data-science-introduction

January 12, 2024

1 numpy practice

It is used for mathematical operations

[1]:
```
import numpy as np
import pandas as pd
import matplotlib as plt                    # importing libraries
```

[2]:
```
a = [1,2,3]
b = [1,2,3]                                 # here we are making two list
```

[3]:
```
a+b                                         # If we want to add two list, it␣
↪would get concatenate
```

[3]:
```
[1, 2, 3, 1, 2, 3]
```

[4]:
```
A=np.array(a)                               #  here we are creating two numpy␣
↪array
B=np.array(b)
```

[5]:
```
print(A)
print(a)
```

```
[1 2 3]
[1, 2, 3]
```

[27]:
```
print(type(A),type(a))
```

```
<class 'numpy.ndarray'> <class 'list'>
```

[28]:
```
A+B                                         # we can easily perform␣
↪arithmetic operations on numpy array
```

[28]:
```
array([2, 4, 6])
```

[29]:
```
A*B
```

[29]:
```
array([1, 4, 9])
```

[30]:
```
A%B
```

[30]:
```
array([0, 0, 0])
```

[31]:
```
print(A)
```

```
[1 2 3]
```

[32]:
```
A[1:]                          # slicing
```

[32]:
```
array([2, 3])
```

[33]:
```
A[0:2]
```

[33]:
```
array([1, 2])
```

[34]:
```
A[1:]  * B[1:]                      # slicing
```

[34]:
```
array([4, 9])
```

[ ]:
```

```

[35]:
```
arr=np.array([[1,2,3],[11,22,33]])
```

[36]:
```
arr.shape                    # this is used to find out the shape of the array
```

[36]:
```
(2, 3)
```

[37]:
```
print(arr)
```

```
[[ 1  2  3]
 [11 22 33]]
```

[38]:
```
arr[1,2]
```

[38]:
```
33
```

[39]:
```
arr[0,1]       # to fetch the value at the second row and first column
```

[39]:
```
2
```

[40]:
```
arr[0]
```

[40]:
```
array([1, 2, 3])
```

[41]:
```
arr1=np.random.rand(3,3)  # random.rand(), randomly generates the array of␣
↪given size with element value between 0-1
```

[42]:
```
arr1
```

[42]:
```
array([[0.99449381, 0.5359684 , 0.7503114 ],
       [0.46911568, 0.48613014, 0.29801159],
       [0.14649855, 0.87285144, 0.76960878]])
```

[43]:
```
arr1[0:2,0:2]
```

[43]:
```
array([[0.99449381, 0.5359684 ],
       [0.46911568, 0.48613014]])
```

[6]:
```
arr2=np.random.randint(50,100,2)     # this of numpy randomly generates the␣
 ↪between the given range
```

[7]:
```
arr2
```

[7]:
```
array([83, 93])
```

[8]:
```
arr3=np.random.randn(3,3)            # this is to generate the array of given␣
 ↪size with some negative values
```

[9]:
```
arr3
```

[9]:
```
array([[-0.99030042,  1.30143104,  0.4390901 ],
       [-1.31694987,  0.88348741,  0.36106244],
       [-0.30317285,  0.31708743, -1.37572524]])
```

[11]:
```
np.arange(0,5,2)        #  It is used to create an array of evenly spaced values␣
 ↪within a specified range.
                    # numpy.arange(start, stop, step)
```

[11]:
```
array([0, 2, 4])
```

[ ]:
```

```

2 Pandas practice

It is used for data manipulation and analysis.

[49]:
```
user_data={
    'Marks_A': np.random.randint(50,100,5),
    'Marks_B':np.random.randint(50,100,5),                    # creating a␣
 ↪dictionary
    'Marks_c':np.random.randint(50,100,5)
}
```

[50]:
```
user_data
```

[50]:
```
{'Marks_A': array([93, 92, 89, 86, 94]),
 'Marks_B': array([86, 85, 56, 67, 91]),
 'Marks_c': array([66, 71, 50, 72, 82])}
```

[51]:
```
user_data.values()                    # to get the values of the dictionary
```

[51]:
```
dict_values([array([93, 92, 89, 86, 94]), array([86, 85, 56, 67, 91]),
array([66, 71, 50, 72, 82])])
```

