In [1]:
```
from IPython.display import Image
Image(filename='AnkurKhanna.PNG', height=340, width=900)
```

DATA Visualization Using MATPLOTLIB

1. Matplotlib is the **most popular plotting library** for Python
2. You can install matplotlib by going to Anaconda C-Prompt and using `pip install matplotlib` OR `conda install matplotlib`

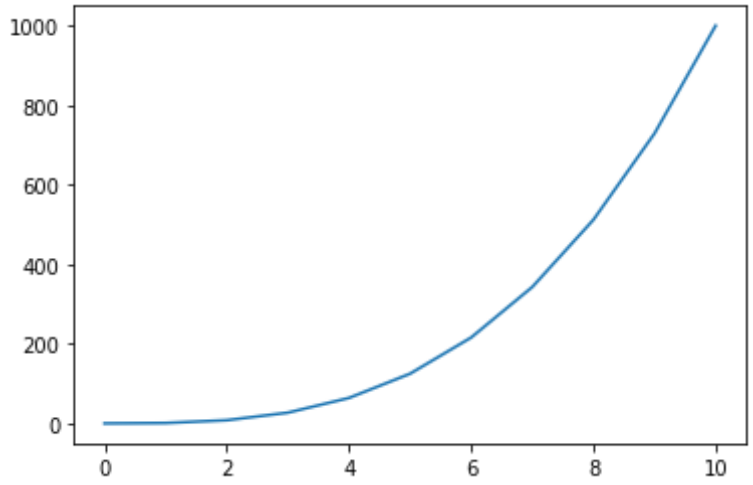
In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [3]:
```
x = np.arange(0,11)
y = x**3
```

LINE CHARTS

In [4]:
```
plt.plot(x,y)
```

Out[4]:
```
[<matplotlib.lines.Line2D at 0x1b00e2834c8>]
```



But these charts are not useful without:

- (a) Axis Labels

- (b) Chart Title

Adding Axis-Labels/Chart Titles to the Charts

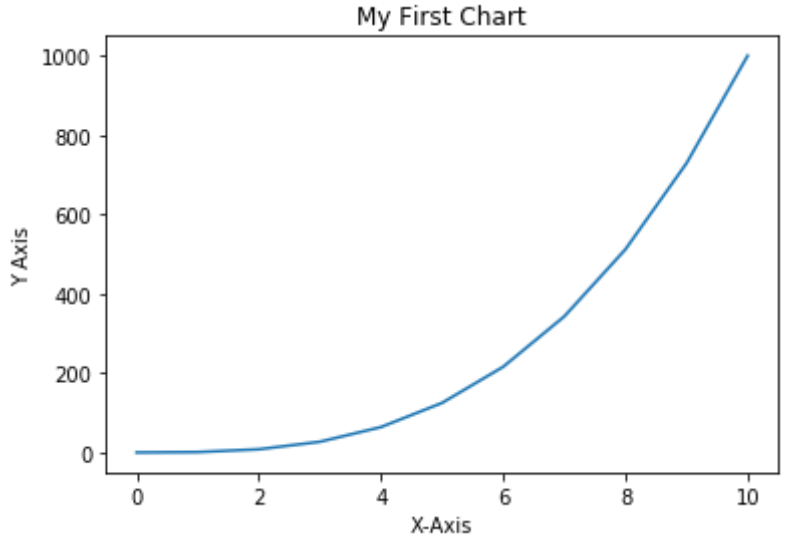
We can add the Axis-labels and the Chart Title using the following functions in Matplotlib.

- plt.title('Title of the Chart')
- plt.ylabel('Y axis Name')
- plt.xlabel('X axis Name')

Using the above functions let us name the axis and the chart title of the above chart. Let's use the following:

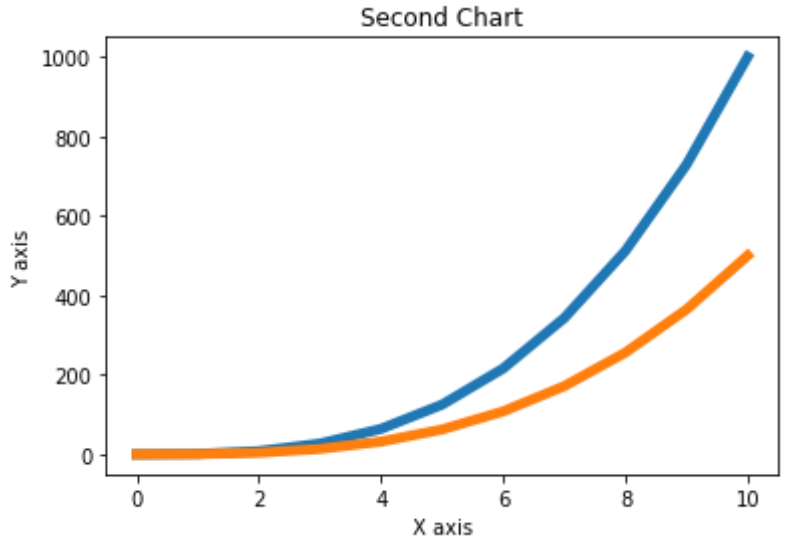
In [5]:
```
plt.plot(x,y)
plt.title('My First Chart') #— Where "Epic Info" is the title/name of t
he chart
plt.ylabel('Y Axis') #— Where "Y-Axis" is the name of the y-axis
```

```
plt.xlabel('X-Axis') #– Where “X-Axis” is the name of the x-axis
plt.show()
```



MULITPLE LINE CHARTS

In [6]:
```
x = np.arange(0,11)
y = x**3

x2 = np.arange(0,11)
y2 = (x**3)/2

plt.plot(x,y,linewidth=5) #-Where linewidth determines the width of the
 line in the chart
plt.plot(x2,y2,linewidth=5)
plt.title('Second Chart')
plt.ylabel('Y axis')
plt.xlabel('X axis')
plt.show()
```



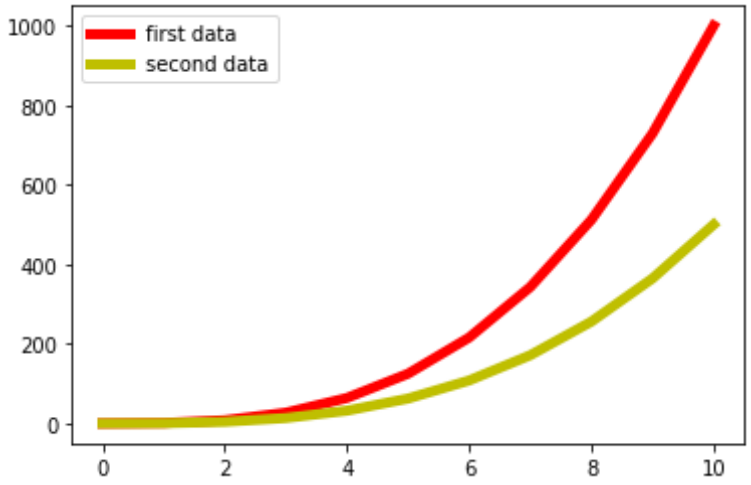
Adding/Editing Colors/Legends to the Chart:

We can also edit the colors of the line chart and add legends to the chart to make them more interactive and informative.

We can edit the color of the lines using the following prompt:

In [7]:
```
plt.plot(x,y,'r', label = 'first data', linewidth=5)
plt.plot(x2,y2,'y', label='second data', linewidth=5)
plt.legend()
```

Out[7]:
```
<matplotlib.legend.Legend at 0x1b01235cb48>
```



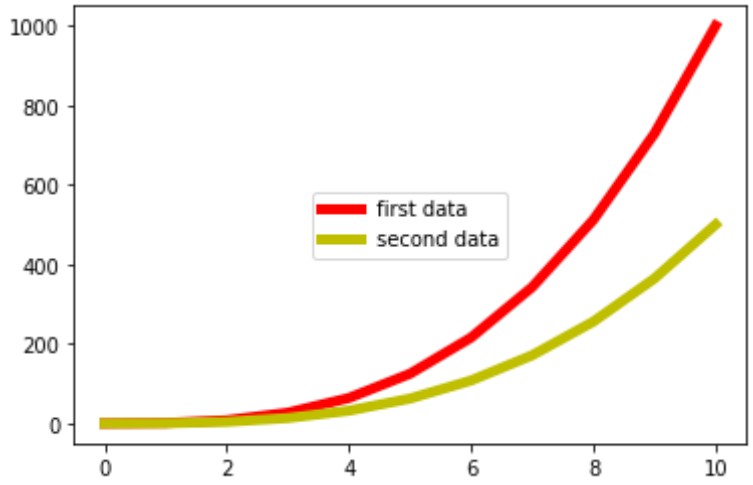
we can also **change the location of the legend** using the following prompts: plt.legend(loc=0)

0 is for default(auto-selection), 1, 2, 3, 4

Let us use one of them for illustration purposes:

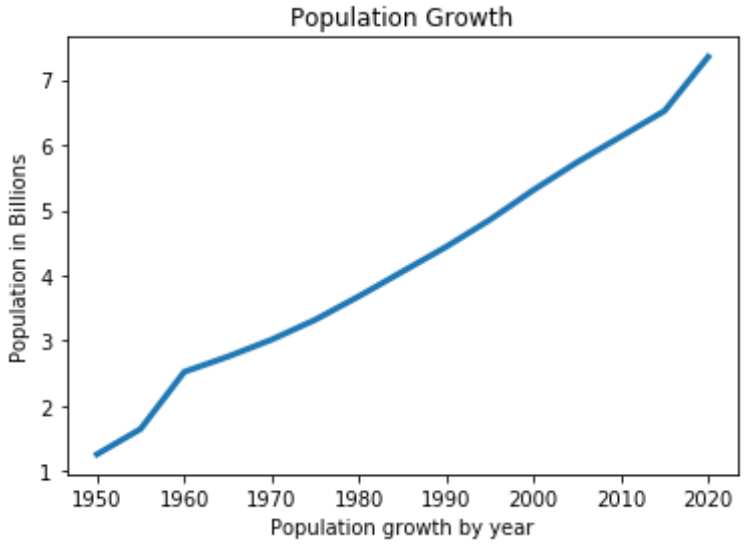
In [8]:
```
plt.plot(x,y,'r', label = 'first data', linewidth=5)
plt.plot(x2,y2,'y', label='second data', linewidth=5)
plt.legend(loc=10)
```

