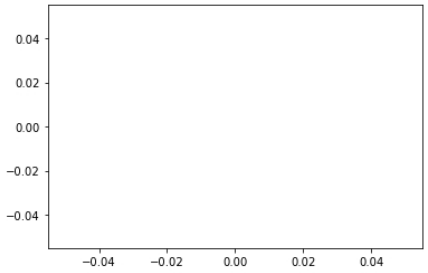
# Introduction to Matplotlib

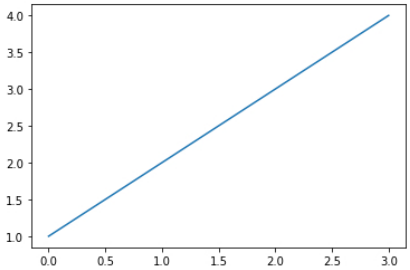
In [1]:
```
%matplotlib inline
import matplotlib.pyplot as plt

import pandas as pd
import numpy as np
```

In [2]:
```
plt.plot()
```

Out[2]: []

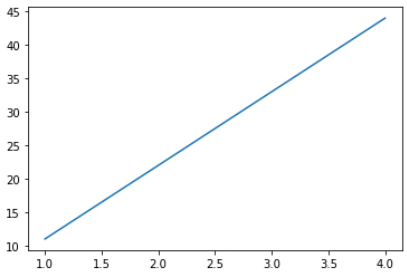



In [3]:
```
plt.plot([1,2,3,4]);
```




In [4]:
```
x = [1,2,3,4]
y = [11,22,33,44]
plt.plot(x,y)
```

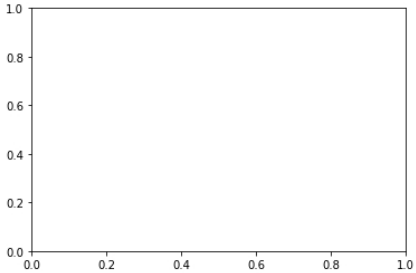
Out[4]: [<matplotlib.lines.Line2D at 0x21c9b6d4d00>]




## method of plotting

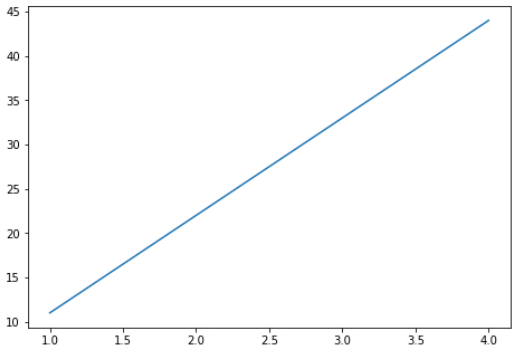
In [5]:
```
#method 1
fig = plt.figure() # creating a figure
ax = fig.add_subplot() # add some axes
plt.show
```

Out[5]: <function matplotlib.pyplot.show(close=None, block=None)>



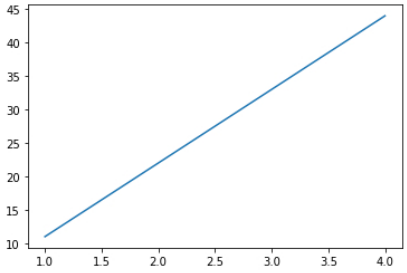

In [6]:
```
#method 2
```

```
fig = plt.figure() # creates a figure
ax = fig.add_axes([1, 1, 1, 1])
ax.plot(x, y) # add some data
plt.show()
```



In [7]:

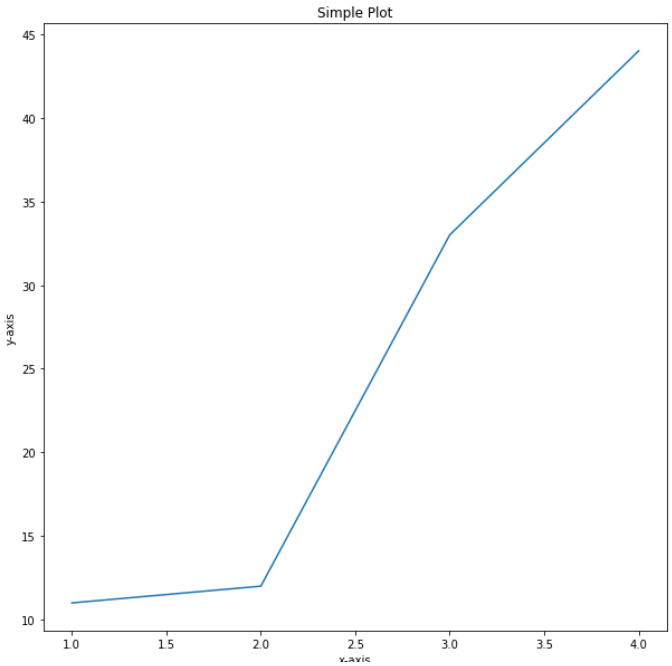
```
#method 3
fig, ax = plt.subplots()
ax.plot(x, y); #add some data
```



## Matplotlib example workflow

In [8]:

```
# 0. import map plot lib and get it ready for plotting in Jupiter
%matplotlib inline
import matplotlib.pyplot as plt

#1. prepare data
x = [1,2,3,4]
y = [11,12,33,44]

#2. Setup plot
fig, ax = plt.subplots(figsize = (10,10)) # (width, height)

#3. plot data
ax.plot(x,y)

#4. Customize plot
ax.set(title = "Simple Plot",
       xlabel = "x-axis",
        ylabel = "y-axis" )

#5. save & show (you save the whole figure)
fig.savefig("./images/sample-plot.png")
```



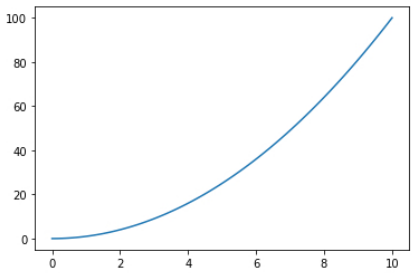
## making figures with NUM pi arrays

In [9]:
```
#create some data
x = np.linspace(0,10,100)
x[:10]
```

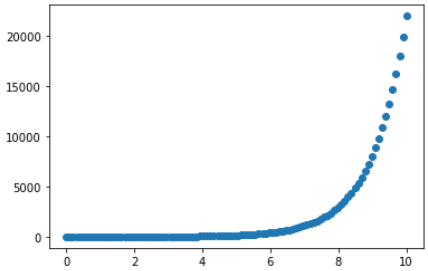
Out[9]:
```
array([0.        , 0.1010101 , 0.2020202 , 0.3030303 , 0.4040404 ,
       0.50505051, 0.60606061, 0.70707071, 0.80808081, 0.90909091])
```