[52]:
```
user_data.keys()
                                                  # to get the keys of the␣
 ↪dictionary
```

[52]:
```
dict_keys(['Marks_A', 'Marks_B', 'Marks_c'])
```

[53]:
```
df=pd.DataFrame(user_data)                        #To create the dataframe from␣
 ↪the given variable
```

[54]:
```
df
```

[54]:

	Marks_A	Marks_B	Marks_c
0	93	86	66
1	92	85	71
2	89	56	50
3	86	67	72
4	94	91	82

[55]:
```
df=pd.DataFrame(user_data,dtype='float')
```

[56]:
```
df
```

[56]:

	Marks_A	Marks_B	Marks_c
0	93.0	86.0	66.0
1	92.0	85.0	71.0
2	89.0	56.0	50.0
3	86.0	67.0	72.0
4	94.0	91.0	82.0

[57]:
```
df.shape          # to get the shape of the Dataframe
```

[57]:
```
(5, 3)
```

[58]:
```
df.head(2)                    # to diplay the upper 2 row of the data frame
```

[58]:
```
   Marks_A  Marks_B  Marks_c
0     93.0     86.0     66.0
1     92.0     85.0     71.0
```

[59]:
```
df.tail(2)                                    # to display the lower two rows␣
 ↪of the data set
```

[59]:
```
   Marks_A  Marks_B  Marks_c
3     86.0     67.0     72.0
4     94.0     91.0     82.0
```

[60]:
```
df.columns                                # to get the columns
```

[60]:
```
Index(['Marks_A', 'Marks_B', 'Marks_c'], dtype='object')
```

[61]:
```
df= df.to_csv('marks.csv')  # to convert the dataframe to csv file
```

[ ]:
```

```

[12]:
```
# To rad the csv file and convert it into the dataframe
```

[62]:
```
my_data=pd.read_csv('marks.csv')
```

[63]:
```
my_data
```

[63]:
```
   Unnamed: 0  Marks_A  Marks_B  Marks_c
0           0     93.0     86.0     66.0
1           1     92.0     85.0     71.0
2           2     89.0     56.0     50.0
3           3     86.0     67.0     72.0
4           4     94.0     91.0     82.0
```

[64]:
```
my_data.info()                                          # this is used to get␣
 ↪the information about the dataframe
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Unnamed: 0  5 non-null      int64
 1   Marks_A     5 non-null      float64
 2   Marks_B     5 non-null      float64
 3   Marks_c     5 non-null      float64
dtypes: float64(3), int64(1)
memory usage: 288.0 bytes
```

[65]:
```
my_data.describe()                          # gives description of the dataframe␣
↪like no. of rows, mean std,
```

[65]:

	Unnamed: 0	Marks_A	Marks_B	Marks_c
count	5.000000	5.000000	5.000000	5.00000
mean	2.000000	90.800000	77.000000	68.20000
std	1.581139	3.271085	14.849242	11.71324
min	0.000000	86.000000	56.000000	50.00000
25%	1.000000	89.000000	67.000000	66.00000
50%	2.000000	92.000000	85.000000	71.00000
75%	3.000000	93.000000	86.000000	72.00000
max	4.000000	94.000000	91.000000	82.00000

[66]:
```
my_data.iloc[1:]                            # slicing the data frame
```

[66]:

	Unnamed: 0	Marks_A	Marks_B	Marks_c
1	1	92.0	85.0	71.0
2	2	89.0	56.0	50.0
3	3	86.0	67.0	72.0
4	4	94.0	91.0	82.0

[67]:
```
x=my_data.iloc[:,1:]  # # it is used to select the rows and column of the␣
↪dataframe
```

[68]:
```
x
```

[68]:

	Marks_A	Marks_B	Marks_c
0	93.0	86.0	66.0
1	92.0	85.0	71.0
2	89.0	56.0	50.0
3	86.0	67.0	72.0
4	94.0	91.0	82.0

[69]:
```
type(x)
```

[69]:
```
pandas.core.frame.DataFrame
```

[70]:
```
y=my_data.iloc[:,1:].values
```

[71]:
```
y
```

[71]:
```
array([[93., 86., 66.],
       [92., 85., 71.],
       [89., 56., 50.],
       [86., 67., 72.],
       [94., 91., 82.]])
```