Out[8]:
```
<matplotlib.legend.Legend at 0x1b0122d4bc8>
```



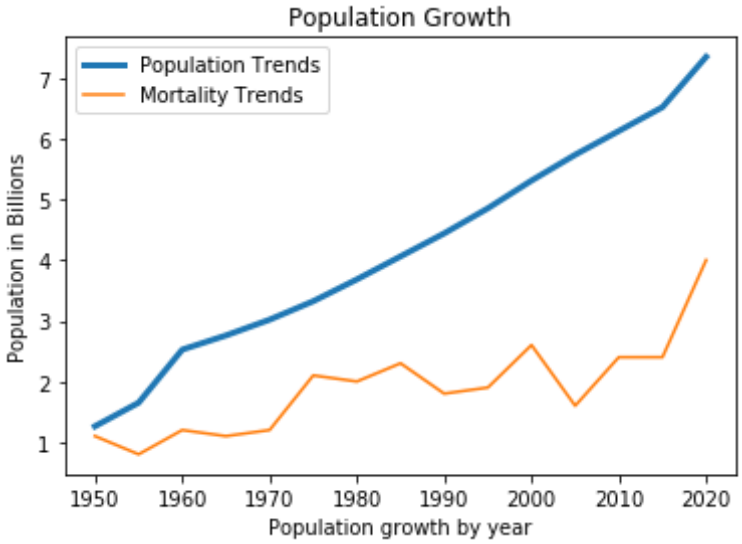
In [9]:
```
years=[1950,1955,1960,1965,1970,1975, 1980,1985,1990,1995,2000,2005,201
0,2015,2020]
pops =np.array([1262,1650,2525,2758,3018,3322,3682,4061,4440,4853,5310,
5735,6127,6520,7349])
pops_b = pops/1000
plt.plot(years,pops_b, linewidth = 3)
plt.ylabel("Population in Billions")
plt.xlabel("Population growth by year")
plt.title("Population Growth")
```

Out[9]:
```
Text(0.5, 1.0, 'Population Growth')
```



In [10]:
```
years=[1950,1955,1960,1965,1970,1975, 1980,1985,1990,1995,2000,2005,201
0,2015,2020]
pops =np.array([1262,1650,2525,2758,3018,3322,3682,4061,4440,4853,5310,
5735,6127,6520,7349])
pops_b = pops/1000
deaths = [1.1,0.8,1.2,1.1,1.2,2.1,2.0,2.3,1.8,1.9,2.6,1.6,2.4,2.4,4.0]
plt.plot(years,pops_b, linewidth = 3, label = 'Population Trends')
plt.plot(years,deaths,label = 'Mortality Trends')
plt.ylabel("Population in Billions")
plt.xlabel("Population growth by year")
plt.title("Population Growth")
plt.legend()
```

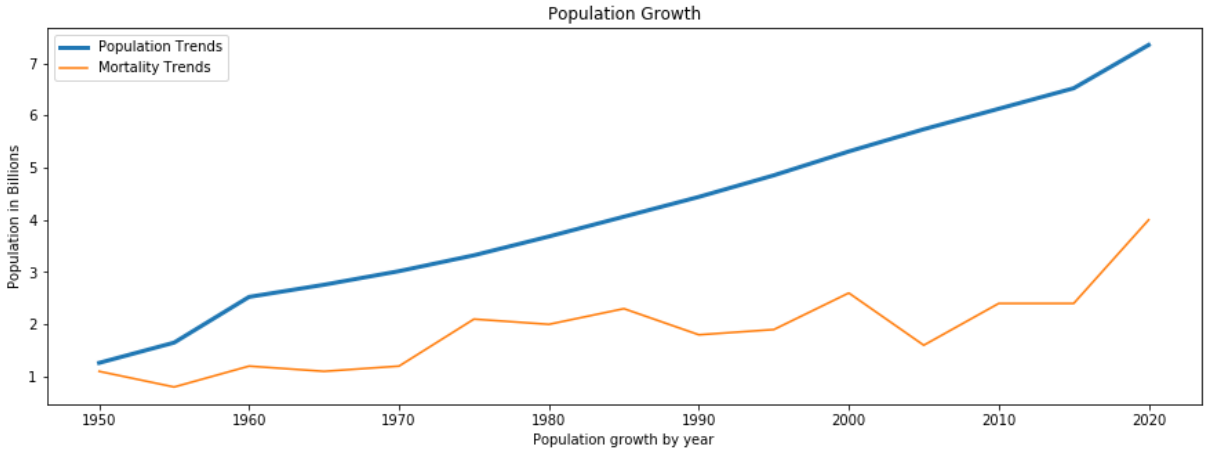
Out[10]:
```
<matplotlib.legend.Legend at 0x1b0123eae48>
```



Adjusting the Size of the Figure

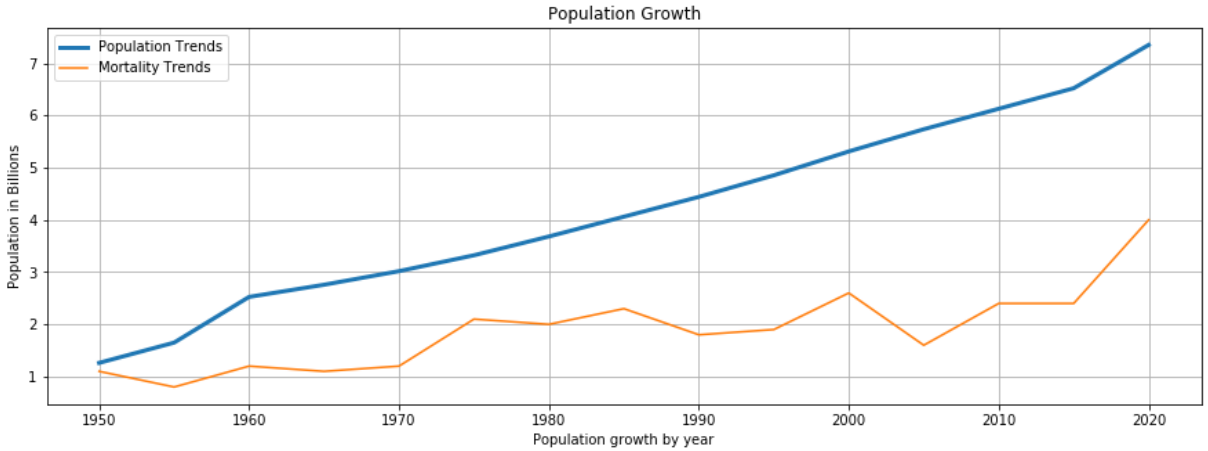
In [11]:
```
plt.figure(figsize=(15,5))
years=[1950,1955,1960,1965,1970,1975, 1980,1985,1990,1995,2000,2005,201
0,2015,2020]
pops =np.array([1262,1650,2525,2758,3018,3322,3682,4061,4440,4853,5310,
5735,6127,6520,7349])
pops_b = pops/1000
deaths = [1.1,0.8,1.2,1.1,1.2,2.1,2.0,2.3,1.8,1.9,2.6,1.6,2.4,2.4,4.0]
plt.plot(years,pops_b, linewidth = 3, label = 'Population Trends')
plt.plot(years,deaths,label = 'Mortality Trends')
plt.ylabel("Population in Billions")
plt.xlabel("Population growth by year")
plt.title("Population Growth")
plt.legend()
```

Out[11]:
```
<matplotlib.legend.Legend at 0x1b012457c08>
```



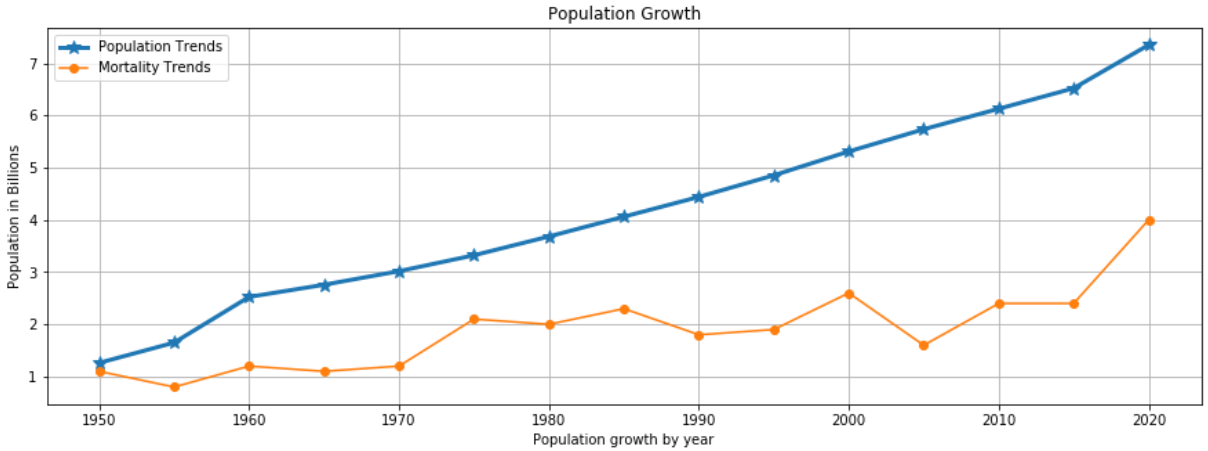
Adding Background Grid

In [12]:
```
plt.figure(figsize=(15,5))
years=[1950,1955,1960,1965,1970,1975, 1980,1985,1990,1995,2000,2005,201
0,2015,2020]
pops =np.array([1262,1650,2525,2758,3018,3322,3682,4061,4440,4853,5310,
5735,6127,6520,7349])
pops_b = pops/1000
deaths = [1.1,0.8,1.2,1.1,1.2,2.1,2.0,2.3,1.8,1.9,2.6,1.6,2.4,2.4,4.0]
plt.plot(years,pops_b, linewidth = 3, label = 'Population Trends')
plt.plot(years,deaths,label = 'Mortality Trends')
plt.ylabel("Population in Billions")
plt.xlabel("Population growth by year")
plt.title("Population Growth")
plt.legend()
plt.grid(True)
```