In [10]:
```
# plot the data and create a line plot
fig, ax = plt.subplots()
ax.plot(x, x**2)
```

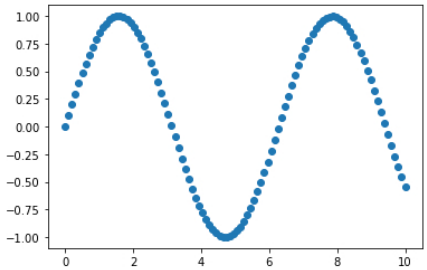
Out[10]:
```
[<matplotlib.lines.Line2D at 0x21c9b82a520>]
```



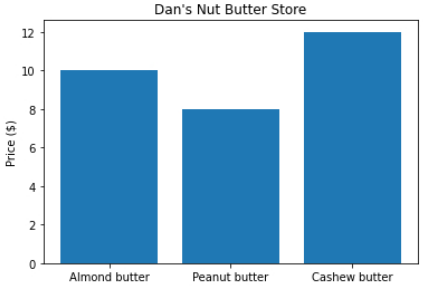

In [11]:
```
#use same data to make a scatter
fig, ax = plt.subplots()
ax.scatter(x, np.exp(x));
```




In [12]:
```
# Another scatter plot
fig, ax = plt.subplots()
ax.scatter(x, np.sin(x));
```

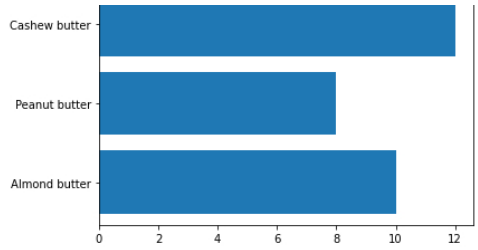



In [13]:
```
# make a plot from dictionary
nut_butter_prices = {"Almond butter": 10,
                     "Peanut butter": 8,
                     "Cashew butter": 12}
fig, ax = plt.subplots()
ax.bar(nut_butter_prices.keys(), nut_butter_prices.values())
ax.set(title = "Dan's Nut Butter Store",
       ylabel = "Price ($)");
```

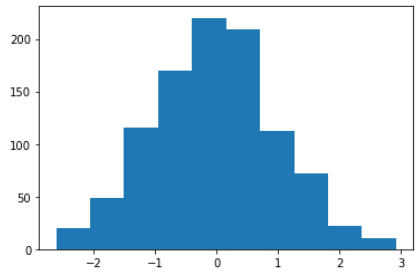



In [14]:
```
fig, ax = plt.subplots()
ax.barh(list(nut_butter_prices.keys()), list(nut_butter_prices.values()))
```

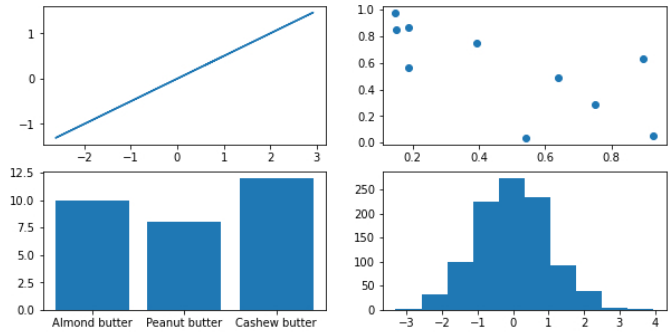
Out[14]:
```
<BarContainer object of 3 artists>
```



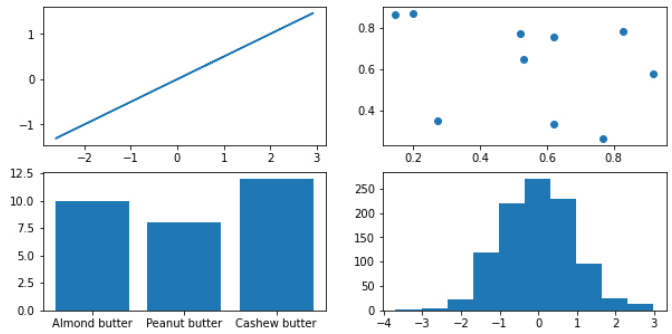

In [15]:
```
# Make some data for histograms and plot it
x = np.random.randn(1000)
fig, ax = plt.subplots()
ax.hist(x);
```




In [16]:
```
# Subplot option 1
fig, ((ax1, ax2),(ax3, ax4)) = plt.subplots(nrows=2,
                                            ncols=2,
                                            figsize=(10,5))

# plot to each different axis / filling
ax1.plot(x, x/2)
ax2.scatter(np.random.random(10), np.random.random(10));
ax3.bar(nut_butter_prices.keys(), nut_butter_prices.values());
ax4.hist(np.random.randn(1000));
```




In [17]:
```
# subplots option 2
fig, ax = plt.subplots(nrows=2,
                       ncols=2,
                       figsize=(10,5))
# plot to each different index
ax[0,0].plot(x, x/2)
ax[0,1].scatter(np.random.random(10), np.random.random(10));
ax[1,0].bar(nut_butter_prices.keys(), nut_butter_prices.values());
ax[1,1].hist(np.random.randn(1000));
```

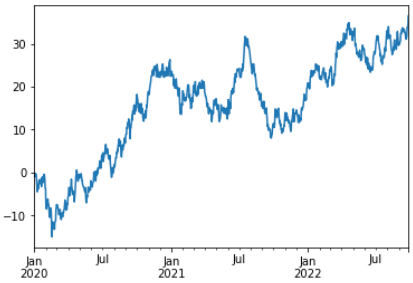



## Plotting From Pandas DataFrames

In [18]:
```
import pandas as pd
```

In [19]:
```
ts = pd.Series(np.random.randn(1000),
               index = pd.date_range("1/1/2020",periods=1000))
ts = ts.cumsum()
ts.plot()
```

Out[19]: <AxesSubplot:>




In [20]:
```
# Make a dataFrame
car_sales = pd.read_csv("7.1 car-sales.csv")
car_sales
```