[72]:
```
type(y)
```

[72]:
```
numpy.ndarray
```

[73]:
```
my_data
```

[73]:

	Unnamed: 0	Marks_A	Marks_B	Marks_c
0	0	93.0	86.0	66.0
1	1	92.0	85.0	71.0
2	2	89.0	56.0	50.0
3	3	86.0	67.0	72.0
4	4	94.0	91.0	82.0

[74]:
```
my_data=my_data.drop(columns='Unnamed: 0')            # to delete a particluar␣
 ↪column form the data frame
```

[75]:
```
my_data
```

[75]:

	Marks_A	Marks_B	Marks_c
0	93.0	86.0	66.0
1	92.0	85.0	71.0
2	89.0	56.0	50.0
3	86.0	67.0	72.0
4	94.0	91.0	82.0

3 matplot lib

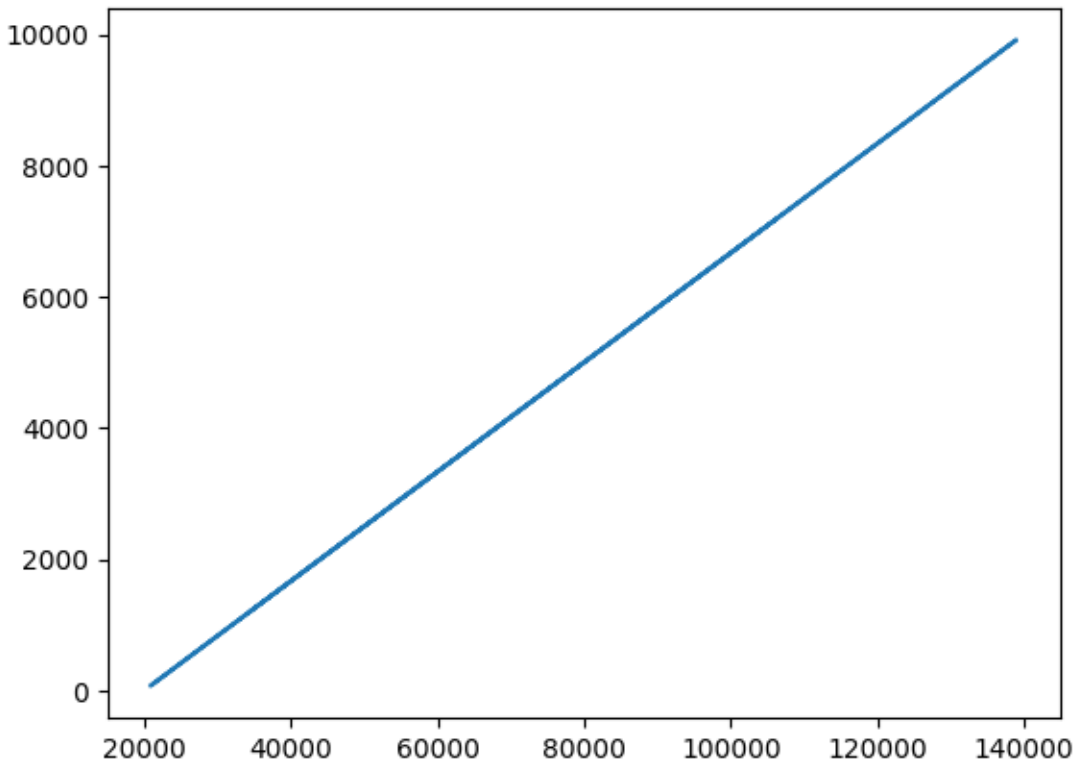
It is used for the visulization.