Adding Markers & Marker Size

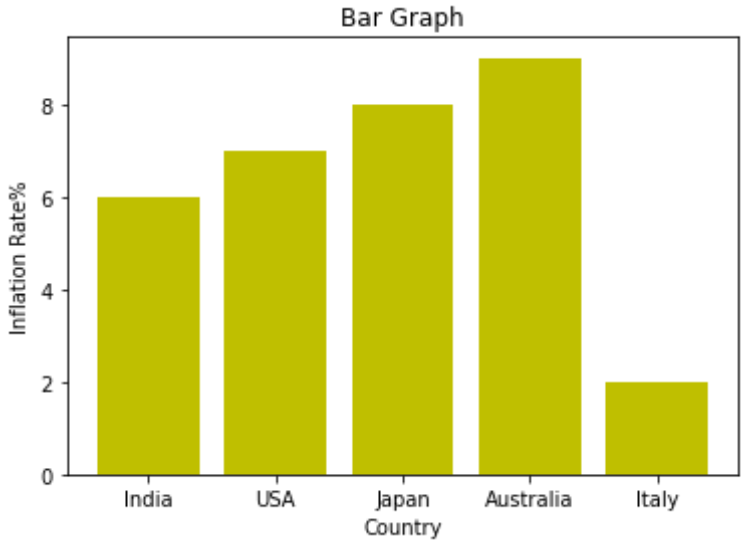
In [13]:
```
plt.figure(figsize=(15,5))
years=[1950,1955,1960,1965,1970,1975, 1980,1985,1990,1995,2000,2005,201
0,2015,2020]
pops =np.array([1262,1650,2525,2758,3018,3322,3682,4061,4440,4853,5310,
5735,6127,6520,7349])
pops_b = pops/1000
deaths = [1.1,0.8,1.2,1.1,1.2,2.1,2.0,2.3,1.8,1.9,2.6,1.6,2.4,2.4,4.0]
plt.plot(years,pops_b, linewidth = 3, label = 'Population Trends', mark
er='*', markersize=10)
plt.plot(years,deaths,label = 'Mortality Trends', marker='o')
plt.ylabel("Population in Billions")
plt.xlabel("Population growth by year")
plt.title("Population Growth")
plt.legend()
plt.grid(True)
```



BAR CHARTS

In [14]:
```
x = ["India",'USA',"Japan",'Australia','Italy']
y = [6,7,8,9,2]
plt.bar(x,y, label='Bars1', color ='y')
plt.xlabel("Country")
plt.ylabel("Inflation Rate%")
plt.title("Bar Graph")
```

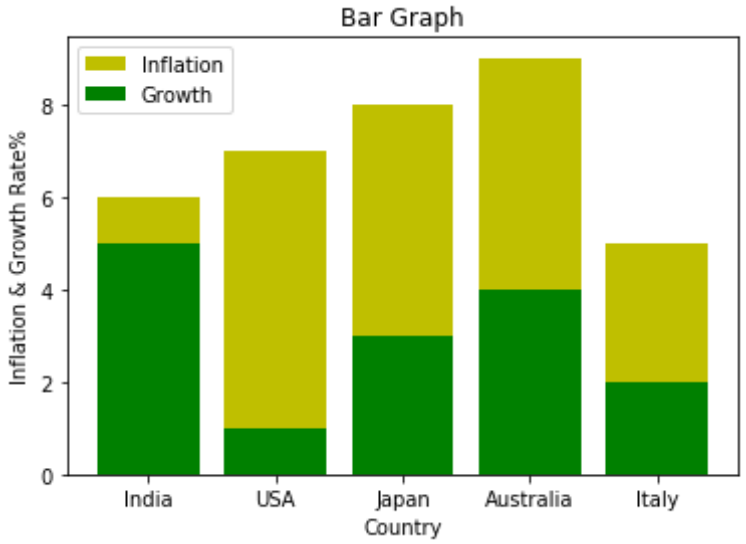
Out[14]:
```
Text(0.5, 1.0, 'Bar Graph')
```



MULTIPLE BARS

In [15]:
```
x = ["India",'USA',"Japan",'Australia','Italy']
y = [6,7,8,9,5]
x2 = ["India",'USA',"Japan",'Australia','Italy']
y2 = [5,1,3,4,2]
plt.bar(x,y, label='Inflation', color ='y')
plt.bar(x2,y2, label='Growth', color ='g')
plt.xlabel("Country")
plt.ylabel("Inflation & Growth Rate%")
plt.title("Bar Graph")
plt.legend()
```

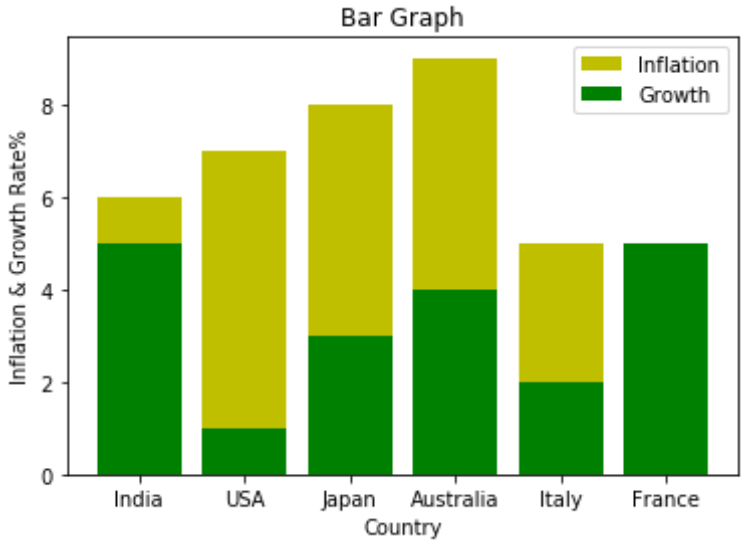
Out[15]:
```
<matplotlib.legend.Legend at 0x1b0125b7e88>
```



Additional Features will appear as Separate Bars

In [16]:
```
x = ["India",'USA',"Japan",'Australia','Italy']
y = [6,7,8,9,5]
x2 = ["India",'USA',"Japan",'Australia','Italy', 'France']
y2 = [5,1,3,4,2,5]
plt.bar(x,y, label='Inflation', color ='y')
plt.bar(x2,y2, label='Growth', color ='g')
plt.xlabel("Country")
plt.ylabel("Inflation & Growth Rate%")
plt.title("Bar Graph")
plt.legend()
```

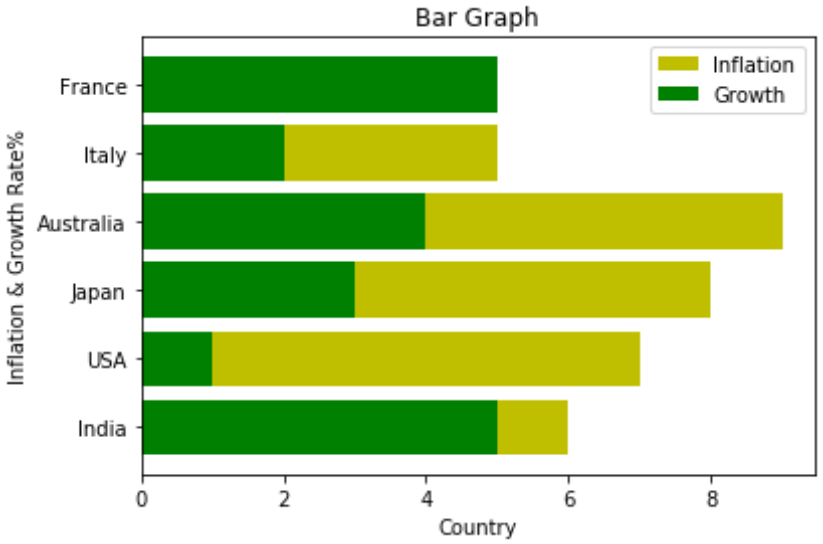
Out[16]:
```
<matplotlib.legend.Legend at 0x1b0127ae308>
```



HORIZONTAL BAR GRAPHS

In [17]:
```
x = ["India",'USA',"Japan",'Australia','Italy']
y = [6,7,8,9,5]
x2 = ["India",'USA',"Japan",'Australia','Italy', 'France']
y2 = [5,1,3,4,2,5]
plt.barh(x,y, label='Inflation', color ='y')
plt.barh(x2,y2, label='Growth', color ='g')
plt.xlabel("Country")
plt.ylabel("Inflation & Growth Rate%")
plt.title("Bar Graph")
plt.legend()
```

