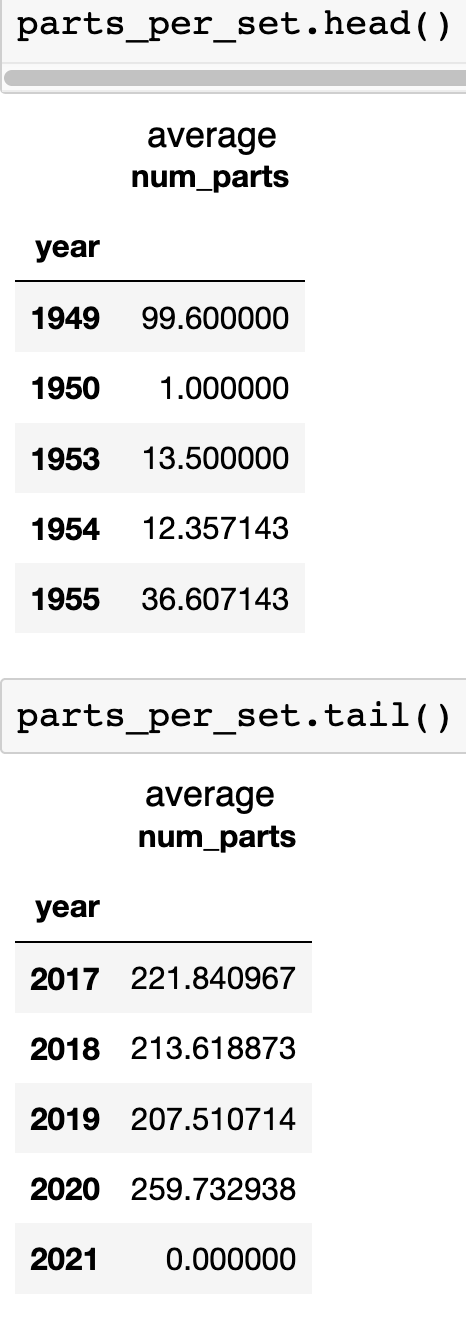
**Scatter Plots: Average Number of Parts per LEGO Set**

**Complexity Over Time**

Have LEGO sets become larger and more complex over time? Let's work out the average number of parts per LEGO set. This is the perfect time to revise how to use the .agg() function.

Challenge

Create a Pandas Series called parts\_per\_set that has the year as the index and contains the average number of parts per LEGO set in that year. Here's what you're looking to create:



.

.

..

...

..

.

**Solution: How to use .groupby() and .agg() together**

Once again, we're going to use the .groupby() and the .agg() function together to work this one out. However, this time we pass a dictionary to the .agg() function so that we will target the num\_parts column with the mean() function. That way, we group our data by year and then we average the number of parts for that year.

1. parts\_per\_set = sets.groupby('year').agg({'num\_parts': pd.Series.mean})

To visualise our parts\_per\_set data, let's create a scatter plot. A scatter plot simply uses dots to represent the values of each data point.

Challenge

See if you can use [the Matplotlib documentation](https://matplotlib.org/3.1.0/api/_as_gen/matplotlib.pyplot.scatter.html) to generate the scatter plot chart. Do you spot a trend in the chart? Again, you'll have to exclude the last two observations.

I'll provide the solution below.

.

.

..

...

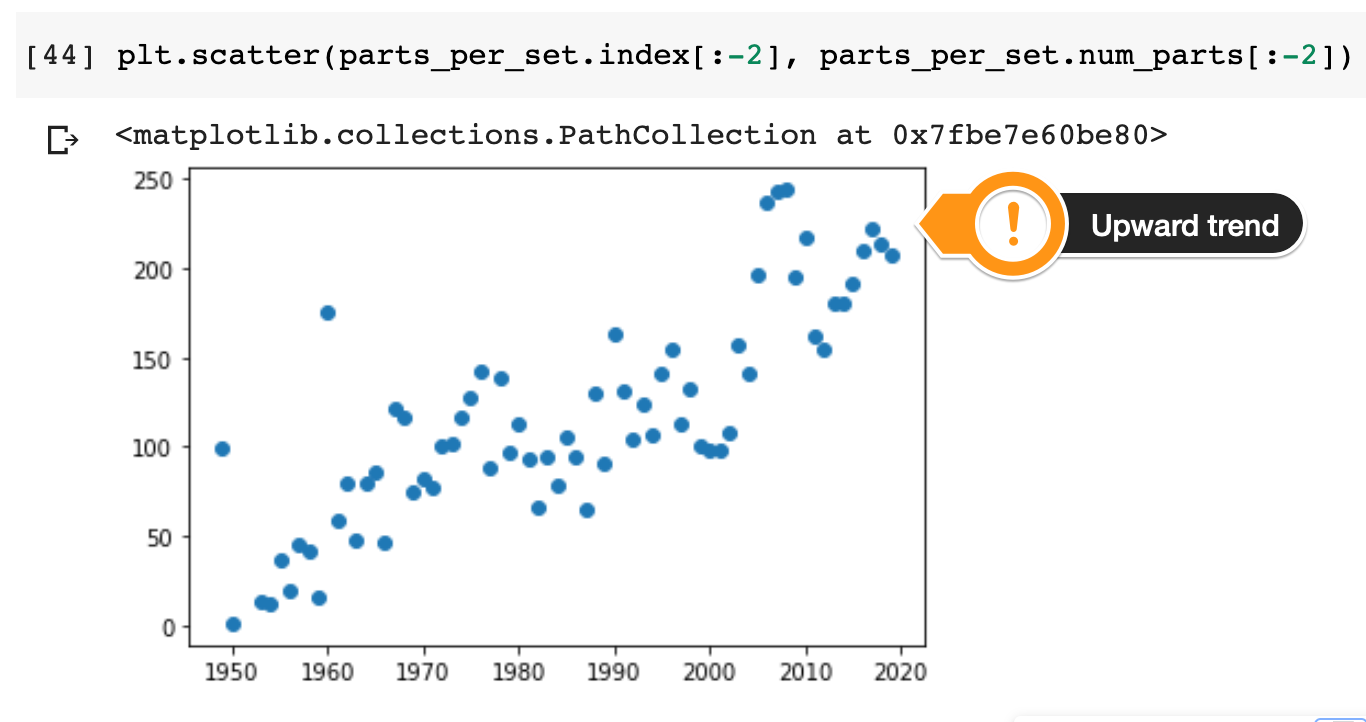
..

.

.

**Solution: Create a Scatter Plot**

We just need to call the .scatter() instead of the .plot() method to create the chart. For the x-values, we'll use the index of the parts\_per\_set Series (the years) and for the y-values, we'll use the values of the series (the column name happens to be num\_parts).



From the chart, we can definitely make out an upward trend in the size and complexity of the LEGO sets based on the average number of parts. In the 2010s the average set contained around 200 individual pieces, which is roughly double what average LEGO set used to contain in the 1960s.