1. **Line plot (1)**

With matplotlib, you can create a bunch of different plots in Python. The most basic plot is the line plot. A general recipe is given here.

import matplotlib.pyplot as plt

plt.plot(x,y)

plt.show()

In the video, you already saw how much the world population has grown over the past years. Will it continue to do so? The world bank has estimates of the world population for the years 1950 up to 2100. The years are loaded in your workspace as a list called year, and the corresponding populations as a list called pop.

* [**print()**](https://docs.python.org/3/library/functions.html#print) the last item from both the year and the pop list to see what the predicted population for the year 2100 is. Use two print() functions.
* Before you can start, you should import matplotlib.pyplot as plt. pyplot is a sub-package of matplotlib, hence the dot.
* Use **[plt.plot()](http://matplotlib.org/api/pyplot_api.html" \l "matplotlib.pyplot.plot" \t "_blank)** to build a line plot. year should be mapped on the horizontal axis, pop on the vertical axis. Don't forget to finish off with the [**show()**](http://matplotlib.org/api/pyplot_api.html#matplotlib.pyplot.show) function to actually display the plot.

# Line Plot (2): Interpretation

Have another look at the plot you created in the previous exercise; it's shown on the right. Based on the plot, in **approximately** what year will there be more than ten billion human beings on this planet?

**Answer is 2060**

1. **Line plot (3)**

Now that you've built your first line plot, let's start working on the data that professor Hans Rosling used to build his beautiful bubble chart. It was collected in 2007. Two lists are available for you:

* life\_exp which contains the life expectancy for each country and
* gdp\_cap, which contains the GDP per capita (i.e. per person) for each country expressed in US Dollars.

GDP stands for Gross Domestic Product. It basically represents the size of the economy of a country. Divide this by the population and you get the GDP per capita.

matplotlib.pyplot is already imported as plt, so you can get started straight away.

* Print the last item from both the list gdp\_cap, and the list life\_exp; it is information about Zimbabwe.
* Build a line chart, with gdp\_cap on the x-axis, and life\_exp on the y-axis. Does it make sense to plot this data on a line plot?
* Don't forget to finish off with a **[plt.show()](http://matplotlib.org/api/pyplot_api.html" \l "matplotlib.pyplot.show" \t "_blank)** command, to actually display the plot.

# Scatter Plot (1)

When you have a time scale along the horizontal axis, the line plot is your friend. But in many other cases, when you're trying to assess if there's a correlation between two variables, for example, the scatter plot is the better choice. Below is an example of how to build a scatter plot.

import matplotlib.pyplot as plt

plt.scatter(x,y)

plt.show()

Let's continue with the gdp\_cap versus life\_exp plot, the GDP and life expectancy data for different countries in 2007. Maybe a scatter plot will be a better alternative?

Again, the matplotlib.pyplot package is available as plt.

* Change the line plot that's coded in the script to a scatter plot.
* A correlation will become clear when you display the GDP per capita on a logarithmic scale. Add the line plt.xscale('log').
* Finish off your script with **[plt.show()](http://matplotlib.org/api/pyplot_api.html" \l "matplotlib.pyplot.show" \t "_blank)** to display the plot.