Out[17]:
```
<matplotlib.legend.Legend at 0x1b012825d48>
```

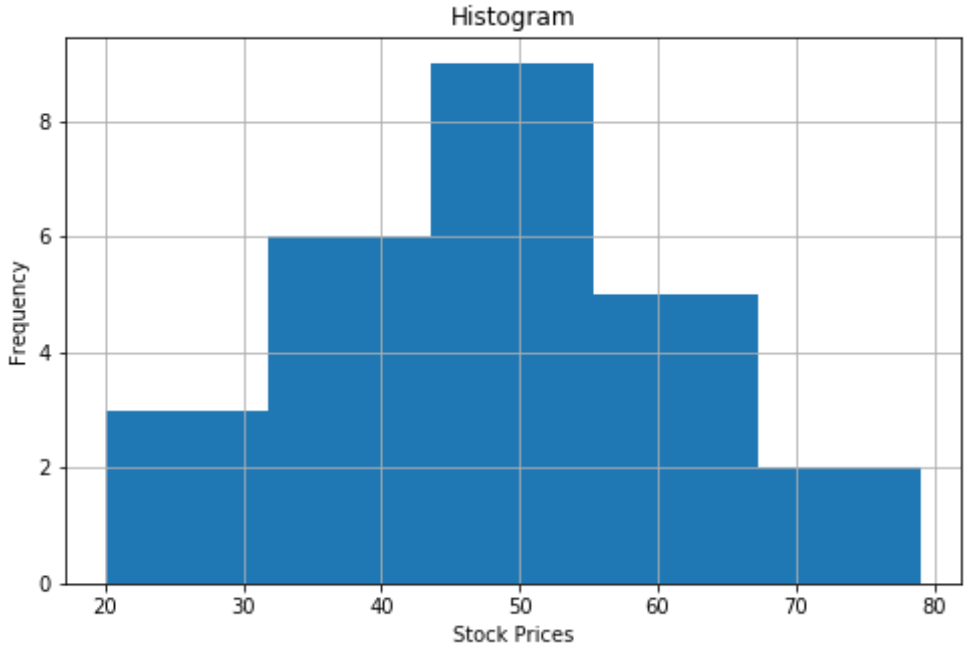


HISTOGRAMS

- Tend to show distribution by grouping segments together. Examples of this might be age groups, or scores on a test. Rather than showing every single age a group might be, maybe we just show people from 20-25, 25-30... and so on
- It uses categorical data similar to bar plot, but groups continuous data into numbers into ranges.
- The ranges of continuous data are known as bins/Intervals.
- One more important point is all the bins must be adjacent to each other.
- Usually bins are of equal size, but they can vary in sizes.
- Example: Consider a histogram of x and y, which are 500 random numbers. In the histogram distribution of x and y are represented with different colors.

In [18]:
```
plt.figure(figsize = (8,5))
stock_prices = [32,67,43,56,45,43,42,46,48,53,73,55,54,56,43,55,54,20,3
3,65,62,51,79,31,27]
plt.hist(stock_prices, bins = 5)
```

```
plt.xlabel("Stock Prices")
plt.ylabel("Frequency")
plt.title("Histogram")
plt.grid(True)
```



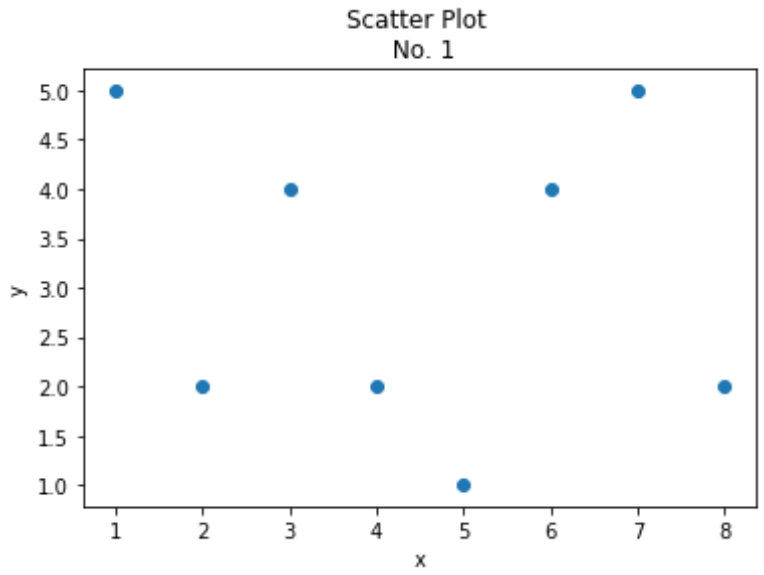
SCATTER PLOTS

Scatter plots are similar to line graphs in that they use horizontal and vertical axes to plot data points. However, they have a very specific purpose. Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation .

In [19]:
```
x = [1,2,3,4,5,6,7,8]
y = [5,2,4,2,1,4,5,2]
plt.scatter(x,y)
plt.xlabel('x', )
```

```
plt.ylabel('y')
plt.title('Scatter Plot \n No. 1')
```

Out[19]: Text(0.5, 1.0, 'Scatter Plot \n No. 1')



PIE CHARTS

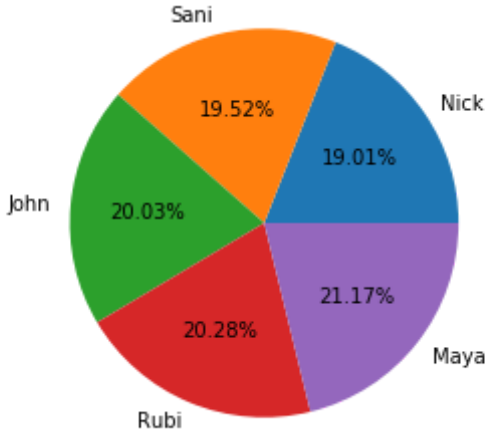
In [20]:
```
raw_data={'names':['Nick','Sani','John','Rubi','Maya'],
          'jan_score':[123,124,125,126,128],
          'feb_score':[23,24,25,27,29],
          'march_score':[3,5,7,6,9]}

df=pd.DataFrame(raw_data,columns=['names','jan_score','feb_score','marc
h_score'])
df['total_score']=df['jan_score']+df['feb_score']+df['march_score']
df
```

Out[20]:

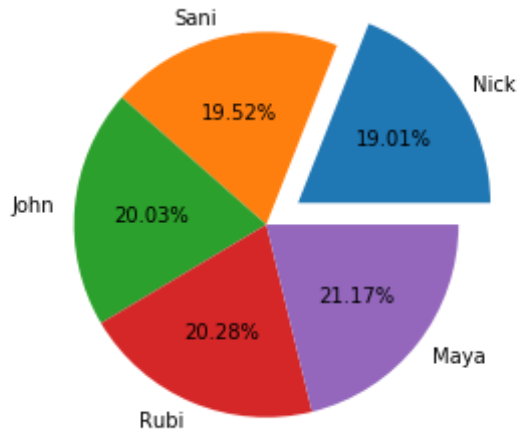
	names	jan_score	feb_score	march_score	total_score
0	Nick	123	23	3	149

	names	jan_score	feb_score	march_score	total_score
1	Sani	124	24	5	153
2	John	125	25	7	157
3	Rubi	126	27	6	159
4	Maya	128	29	9	166

In [21]:
```
plt.pie(df['total_score'],labels=df['names'],autopct='%.2f%%')
plt.axis('equal')
plt.axis('equal')
plt.show()
```



In [22]:
```
plt.pie(df['total_score'],labels=df['names'],autopct='%.2f%%', explode=
(0.2,0,0,0,0))
plt.axis('equal')
plt.axis('equal')
plt.show()
```

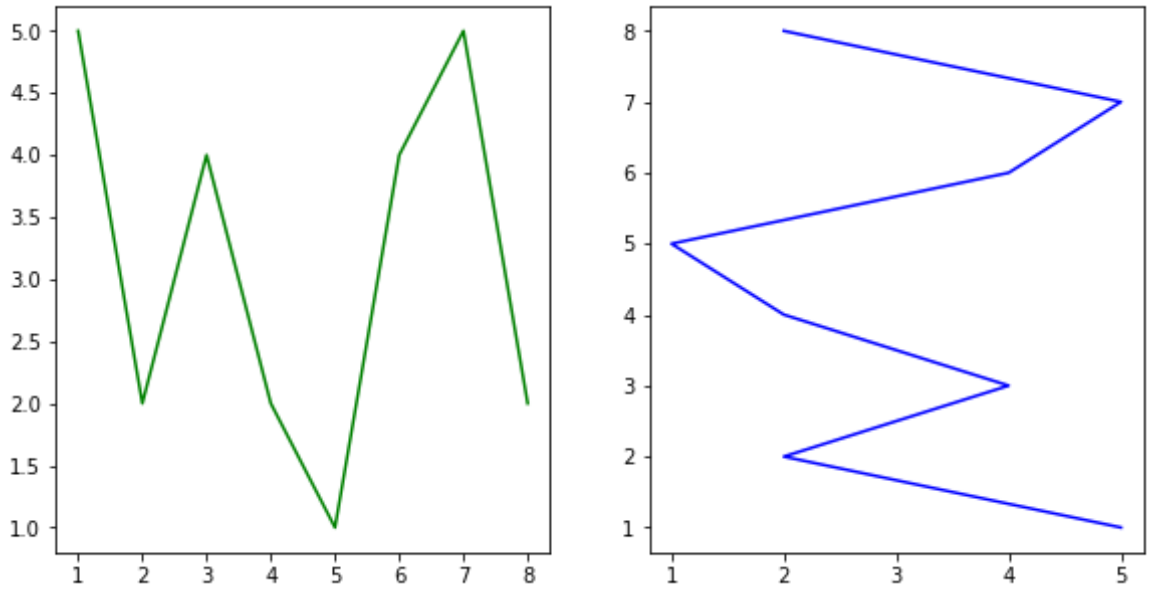


SUBPLOTS

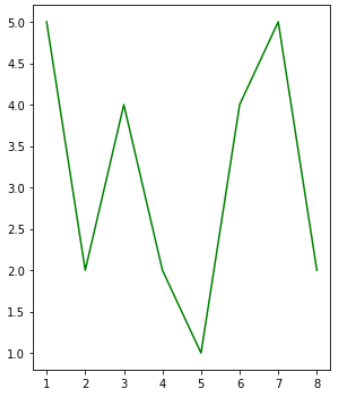
HORIZONTAL SUBPLOTS

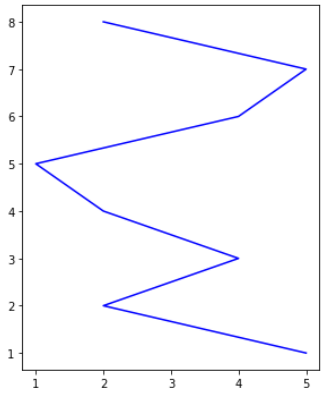
In [23]:
```
plt.figure(figsize=(15,5))
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([5,2,4,2,1,4,5,2])

plt.subplot(1,3,1)
plt.plot(x,y,'g')

plt.subplot(1,3,2)
plt.plot(y,x,'b')
```