Out[20]:

| | Make | Colour | Odometer (KM) | Doors | Price |
|---|---|---|---|---|---|
| 0 | Toyota | White | 150043 | 4 | $4,000.00 |
| 1 | Honda | Red | 87899 | 4 | $5,000.00 |
| 2 | Toyota | Blue | 32549 | 3 | $7,000.00 |
| 3 | BMW | Black | 11179 | 5 | $22,000.00 |
| 4 | Nissan | White | 213095 | 4 | $3,500.00 |
| 5 | Toyota | Green | 99213 | 4 | $4,500.00 |
| 6 | Honda | Blue | 45698 | 4 | $7,500.00 |
| 7 | Honda | Blue | 54738 | 4 | $7,000.00 |
| 8 | Toyota | White | 60000 | 4 | $6,250.00 |
| 9 | Nissan | White | 31600 | 4 | $9,700.00 |

In [21]:
```
car_sales["Price"] = car_sales["Price"].str.replace('[\$\,\.]','')
car_sales
```

```
<ipython-input-21-dbb1dc823e29>:1: FutureWarning: The default value of regex will change from True to False in a future vers
ion.
  car_sales["Price"] = car_sales["Price"].str.replace('[\$\,\.]','')
```

Out[21]:

| | Make | Colour | Odometer (KM) | Doors | Price |
|---|---|---|---|---|---|
| 0 | Toyota | White | 150043 | 4 | 400000 |
| 1 | Honda | Red | 87899 | 4 | 500000 |
| 2 | Toyota | Blue | 32549 | 3 | 700000 |
| 3 | BMW | Black | 11179 | 5 | 2200000 |
| 4 | Nissan | White | 213095 | 4 | 350000 |
| 5 | Toyota | Green | 99213 | 4 | 450000 |
| 6 | Honda | Blue | 45698 | 4 | 750000 |
| 7 | Honda | Blue | 54738 | 4 | 700000 |
| 8 | Toyota | White | 60000 | 4 | 625000 |
| 9 | Nissan | White | 31600 | 4 | 970000 |

In [22]:
```
# Remove laast two zeros
car_sales["Price"] = car_sales["Price"].str[:-2]
car_sales
```

Out[22]:

| | Make | Colour | Odometer (KM) | Doors | Price |
|---|---|---|---|---|---|
| 0 | Toyota | White | 150043 | 4 | 4000 |
| 1 | Honda | Red | 87899 | 4 | 5000 |
| 2 | Toyota | Blue | 32549 | 3 | 7000 |
| 3 | BMW | Black | 11179 | 5 | 22000 |
| 4 | Nissan | White | 213095 | 4 | 3500 |
| 5 | Toyota | Green | 99213 | 4 | 4500 |
| 6 | Honda | Blue | 45698 | 4 | 7500 |
| 7 | Honda | Blue | 54738 | 4 | 7000 |
| 8 | Toyota | White | 60000 | 4 | 6250 |
| 9 | Nissan | White | 31600 | 4 | 9700 |

In [23]:
```
car_sales["Sale Date"] = pd.date_range("1/1/2020", periods = len(car_sales))
```

In [24]:
```
car_sales
```

Out[24]:

| | Make | Colour | Odometer (KM) | Doors | Price | Sale Date |
|---|---|---|---|---|---|---|
| 0 | Toyota | White | 150043 | 4 | 4000 | 2020-01-01 |
| 1 | Honda | Red | 87899 | 4 | 5000 | 2020-01-02 |
| 2 | Toyota | Blue | 32549 | 3 | 7000 | 2020-01-03 |
| 3 | BMW | Black | 11179 | 5 | 22000 | 2020-01-04 |
| 4 | Nissan | White | 213095 | 4 | 3500 | 2020-01-05 |
| 5 | Toyota | Green | 99213 | 4 | 4500 | 2020-01-06 |
| 6 | Honda | Blue | 45698 | 4 | 7500 | 2020-01-07 |
| 7 | Honda | Blue | 54738 | 4 | 7000 | 2020-01-08 |

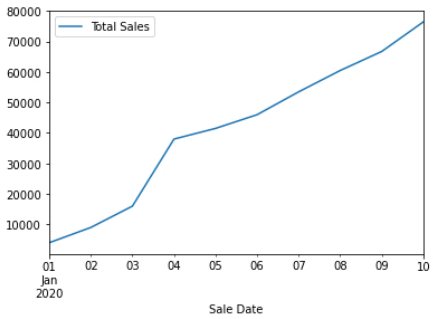
| | | | | | | |
|---|---|---|---|---|---|---|
| 8 | Toyota | White | 60000 | 4 | 6250 | 2020-01-09 |
| 9 | Nissan | White | 31600 | 4 | 9700 | 2020-01-10 |

In [85]:
```
car_sales["Price"] = car_sales["Price"].astype(int)
car_sales["Total Sales"] = car_sales["Price"].cumsum()
car_sales
```

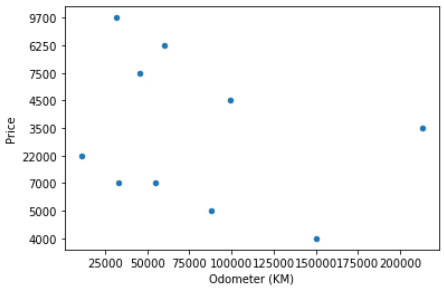
Out[85]:

| | Make | Colour | Odometer (KM) | Doors | Price | Sale Date | Total Sales |
|---|---|---|---|---|---|---|---|
| 0 | Toyota | White | 150043 | 4 | 4000 | 2020-01-01 | 4000 |
| 1 | Honda | Red | 87899 | 4 | 5000 | 2020-01-02 | 9000 |
| 2 | Toyota | Blue | 32549 | 3 | 7000 | 2020-01-03 | 16000 |
| 3 | BMW | Black | 11179 | 5 | 22000 | 2020-01-04 | 38000 |
| 4 | Nissan | White | 213095 | 4 | 3500 | 2020-01-05 | 41500 |
| 5 | Toyota | Green | 99213 | 4 | 4500 | 2020-01-06 | 46000 |
| 6 | Honda | Blue | 45698 | 4 | 7500 | 2020-01-07 | 53500 |
| 7 | Honda | Blue | 54738 | 4 | 7000 | 2020-01-08 | 60500 |
| 8 | Toyota | White | 60000 | 4 | 6250 | 2020-01-09 | 66750 |
| 9 | Nissan | White | 31600 | 4 | 9700 | 2020-01-10 | 76450 |

In [26]:
```
# Let's plot the total sales
car_sales.plot(x="Sale Date", y="Total Sales");
```




In [27]:
```
#Plot scatter plot
car_sales.plot(x = "Odometer (KM)", y="Price",kind = "scatter");
```

