# Solution Review: Visualizing Auto MPG Dataset

This lesson provides the solutions to the previous challenges.

**WE'LL COVER THE FOLLOWING**

- Scatter plot
- Bar plot
- Line plot

## Scatter plot

```
import pandas as pd
import seaborn as sns

# Load data
def read_csv():
    # Define the column names as a list
    names = ["mpg", "cylinders", "displacement", "horsepower", "weight", "acceleration", "moc
    # Read in the CSV file from the webpage using the defined column names
    df = pd.read_csv("auto-mpg.data", header=None, names=names, delim_whitespace=True)
    return df

# Create the scatter plot
def scatter_plot(df):
    sns.lmplot(x="displacement", y="acceleration", data = df)
    # Remove excess chart lines and ticks for a nicer looking plot
    sns.despine()

# calling function
scatter_plot(read_csv())
```

According to the problem statement, we need to find the relationship between `acceleration` and `displacement`. To plot a visualization, we import `seaborn` module at **line 2**. Before doing it, we have to read the data first. There's no need to explain how to read the data, as we studied that in detail previously. Dataset is read from **line 5** to **line 10**.

Moving towards the main implementation, look at the header of the `scatter_plot(df)` function at **line 13**. It takes *one* arguments as input:

- `df`: A dataframe containing the dataset in the form of a matrix.

**Line 14** is the most important line. We are using a built-in function `lmplot` from `seaborn` library which takes *three* main argument:

- `x`: Column whose values are to plotted at the x-axis
- `y`: Column whose values are to plotted at the y-axis
- `data`: Dataframe containing data in the form of a matrix

We set `displacement` as `x` and `acceleration` as `y` according to the problem statement. At **line 16**, we are just trying to make the plot look clean using function `despine()`.

At **line 19**, we are calling the function `scatter_plot(read_csv())`. First control will transfer to `read_csv()` at **line 5** and we'll get a dataframe. Then control will go to **line 13** and plot will be visualized.

The plot clearly shows the negative correlation between these two data values, which means `acceleration` is the inverse of `displacement`.

## Bar plot

```
import pandas as pd
import seaborn as sns

# Load data
def read_csv():
    # Define the column names as a list
    names = ["mpg", "cylinders", "displacement", "horsepower", "weight", "acceleration", "mod
    # Read in the CSV file from the webpage using the defined column names
    df = pd.read_csv("auto-mpg.data", header=None, names=names, delim_whitespace=True)
    return df


def bar_plot(df):
    df = df[df.car_name.str.contains('ford')]
    df = df[df["model_year"].isin([75])]

    # Create the bar plot
    sns.barplot(df['car_name'], df['cylinders'])
    # Remove excess chart lines and ticks for a nicer looking plot
    sns.despine()
```

```
bar_plot(read_csv())
```

According to the problem statement, we need to compare the `cylinders` of all the cars from **1975** `model_year` and `ford` company. To plot a visualization, we import `seaborn` module at **line 2**. Before doing it we definitely have to read the data first. There's no need to explain how to read the data, as we studied that in detail previously. Dataset is read from **line 5** to **line 10**.

Moving towards the main implementation, look at the header of the `bar_plot(df)` function at **line 13**. It takes *one* arguments as input:

- `df`: A dataframe containing the dataset in the form of a matrix.

Before plotting, we need to filter out the data we need. Look at **line 14**. Here we are considering all the rows for which the value in `car_name` would have `ford` as a substring. For example, **chevrolet chevelle** will be discarded and **ford torino** will be considered as a data value. Next, we are keeping all the rows for which `model_year` is **1975**. So all the instances having `model_year` other than **1975** will be discarded.

**Line 18** is the most important line. We are using a built-in function `barplot` from `seaborn` library which takes *two* main argument:

- **column1**: Column whose values are to plotted at the x-axis
- **column2**: Column whose values are to plotted at the y-axis

We set `car_name` as `x` and `cylinders` as `y` according to the problem statement. At **line 20**, we are just trying to make the plot look clean using function `despine()`.

At **line 22**, we are calling the function `bar_plot(read_csv())`. First control will transfer to `read_csv()` at **line 5** and we'll get a dataframe. Then control will go to **line 13** and plot will be visualized.

The plot clearly shows that the `ford` company launches following *four* different models in **1975**:

- `ford maverick` having **6** cylinders

- `ford ltd` having **8** cylinders
- `ford mustang ii` having **8** cylinders
- `ford pinto` having **5** cylinders

## Line plot #

```
import seaborn as sns            # importing seaborn functionality
import pandas as pd

# Load dataset
def read_csv():
    # Define the column names as a list
    names = ["mpg", "cylinders", "displacement", "horsepower", "weight", "acceleration", "mod
    # Read in the CSV file from the webpage using the defined column names
    df = pd.read_csv("auto-mpg.data", header=None, names=names, delim_whitespace=True)
    return df

def line_plot(df):

    # filtering the dataset to obtain the January records for all years
    df=df[df.car_name.str.contains('ford')]

    #plotting a line graph
    sns.lineplot(df.model_year, df.weight)

line_plot(read_csv())
```

According to the problem statement, we need to plot the change in the `weight` of all the `ford` cars. To plot a visualization, we import the `seaborn` module at **line 2**. Before doing it, we have to read the data first. There's no need to explain how to read the data, as we studied that in detail previously. Dataset is read from **line 5** to **line 10**.

Moving towards the main implementation, look at the header of the `line_plot(df)` function at **line 12**. It takes *one* arguments as input:

- `df`: A dataframe containing the dataset in the form of a matrix.

Before plotting, we need to filter out the data we need. Look at **line 15**. Here we are considering all the rows for which the value in `car_name` would have `ford` as a substring. For example, **chevrolet chevelle** will be discarded, and **ford torino** will be considered as a data value.

**Line 18** is the most important line. We are using a built-in function `lineplot()` from `seaborn` library which takes *two* main argument:

- **column1**: Column whose values are to plotted at the x-axis
- **column2**: Column whose values are to plotted at the y-axis

We set `model_year` as `x` and `weight` as `y` according to the problem statement.

At **line 20**, we are calling the function `line_plot(read_csv())`. First control will transfer to `read_csv()` at **line 5** and we'll get a dataframe. Then control will go to **line 12** and plot will be visualized.

The plot marks the points, each for one `model_year`. The point represents the average `weight` of all the cars from the `ford` company in a year. Then those points are connected via a *line* to form a plot.

That's all about the solutions. In the next lesson, there's a quick quiz for you to evaluate your concepts regarding visualization. Good Luck!