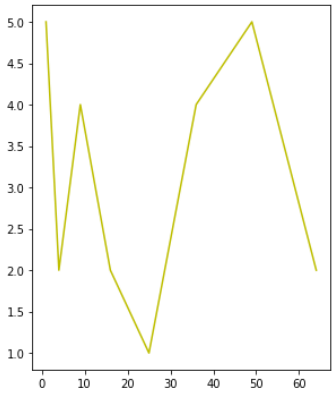
Out[23]:
```
[<matplotlib.lines.Line2D at 0x1b012957ec8>]
```



In [24]:
```
plt.figure(figsize=(15,5))
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([5,2,4,2,1,4,5,2])

plt.subplot(1,3,1)
plt.plot(x,y,'g')

plt.subplot(1,3,2)
plt.plot(y,x,'b')

plt.subplot(1,3,3)
plt.plot(x**2,y,'y')

plt.tight_layout(w_pad=10)
```



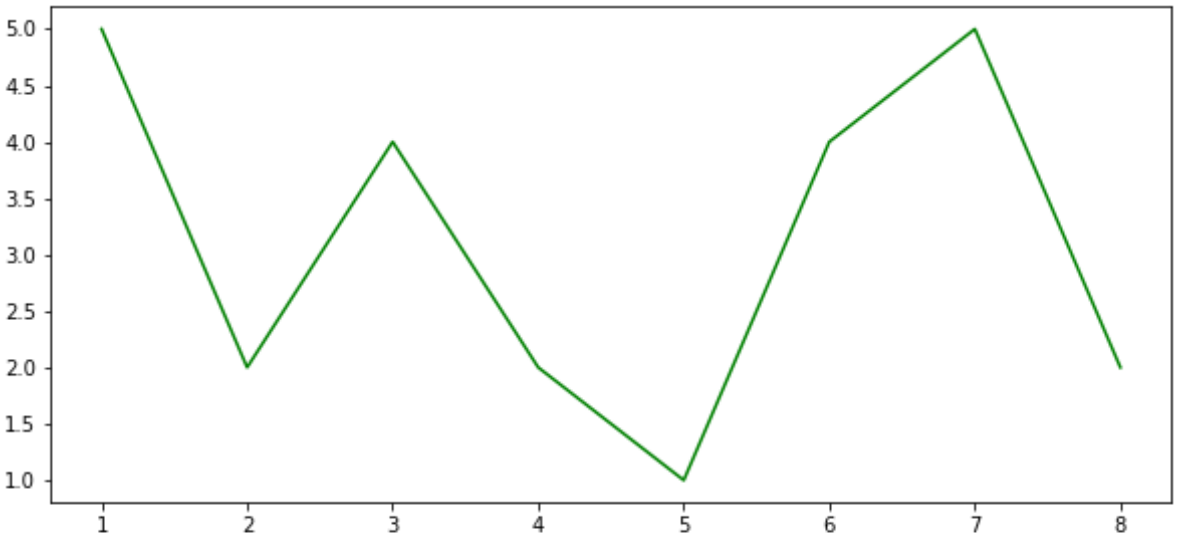


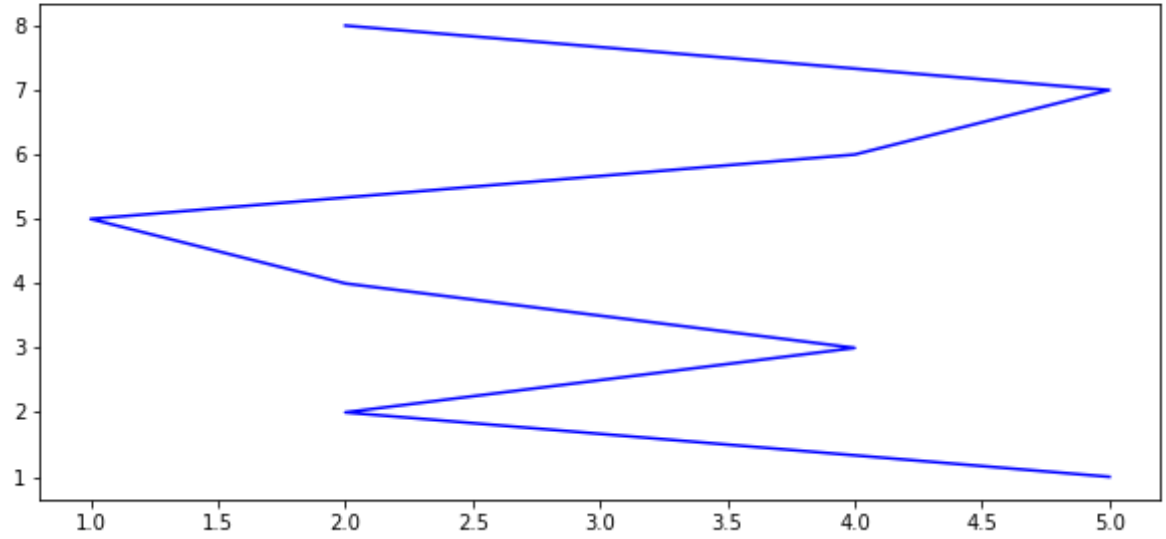


VERTICAL SUBPLOTS

In [25]:
```
plt.figure(figsize=(10,10))
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([5,2,4,2,1,4,5,2])

plt.subplot(2,1,1)
plt.plot(x,y,'g')

plt.subplot(2,1,2)
plt.plot(y,x,'b')
```