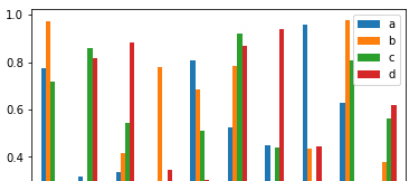


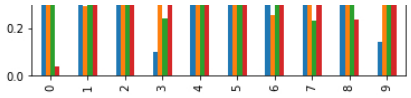

In [28]:
```
#################################################################
# How about a bar graph
x = np.random.rand(10,4)
x

# Turn it into a dataFrame
df = pd.DataFrame(x, columns = ['a','b','c','d'])
df
```

Out[28]:

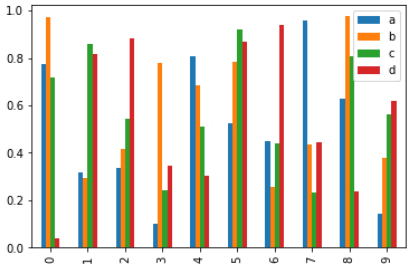
| | a | b | c | d |
|---|---|---|---|---|
| 0 | 0.772459 | 0.972177 | 0.716960 | 0.040040 |
| 1 | 0.319596 | 0.293884 | 0.857776 | 0.815723 |
| 2 | 0.337049 | 0.415053 | 0.542081 | 0.882600 |
| 3 | 0.103028 | 0.779854 | 0.240274 | 0.347462 |
| 4 | 0.809338 | 0.682590 | 0.512807 | 0.302645 |
| 5 | 0.524258 | 0.781529 | 0.918177 | 0.868655 |
| 6 | 0.447140 | 0.255052 | 0.440325 | 0.941576 |
| 7 | 0.959269 | 0.434213 | 0.233764 | 0.445320 |
| 8 | 0.630352 | 0.975356 | 0.806836 | 0.239339 |
| 9 | 0.144180 | 0.380845 | 0.562812 | 0.616797 |

In [29]:
```
df.plot.bar();
```





In [30]:
```
df.plot(kind="bar");
```

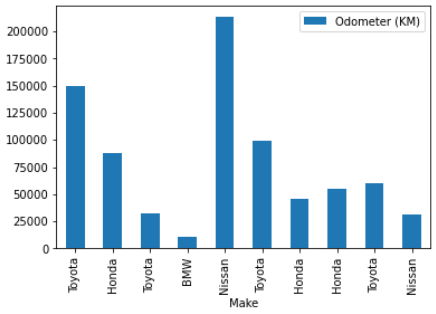



In [31]:
```
car_sales
```

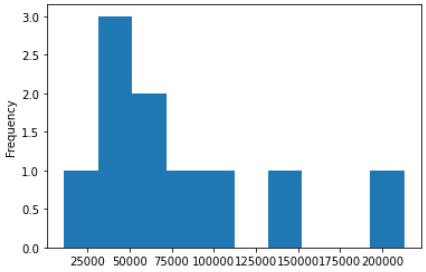
Out[31]:

| | Make | Colour | Odometer (KM) | Doors | Price | Sale Date | Total Sales |
|---|---|---|---|---|---|---|---|
| 0 | Toyota | White | 150043 | 4 | 4000 | 2020-01-01 | 4000 |
| 1 | Honda | Red | 87899 | 4 | 5000 | 2020-01-02 | 9000 |
| 2 | Toyota | Blue | 32549 | 3 | 7000 | 2020-01-03 | 16000 |
| 3 | BMW | Black | 11179 | 5 | 22000 | 2020-01-04 | 38000 |
| 4 | Nissan | White | 213095 | 4 | 3500 | 2020-01-05 | 41500 |
| 5 | Toyota | Green | 99213 | 4 | 4500 | 2020-01-06 | 46000 |
| 6 | Honda | Blue | 45698 | 4 | 7500 | 2020-01-07 | 53500 |
| 7 | Honda | Blue | 54738 | 4 | 7000 | 2020-01-08 | 60500 |
| 8 | Toyota | White | 60000 | 4 | 6250 | 2020-01-09 | 66750 |
| 9 | Nissan | White | 31600 | 4 | 9700 | 2020-01-10 | 76450 |

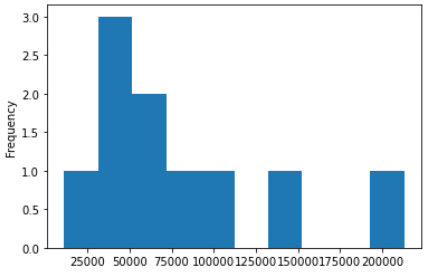
In [32]:
```
car_sales.plot(x="Make" , y = "Odometer (KM)",kind="bar");
```



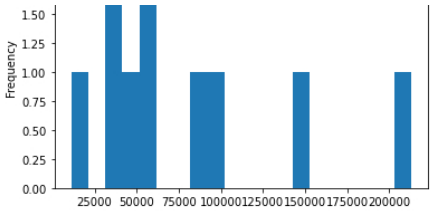

In [33]:
```
# how about histogram
car_sales["Odometer (KM)"].plot.hist();
```




In [34]:
```
car_sales["Odometer (KM)"].plot(kind="hist");
```




In [35]:
```
car_sales["Odometer (KM)"].plot.hist(bins = 20);
```




## Ploting on heart diseases

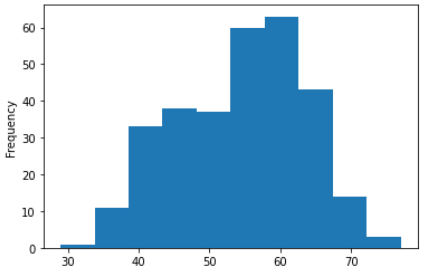
In [37]:
```
heart_disease = pd.read_csv("11.2 heart-disease.csv")
heart_disease
```

Out[37]:

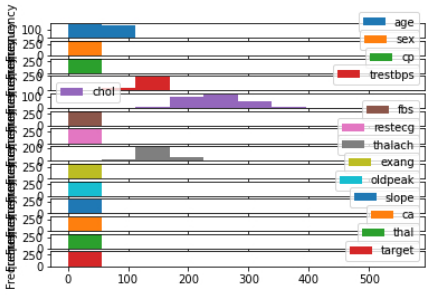
| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63 | 1 | 3 | 145 | 233 | 1 | 0 | 150 | 0 | 2.3 | 0 | 0 | 1 | 1 |
| **1** | 37 | 1 | 2 | 130 | 250 | 0 | 1 | 187 | 0 | 3.5 | 0 | 0 | 2 | 1 |
| **2** | 41 | 0 | 1 | 130 | 204 | 0 | 0 | 172 | 0 | 1.4 | 2 | 0 | 2 | 1 |
| **3** | 56 | 1 | 1 | 120 | 236 | 0 | 1 | 178 | 0 | 0.8 | 2 | 0 | 2 | 1 |
| **4** | 57 | 0 | 0 | 120 | 354 | 0 | 1 | 163 | 1 | 0.6 | 2 | 0 | 2 | 1 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **298** | 57 | 0 | 0 | 140 | 241 | 0 | 1 | 123 | 1 | 0.2 | 1 | 0 | 3 | 0 |
| **299** | 45 | 1 | 3 | 110 | 264 | 0 | 1 | 132 | 0 | 1.2 | 1 | 0 | 3 | 0 |
| **300** | 68 | 1 | 0 | 144 | 193 | 1 | 1 | 141 | 0 | 3.4 | 1 | 2 | 3 | 0 |
| **301** | 57 | 1 | 0 | 130 | 131 | 0 | 1 | 115 | 1 | 1.2 | 1 | 1 | 3 | 0 |
| **302** | 57 | 0 | 1 | 130 | 236 | 0 | 0 | 174 | 0 | 0.0 | 1 | 1 | 2 | 0 |

303 rows × 14 columns

In [38]:
```
# Create a histogram of age
heart_disease["age"].plot.hist();
```




In [40]:
```
heart_disease.plot.hist(subplots=True);
```




### Which one should you use? (pyplot vs matplotlib OO method?)

- when plotting something quickly, okay to use the pyplot method.
- when plotting something more advanced, use the OO method.

In [41]:
```
heart_disease
```

Out[41]:

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63 | 1 | 3 | 145 | 233 | 1 | 0 | 150 | 0 | 2.3 | 0 | 0 | 1 | 1 |
| **1** | 37 | 1 | 2 | 130 | 250 | 0 | 1 | 187 | 0 | 3.5 | 0 | 0 | 2 | 1 |
| **2** | 41 | 0 | 1 | 130 | 204 | 0 | 0 | 172 | 0 | 1.4 | 2 | 0 | 2 | 1 |
| **3** | 56 | 1 | 1 | 120 | 236 | 0 | 1 | 178 | 0 | 0.8 | 2 | 0 | 2 | 1 |
| **4** | 57 | 0 | 0 | 120 | 354 | 0 | 1 | 163 | 1 | 0.6 | 2 | 0 | 2 | 1 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **298** | 57 | 0 | 0 | 140 | 241 | 0 | 1 | 123 | 1 | 0.2 | 1 | 0 | 3 | 0 |
| **299** | 45 | 1 | 3 | 110 | 264 | 0 | 1 | 132 | 0 | 1.2 | 1 | 0 | 3 | 0 |
| **300** | 68 | 1 | 0 | 144 | 193 | 1 | 1 | 141 | 0 | 3.4 | 1 | 2 | 3 | 0 |
| **301** | 57 | 1 | 0 | 130 | 131 | 0 | 1 | 115 | 1 | 1.2 | 1 | 1 | 3 | 0 |
| **302** | 57 | 0 | 1 | 130 | 236 | 0 | 0 | 174 | 0 | 0.0 | 1 | 1 | 2 | 0 |

303 rows × 14 columns

In [42]:
```
over_50 = heart_disease[heart_disease["age"] > 50]
over_50
```

Out[42]:

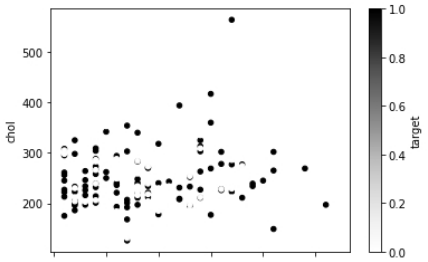
| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63 | 1 | 3 | 145 | 233 | 1 | 0 | 150 | 0 | 2.3 | 0 | 0 | 1 | 1 |
| **3** | 56 | 1 | 1 | 120 | 236 | 0 | 1 | 178 | 0 | 0.8 | 2 | 0 | 2 | 1 |
| **4** | 57 | 0 | 0 | 120 | 354 | 0 | 1 | 163 | 1 | 0.6 | 2 | 0 | 2 | 1 |
| **5** | 57 | 1 | 0 | 140 | 192 | 0 | 1 | 148 | 0 | 0.4 | 1 | 0 | 1 | 1 |
| **6** | 56 | 0 | 1 | 140 | 294 | 0 | 0 | 153 | 0 | 1.3 | 1 | 0 | 2 | 1 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **297** | 59 | 1 | 0 | 164 | 176 | 1 | 0 | 90 | 0 | 1.0 | 1 | 2 | 1 | 0 |
| **298** | 57 | 0 | 0 | 140 | 241 | 0 | 1 | 123 | 1 | 0.2 | 1 | 0 | 3 | 0 |
| **300** | 68 | 1 | 0 | 144 | 193 | 1 | 1 | 141 | 0 | 3.4 | 1 | 2 | 3 | 0 |
| **301** | 57 | 1 | 0 | 130 | 131 | 0 | 1 | 115 | 1 | 1.2 | 1 | 1 | 3 | 0 |
| **302** | 57 | 0 | 1 | 130 | 236 | 0 | 0 | 174 | 0 | 0.0 | 1 | 1 | 2 | 0 |

208 rows × 14 columns

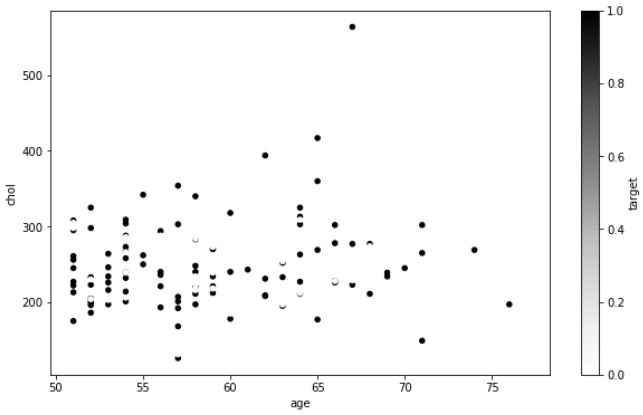
In [43]:
```
len(over_50)
```

Out[43]: 208

In [46]:
```
# Pyplot method
over_50.plot(kind = "scatter",
             x = 'age',
             y = 'chol',
             c = 'target');
```