[76]:
```
import matplotlib.pyplot as plt
```

[77]:
```
x=np.random.randint(1,10000,200)
y= 12 * x + 20000
```

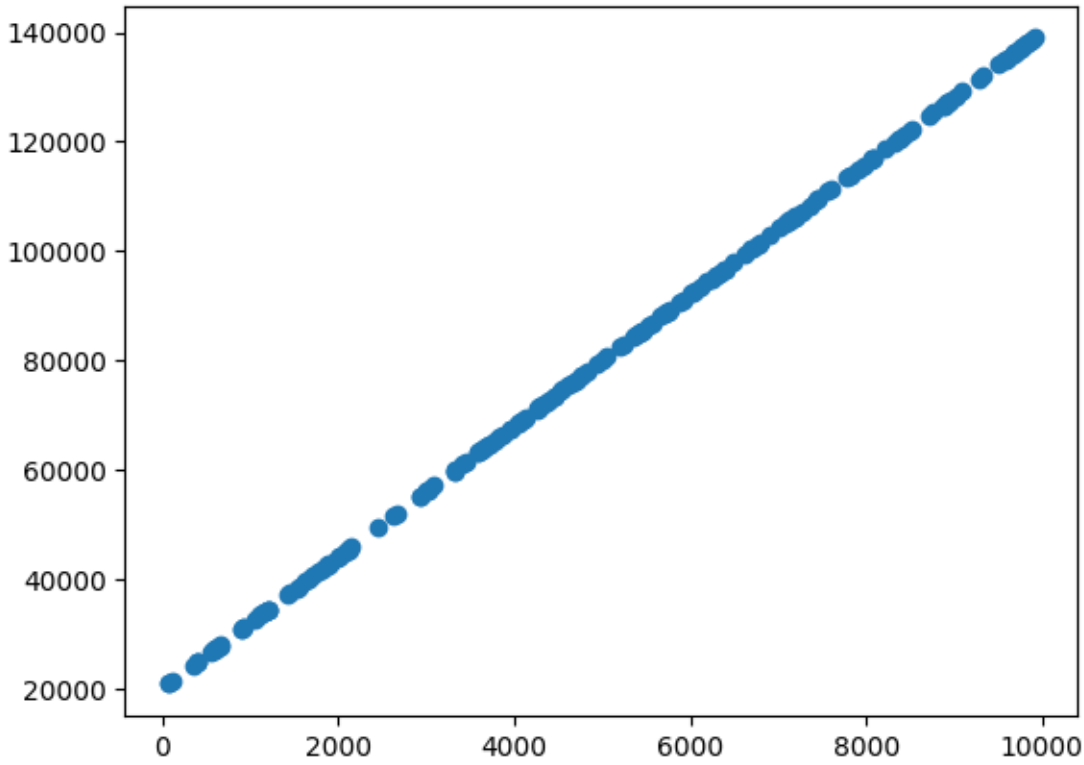
[78]:
```
plt.plot(y,x)
```

[78]:
```
[<matplotlib.lines.Line2D at 0x2366c4cacd0>]
```



[79]:
```
plt.scatter(x,y)
```

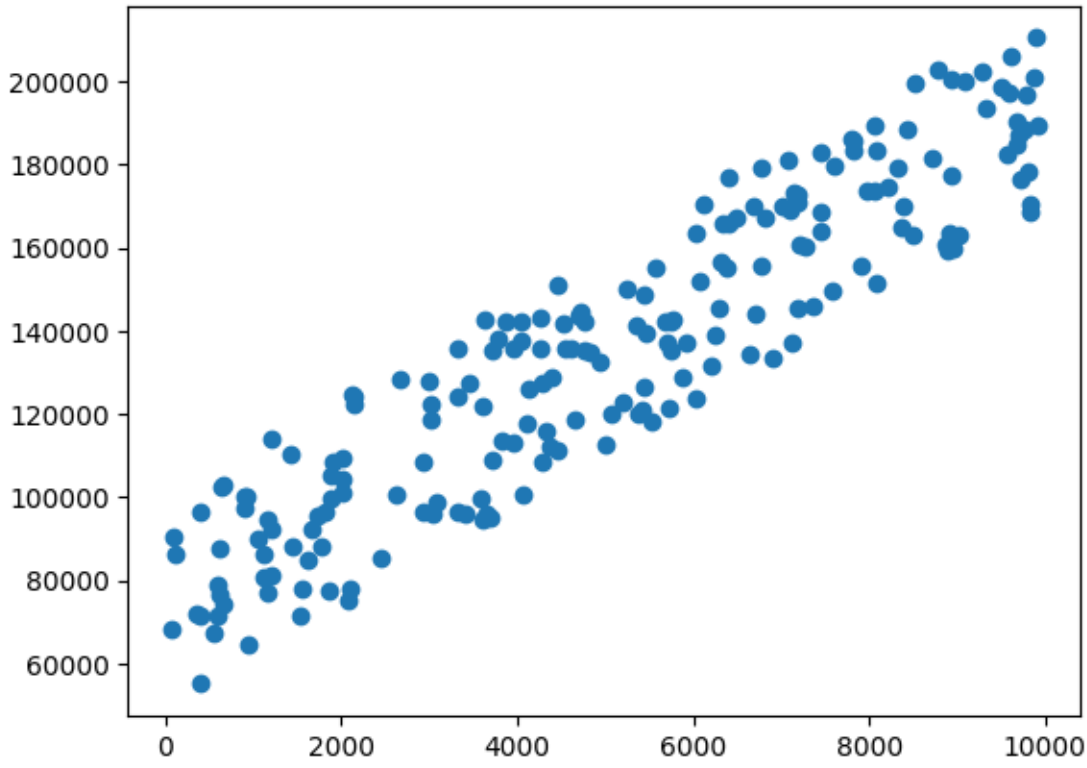
[79]:
```
<matplotlib.collections.PathCollection at 0x2366e58a3d0>
```