Out[25]:
```
[<matplotlib.lines.Line2D at 0x1b012ae8f48>]
```

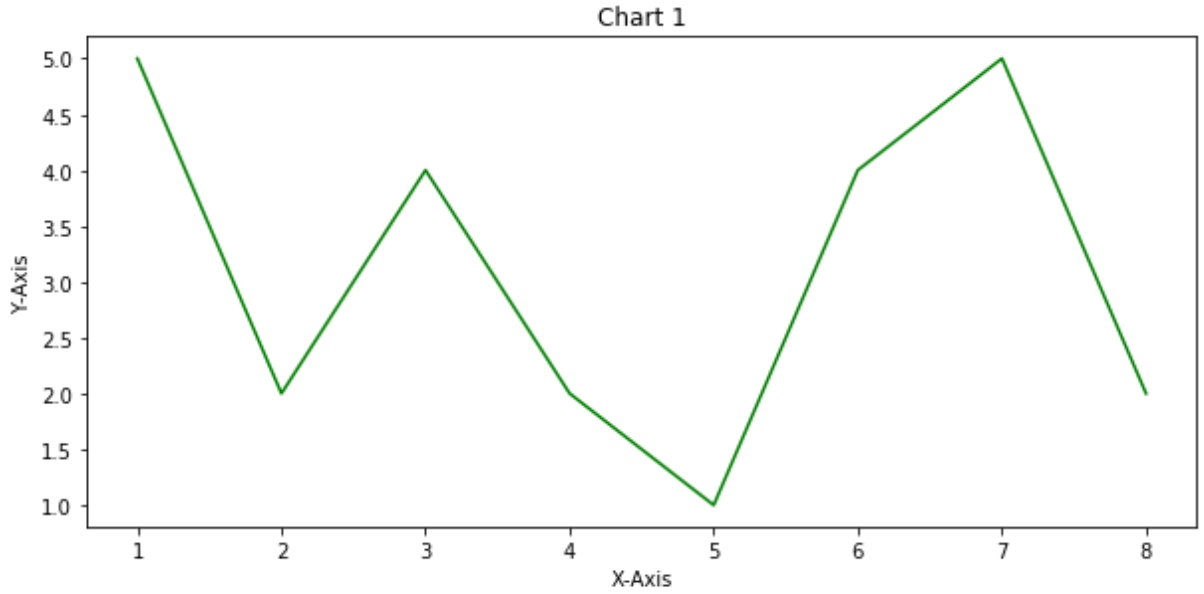


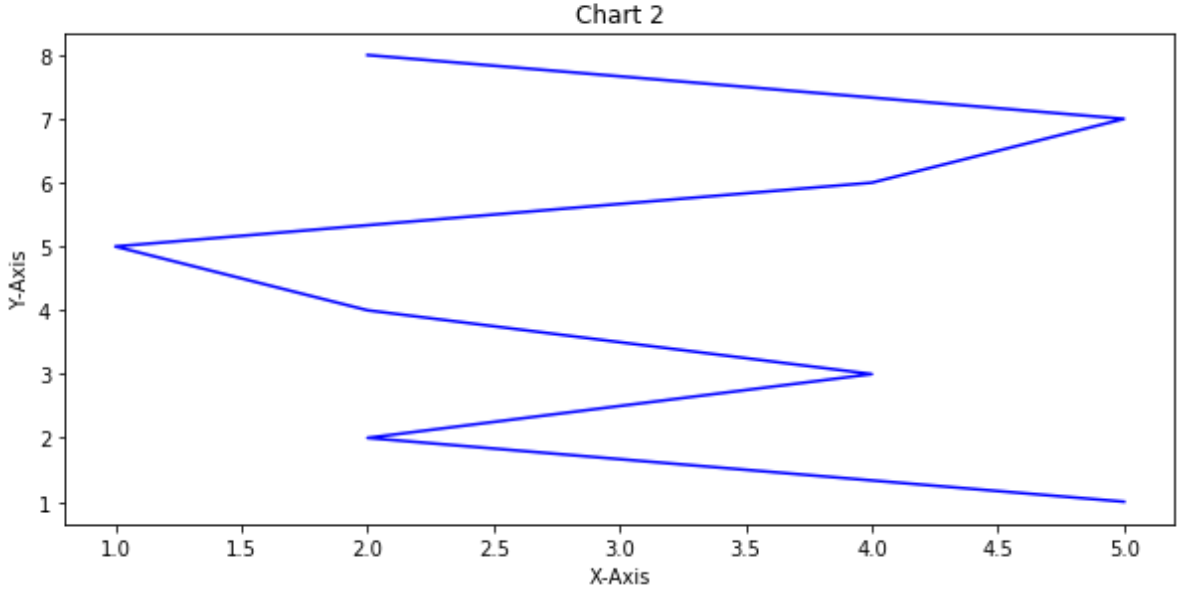


In [26]:
```
plt.figure(figsize=(10,10))
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([5,2,4,2,1,4,5,2])
```

```
plt.subplot(2,1,1)
plt.plot(x,y,'g')
plt.title('Chart 1')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.subplot(2,1,2)
plt.plot(y,x,'b')
plt.title('Chart 2')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')
```

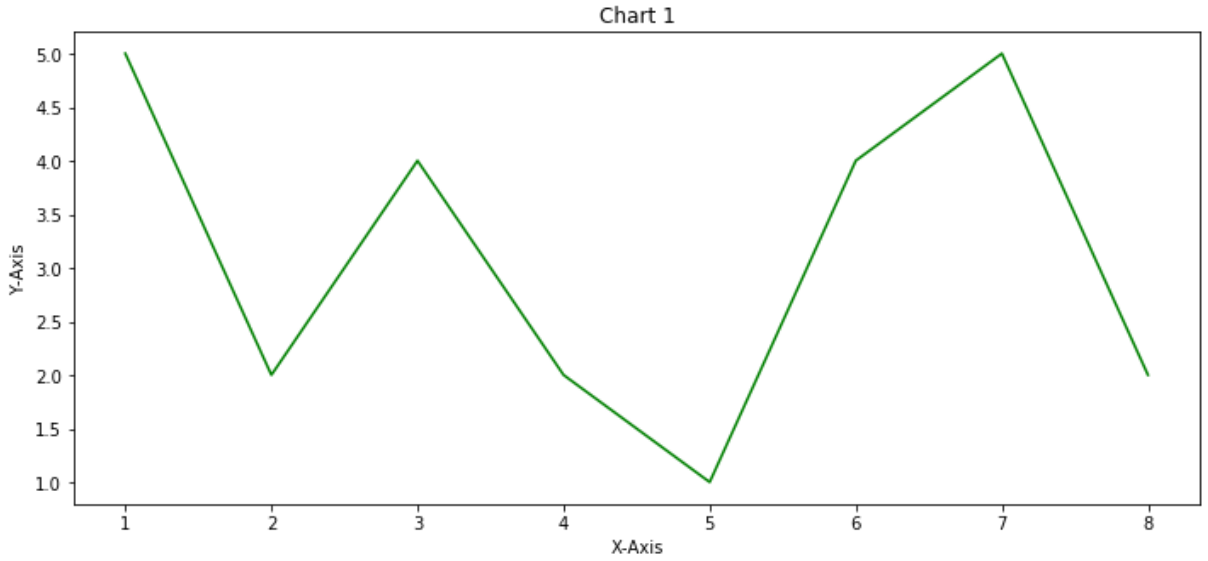
Out[26]: Text(0, 0.5, 'Y-Axis')

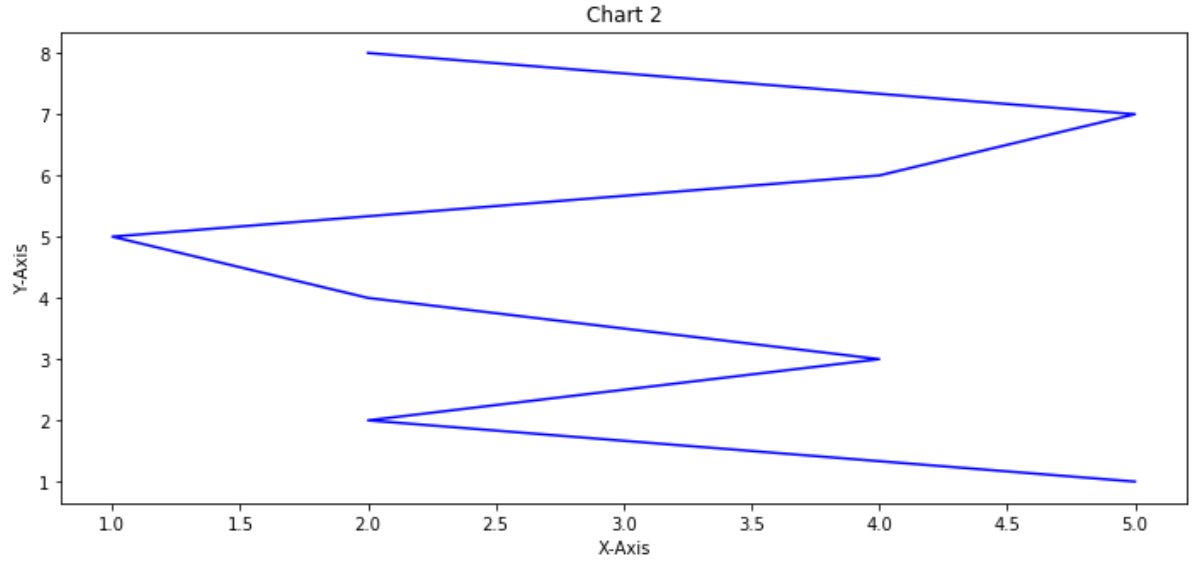




In [27]:
```
plt.figure(figsize=(10,10))
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([5,2,4,2,1,4,5,2])
```

```
plt.subplot(2,1,1)
plt.plot(x,y,'g')
plt.title('Chart 1')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.subplot(2,1,2)
plt.plot(y,x,'b')
plt.title('Chart 2')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.tight_layout(h_pad=5)
```

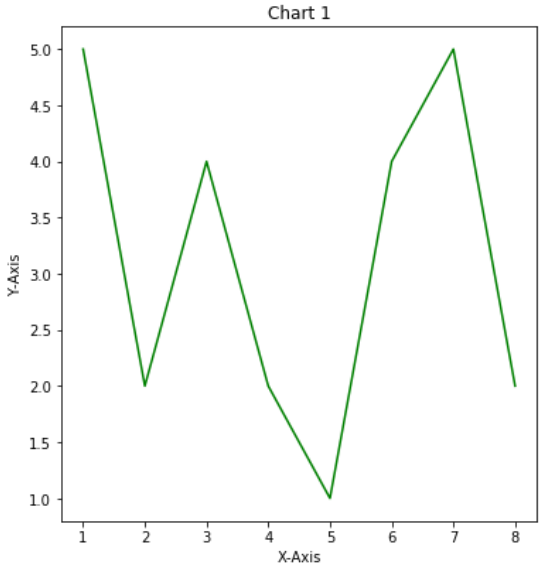


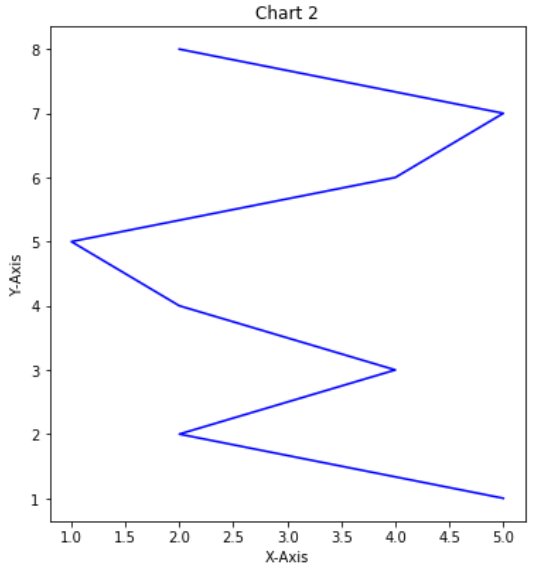


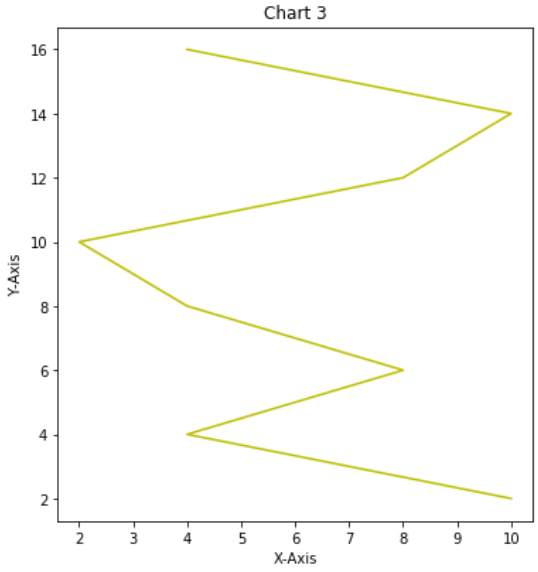
SUBPLOTS 2X2

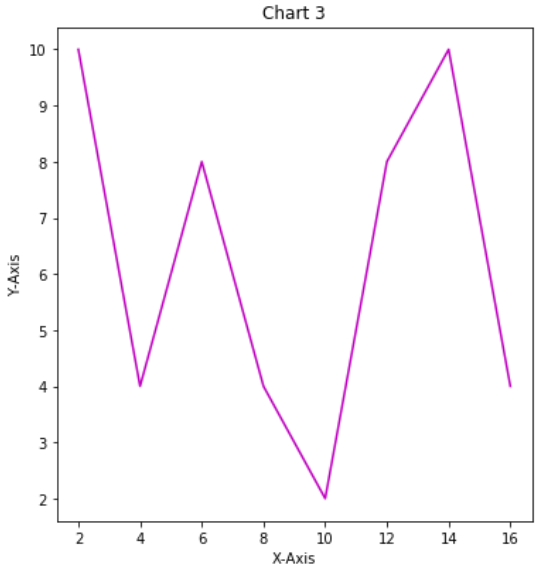
In [28]:

```
plt.figure(figsize=(12,12))
x = np.array([1,2,3,4,5,6,7,8])
y = np.array([5,2,4,2,1,4,5,2])

plt.subplot(2,2,1)
plt.plot(x,y,'g')
plt.title('Chart 1')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.subplot(2,2,2)
plt.plot(y,x,'b')
plt.title('Chart 2')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.subplot(2,2,3)
plt.plot(y*2,x*2,'y')
plt.title('Chart 3')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.subplot(2,2,4)
plt.plot(x*2,y*2,'m')
plt.title('Chart 3')
plt.xlabel('X-Axis')
plt.ylabel('Y-Axis')

plt.tight_layout(h_pad=5, w_pad=10)
```









Basic Data Visualizations USING PANDAS

In [29]:
```
df = pd.read_csv('cargame.csv')
```

In [30]:
```
df.head()
```

Out[30]:

	Gender	Name of Car	Pull Distance	Distance	Time
0	F	Car	11.8	53.0	3.57
1	F	Car	1.9	15.7	2.65
2	F	Car	3.9	31.0	2.30
3	F	Car	10.2	22.8	5.20
4	F	Car	5.0	6.0	2.53

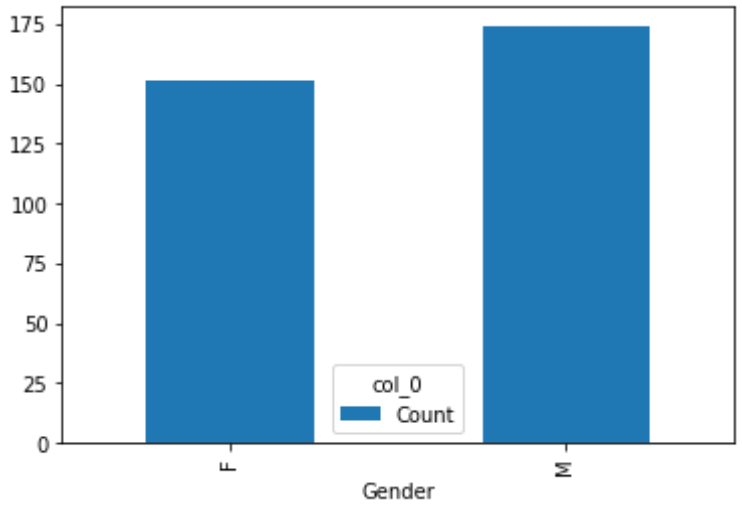
In [31]:
```
pd.crosstab(df['Gender'], columns='Count')
```

Out[31]:

col_0	Count
Gender	
F	151
M	174

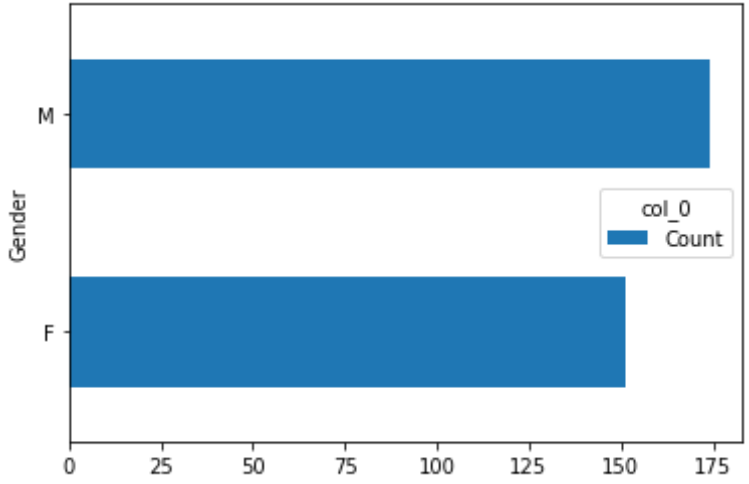
In [32]:
```
pd.crosstab(df['Gender'], columns='Count').plot(kind='bar')
```

Out[32]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1b01291c2c8>
```



In [33]:
```
pd.crosstab(df['Gender'], columns='Count').plot(kind='barh')
```

Out[33]: <matplotlib.axes._subplots.AxesSubplot at 0x1b0124c29c8>



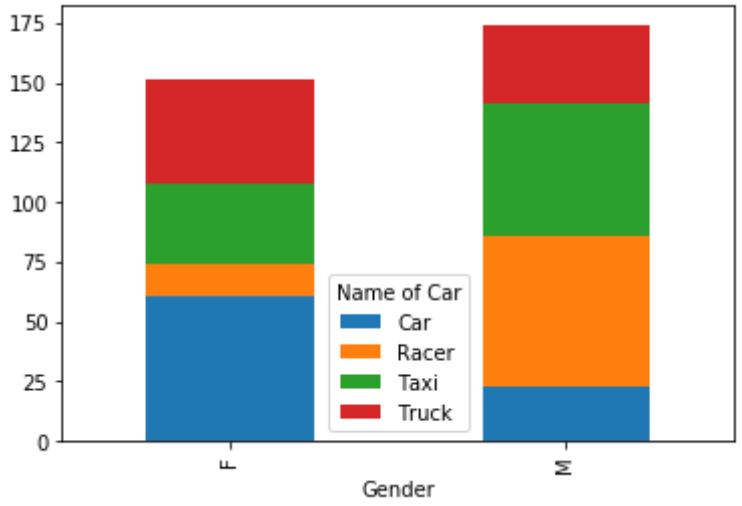
In [34]:
```
pd.crosstab(df['Gender'],df['Name of Car'])
```

Out[34]:

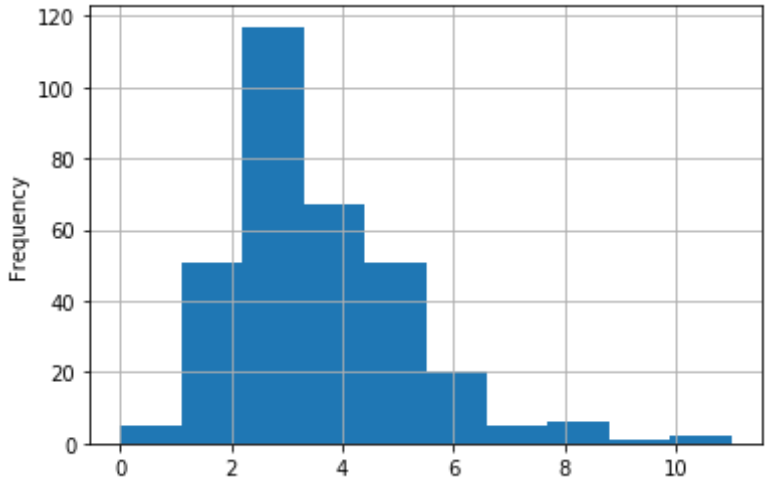
Name of Car	Car	Racer	Taxi	Truck
Gender				
F	61	13	34	43
M	23	63	55	33

In [35]:
```
pd.crosstab(df['Gender'],df['Name of Car']).plot(kind='bar', stacked=True)
```

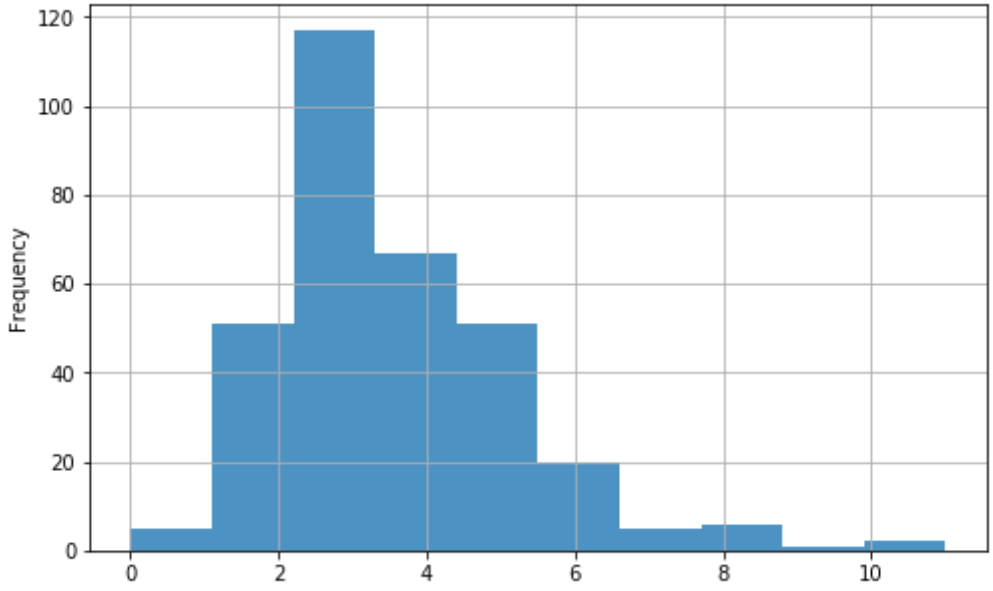
Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x1b0129e9108>



In [36]:
```
df['Time'].plot(kind='hist')
plt.grid(True)
```

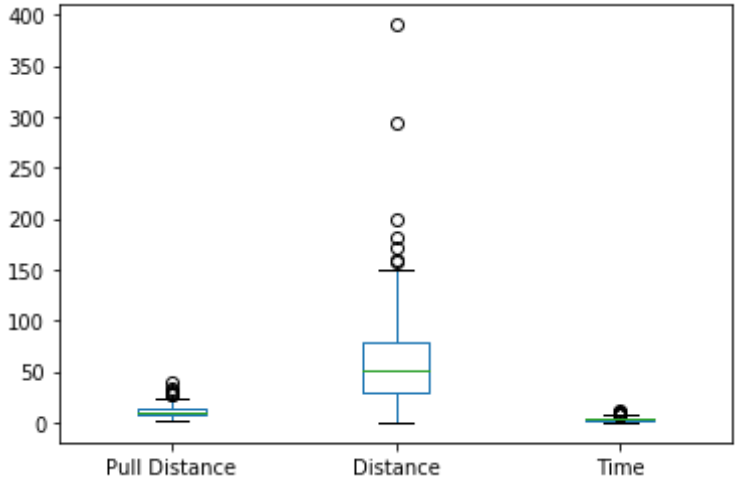


In [37]:
```
plt.figure(figsize=(8,5))
df['Time'].plot(kind='hist', alpha=0.8)
plt.grid(True)
```



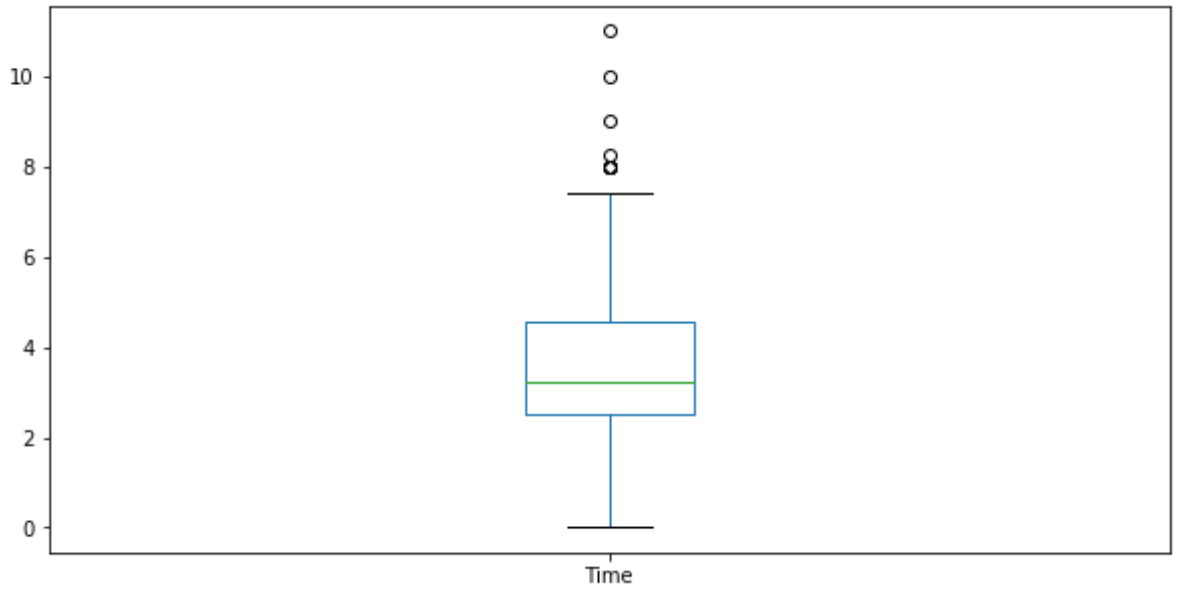
In [38]:
```
plt.figure(figsize=(15,15))
df.plot(kind='box')
```

Out[38]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1b013f441c8>

<Figure size 1080x1080 with 0 Axes>
```

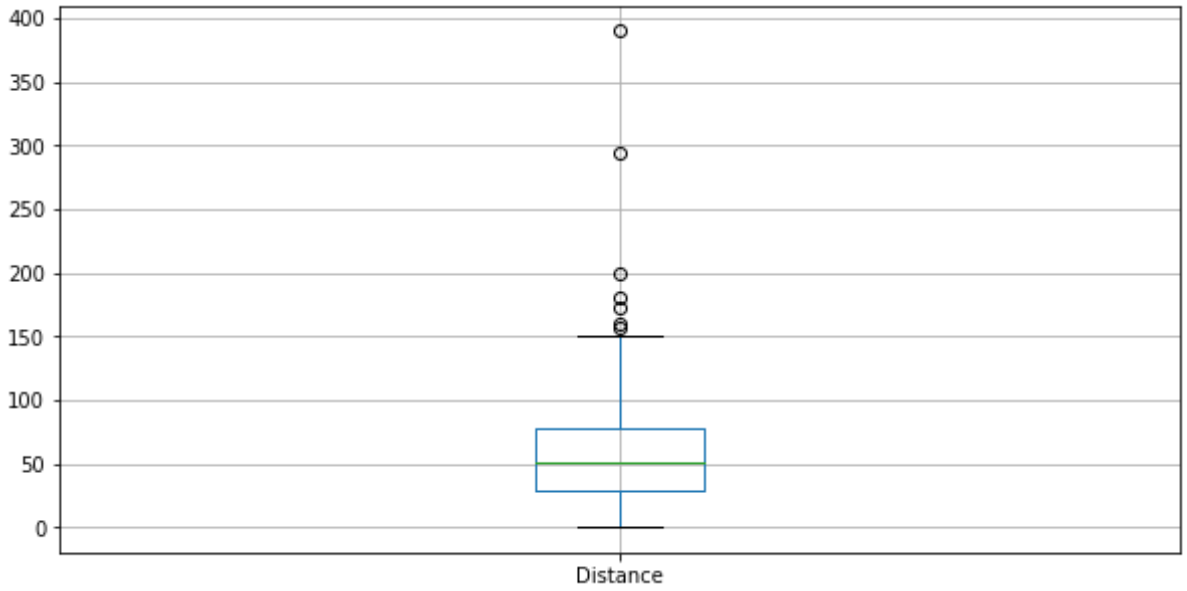


In [39]:
```
plt.figure(figsize=(10,5))
df['Time'].plot(kind='box')
```

Out[39]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1b01435aac8>
```

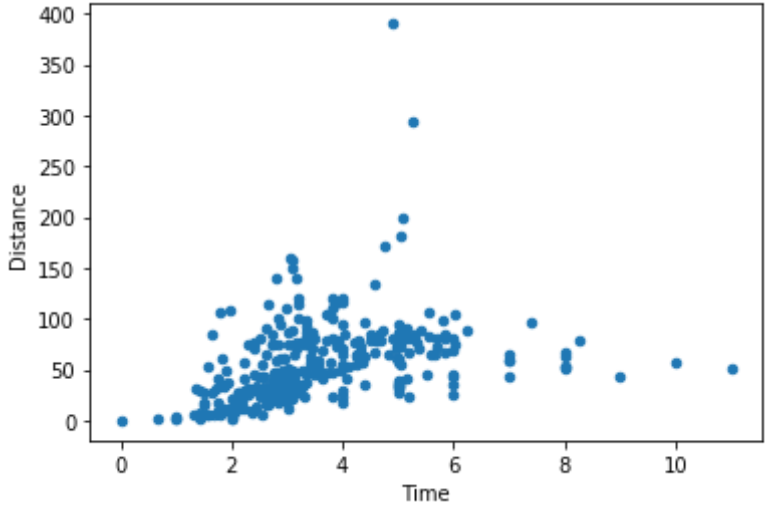


In [40]:
```
plt.figure(figsize=(10,5))
df['Distance'].plot(kind='box')
plt.grid(True)
```



In [41]:
```
df.plot(kind='scatter', x='Time',y='Distance')
```

Out[41]: <matplotlib.axes._subplots.AxesSubplot at 0x1b01442dc88>



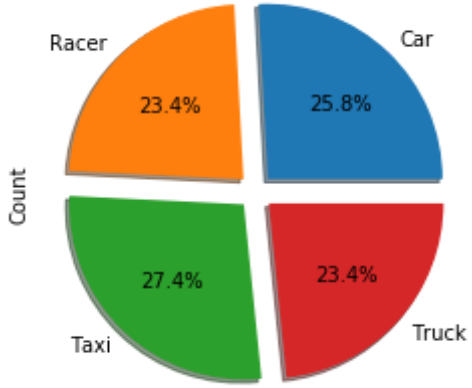
In [42]:
```
pd.crosstab(df['Name of Car'], columns='Count')
```

Out[42]:

Name of Car	col_0 Count
Car	84
Racer	76
Taxi	89
Truck	76

In [43]:
```
pd.crosstab(df['Name of Car'], columns='Count')['Count'].plot(kind='pie', autopct='%0.1f%%', shadow=True,
                                                              explode=(0.1,0.1,0.1,0.1))
```

Out[43]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1b013f03448>
```



In [ ]:
```

```