RU            200         Russia          True         RUSSIA
MOR            70        Morocco          True        MOROCCO
EG             45          Egypt          True          EGYPT
```

Exercise 13

In [71]:

```
import pandas as pd
brics = pd.read_csv('brics.csv', index_col = 0)
for lab,row in brics.iterrows():
    # Add a new column called name_length
    brics.loc[lab , "name_length"] =len(row["country"])

# Print brics
print(brics)
```

```
        country    capital    area  population  name_length
BR       Brazil   Brasilia   8.516      200.40          6.0
RU       Russia     Moscow  17.100      143.50          6.0
IN        India  New Delhi   3.286     1252.00          5.0
CH        China    Beijing   9.597     1357.00          5.0
SA  South Africa   Pretoria   1.221       52.98         12.0
```

In [72]:

```
brics["name_length"] = brics["country"].apply(len)
print(brics)
```

```
        country    capital    area  population  name_length
BR       Brazil   Brasilia   8.516      200.40            6
RU       Russia     Moscow  17.100      143.50            6
IN        India  New Delhi   3.286     1252.00            5
CH        China    Beijing   9.597     1357.00            5
SA  South Africa   Pretoria   1.221       52.98           12
```

Exercise 14

In [73]:

```
import pandas as pd

# Import cars data
cars = pd.read_csv('cars.csv', index_col = 0)

# Add a new column called COUNTRY
cars['COUNTRY'] = cars['country'].apply(str.upper)

# Print cars
print(cars)
```

```
     cars_per_cap        country  drives_right        COUNTRY
US            809  United States          True  UNITED STATES
AUS           731      Australia         False      AUSTRALIA
JAP           588          Japan         False          JAPAN
IN             18          India         False          INDIA
RU            200         Russia          True         RUSSIA
MOR            70        Morocco          True        MOROCCO
EG             45          Egypt          True          EGYPT
```

In [ ]:

```
1
```