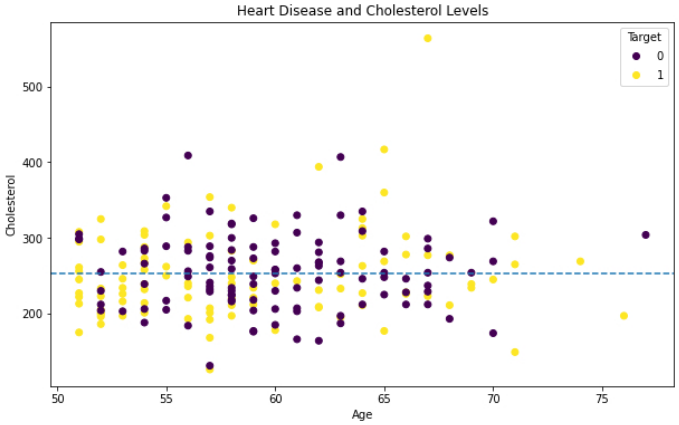



In [54]:
```
# OO method
fix , ax = plt.subplots(figsize=(10,6))
over_50.plot(kind='scatter',
             x = 'age',
             y = 'chol',
             c = 'target',
             ax = ax);

#we can also set limits
#ax.set_xlim([45, 100]);
#ax.set_ylim([45, 100])
```




In [61]:
```
## OO method from scratch
fig, ax = plt.subplots(figsize = (10,6))

# Plot the data
scatter = ax.scatter(x = over_50["age"],
                     y = over_50["chol"],
                     c = over_50["target"]);

# Customize the plot
ax.set(title = "Heart Disease and Cholesterol Levels",
       xlabel="Age",
       ylabel="Cholesterol")

#Add a legent
ax.legend(*scatter.legend_elements(), title="Target");

# Add a horizontal line
```

```
# Add a horizontal line
ax.axhline(over_50["chol"].mean(),
           linestyle = '--');
```



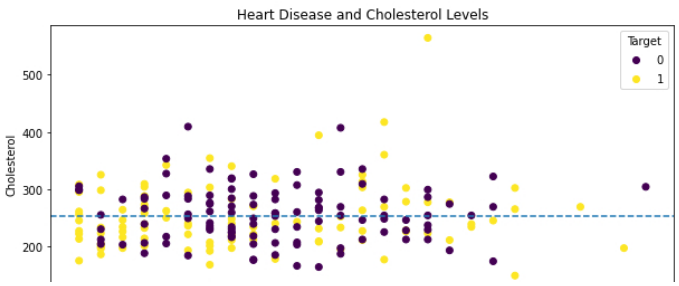

In [81]:
```
over_50.head()
```

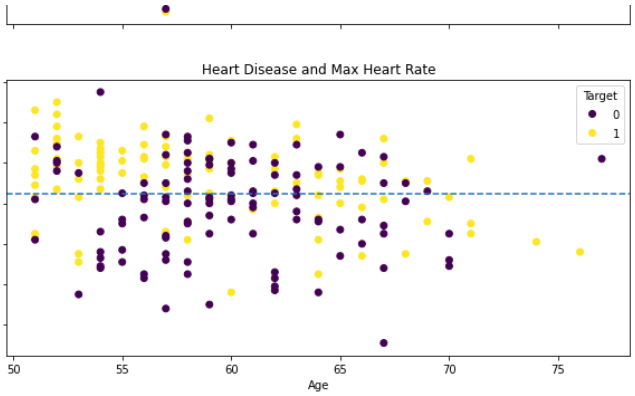
Out[81]:

| | age | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | target |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 63 | 1 | 3 | 145 | 233 | 1 | 0 | 150 | 0 | 2.3 | 0 | 0 | 1 | 1 |
| **3** | 56 | 1 | 1 | 120 | 236 | 0 | 1 | 178 | 0 | 0.8 | 2 | 0 | 2 | 1 |
| **4** | 57 | 0 | 0 | 120 | 354 | 0 | 1 | 163 | 1 | 0.6 | 2 | 0 | 2 | 1 |
| **5** | 57 | 1 | 0 | 140 | 192 | 0 | 1 | 148 | 0 | 0.4 | 1 | 0 | 1 | 1 |
| **6** | 56 | 0 | 1 | 140 | 294 | 0 | 0 | 153 | 0 | 1.3 | 1 | 0 | 2 | 1 |

In [78]:
```
# Subplot of chol, age, thalach
fig, (ax0, ax1) = plt.subplots(nrows= 2 ,
                               ncols = 1,
                               figsize = (10,10),
                               sharex=True)

# Add data to ax0
scatter = ax0.scatter(x = over_50["age"],
                      y = over_50["chol"],
                      c = over_50["target"]);

#Customize ax0
ax0.set(title = "Heart Disease and Cholesterol Levels",
     #xlabel="Age", because of sharex=True

      ylabel="Cholesterol")

#Add a legent to ax0
ax0.legend(*scatter.legend_elements(), title="Target");

# Add a horizontal line
ax0.axhline(over_50["chol"].mean(),
           linestyle = '--');




### ax1
# Add data to ax1
scatter = ax1.scatter(x = over_50["age"],
                      y = over_50["thalach"],
                      c = over_50["target"]);

#Customize ax1
ax1.set(title = "Heart Disease and Max Heart Rate",
      xlabel="Age",
      ylabel="Max Heart Rate")

#Add a legent to ax1
ax1.legend(*scatter.legend_elements(), title="Target");

# Add a horizontal line at ax1
ax1.axhline(y = over_50["thalach"].mean(),
           linestyle = '--');

#Add a title to the figure
fig.suptitle("Heart Disease Analysis", fontsize = 16, fontweight="bold");
```

**Heart Disease Analysis**






## Customizing Matplotlob plots and getting stylish

In [82]:
```
# See the different styles avilable
plt.style.available
```

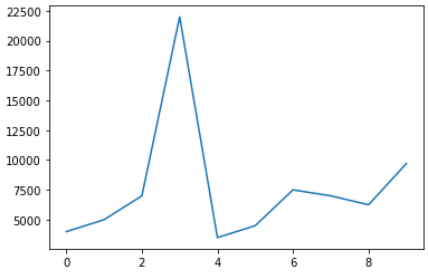
Out[82]:
```
['Solarize_Light2',
 '_classic_test_patch',
 'bmh',
 'classic',
 'dark_background',
 'fast',
 'fivethirtyeight',
 'ggplot',
 'grayscale',
 'seaborn',
 'seaborn-bright',
 'seaborn-colorblind',
 'seaborn-dark',
 'seaborn-dark-palette',
 'seaborn-darkgrid',
 'seaborn-deep',
 'seaborn-muted',
 'seaborn-notebook',
 'seaborn-paper',
 'seaborn-pastel',
 'seaborn-poster',
 'seaborn-talk',
 'seaborn-ticks',
 'seaborn-white',
 'seaborn-whitegrid',
 'tableau-colorblind10']
```

In [83]:
```
car_sales
```

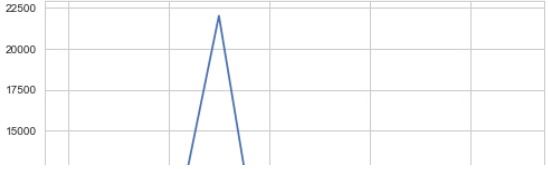
Out[83]:

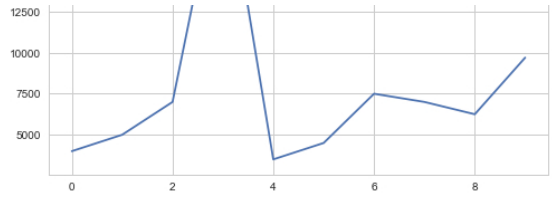
| | Make | Colour | Odometer (KM) | Doors | Price | Sale Date | Total Sales |
|---|---|---|---|---|---|---|---|
| 0 | Toyota | White | 150043 | 4 | 4000 | 2020-01-01 | 4000 |
| 1 | Honda | Red | 87899 | 4 | 5000 | 2020-01-02 | 9000 |
| 2 | Toyota | Blue | 32549 | 3 | 7000 | 2020-01-03 | 16000 |
| 3 | BMW | Black | 11179 | 5 | 22000 | 2020-01-04 | 38000 |
| 4 | Nissan | White | 213095 | 4 | 3500 | 2020-01-05 | 41500 |
| 5 | Toyota | Green | 99213 | 4 | 4500 | 2020-01-06 | 46000 |
| 6 | Honda | Blue | 45698 | 4 | 7500 | 2020-01-07 | 53500 |
| 7 | Honda | Blue | 54738 | 4 | 7000 | 2020-01-08 | 60500 |
| 8 | Toyota | White | 60000 | 4 | 6250 | 2020-01-09 | 66750 |
| 9 | Nissan | White | 31600 | 4 | 9700 | 2020-01-10 | 76450 |

In [86]:
```
car_sales["Price"].plot();
```

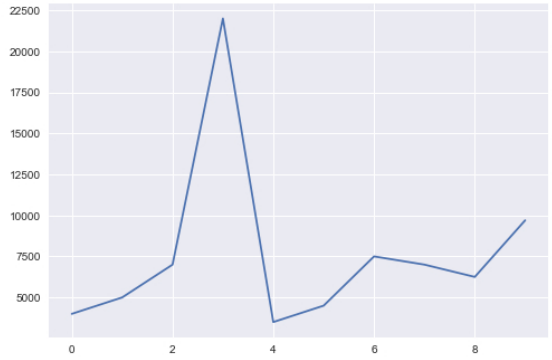



In [90]:
```
plt.style.use('seaborn-whitegrid')
car_sales["Price"].plot();
```

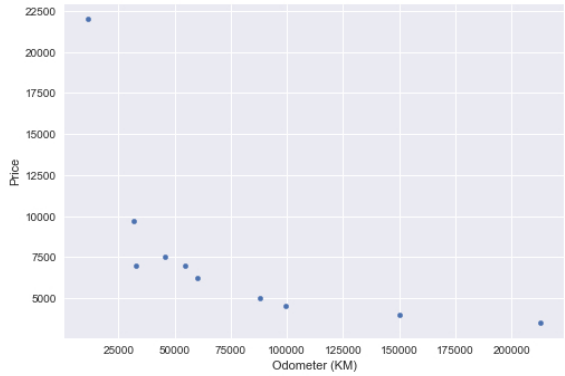




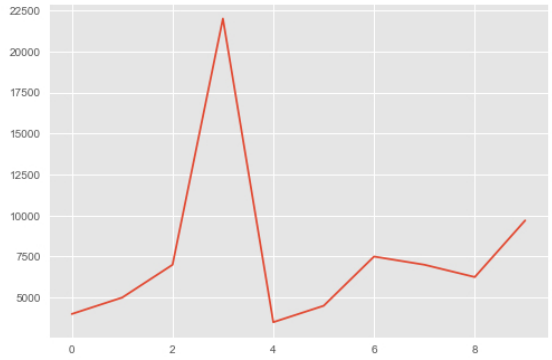

In [92]:
```
plt.style.use('seaborn')
car_sales["Price"].plot();
```




In [93]:
```
car_sales.plot(x = "Odometer (KM)",y = "Price", kind = "scatter");
```




In [94]:
```
plt.style.use('ggplot')
car_sales["Price"].plot();
```




In [96]:
```
# create some data
x = np.random.randn(10, 4)
x
```

Out[96]:
```
array([[ 2.28098942e-01, -2.09675908e-01,  5.82118225e-01,
        -2.46393106e+00],
       [-1.15120591e+00, -1.36760690e-02,  6.70629307e-01,
         1.72148248e-03],
       [ 3.40715733e-01, -6.15938156e-01,  2.81613164e-01,
         2.19124224e+00],
       [ 1.43882828e-01, -1.36321307e+00,  8.01687089e-01,
         1.18643405e+00],
       [-1.67557561e+00,  9.53229057e-01,  3.62746564e-01,
         1.04027738e+00],
       [ 1.38293763e+00,  4.88261085e-01, -4.77871496e-01,
        -1.58033421e+00],
       [-3.42936084e-01,  3.83328211e-02, -7.65508643e-02,
         5.73646078e-01],
       [-3.00764074e-01,  1.30209377e+00, -5.21951930e-01,
        -6.15622712e-02],
       [ 3.40892758e-02,  4.53240449e-01,  6.66522307e-01,
         1.06308100e-01],
       [ 6.92948677e-01,  1.29576497e+00,  3.44342336e-01,
        -3.16970525e-01]])
```

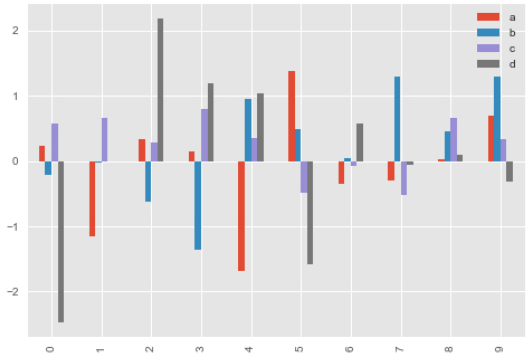
In [99]:
```
df = pd.DataFrame(x, columns=['a','b','c','d'])
df
```

Out[99]:

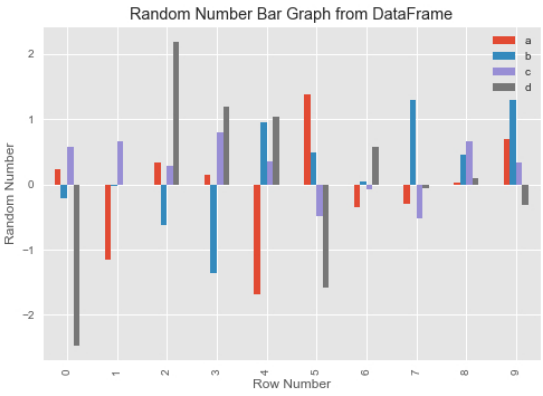
| | a | b | c | d |
|---|---|---|---|---|
| 0 | 0.228099 | -0.209676 | 0.582118 | -2.463931 |
| 1 | -1.151206 | -0.013676 | 0.670629 | 0.001721 |
| 2 | 0.340716 | -0.615938 | 0.281613 | 2.191242 |
| 3 | 0.143883 | -1.363213 | 0.801687 | 1.186434 |
| 4 | -1.675576 | 0.953229 | 0.362747 | 1.040277 |
| 5 | 1.382938 | 0.488261 | -0.477871 | -1.580334 |
| 6 | -0.342936 | 0.038333 | -0.076551 | 0.573646 |
| 7 | -0.300764 | 1.302094 | -0.521952 | -0.061562 |
| 8 | 0.034089 | 0.453240 | 0.666522 | 0.106308 |
| 9 | 0.692949 | 1.295765 | 0.344342 | -0.316971 |

In [102]:
```
ax = df.plot(kind = 'bar')
type(ax)
```

Out[102]: matplotlib.axes._subplots.AxesSubplot



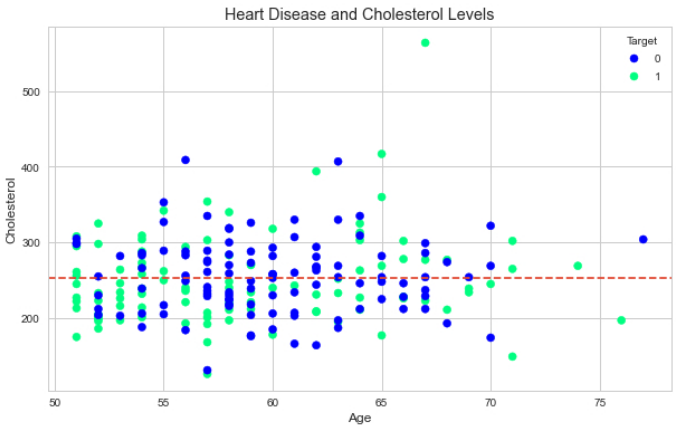

In [101]:
```
# Customize our plot with the set() method
ax = df.plot(kind = 'bar')
# Add some labels and a title
ax.set(title = "Random Number Bar Graph from DataFrame",
       xlabel = 'Row Number',
       ylabel = 'Random Number')

# Make the legend visible
ax.legend().set_visible(True)
```




## change the style again but from within an existing style.

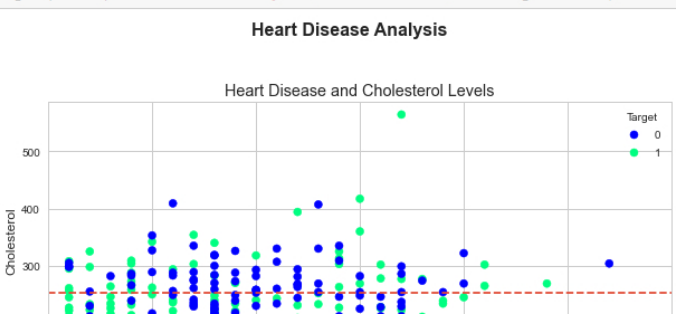
In [109]:
```
# Set the style
plt.style.use('seaborn-whitegrid')

## OO method from scratch
fig, ax = plt.subplots(figsize = (10,6))

# matplotlib colors tutorial for more color scheme

# Plot the data
scatter = ax.scatter(x = over_50["age"],
                     y = over_50["chol"],
                     c = over_50["target"],
                     cmap = "winter"); # this changes the color scheme
                            #"summer"
                            #"plasma'
# Customize the plot
ax.set(title = "Heart Disease and Cholesterol Levels",
       xlabel="Age",
       ylabel="Cholesterol")

#Add a Legent
```

```
ax.legend(*scatter.legend_elements(), title="Target");

# Add a horizontal line
ax.axhline(over_50["chol"].mean(),
           linestyle = '--');
```




In [116]:

```
# Customizing the y and x axis limitations

# Subplot of chol, age, thalach
fig, (ax0, ax1) = plt.subplots(nrows= 2 ,
                               ncols = 1,
                               figsize = (10,10),
                               sharex=True)

# Add data to ax0
scatter = ax0.scatter(x = over_50["age"],
                      y = over_50["chol"],
                      c = over_50["target"],
                       cmap= "winter");

#Customize ax0
ax0.set(title = "Heart Disease and Cholesterol Levels",

     #xlabel="Age", because of sharex=True

      ylabel="Cholesterol")

#### change the x axis limits
ax0.set_xlim([50, 80])

#Add a legent to ax0
ax0.legend(*scatter.legend_elements(), title="Target");

# Add a horizontal line
ax0.axhline(over_50["chol"].mean(),
           linestyle = '--');



### ax1
# Add data to ax1
scatter = ax1.scatter(x = over_50["age"],
                      y = over_50["thalach"],
                      c = over_50["target"],
                       cmap= "winter");

#Customize ax1
ax1.set(title = "Heart Disease and Max Heart Rate",
      xlabel="Age",
      ylabel="Max Heart Rate")

#### Chande ax1 axis limits
ax1.set_xlim([50,80])
ax1.set_ylim([60,200])

#Add a legent to ax1
ax1.legend(*scatter.legend_elements(), title="Target");

# Add a horizontal line at ax1
ax1.axhline(y = over_50["thalach"].mean(),
           linestyle = '--');

#Add a title to the figure
fig.suptitle("Heart Disease Analysis", fontsize = 16, fontweight="bold");
```




## Save figure / plot

In [117]:
```
fig.savefig("Heart-disease-analysis-plot-saved-with-code")
```

In [ ]:
```

```