[80]:
```
Y= 12 * x + np.random.randint(50000,100000,200)
```

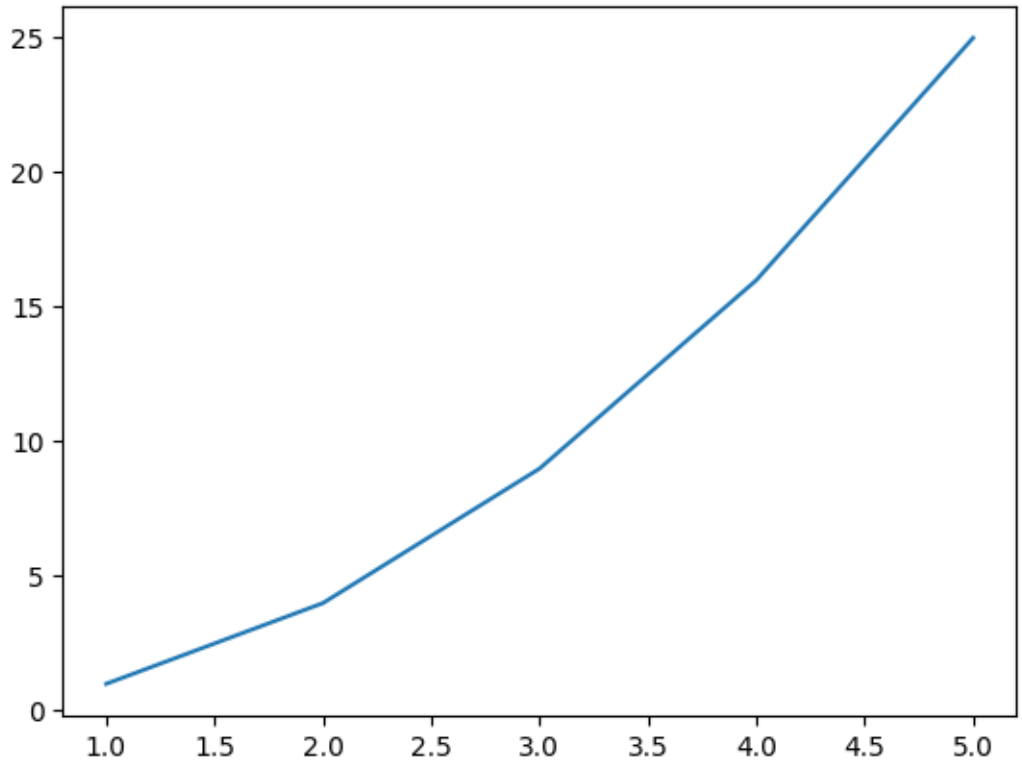
[81]:
```
plt.scatter(x,Y)
```

[81]: <matplotlib.collections.PathCollection at 0x2366e6c2950>



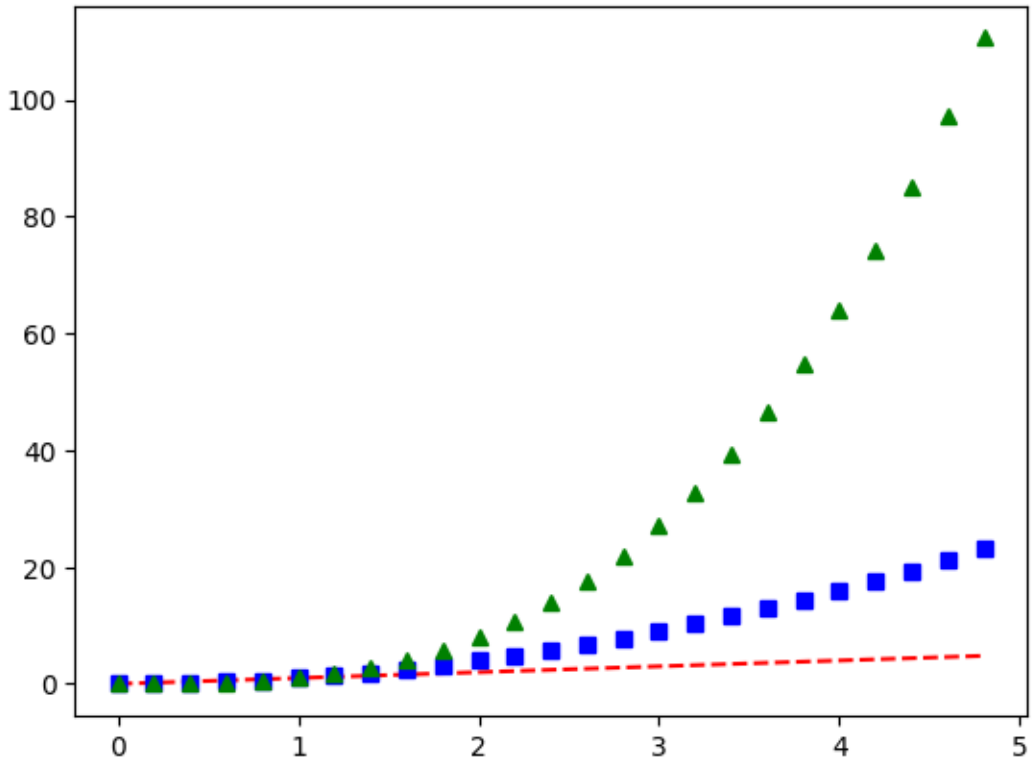
[82]:
```
plt.plot([1,2,3,4,5],[1,4,9,16,25])
```

[82]: [<matplotlib.lines.Line2D at 0x2366e740050>]



[83]:
```
t = np.arange(0., 5., 0.2)
print(t)
# red dashes, blue squares and green triangles
plt.plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
plt.show()
```

```
[0.  0.2 0.4 0.6 0.8 1.  1.2 1.4 1.6 1.8 2.  2.2 2.4 2.6 2.8 3.  3.2 3.4
 3.6 3.8 4.  4.2 4.4 4.6 4.8]
```



[84]:
```
company=['google','apple','sumsang','moto']
x=np.arange(len(company))

y=[100,150,90,120]
```

