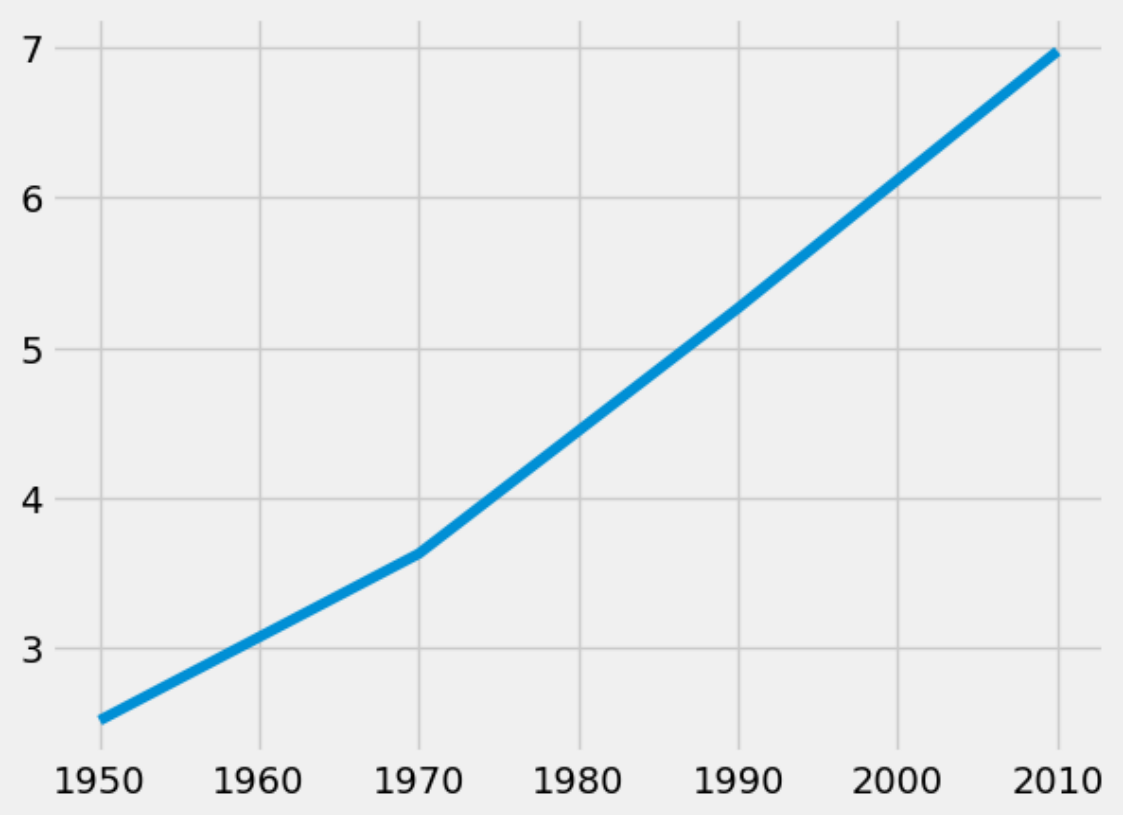
6511286_JanPoonthong_541_W2

June 24, 2023

[478]:
```
import pandas as pd
import matplotlib.pyplot as plt

year = [1950, 1970, 1990, 2010]
pop = [2.519, 3.629, 5.263, 6.972]

plt.plot(year, pop)
```

[478]:
```
[<matplotlib.lines.Line2D at 0x17e8adc00>]
```



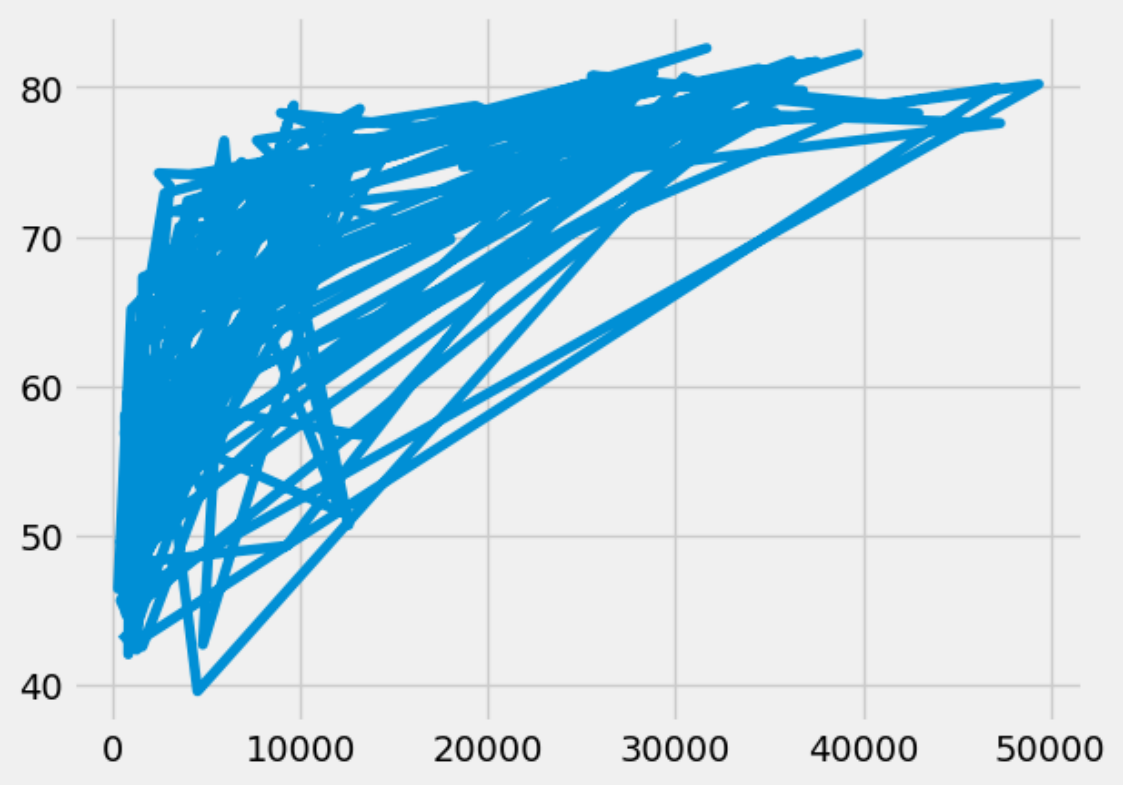
[479]:
```
gdp_cap = [974.5803384, 5937.029525999999, 6223.367465, 4797.231267, 12779.
↪37964, 34435.367439999995, 36126.4927, 29796.04834, 1391.253792, 33692.
↪60508, 1441.284873, 3822.137084, 7446.298803, 12569.85177, 9065.800825,␣
↪10680.79282, 1217.032994, 430.0706916, 1713.778686, 2042.09524, 36319.23501,␣
↪706.016537, 1704.063724, 13171.63885, 4959.114854, 7006.580419, 986.1478792,␣
↪277.5518587, 3632.557798, 9645.06142, 1544.750112, 14619.222719999998, 8948.
↪102923, 22833.30851, 35278.41874, 2082.4815670000003, 6025.374752000001,␣
↪6873.262326000001, 5581.180998, 5728.353514, 12154.08975, 641.3695236000001,␣
↪690.8055759, 33207.0844, 30470.0167, 13206.48452, 752.7497265, 32170.37442,␣
↪1327.60891, 27538.41188, 5186.050003, 942.6542111, 579.2317429999999, 1201.
↪637154, 3548.3308460000003, 39724.97867, 18008.94444, 36180.78919, 2452.
↪210407, 3540.651564, 11605.71449, 4471.061906, 40675.99635, 25523.2771,␣
↪28569.7197, 7320.880262000001, 31656.06806, 4519.461171, 1463.249282, 1593.
↪06548, 23348.139730000003, 47306.98978, 10461.05868, 1569.331442, 414.
↪5073415, 12057.49928, 1044.770126, 759.3499101, 12451.6558, 1042.581557,␣
↪1803.151496, 10956.99112, 11977.57496, 3095.7722710000003, 9253.896111, 3820.
↪17523, 823.6856205, 944.0, 4811.060429, 1091.359778, 36797.93332, 25185.
↪00911, 2749.320965, 619.6768923999999, 2013.977305, 49357.19017, 22316.
↪19287, 2605.94758, 9809.185636, 4172.838464, 7408.905561, 3190.481016, 15389.
↪924680000002, 20509.64777, 19328.70901, 7670.122558, 10808.47561, 863.
↪0884639000001, 1598.435089, 21654.83194, 1712.472136, 9786.534714, 862.
↪5407561000001, 47143.17964, 18678.31435, 25768.25759, 926.1410683, 9269.
↪657808, 28821.0637, 3970.095407, 2602.394995, 4513.480643, 33859.74835,␣
↪37506.41907, 4184.548089, 28718.27684, 1107.482182, 7458.396326999999, 882.
↪9699437999999, 18008.50924, 7092.923025, 8458.276384, 1056.380121, 33203.
↪26128, 42951.65309, 10611.46299, 11415.80569, 2441.576404, 3025.349798, 2280.
↪769906, 1271.211593, 469.70929810000007]

life_exp = [43.828, 76.423, 72.301, 42.731, 75.32, 81.235, 79.829, 75.635, 64.
↪062, 79.441, 56.728, 65.554, 74.852, 50.728, 72.39, 73.005, 52.295, 49.58,␣
↪59.723, 50.43, 80.653, 44.74100000000001, 50.651, 78.553, 72.961, 72.889, 65.
↪152, 46.462, 55.322, 78.782, 48.328, 75.748, 78.273, 76.486, 78.332, 54.791,␣
↪72.235, 74.994, 71.33800000000001, 71.878, 51.57899999999999, 58.04, 52.947,␣
↪79.313, 80.657, 56.735, 59.448, 79.406, 60.022, 79.483, 70.259, 56.007, 46.
↪388000000000005, 60.916, 70.19800000000001, 82.208, 73.33800000000001, 81.
↪757, 64.69800000000001, 70.65, 70.964, 59.545, 78.885, 80.745, 80.546, 72.
↪567, 82.603, 72.535, 54.11, 67.297, 78.623, 77.58800000000001, 71.993, 42.
↪592, 45.678, 73.952, 59.443000000000005, 48.303, 74.241, 54.467, 64.164, 72.
↪801, 76.195, 66.803, 74.543, 71.164, 42.082, 62.069, 52.906000000000006, 63.
↪785, 79.762, 80.204, 72.899, 56.867, 46.859, 80.196, 75.64, 65.483, 75.
↪53699999999999, 71.752, 71.421, 71.688, 75.563, 78.098, 78.74600000000001,␣
↪76.442, 72.476, 46.242, 65.528, 72.777, 63.062, 74.002, 42.568000000000005,␣
↪79.972, 74.663, 77.926, 48.159, 49.339, 80.941, 72.396, 58.556, 39.613, 80.
↪884, 81.70100000000001, 74.143, 78.4, 52.517, 70.616, 58.42, 69.819, 73.923,␣
↪71.777, 51.542, 79.425, 78.242, 76.384, 73.747, 74.249, 73.422, 62.698, 42.
↪38399999999999, 43.487]
```

```
pop = [31.889923, 3.600523, 33.333216, 12.420476, 40.301927, 20.434176, 8.
 ↪199783, 0.708573, 150.448339, 10.392226, 8.078314, 9.119152, 4.552198, 1.
 ↪639131, 190.010647, 7.322858, 14.326203, 8.390505, 14.131858, 17.696293, 33.
 ↪390141, 4.369038, 10.238807, 16.284741, 1318.683096, 44.22755, 0.71096, 64.
 ↪606759, 3.80061, 4.133884, 18.013409, 4.493312, 11.416987, 10.228744, 5.
 ↪46812, 0.496374, 9.319622, 13.75568, 80.264543, 6.939688, 0.551201, 4.
 ↪906585, 76.511887, 5.23846, 61.083916, 1.454867, 1.688359, 82.400996, 22.
 ↪873338, 10.70629, 12.572928, 9.947814, 1.472041, 8.502814, 7.483763, 6.
 ↪980412, 9.956108, 0.301931, 1110.396331, 223.547, 69.45357, 27.499638, 4.
 ↪109086, 6.426679, 58.147733, 2.780132, 127.467972, 6.053193, 35.610177, 23.
 ↪301725, 49.04479, 2.505559, 3.921278, 2.012649, 3.193942, 6.036914, 19.
 ↪167654, 13.327079, 24.821286, 12.031795, 3.270065, 1.250882, 108.700891, 2.
 ↪874127, 0.684736, 33.757175, 19.951656, 47.76198, 2.05508, 28.90179, 16.
 ↪570613, 4.115771, 5.675356, 12.894865, 135.031164, 4.627926, 3.204897, 169.
 ↪270617, 3.242173, 6.667147, 28.674757, 91.077287, 38.518241, 10.642836, 3.
 ↪942491, 0.798094, 22.276056, 8.860588, 0.199579, 27.601038, 12.267493, 10.
 ↪150265, 6.144562, 4.553009, 5.447502, 2.009245, 9.118773, 43.997828, 40.
 ↪448191, 20.378239, 42.292929, 1.133066, 9.031088, 7.554661, 19.314747, 23.
 ↪174294, 38.13964, 65.068149, 5.701579, 1.056608, 10.276158, 71.158647, 29.
 ↪170398, 60.776238, 301.139947, 3.447496, 26.084662, 85.262356, 4.018332, 22.
 ↪211743, 11.746035, 12.311143]

life_exp1950 = [28.8, 55.23, 43.08, 30.02, 62.48, 69.12, 66.8, 50.94, 37.48, 68.
 ↪0, 38.22, 40.41, 53.82, 47.62, 50.92, 59.6, 31.98, 39.03, 39.42, 38.52, 68.
 ↪75, 35.46, 38.09, 54.74, 44.0, 50.64, 40.72, 39.14, 42.11, 57.21, 40.48, 61.
 ↪21, 59.42, 66.87, 70.78, 34.81, 45.93, 48.36, 41.89, 45.26, 34.48, 35.93, 34.
 ↪08, 66.55, 67.41, 37.0, 30.0, 67.5, 43.15, 65.86, 42.02, 33.61, 32.5, 37.58,␣
 ↪41.91, 60.96, 64.03, 72.49, 37.37, 37.47, 44.87, 45.32, 66.91, 65.39, 65.94,␣
 ↪58.53, 63.03, 43.16, 42.27, 50.06, 47.45, 55.56, 55.93, 42.14, 38.48, 42.72,␣
 ↪36.68, 36.26, 48.46, 33.68, 40.54, 50.99, 50.79, 42.24, 59.16, 42.87, 31.29,␣
 ↪36.32, 41.72, 36.16, 72.13, 69.39, 42.31, 37.44, 36.32, 72.67, 37.58, 43.44,␣
 ↪55.19, 62.65, 43.9, 47.75, 61.31, 59.82, 64.28, 52.72, 61.05, 40.0, 46.47,␣
 ↪39.88, 37.28, 58.0, 30.33, 60.4, 64.36, 65.57, 32.98, 45.01, 64.94, 57.59,␣
 ↪38.64, 41.41, 71.86, 69.62, 45.88, 58.5, 41.22, 50.85, 38.6, 59.1, 44.6, 43.
 ↪58, 39.98, 69.18, 68.44, 66.07, 55.09, 40.41, 43.16, 32.55, 42.04, 48.45]
```

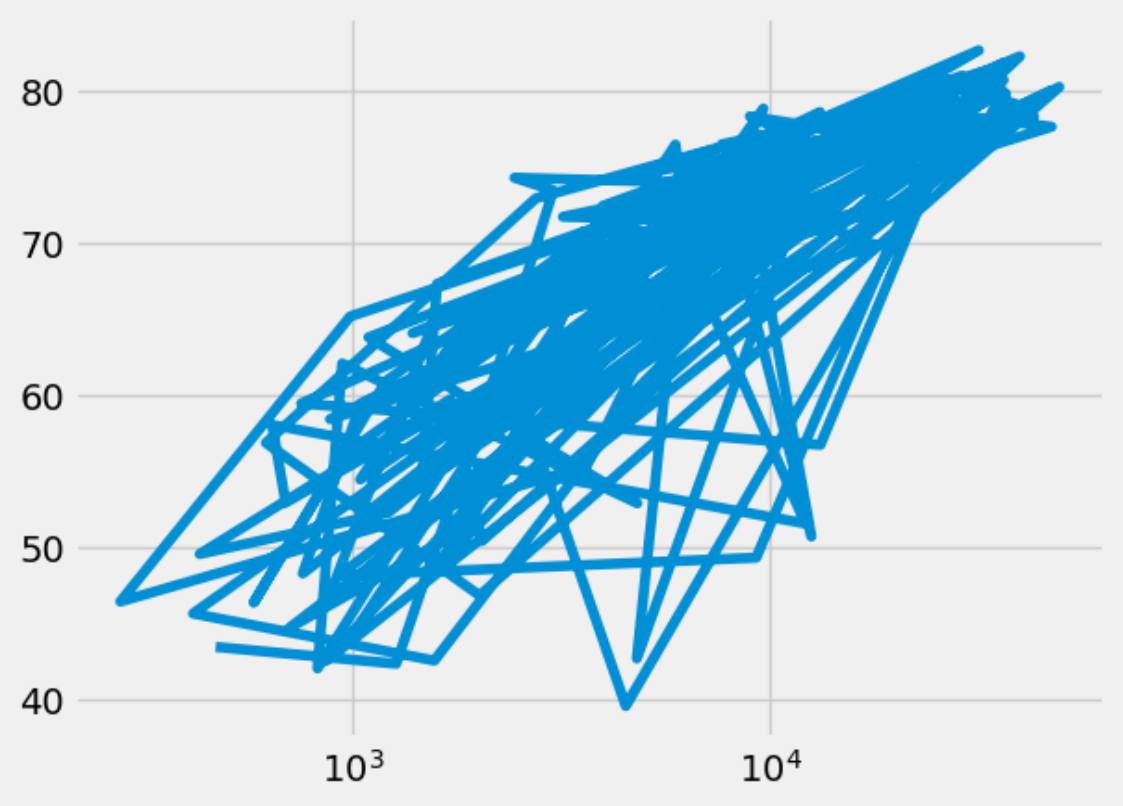
```
col = ['red', 'green', 'blue', 'blue', 'yellow', 'black', 'green', 'red',␣
 ↪'red', 'green', 'blue', 'yellow', 'green', 'blue', 'yellow', 'green',␣
 ↪'blue', 'blue', 'red', 'blue', 'yellow', 'blue', 'blue', 'yellow', 'red',␣
 ↪'yellow', 'blue', 'blue', 'blue', 'yellow', 'blue', 'green', 'yellow',␣
 ↪'green', 'green', 'blue', 'yellow', 'yellow', 'blue', 'yellow', 'blue',␣
 ↪'blue', 'blue', 'green', 'green', 'blue', 'blue', 'green', 'blue', 'green',␣
 ↪'yellow', 'blue', 'blue', 'yellow', 'yellow', 'red', 'green', 'green',␣
 ↪'red', 'red', 'red', 'red', 'green', 'red', 'green', 'yellow', 'red', 'red',␣
 ↪'blue', 'red', 'red', 'red', 'red', 'blue', 'blue', 'blue', 'blue', 'blue',␣
 ↪'red', 'blue', 'blue', 'blue', 'yellow', 'red', 'green', 'blue', 'blue',␣
 ↪'red', 'blue', 'red', 'green', 'black', 'yellow', 'blue', 'blue', 'green',␣
 ↪'red', 'red', 'yellow', 'yellow', 'yellow', 'red', 'green', 'green',␣
 ↪'yellow', 'blue', 'green', 'blue', 'blue', 'red', 'blue', 'green', 'blue',␣
 ↪'red', 'green', 'green', 'blue', 'blue', 'green', 'red', 'blue', 'blue',␣
 ↪'green', 'green', 'red', 'red', 'blue', 'red', 'blue', 'yellow', 'blue',␣
 ↪'green', 'blue', 'green', 'yellow', 'yellow', 'yellow', 'red', 'red', 'red',␣
 ↪'blue', 'blue']
```

[480]:
```
plt.plot(gdp_cap, life_exp)
```

[480]:
```
[<matplotlib.lines.Line2D at 0x17f2f6d10>]
```

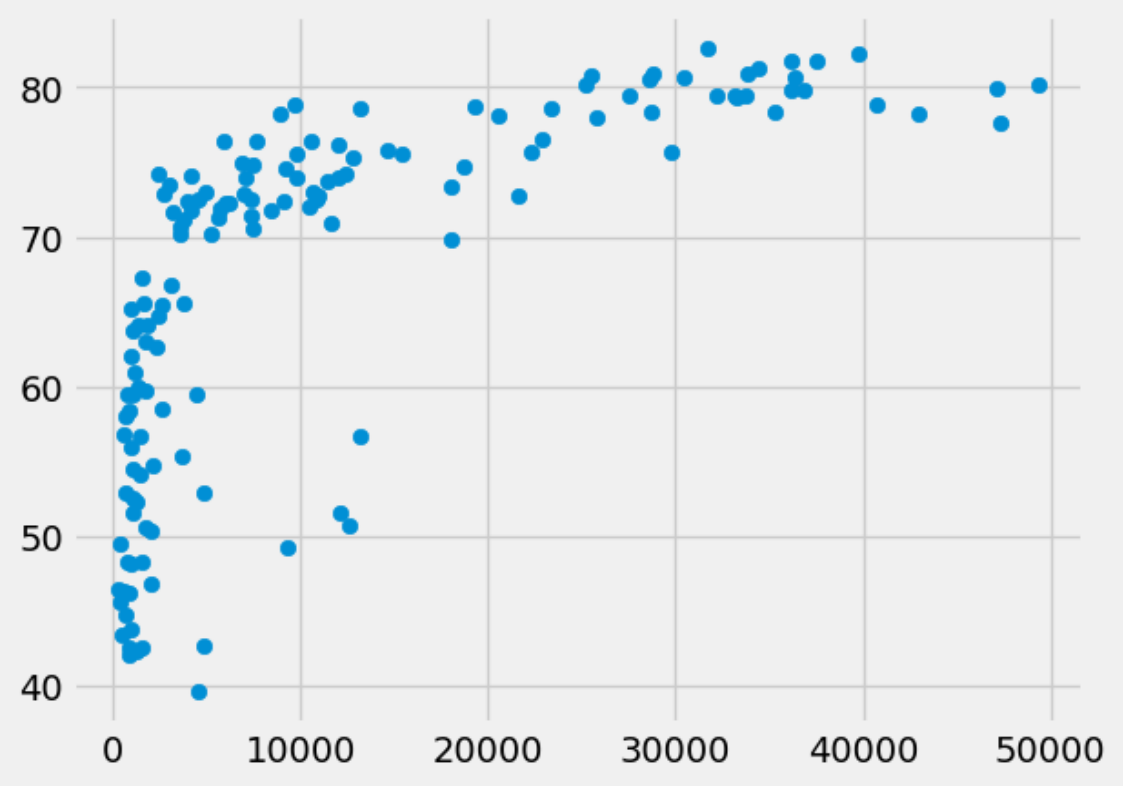


[481]:
```
# Testing with log - mine
plt.plot(gdp_cap, life_exp)
plt.xscale('log')
```

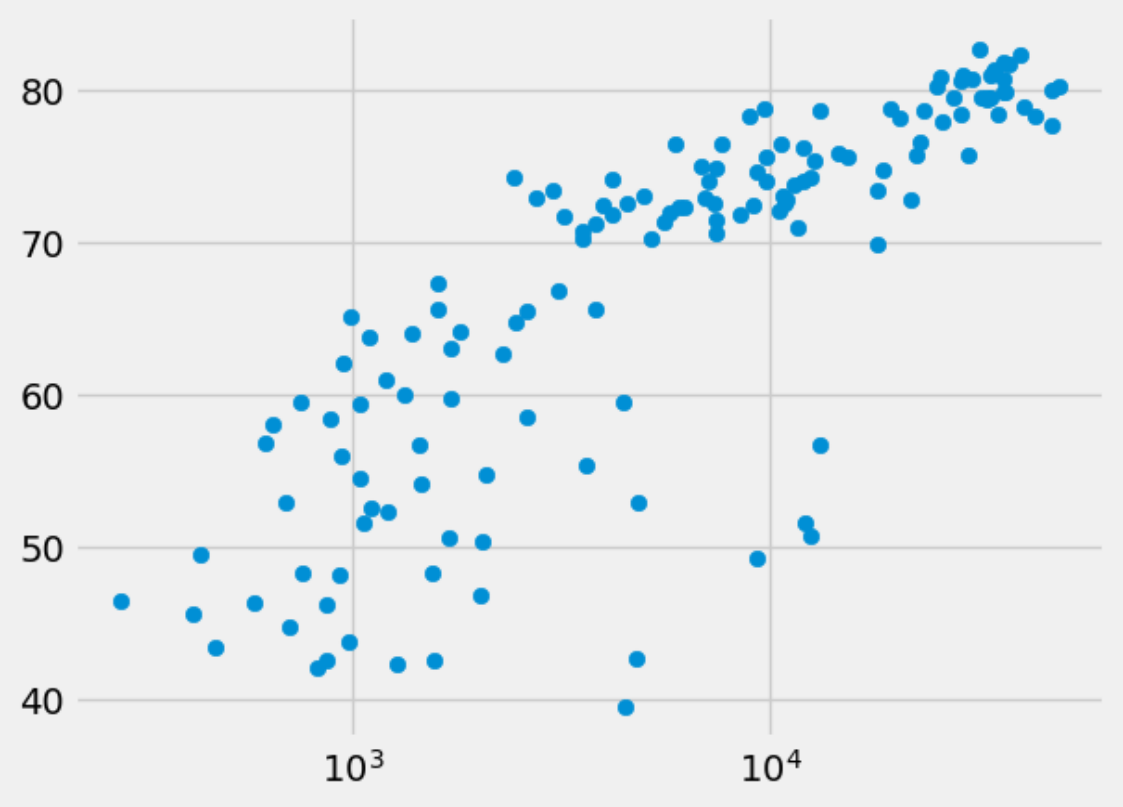


[482]:
```
plt.scatter(gdp_cap, life_exp)
```

[482]:
```
<matplotlib.collections.PathCollection at 0x17f4624a0>
```

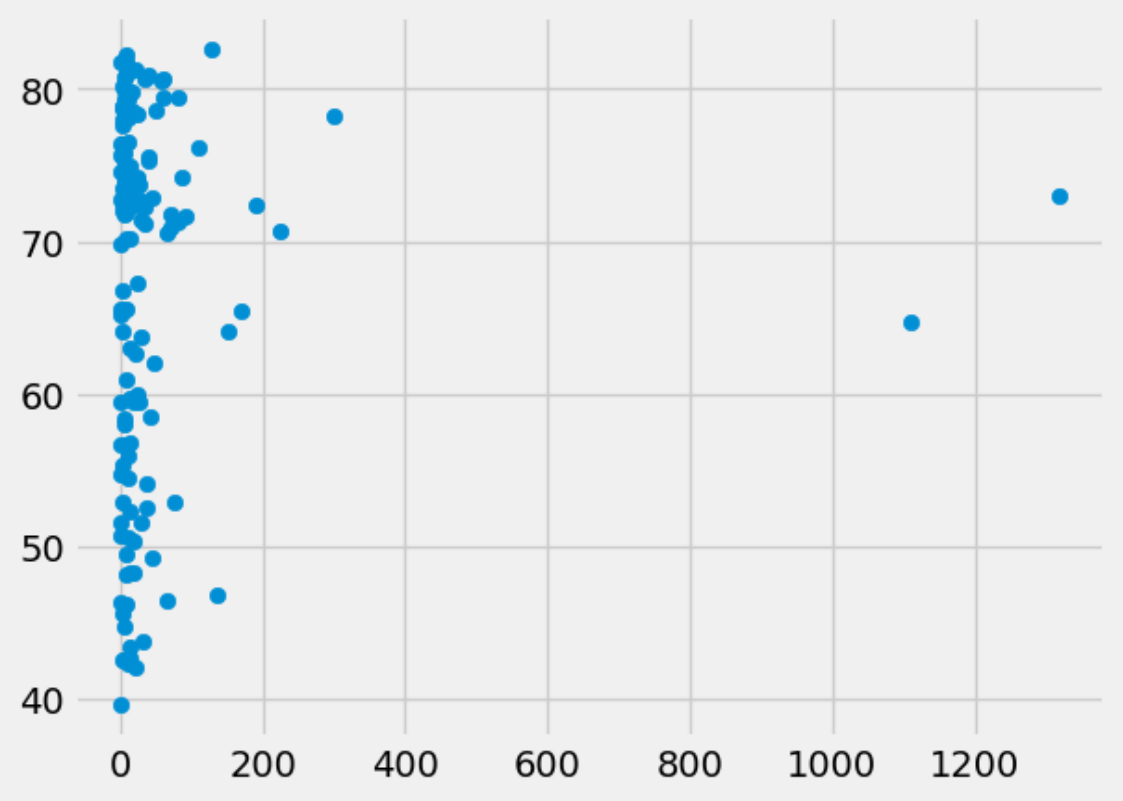


[483]:
```
# Testing with log
plt.scatter(gdp_cap, life_exp)
plt.xscale('log')
```



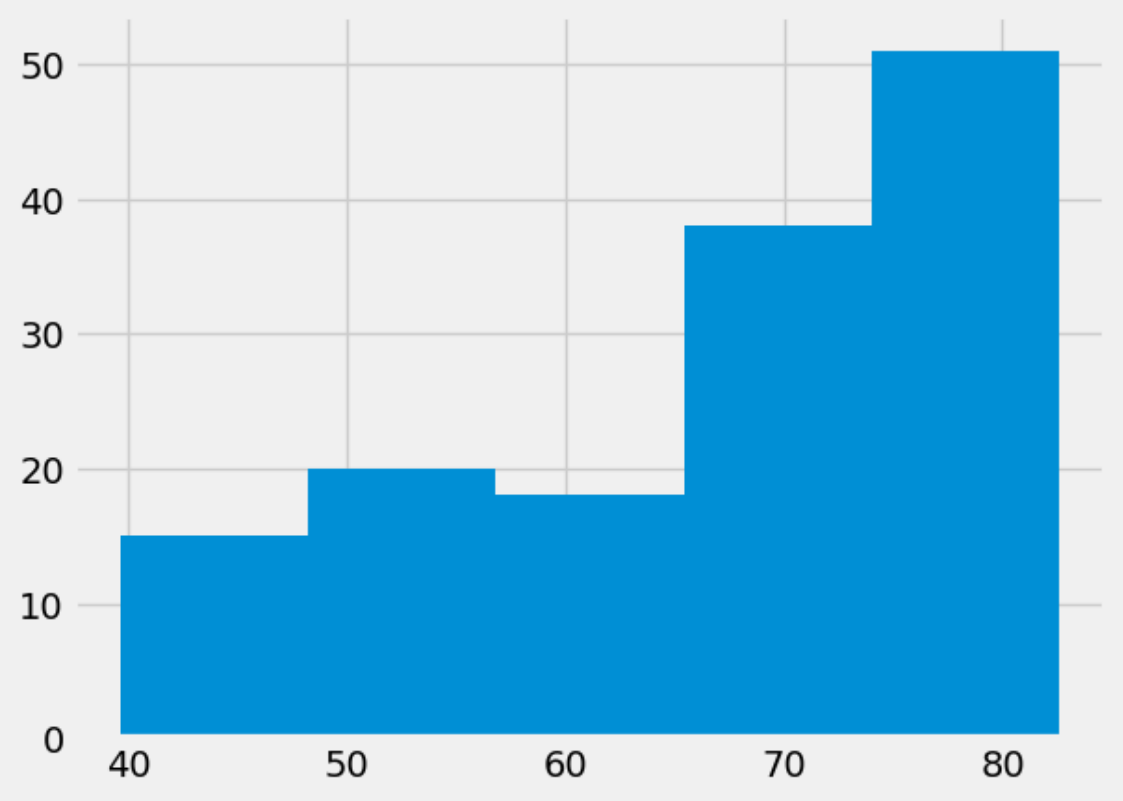
[484]:
```
# Ex 1
plt.scatter(pop, life_exp)
```

[484]: <matplotlib.collections.PathCollection at 0x17f591720>



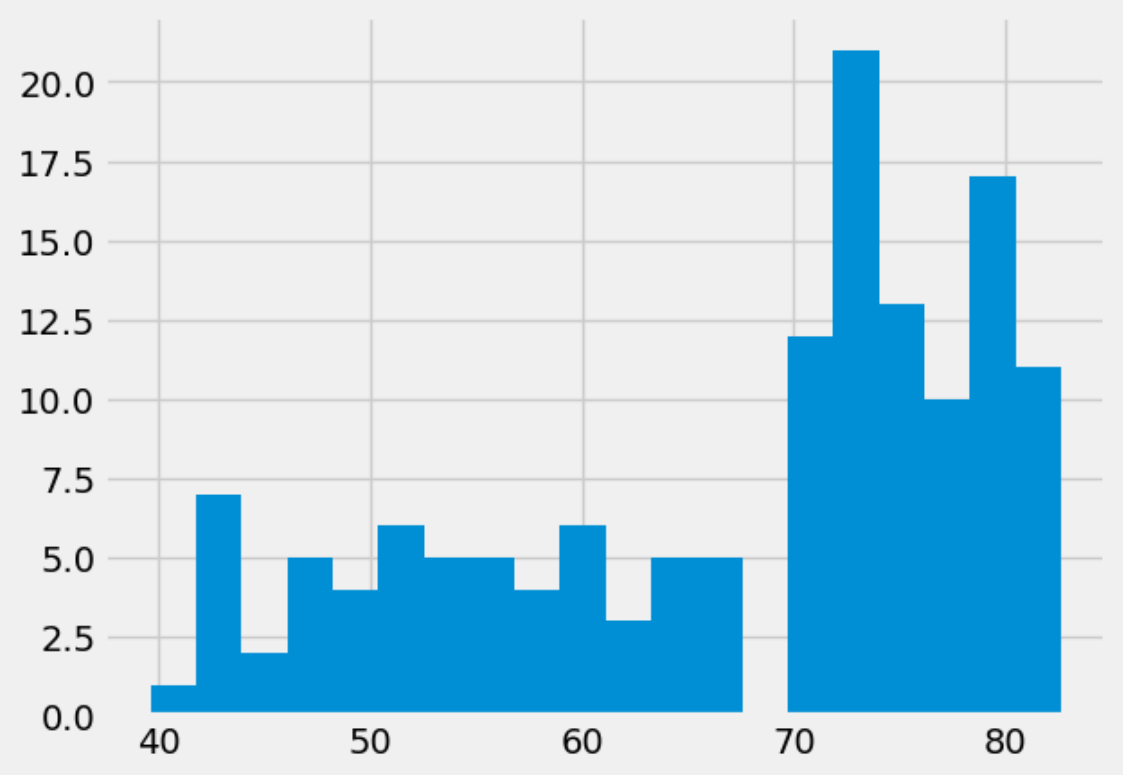
[485]:
```
# Ex 2
plt.hist(life_exp, bins=5)
```

[485]:
```
(array([15., 20., 18., 38., 51.]),
 array([39.613, 48.211, 56.809, 65.407, 74.005, 82.603]),
 <BarContainer object of 5 artists>)
```



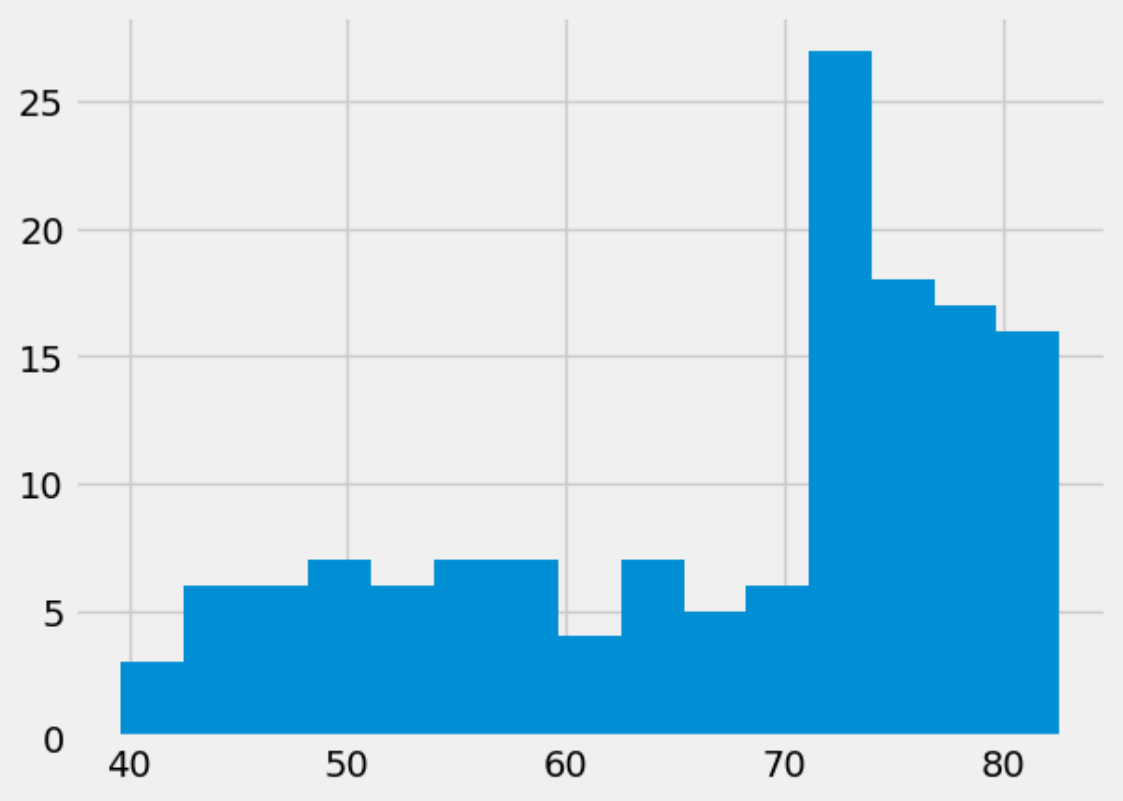
[486]:
```
# Ex 2
plt.hist(life_exp, bins=20)
```

[486]:
```
(array([ 1.,  7.,  2.,  5.,  4.,  6.,  5.,  5.,  4.,  6.,  3.,  5.,  5.,
         0., 12., 21., 13., 10., 17., 11.]),
 array([39.613 , 41.7625, 43.912 , 46.0615, 48.211 , 50.3605, 52.51  ,
        54.6595, 56.809 , 58.9585, 61.108 , 63.2575, 65.407 , 67.5565,
        69.706 , 71.8555, 74.005 , 76.1545, 78.304 , 80.4535, 82.603 ]),
 <BarContainer object of 20 artists>)
```



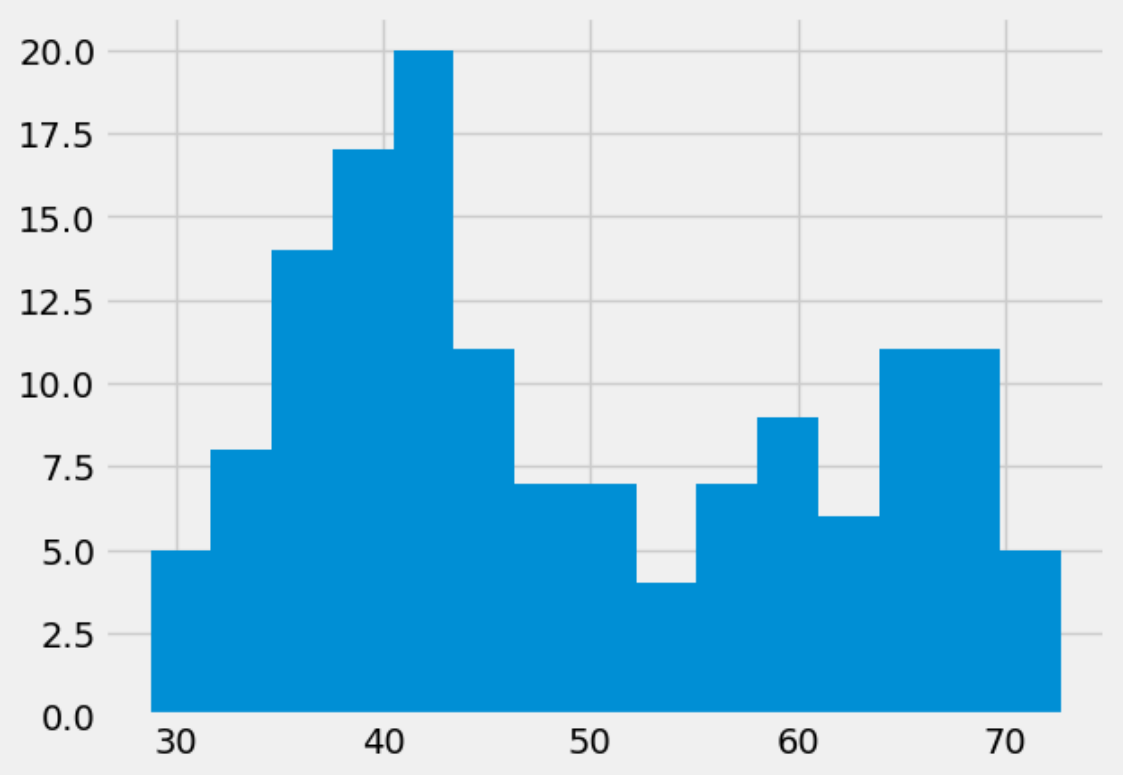
[487]:
```
# Ex 3
plt.hist(life_exp, bins=15)
```

[487]:
```
(array([ 3.,  6.,  6.,  7.,  6.,  7.,  7.,  4.,  7.,  5.,  6., 27., 18.,
        17., 16.]),
 array([39.613, 42.479, 45.345, 48.211, 51.077, 53.943, 56.809, 59.675,
        62.541, 65.407, 68.273, 71.139, 74.005, 76.871, 79.737, 82.603]),
 <BarContainer object of 15 artists>)
```



[488]:
```
# Ex 3
plt.hist(life_exp1950, bins=15)
```

[488]:
```
(array([ 5.,  8., 14., 17., 20., 11.,  7.,  7.,  4.,  7.,  9.,  6., 11.,
        11.,  5.]),
 array([28.8       , 31.72466667, 34.64933333, 37.574     , 40.49866667,
        43.42333333, 46.348     , 49.27266667, 52.19733333, 55.122     ,
        58.04666667, 60.97133333, 63.896     , 66.82066667, 69.74533333,
        72.67      ]),
 <BarContainer object of 15 artists>)
```



1 Ex 3

1.1 Answer

Observation: life_exp for 1950 has life expectancy is around 40 to 50 but has imporove in 2007 life_exp for 2007 has life expectancy is around 70 to 80

Bins: life_exp for 1950 has highest frequency of bins around 15 to 20

life_exp for 2007 has highest frequency of bins around 15 to 20

2 Ex 4

Histogram

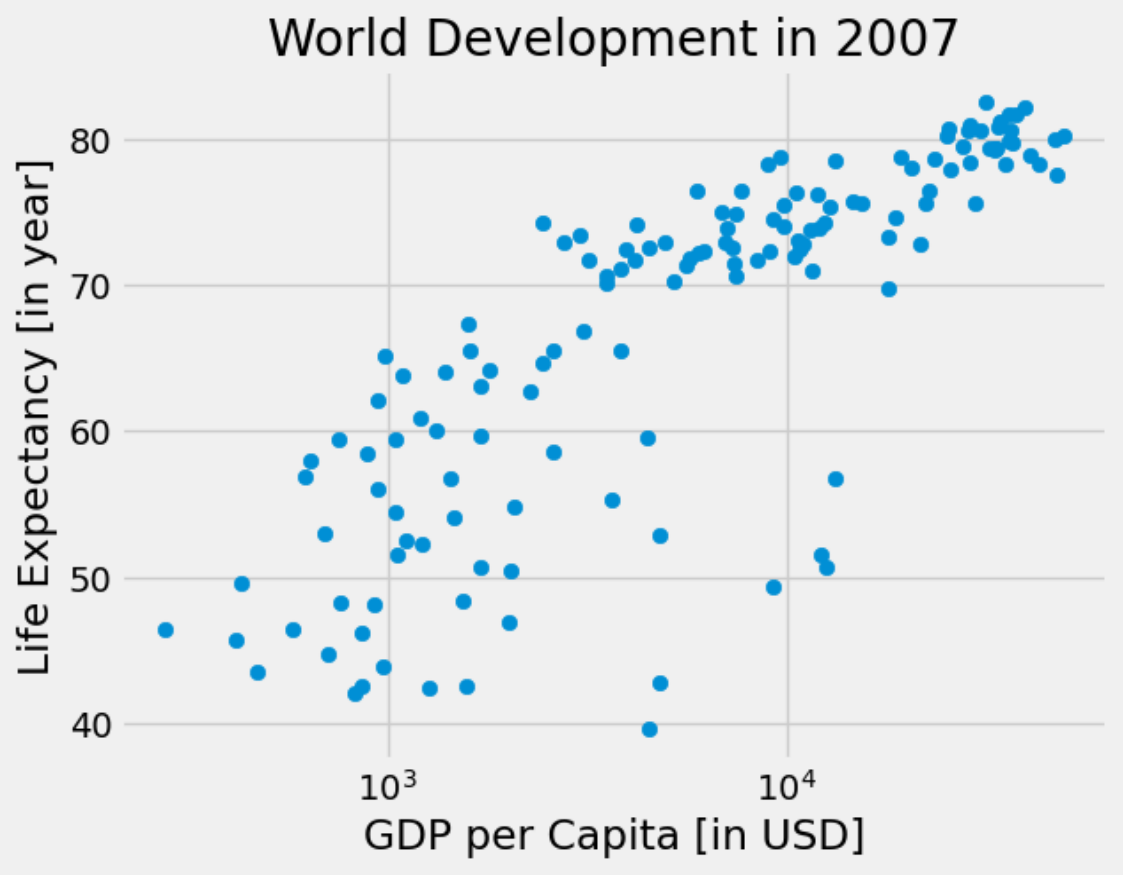
3 Ex 5

Scatter Plot

[489]:
```
# Ex 6
plt.scatter(gdp_cap, life_exp)
plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
```

```
plt.ylabel('Life Expectancy [in year]')
plt.title('World Development in 2007')
```

[489]: `Text(0.5, 1.0, 'World Development in 2007')`

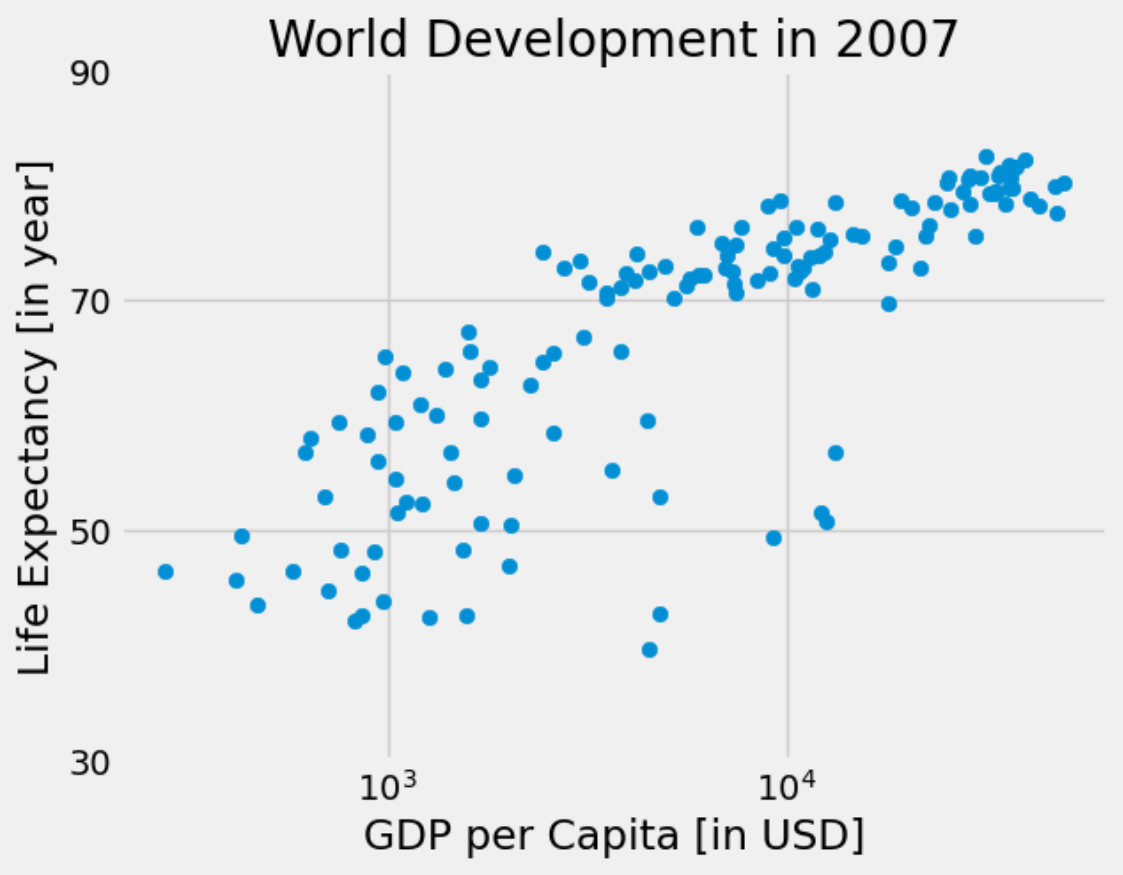


[490]:
```
# Ex 6
plt.scatter(gdp_cap, life_exp)
plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in year]')
plt.title('World Development in 2007')
tick_val = [1000, 10000, 100000]
tick_lab = ['1k', '10k', '100k']
plt.xticks(tick_val, tick_lab)
```

[490]:
```
([<matplotlib.axis.XTick at 0x280a32b90>,
  <matplotlib.axis.XTick at 0x280a32b60>,
  <matplotlib.axis.XTick at 0x280a5d6c0>],
 [Text(1000, 0, '1k'), Text(10000, 0, '10k'), Text(100000, 0, '100k')])
```

[491]:
```
# Ex 7
plt.scatter(gdp_cap, life_exp)
plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in year]')
plt.title('World Development in 2007')
tick_val = [30, 50, 70, 90]
tick_lab = ['30', '50', '70', '90']
plt.yticks(tick_val, tick_lab)
```

[491]:
```
([<matplotlib.axis.YTick at 0x280c36e30>,
  <matplotlib.axis.YTick at 0x280c36770>,
  <matplotlib.axis.YTick at 0x280c342e0>,
  <matplotlib.axis.YTick at 0x280c529e0>],
 [Text(0, 30, '30'), Text(0, 50, '50'), Text(0, 70, '70'), Text(0, 90, '90')])
```

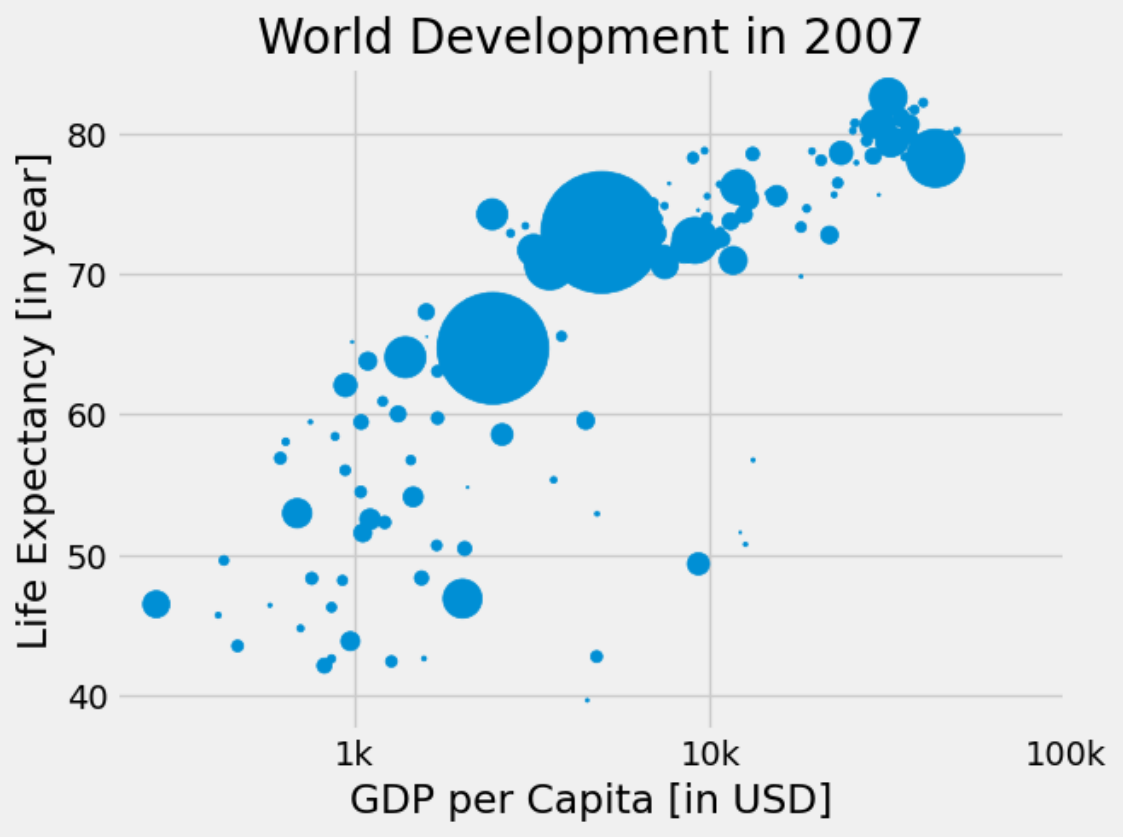


[492]:
```
plt.scatter(gdp_cap, life_exp)
plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in year]')
plt.title('World Development in 2007')

xtick_val = [1000, 10000, 100000]
xtick_lab = ['1k', '10k', '100k']

ytick_val = [30, 50, 70, 90]
ytick_lab = ['30', '50', '70', '90']

plt.yticks(ytick_val, ytick_lab)
plt.xticks(xtick_val, xtick_lab)
```

[492]:
```
([<matplotlib.axis.XTick at 0x17f593940>,
  <matplotlib.axis.XTick at 0x17f593970>,
  <matplotlib.axis.XTick at 0x17f4fb520>],
 [Text(1000, 0, '1k'), Text(10000, 0, '10k'), Text(100000, 0, '100k')])
```

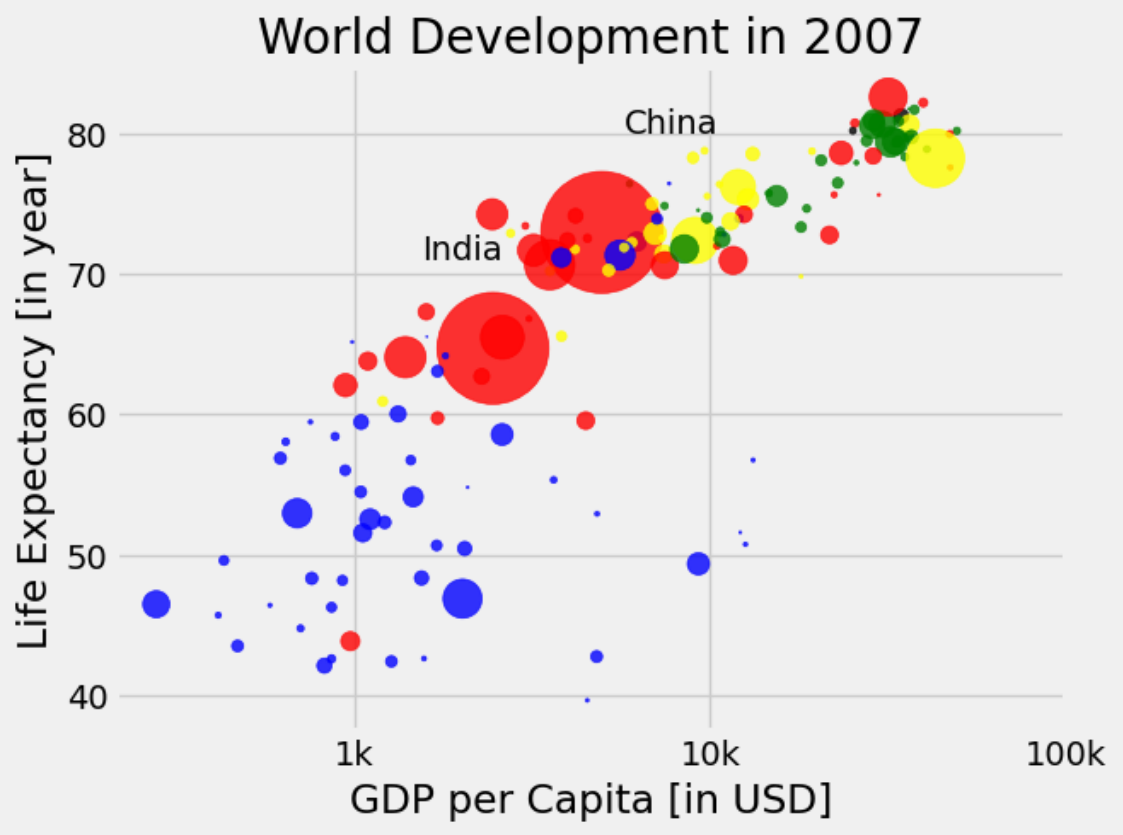
[493]:
```
import numpy as np
np_pop = np.array(pop)
np_pop = np_pop * 2
plt.scatter(gdp_cap, life_exp, s=np_pop)

plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in year]')
plt.title('World Development in 2007')

xtick_val = [1000, 10000, 100000]
xtick_lab = ['1k', '10k', '100k']

plt.xticks(xtick_val, xtick_lab)
```

[493]:
```
([<matplotlib.axis.XTick at 0x17f4101f0>,
  <matplotlib.axis.XTick at 0x17f410310>,
  <matplotlib.axis.XTick at 0x17f34b430>],
 [Text(1000, 0, '1k'), Text(10000, 0, '10k'), Text(100000, 0, '100k')])
```



[494]:
```
import numpy as np
np_pop = np.array(pop)
np_pop = np_pop * 2
plt.scatter(gdp_cap, life_exp, s=np_pop, c=col, alpha=0.8)

plt.xscale('log')
plt.xlabel('GDP per Capita [in USD]')
plt.ylabel('Life Expectancy [in year]')
plt.title('World Development in 2007')

xtick_val = [1000, 10000, 100000]
xtick_lab = ['1k', '10k', '100k']

plt.text(1550, 71, 'India')
plt.text(5700, 80, 'China')
plt.grid(1)

plt.xticks(xtick_val, xtick_lab)
```

[494]:
```
([<matplotlib.axis.XTick at 0x17d529840>,
  <matplotlib.axis.XTick at 0x17d529cf0>,
  <matplotlib.axis.XTick at 0x17d694be0>],
 [Text(1000, 0, '1k'), Text(10000, 0, '10k'), Text(100000, 0, '100k')])
```



4 Ex 8

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

There is a negative correlation between GDP per capita and life expectancy: False

China has both a lower GDP per capita and lower life expectancy compared to India: False

[495]:
```
import pandas as pd
brics = pd.read_csv('week-3-data/brics.csv', index_col=0)
brics.head()
```

[495]:

	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
```

[496]:
```
brics.loc['BR':'IN', 'capital':'population']
```

[496]:
```
       capital    area  population
BR   Brasilia   8.516       200.4
RU     Moscow  17.100       143.5
IN  New Delhi   3.286      1252.0
```

[497]:
```
brics.iloc[:4,1:]
```

[497]:
```
       capital    area  population
BR   Brasilia   8.516       200.4
RU     Moscow  17.100       143.5
IN  New Delhi   3.286      1252.0
CH    Beijing   9.597      1357.0
```

[498]:
```
brics['area'] # series
```

[498]:
```
BR     8.516
RU    17.100
IN     3.286
CH     9.597
SA     1.221
Name: area, dtype: float64
```

[499]:
```
brics[['area', 'country']] # data frame allow 1 or more column
```

[499]:
```
       area       country
BR   8.516        Brazil
RU  17.100        Russia
IN   3.286         India
CH   9.597         China
SA   1.221  South Africa
```

[500]:
```
result = brics['area'] > 8
brics[result]
```

[500]:
```
    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
```

[501]:
```
brics[result][['country', 'area']] # if I want two coloumn showing the data␣
↪that is greater than 8
```

[501]:
```
   country    area
BR  Brazil   8.516
RU  Russia  17.100
CH   China   9.597
```

5 Ex 9

[502]:
```
more_than_200 = brics['population'] >= 200
brics[more_than_200][['country', 'population']]
```

[502]:
```
   country  population
BR  Brazil       200.4
IN   India      1252.0
CH   China      1357.0
```

[503]:
```
brics[(brics['area'] > 8) & (brics['area'] < 12)]
```

[503]:
```
   country   capital   area  population
BR  Brazil  Brasilia  8.516       200.4
CH   China   Beijing  9.597      1357.0
```

[504]:
```
area810 = np.logical_and(brics['area'] > 8, brics['area'] < 12)
brics[area810]
```

[504]:
```
   country   capital   area  population
BR  Brazil  Brasilia  8.516       200.4
CH   China   Beijing  9.597      1357.0
```

6 Ex 10

[505]:
```
brics[(brics['population'] > 1000) | (brics['area'] < 8)][['capital']]
```

[505]:
```
      capital
IN  New Delhi
CH    Beijing
SA   Pretoria
```

7 Ex 11

[506]:
```
cars = pd.read_csv('week-3-data/cars.csv', index_col=0)
cpc = cars['cars_per_cap']
cpc
```

[506]:
```
US     809
AUS    731
JAP    588
```

```
IN      18
RU     200
MOR     70
EG      45
Name: cars_per_cap, dtype: int64
```

[507]:
```
result = cars['cars_per_cap'] > 500
many_cars = cars[result]
many_cars
```

[507]:
```
     cars_per_cap        country  drives_right
US            809  United States          True
AUS           731      Australia         False
JAP           588          Japan         False
```

8 Ex 12

[508]:
```
cpc100500 = cars[(cars['cars_per_cap'] > 100) & (cars['cars_per_cap'] < 500)]
cpc100500
```

[508]:
```
    cars_per_cap country  drives_right
RU           200  Russia          True
```

[509]:
```
cpc100500 = np.logical_and((cars['cars_per_cap'] > 100), (cars['cars_per_cap']␣
↪< 500))
cars[cpc100500]
```

[509]:
```
    cars_per_cap country  drives_right
RU           200  Russia          True
```

[510]:
```
for key in cars:
    print(key)
```

```
cars_per_cap
country
drives_right
```

[511]:
```
for key, value in cars.iterrows():
    print(key, 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
```

[512]:
```
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
```

[513]:
```
for lab, row in cars.iterrows():
    cars.loc[lab, "COUNTRY"] = row['country'].upper()
cars
```

[513]:

	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

9 Ex 13

[514]:
```
brics = pd.read_csv('week-3-data/brics.csv', index_col=0)

for lab, row in brics.iterrows():
    brics.loc[lab, "name_length"] = len(row['country'])
brics
```

[514]:

	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

[515]:
```
brics['name_length'] = brics['country'].apply(len)
brics
```

[515]:

	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

10 Ex 14

[516]:
```
cars = pd.read_csv('week-3-data/cars.csv', index_col=0)

cars['COUNTRY'] = cars['country'].apply(str.upper)
cars
```

[516]:

	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

[517]:
```
cars['name_length'] = cars['country'].apply(len)
cars
```

[517]:

	cars_per_cap	country	drives_right	COUNTRY	name_length
US	809	United States	True	UNITED STATES	13
AUS	731	Australia	False	AUSTRALIA	9

```
JAP          588          Japan          False          JAPAN          5
IN            18          India          False          INDIA          5
RU           200         Russia           True         RUSSIA          6
MOR           70        Morocco           True        MOROCCO          7
EG            45          Egypt           True          EGYPT          5
```

[518]:
```
years = [2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020]
durations = [103,101,99,100,100,95,95,96,93,90]
movie_dict = {"years": years, "durations": durations}

movie_dict
```

[518]:
```
{'years': [2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020],
 'durations': [103, 101, 99, 100, 100, 95, 95, 96, 93, 90]}
```

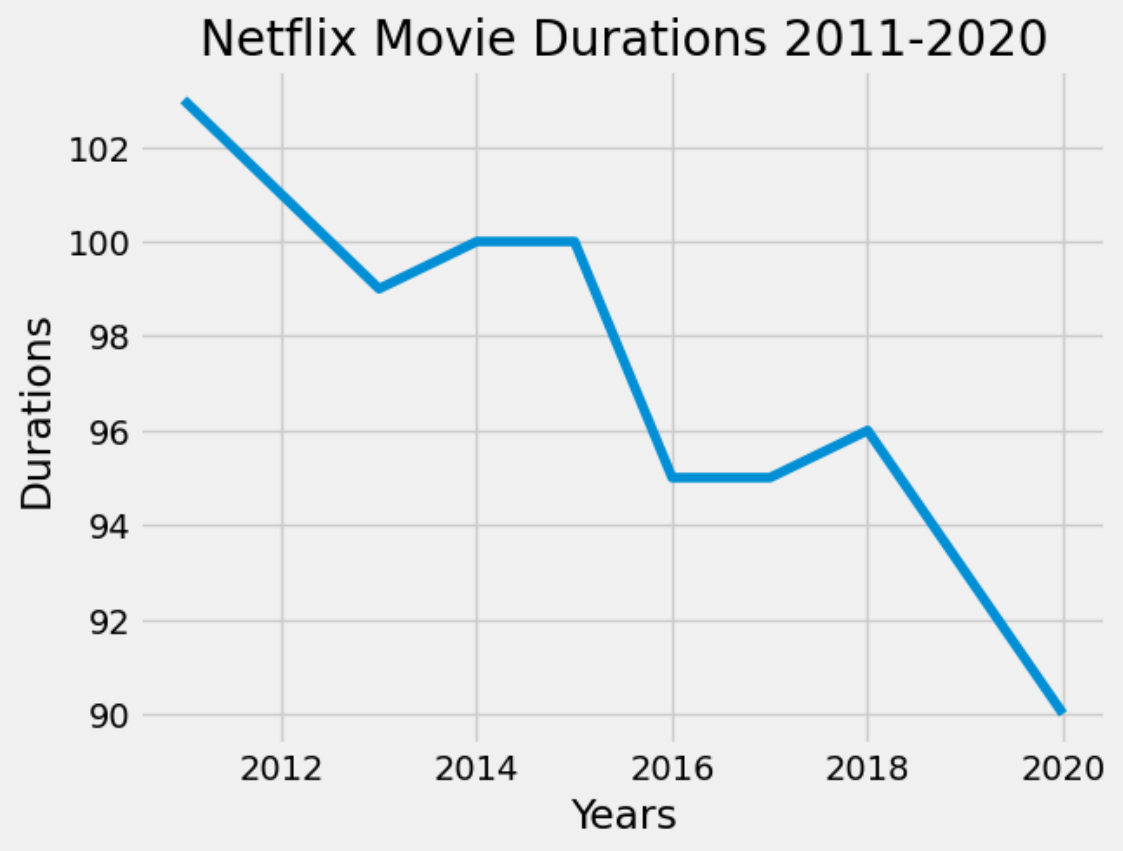
[519]:
```
durations_df = pd.DataFrame(movie_dict)
durations_df
```

[519]:

	years	durations
0	2011	103
1	2012	101
2	2013	99
3	2014	100
4	2015	100
5	2016	95
6	2017	95
7	2018	96
8	2019	93
9	2020	90

[520]:
```
fig = plt.figure()

plt.plot(durations_df['years'], durations_df['durations'])
plt.title('Netflix Movie Durations 2011-2020')
plt.xlabel("Years")
plt.ylabel("Durations")
```

[520]:
```
Text(0, 0.5, 'Durations')
```



[521]:
```
netflix_df = pd.read_csv(r"week-3-data/netflix_data.csv")
netflix_df.head()
```

[521]:

	show_id	type	title	director	cast	country	date_added	release_year	duration
0	s1	TV Show	3%	NaN	João Miguel, Bianca Comparato, Michel Gomes, R…	Brazil	August 14, 2020	2020	4
1	s2	Movie	7:19	Jorge Michel Grau	Demián Bichir, Héctor Bonilla, Oscar Serrano, …	Mexico	December 23, 2016	2016	93
2	s3	Movie	23:59	Gilbert Chan	Tedd Chan, Stella Chung, Henley Hii, Lawrence …	Singapore			
3	s4	Movie	9	Shane Acker	Elijah Wood, John C. Reilly, Jennifer Connelly…	United States			
4	s5	Movie	21	Robert Luketic	Jim Sturgess, Kevin Spacey, Kate Bosworth, Aar…	United States			

```
2  December 20, 2018          2011        78
3  November 16, 2017          2009        80
4    January 1, 2020          2008       123

                                        description             genre
0  In a future where the elite inhabit an island …  International TV
1  After a devastating earthquake hits Mexico Cit…            Dramas
2  When an army recruit is found dead, his fellow…     Horror Movies
3  In a postapocalyptic world, rag-doll robots hi…            Action
4  A brilliant group of students become card-coun…            Dramas
```

[522]:
```
netflix_df[netflix_df["type"] == "Movie"][['title', 'country', 'genre',␣
↪'release_year', 'duration']]
```

[522]:
```
                                      title         country          genre  \
1                                      7:19          Mexico         Dramas
2                                     23:59       Singapore  Horror Movies
3                                         9   United States         Action
4                                        21   United States         Dramas
6                                       122           Egypt  Horror Movies
…                                       …             …              …
7781                                   Zoom   United States       Children
7782                                   Zozo          Sweden         Dramas
7783                                 Zubaan           India         Dramas
7784                 Zulu Man in Japan              NaN  Documentaries
7786  ZZ TOP: THAT LITTLE OL' BAND FROM TEXAS  United Kingdom  Documentaries

      release_year  duration
1             2016        93
2             2011        78
3             2009        80
4             2008       123
6             2019        95
…              …         …
7781          2006        88
7782          2005        99
7783          2015       111
7784          2019        44
7786          2019        90

[5377 rows x 5 columns]
```

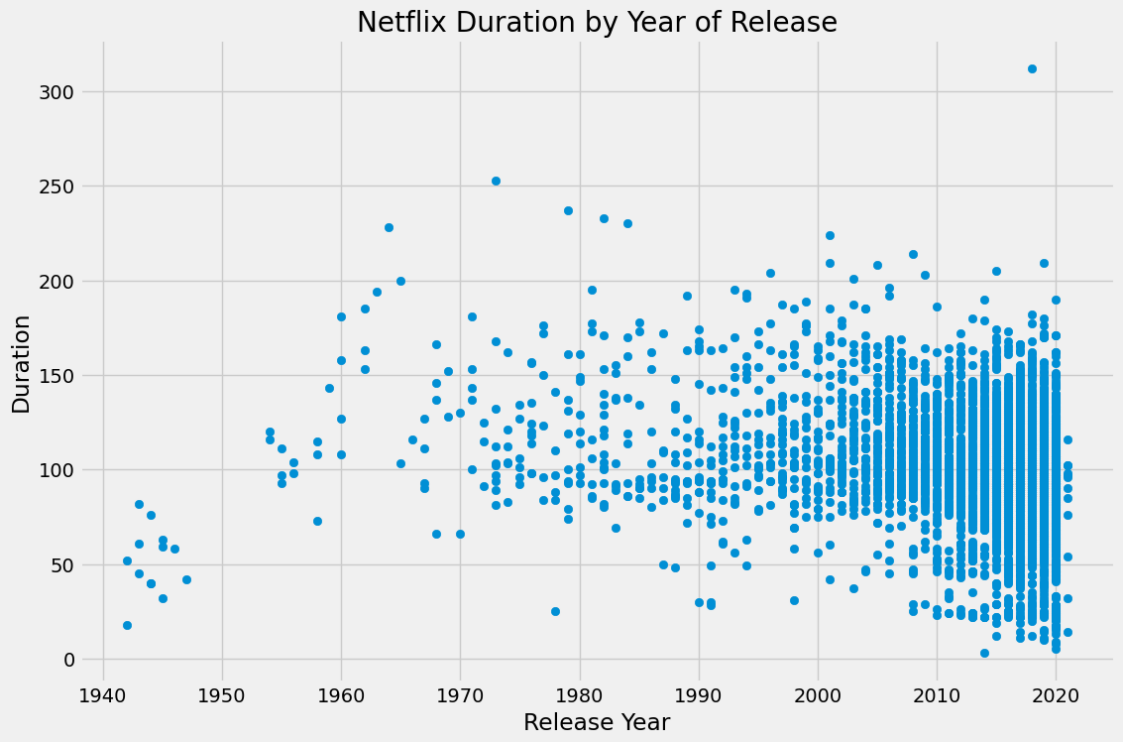
[523]:
```
netflix_df_movies_only = netflix_df.query("type == \"Movie\"")

netflix_movies_col_subset = netflix_df_movies_only[['title', 'country',␣
↪'genre', 'release_year', 'duration']]
netflix_movies_col_subset.head()
```

[523]:

	title	country	genre	release_year	duration
1	7:19	Mexico	Dramas	2016	93
2	23:59	Singapore	Horror Movies	2011	78
3	9	United States	Action	2009	80
4	21	United States	Dramas	2008	123
6	122	Egypt	Horror Movies	2019	95

[524]:
```
fig = plt.figure(figsize=(12, 8))

plt.scatter(netflix_movies_col_subset.release_year, netflix_movies_col_subset.
 ↪duration)
plt.title("Netflix Duration by Year of Release")
plt.xlabel("Release Year")
plt.ylabel("Duration")
```

[524]:
```
Text(0, 0.5, 'Duration')
```



[525]:
```
short_movie = netflix_movies_col_subset['duration'] < 60
netflix_movies_col_subset[short_movie].head(10)
```

[525]:

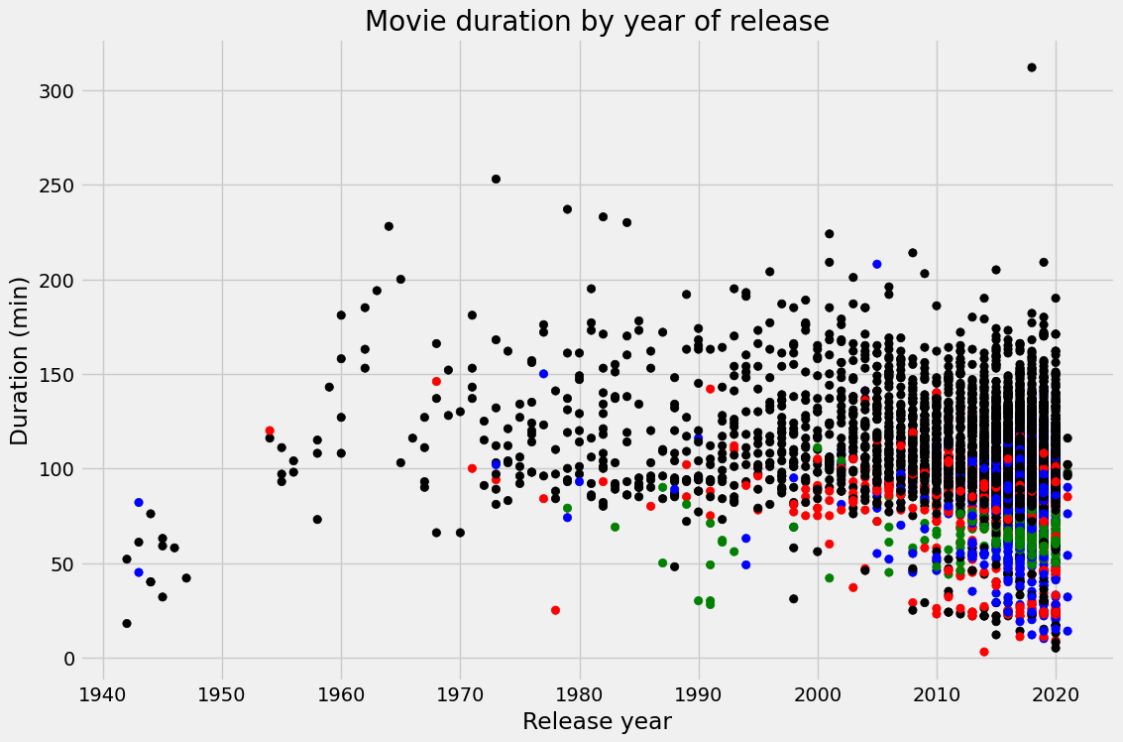
	title	country	\
35	#Rucker50	United States	
55	100 Things to do Before High School	United States	

```
67   13TH: A Conversation with Oprah Winfrey & Ava …            NaN
101                                3 Seconds Divorce         Canada
146                                   A 3 Minute Hug         Mexico
162  A Christmas Special: Miraculous: Tales of Lady…         France
171                          A Family Reunion Christmas  United States
177                   A Go! Go! Cory Carson Christmas  United States
178                   A Go! Go! Cory Carson Halloween            NaN
179                 A Go! Go! Cory Carson Summer Camp            NaN

             genre  release_year  duration
35   Documentaries          2016        56
55   Uncategorized          2014        44
67   Uncategorized          2017        37
101  Documentaries          2018        53
146  Documentaries          2019        28
162  Uncategorized          2016        22
171  Uncategorized          2019        29
177       Children          2020        22
178       Children          2020        22
179       Children          2020        21
```

[526]:
```
color = []

for lab, row in netflix_movies_col_subset.iterrows():
    if row["genre"] == "Children":
        color.append('red')
    elif row['genre'] == 'Documentaries':
        color.append('blue')
    elif row['genre'] == 'Stand-Up':
        color.append('green')
    else:
        color.append('black')
```

[527]:
```
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(12, 8))
plt.scatter(netflix_movies_col_subset.release_year, netflix_movies_col_subset.
 ↪duration, c=color)

plt.title("Movie duration by year of release")
plt.xlabel("Release year")
plt.ylabel("Duration (min)")
```

[527]:
```
Text(0, 0.5, 'Duration (min)')
```



11 NETFLIX

[528]:
```
netflix_df = pd.read_csv("week-3-data/netflix_data.csv")
netflix_df_movies_only = netflix_df.query("type == \"Movie\"")

netflix_us_only = netflix_df_movies_only[(netflix_df_movies_only["country"] ==␣
↪"United States")]
netflix_us_only
```

[528]:
```
     show_id   type                          title         director  \
3         s4  Movie                              9      Shane Acker
4         s5  Movie                             21   Robert Luketic
7         s8  Movie                            187   Kevin Reynolds
10       s11  Movie                           1922     Zak Hilditch
14       s15  Movie                           3022       John Suits
…       …    …                                …               …
7758   s7759  Movie  Zack and Miri Make a Porno      Kevin Smith
7771   s7772  Movie                           Zion       Floyd Russ
7774   s7775  Movie                         Zodiac    David Fincher
7778   s7779  Movie                      Zombieland  Ruben Fleischer
7781   s7782  Movie                           Zoom     Peter Hewitt

                                                  cast        country  \
```

```
3     Elijah Wood, John C. Reilly, Jennifer Connelly…  United States
4     Jim Sturgess, Kevin Spacey, Kate Bosworth, Aar…  United States
7     Samuel L. Jackson, John Heard, Kelly Rowan, Cl…  United States
10    Thomas Jane, Molly Parker, Dylan Schmid, Kaitl…  United States
14    Omar Epps, Kate Walsh, Miranda Cosgrove, Angus…  United States
…                                                   …              …
7758  Seth Rogen, Elizabeth Banks, Craig Robinson, J…  United States
7771                                       Zion Clark  United States
7774  Mark Ruffalo, Jake Gyllenhaal, Robert Downey J…  United States
7778  Jesse Eisenberg, Woody Harrelson, Emma Stone, …  United States
7781  Tim Allen, Courteney Cox, Chevy Chase, Kate Ma…  United States

             date_added  release_year  duration  \
3     November 16, 2017          2009        80
4       January 1, 2020          2008       123
7      November 1, 2019          1997       119
10     October 20, 2017          2017       103
14       March 19, 2020          2019        91
…                   …             …         …
7758    October 1, 2018          2008       101
7771    August 10, 2018          2018        12
7774  November 20, 2019          2007       158
7778   November 1, 2019          2009        88
7781    January 11, 2020          2006        88

                                              description              genre
3     In a postapocalyptic world, rag-doll robots hi…              Action
4     A brilliant group of students become card-coun…              Dramas
7     After one of his high school students attacks …              Dramas
10    A farmer pens a confession admitting to his wi…              Dramas
14    Stranded when the Earth is suddenly destroyed …  Independent Movies
…                                                   …                   …
7758  Zack and Miri make and star in an adult film t…            Comedies
7771  Born without legs and stuck in foster care for…       Documentaries
7774  A political cartoonist, a crime reporter and a…         Cult Movies
7778  Looking to survive in a world taken over by zo…            Comedies
7781  Dragged from civilian life, a former superhero…            Children

[2100 rows x 11 columns]
```

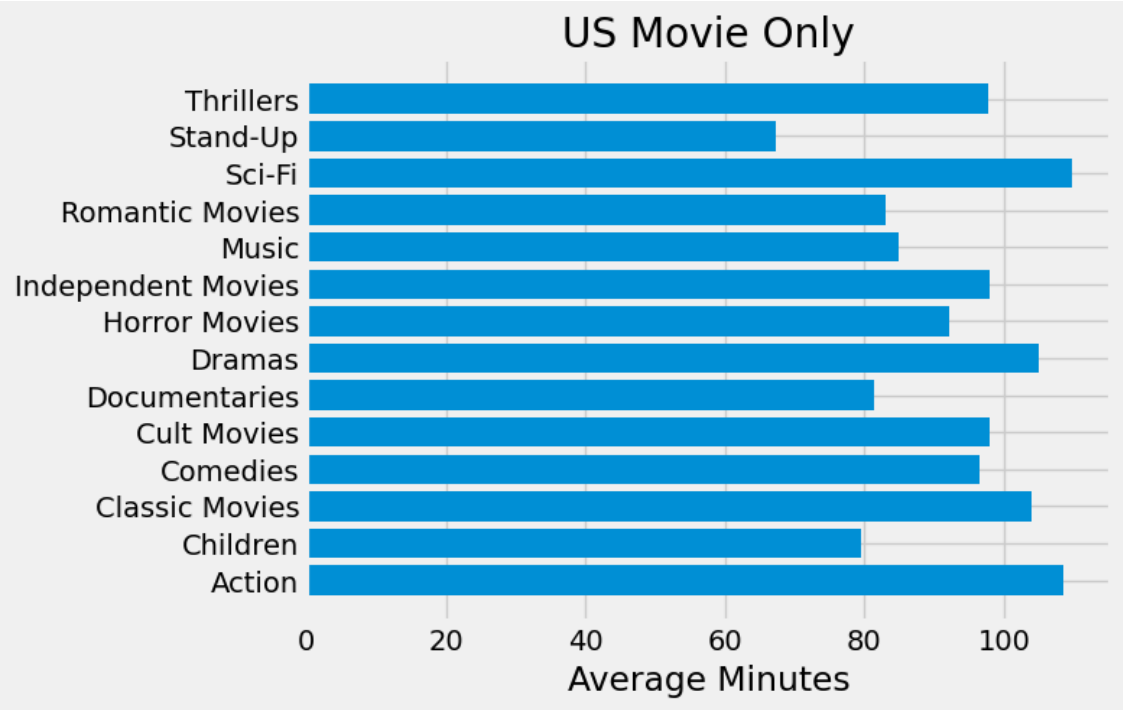
[529]:
```
long_genre = netflix_us_only.groupby('genre')[['release_year', 'duration']].
 ↪mean()
result = long_genre['duration'] > 50
long_genre = long_genre[result][['release_year', 'duration']]
long_genre
```

[529]:

genre	release_year	duration
Action	2008.922449	108.428571
Children	2011.917241	79.441379
Classic Movies	1968.404762	103.880952
Comedies	2012.445122	96.576220
Cult Movies	1990.111111	97.888889
Documentaries	2016.128463	81.372796
Dramas	2012.984085	104.965517
Horror Movies	2014.414414	92.117117
Independent Movies	2016.000000	98.000000
Music	2016.600000	85.000000
Romantic Movies	2017.500000	83.000000
Sci-Fi	2011.833333	109.833333
Stand-Up	2014.449275	67.256039
Thrillers	2013.300000	97.775000

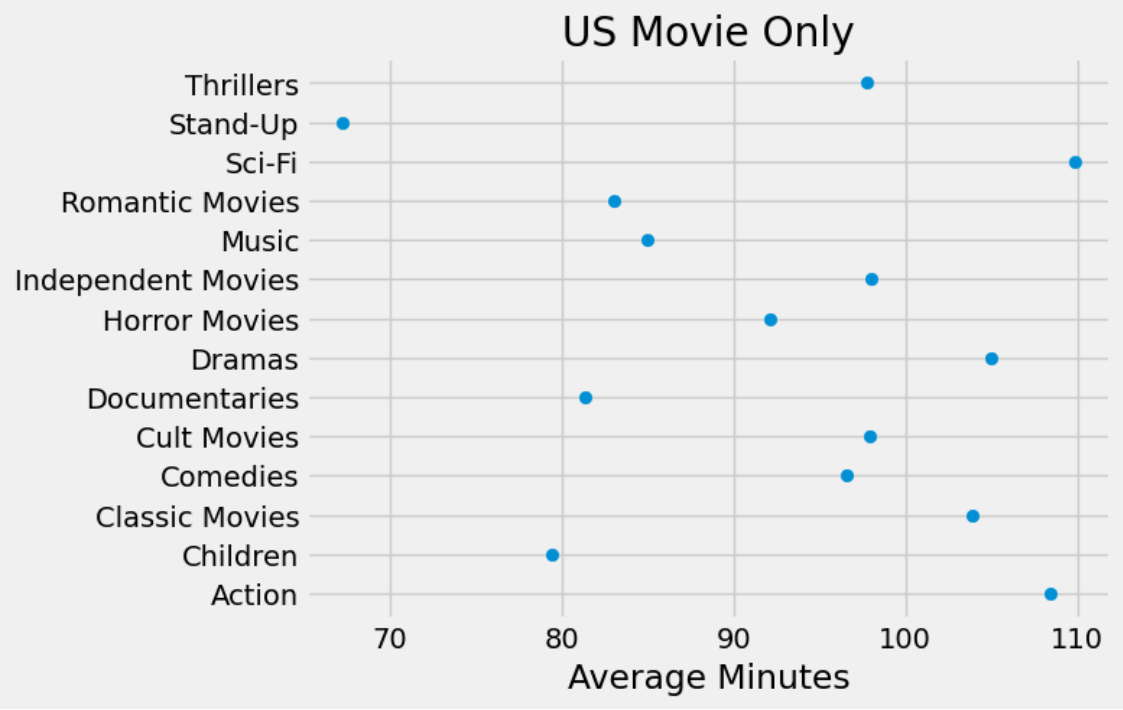
[530]:
```
x = long_genre.index
plt.barh(x, long_genre.duration)
plt.title("US Movie Only")
plt.xlabel("Average Minutes")
```

[530]: Text(0.5, 0, 'Average Minutes')



[531]:
```
plt.scatter(long_genre.duration, x)
plt.title("US Movie Only")
plt.xlabel("Average Minutes")
# plt.xticks([70, 80, 90, 100, 110], ['70', '80', '90', '100', '110'])
```

[531]:
```
Text(0.5, 0, 'Average Minutes')
```



[532]:
```
data_2016_to_2020 = netflix_df[(netflix_df['release_year'] >= 2016) &␣
↪(netflix_df['release_year'] <= 2020)][['title', 'country', 'genre',␣
↪'release_year', 'duration']]
data_2016_to_2020
```

[532]:

	title	country	\
0	3%	Brazil	
1	7:19	Mexico	
5	46	Turkey	
6	122	Egypt	
8	706	India	
…	…	…	
7779	Zona Rosa	Mexico	
7780	Zoo	India	
7784	Zulu Man in Japan	NaN	
7785	Zumbo's Just Desserts	Australia	
7786	ZZ TOP: THAT LITTLE OL' BAND FROM TEXAS	United Kingdom	

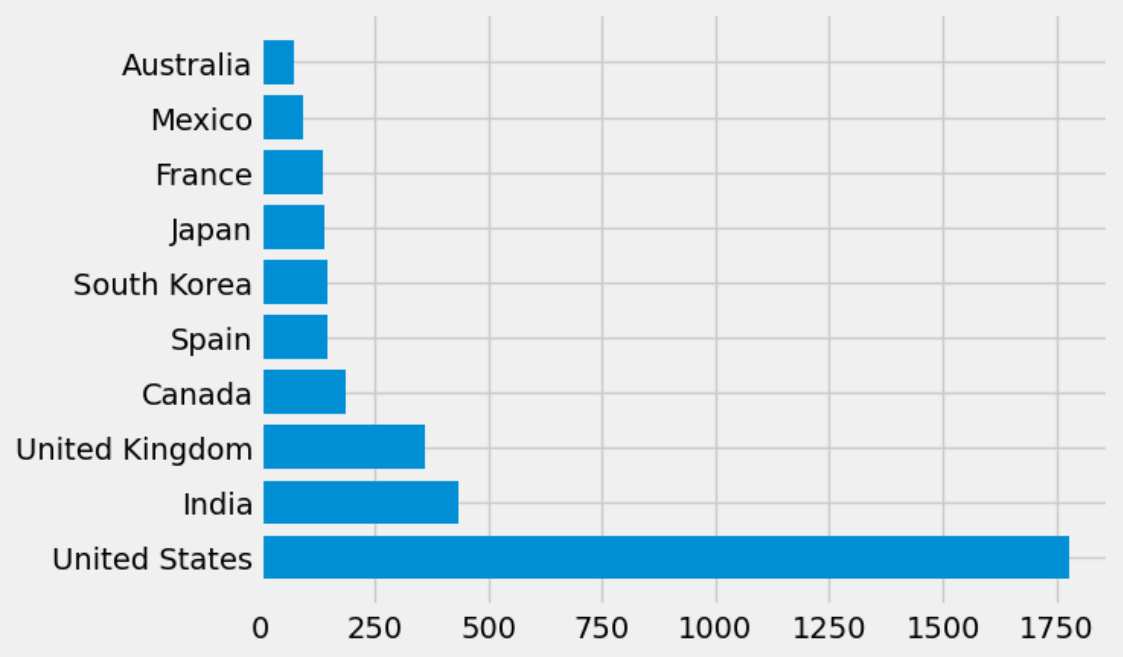
```
                 genre  release_year  duration
0     International TV          2020         4
1              Dramas          2016        93
5     International TV          2016         1
6        Horror Movies          2019        95
8        Horror Movies          2019       118
…                  …           …         …
7779  International TV          2019         1
7780           Dramas          2018        94
7784     Documentaries          2019        44
7785  International TV          2019         1
7786     Documentaries          2019        90

[4879 rows x 5 columns]
```

[538]:
```
count_country = data_2016_to_2020.groupby('country').count().
 ↪sort_values(by="title", ascending=False).head(10).reset_index()
count_country
```

[538]:
```
          country  title  genre  release_year  duration
0   United States   1776   1776          1776      1776
1           India    434    434           434       434
2  United Kingdom    360    360           360       360
3          Canada    186    186           186       186
4           Spain    148    148           148       148
5     South Korea    147    147           147       147
6           Japan    140    140           140       140
7          France    136    136           136       136
8          Mexico     94     94            94        94
9       Australia     74     74            74        74
```

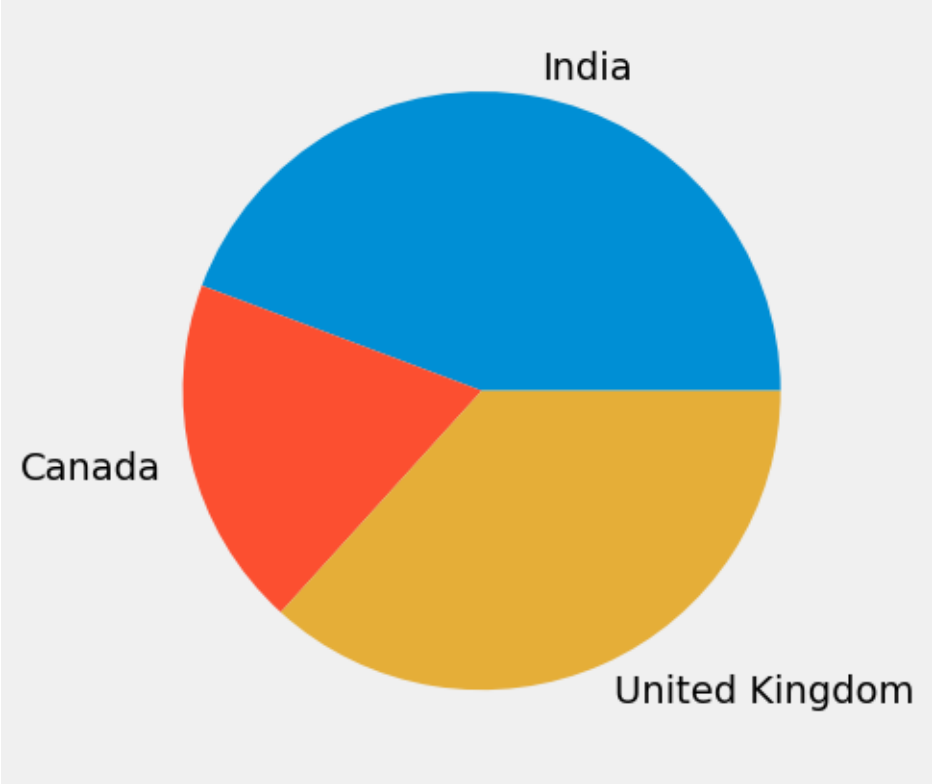
[534]:
```
x = count_country.index
plt.barh(count_country.country, count_country.duration)
```

[534]:
```
<BarContainer object of 10 artists>
```



[535]:
```
# count_country[count_country['country'] == "India"]
india = count_country[count_country['country'] == "India"][['duration']].
 ↪duration.values[0]
canada = count_country[count_country['country'] == "Canada"][['duration']].
 ↪duration.values[0]
uk = count_country[count_country['country'] == "United Kingdom"][['duration']].
 ↪duration.values[0]


y = np.array([india, canada, uk])
mylabels = ["India", "Canada", "United Kingdom"]

plt.pie(y, labels=mylabels)
```

[535]:
```
([<matplotlib.patches.Wedge at 0x28151f550>,
  <matplotlib.patches.Wedge at 0x28151f460>,
  <matplotlib.patches.Wedge at 0x28151fca0>],
 [Text(0.19641257269743187, 1.0823225495601467, 'India'),
  Text(-1.0691939422638115, -0.25850399189639145, 'Canada'),
  Text(0.4452616659030199, -1.0058538904220968, 'United Kingdom')])
```



[536]:
```
f = plt.figure(figsize=(18,18))

my_list = []
for lab, row in count_country.iterrows():
    my_list.append(row["title"])

print(my_list)

def func(pct, allvals):
    absolute = int(np.round(pct/100.*np.sum(allvals)))
    print(absolute)
    return f"{pct:.1f}%\n({absolute:d})"

plt.pie(count_country.duration, labels=count_country.country, autopct=lambda␣
 ↪pct: func(pct, my_list),
        pctdistance=0.85, startangle=90)
centre_circle = plt.Circle((0, 0), 0.55, fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.title("Number of titles released by top 10 countries")
```

```
[1776, 434, 360, 186, 148, 147, 140, 136, 94, 74]
1776
```

```
434
360
186
148
147
140
136
94
74
```

[536]: `Text(0.5, 1.0, 'Number of titles released by top 10 countries')`