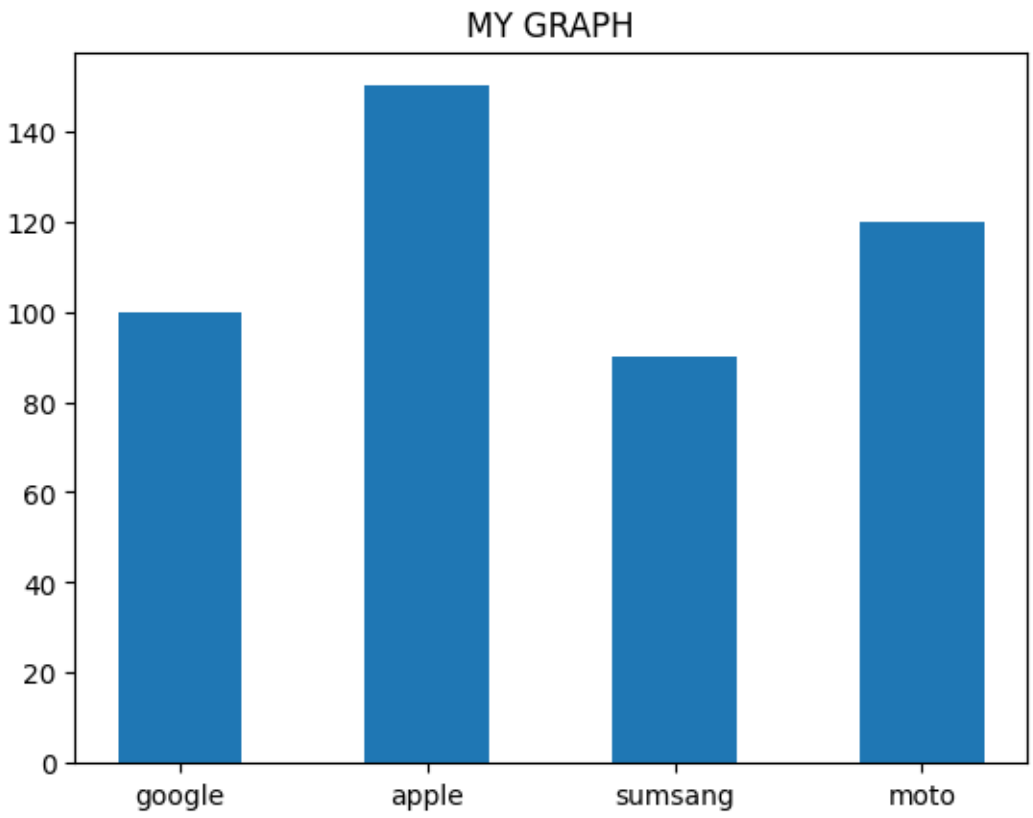
[4]:
```
import numpy as np
X=np.arange(6)
print(X)
```

```
[0 1 2 3 4 5]
```

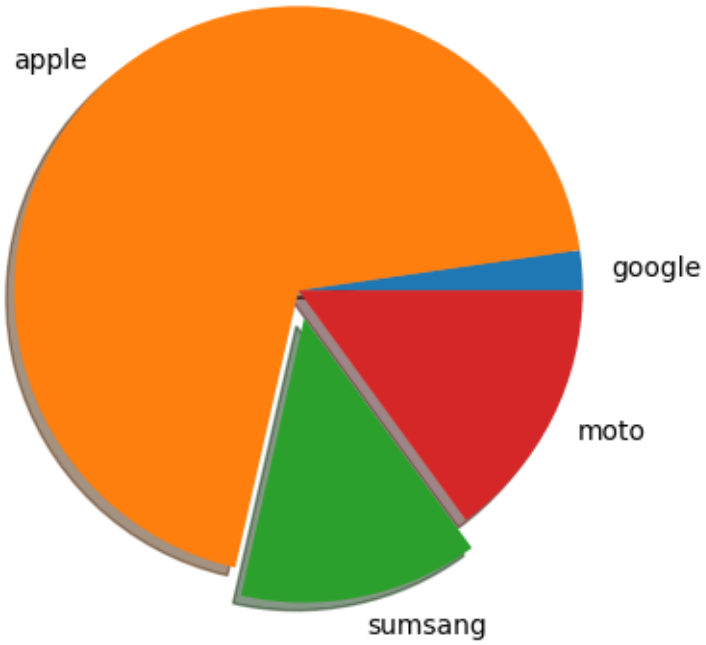
[85]:
```
print(x)
```

```
[0 1 2 3]
```

[86]:
```
plt.bar(x,y,width=0.5)
plt.title('MY GRAPH')
plt.xticks(x,company)
plt.show()
```



[87]:
```
plt.pie([10,305,60,66],labels=company,explode=[0,0,0.1,0],shadow=True)
plt.show()
```


apple
google
moto
sumsang

[ ]: