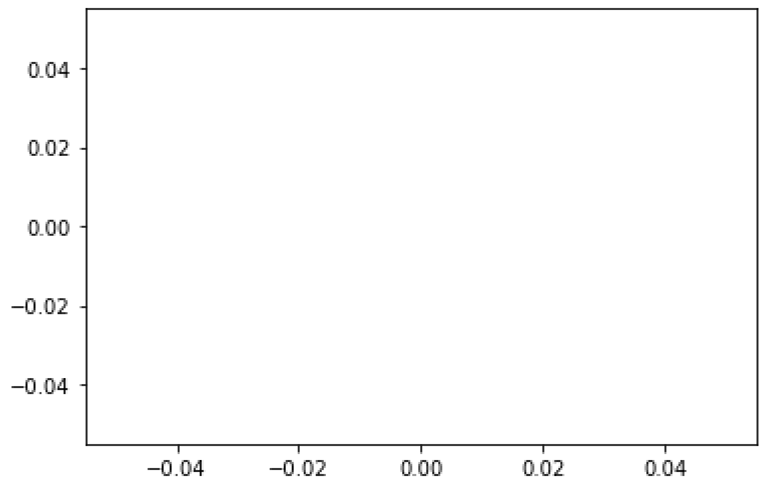
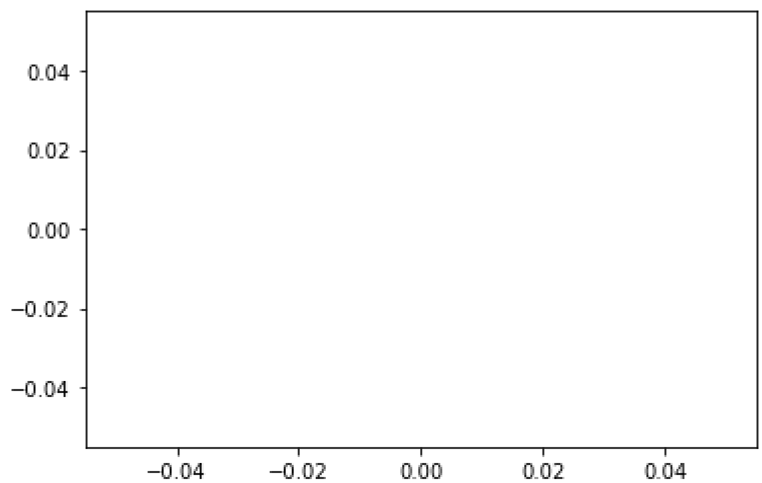
In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
plt.plot()
```

Out[3]: []



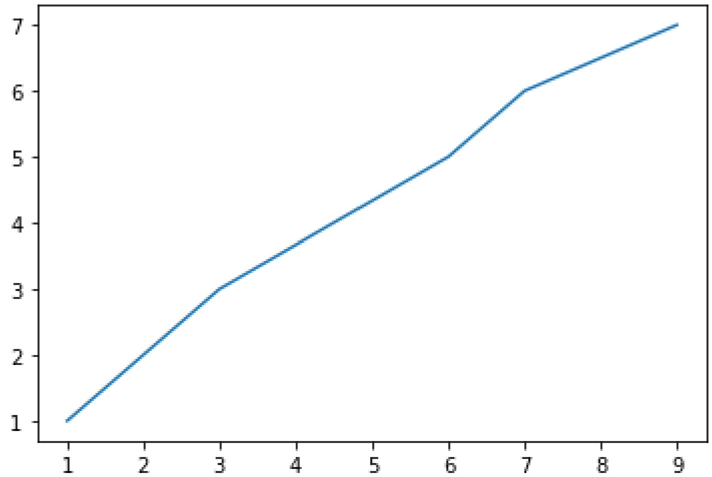
In [4]:
```
plt.plot()
plt.show()
```



In [5]:

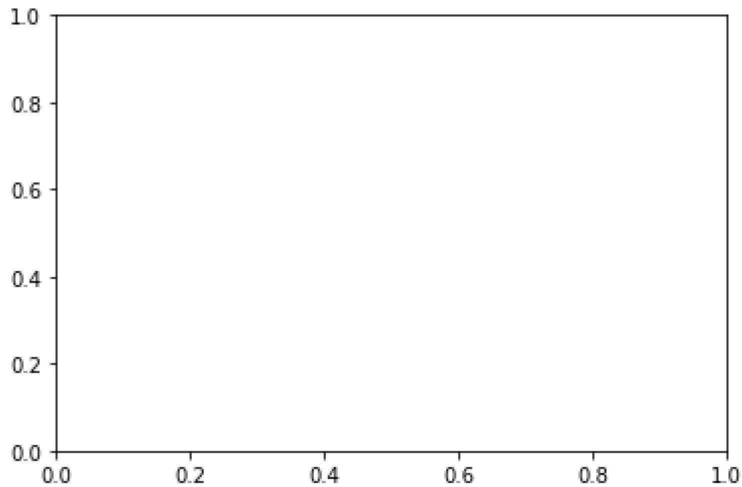
```
x = [1,2,3,6,7,9]
y = [1,2,3,5,6,7]
plt.plot(x,y)
```

Out[5]: [<matplotlib.lines.Line2D at 0x906daf0>]



Methods to Plot

In [6]:
```
figure = plt.figure()
sub_plot = figure.add_subplot()
plt.show()
```

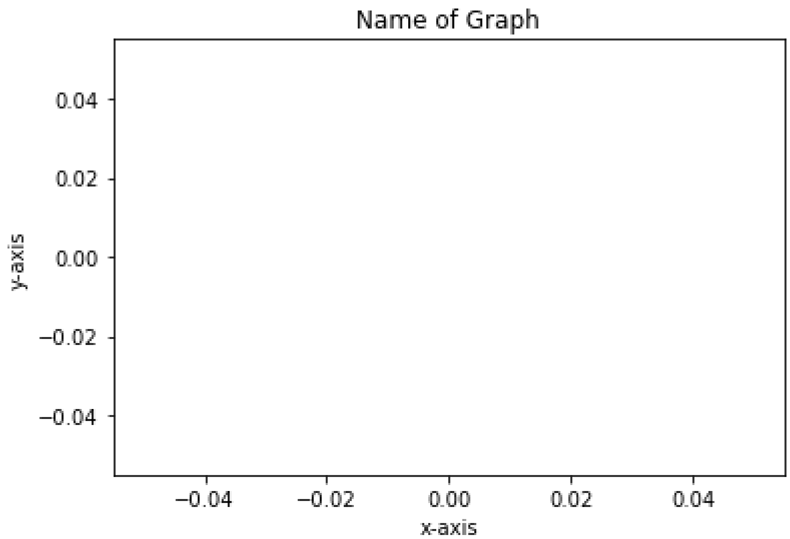


In [7]:
```
figure = plt.figure()
sub_plot = figure.add_axes(1,2,3,4)
plt.show()
```

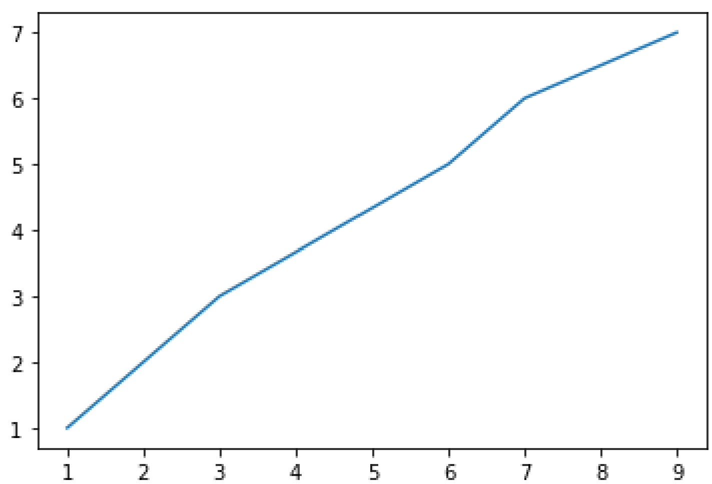
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-7-c53d37eacd79> in <module>
      1 figure = plt.figure()
----> 2 sub_plot = figure.add_axes(1,2,3,4)
      3 plt.show()

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\matplotlib\figure.py
 in add_axes(self, *args, **kwargs)
   1248
   1249             # create the new axes using the axes class given
-> 1250             a = projection_class(self, rect, **kwargs)
   1251
   1252         return self._add_axes_internal(key, a)

C:\Users\AI.khan\desktop\f_project\venv\lib\site-packages\matplotlib\axes\_bas
e.py in __init__(self, fig, rect, facecolor, frameon, sharex, sharey, label, xs
cale, yscale, box_aspect, **kwargs)
    481             self._position = rect
    482         else:
--> 483             self._position = mtransforms.Bbox.from_bounds(*rect)
    484         if self._position.width < 0 or self._position.height < 0:
    485             raise ValueError('Width and height specified must be non-ne
gative')

TypeError: from_bounds() argument after * must be an iterable, not int
```

```
<Figure size 432x288 with 0 Axes>
```

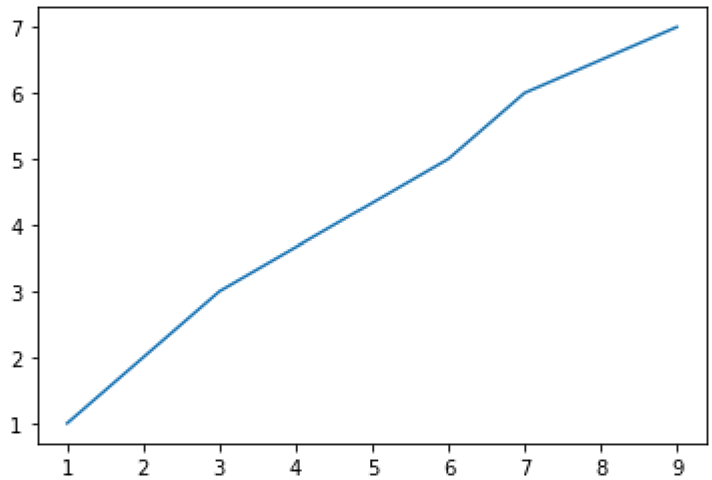
In [8]:
```
home, sub_plot = plt.subplots()
sub_plot.plot()
sub_plot.set(title = 'Name of Graph',
             xlabel = 'x-axis',
             ylabel = 'y-axis')
plt.show()
```



In [9]:
```
figure = plt.figure()
sub_plot = figure.add_subplot()
sub_plot.plot(x,y)
plt.show()
```

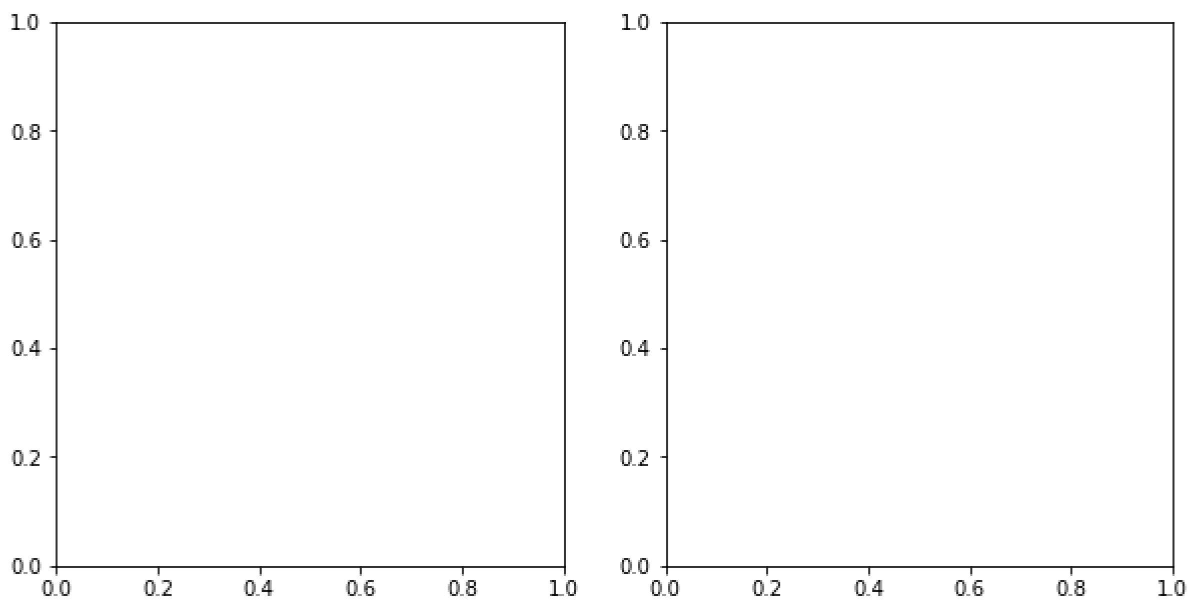


In [10]:
```
home, sub_plot = plt.subplots()
sub_plot.plot(x,y)
plt.show()
```

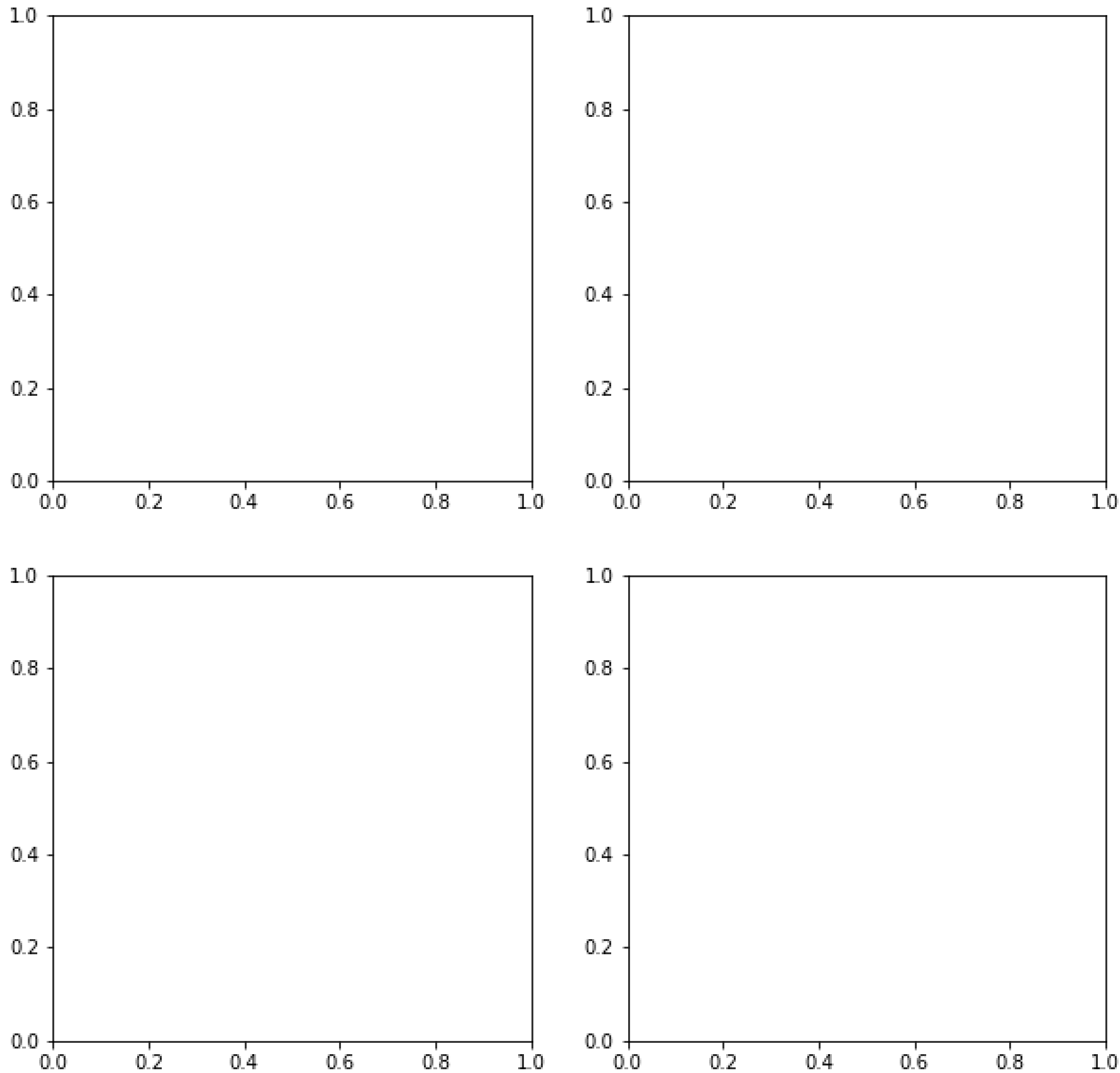


Multiple Plots in Single Figure

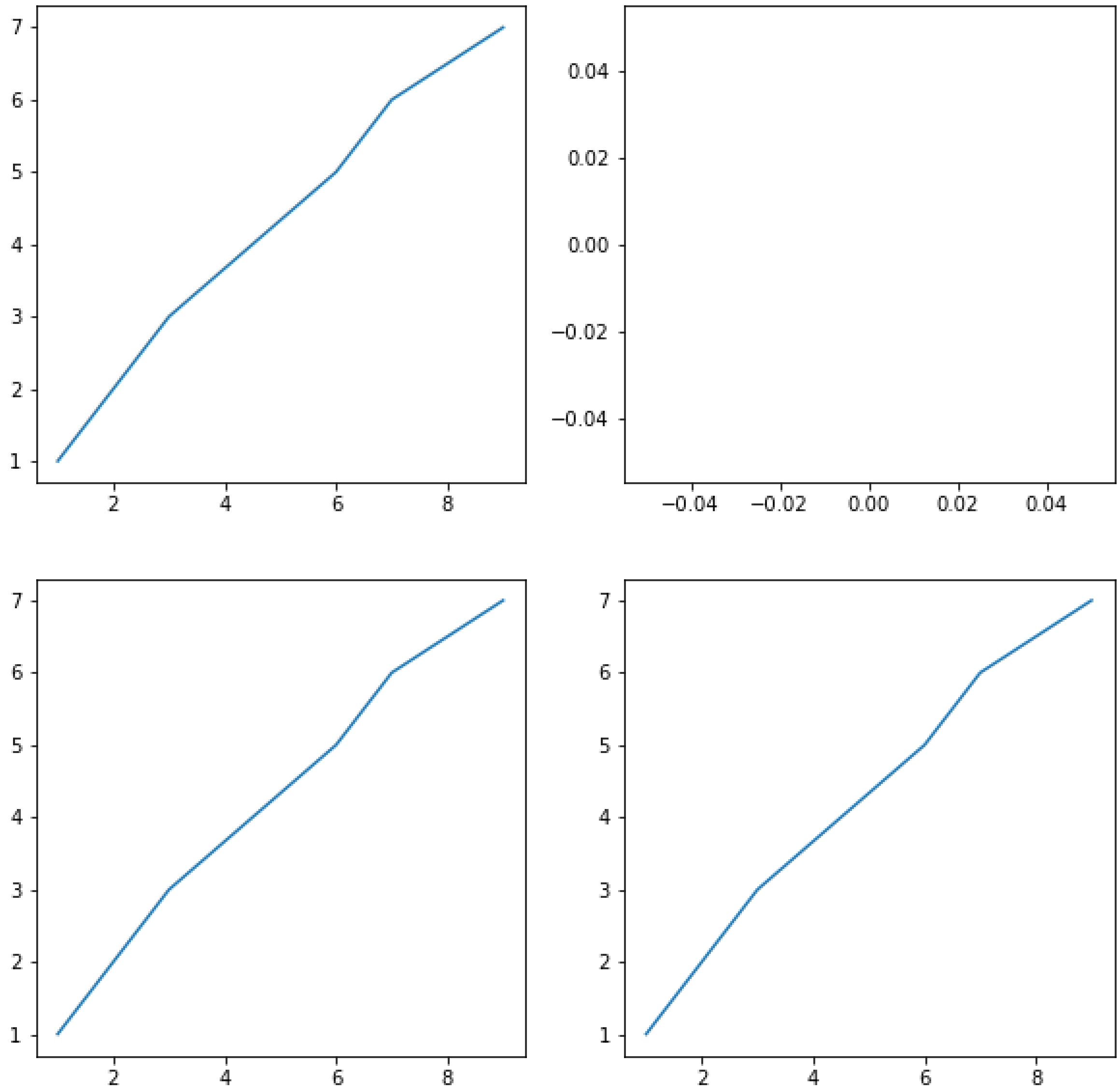
In [11]:
```
figure,(sub_plot_1,sub_plot_2) = plt.subplots(nrows=1,ncols=2, figsize=(10,5))
```



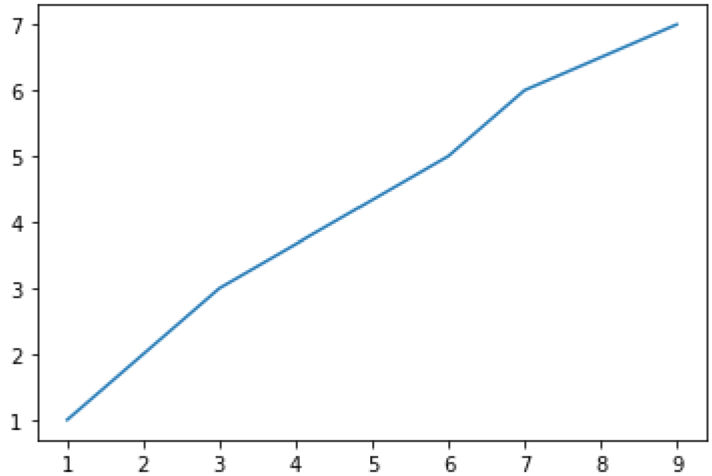
In [12]:
```
figure,((sub_plot_1,sub_plot_2),(sub_plot_3,sub_plot_4)) = plt.subplots(nrows=2,r
```



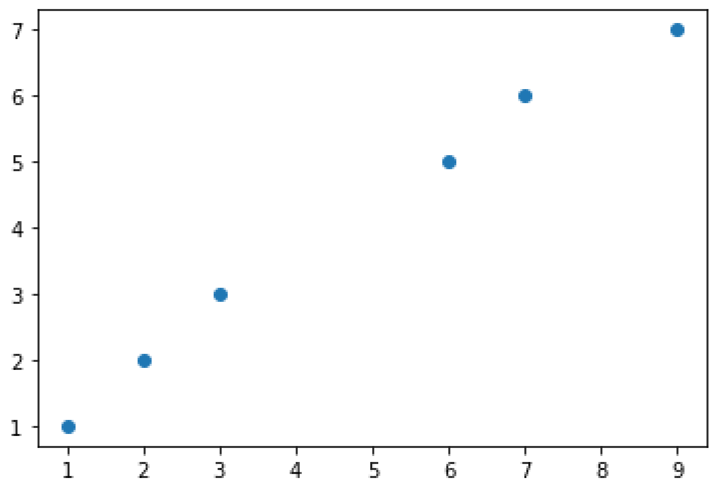
In [13]:
```
figure,((sub_plot_1,sub_plot_2),(sub_plot_3,sub_plot_4)) = plt.subplots(nrows=2,r
sub_plot_1.plot(x,y)
sub_plot_2.plot()
sub_plot_3.plot(x,y)
sub_plot_4.plot(x,y)
plt.show()
```



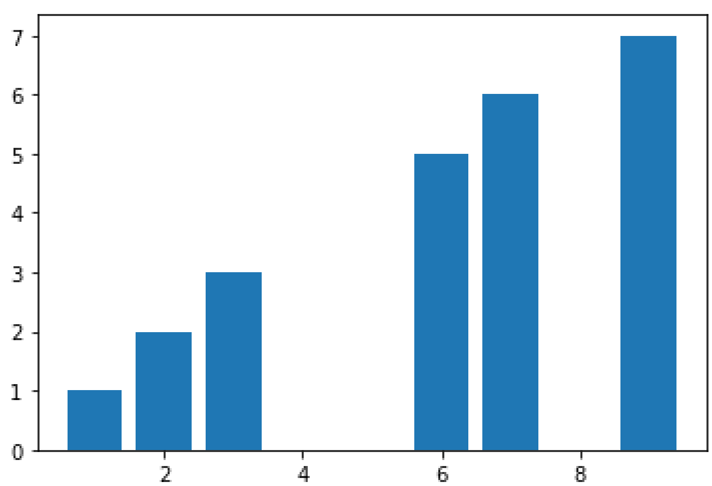
In [14]:
```
figure, sub_plot = plt.subplots()
sub_plot.plot(x,y); # use semi-colon to remove path address
```



In [15]:
```
figure, sub_plot=plt.subplots()
sub_plot.scatter(x,y);
```

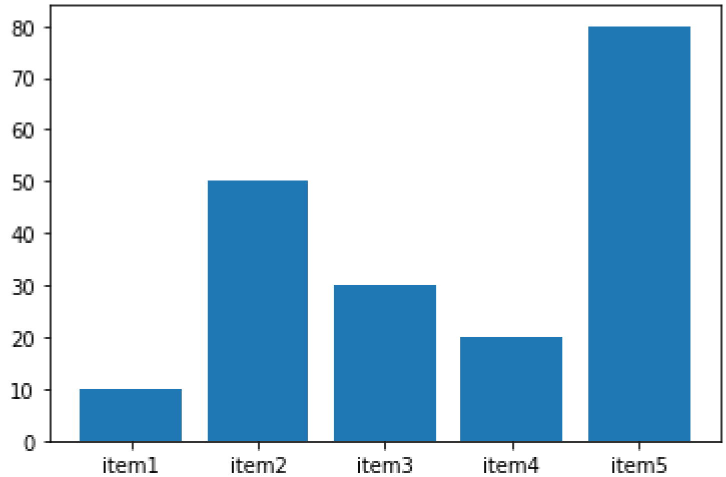


In [16]:
```
figure, sub_plot=plt.subplots()
sub_plot.bar(x,y);
```

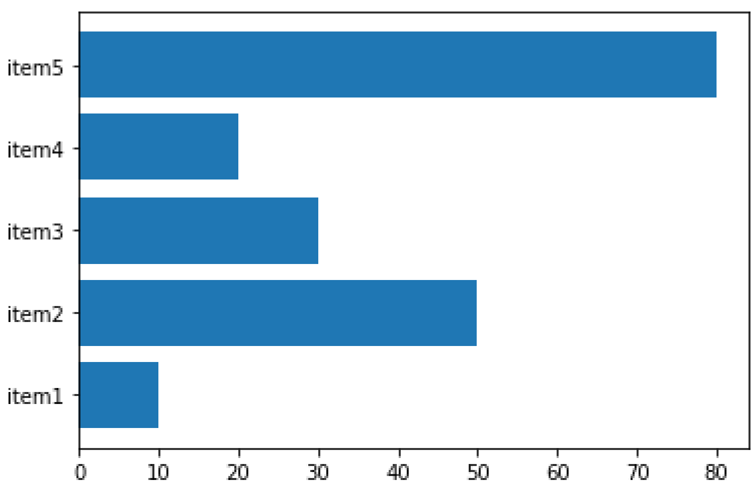


In [ ]:

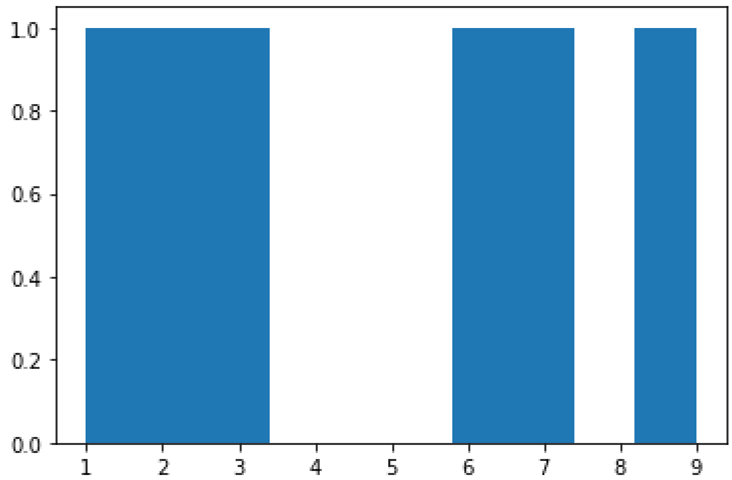
In [17]:
```
shop = { 'item1':10,
       'item2':50,'item3':30,'item4':20, 'item5':80}
figure, sub_plot=plt.subplots()
sub_plot.bar(shop.keys(), shop.values());
```



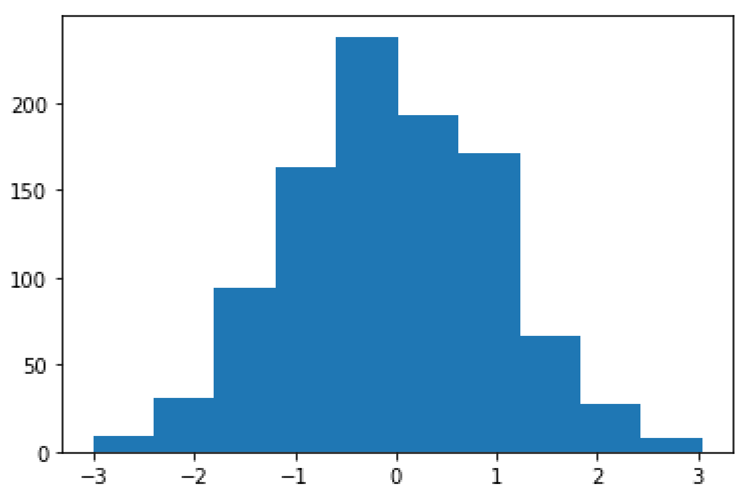
In [18]:
```
shop = { 'item1':10,
       'item2':50,'item3':30,'item4':20, 'item5':80}
figure, sub_plot=plt.subplots()
sub_plot.barh(list(shop.keys()), list(shop.values()));
```



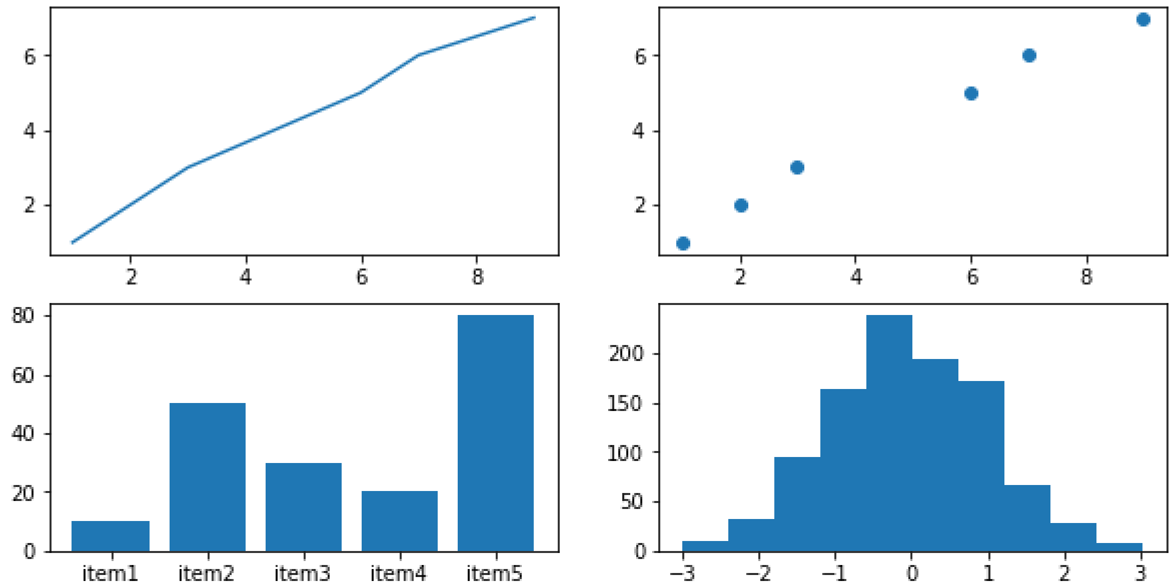
In [19]:
```
figure, sub_plot=plt.subplots()
sub_plot.hist(x);
```



In [20]:
```
a =np.random.randn(1000)
figure, sub_plot=plt.subplots()
sub_plot.hist(a);
```



In [21]:
```
figure, ((sub_plot1,sub_plot2),(sub_plot3,sub_plot4))=plt.subplots(nrows=2,ncols=
sub_plot1.plot(x,y);
sub_plot2.scatter(x,y);
sub_plot3.bar(shop.keys(),shop.values());
sub_plot4.hist(a);
```



In [22]:
```
data = pd.read_csv('marine-economy.csv')
data
```

Out[22]:

	year	category	variable	units	magnitude	source	data_value	flag
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F

In [23]:
```
type(data['flag'][0])
```

Out[23]:
```
str
```

In [24]:
```
added_column=pd.Series(['$400.00','$400.00','$400.00','$400.00','$400.00','$400.0
data['added_column']= added_column
```

In [25]:
```
data
```

Out[25]:

	year	category	variable	units	magnitude	source	data_value	flag	added_colu
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	$40(
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F	$40(
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	$40(
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F	$40(
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	$40(
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	$40(
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F	$40(
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	$40(
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	$40(

In [26]:
```
type(data['added_column'][0])
```

Out[26]:
```
str
```

In [27]:

```
data['added_column']=data['added_column'].str.replace('[\$\.]','')
data
```

Out[27]:

	year	category	variable	units	magnitude	source	data_value	flag	added_colu
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	40
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F	40
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	40
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F	40
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	40
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	40
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F	40
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	40
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	40

In [28]:
```
data['added_column']=data['added_column'].str[:-2]
data
```

Out[28]:

	year	category	variable	units	magnitude	source	data_value	flag	added_colu
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F	
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F	
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F	
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	

In [29]:
```
data['added_column']=data['added_column'].astype(int)
type(data['added_column'][0])
```

Out[29]: numpy.int32

In [30]:
```
data['n_added_column']=data['added_column']
data
```

Out[30]:

	year	category	variable	units	magnitude	source	data_value	flag	added_colu
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F	
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F	
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F	
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	

In [31]:
```
type(data['n_added_column'][0])
```

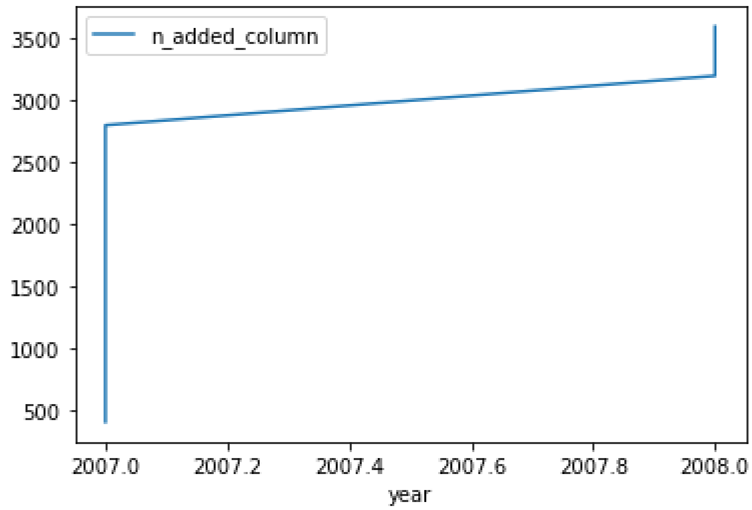
Out[31]:
```
numpy.int32
```

In [32]:
```
data['n_added_column']=data['n_added_column'].cumsum()
data
```

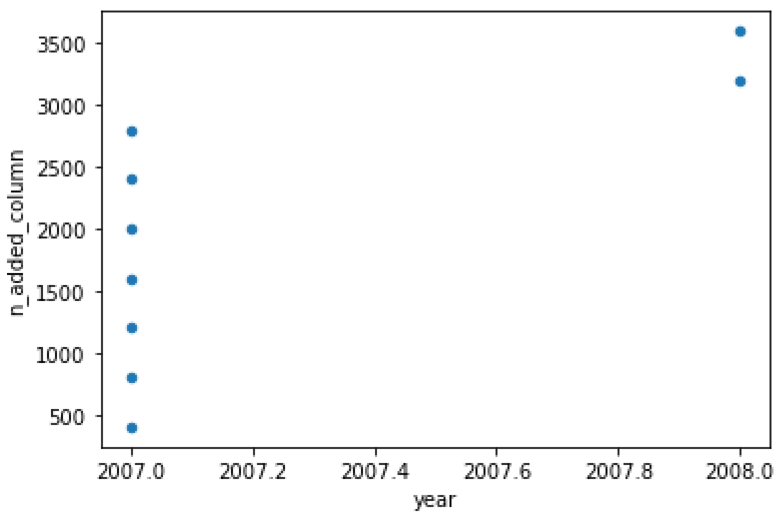
Out[32]:

	year	category	variable	units	magnitude	source	data_value	flag	added_colu
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F	
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F	
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F	
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	

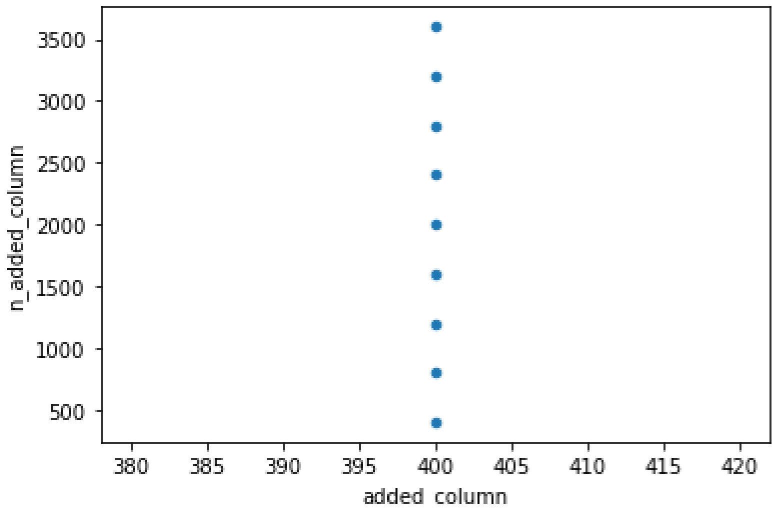
In [33]:
```
data.plot(x='year', y='n_added_column');
```



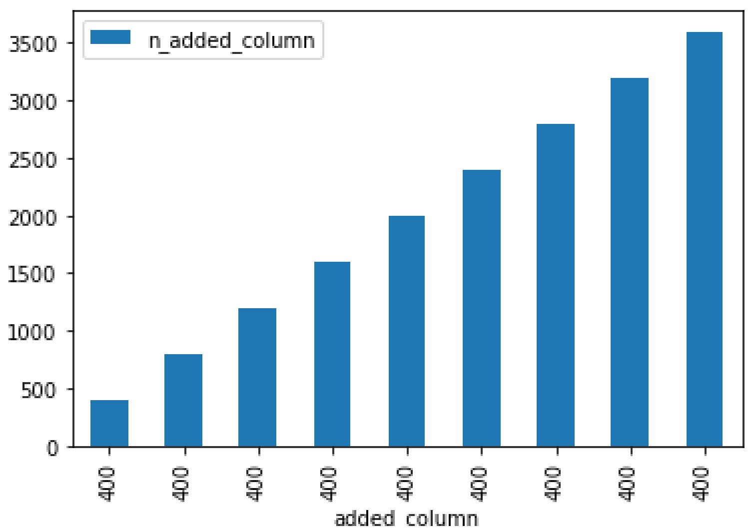
In [34]:
```
data.plot(x='year', y='n_added_column',kind='scatter');
```



In [35]:
```
data.plot(x='added_column', y='n_added_column',kind='scatter');
```



In [36]:
```
data.plot(x='added_column', y='n_added_column',kind='bar');
```



In [37]:
```
x=np.random.rand(4,4)
x
df=pd.DataFrame(x,index=['a','b','c','d'],columns=['a','b','c','d'])
df
```

Out[37]:

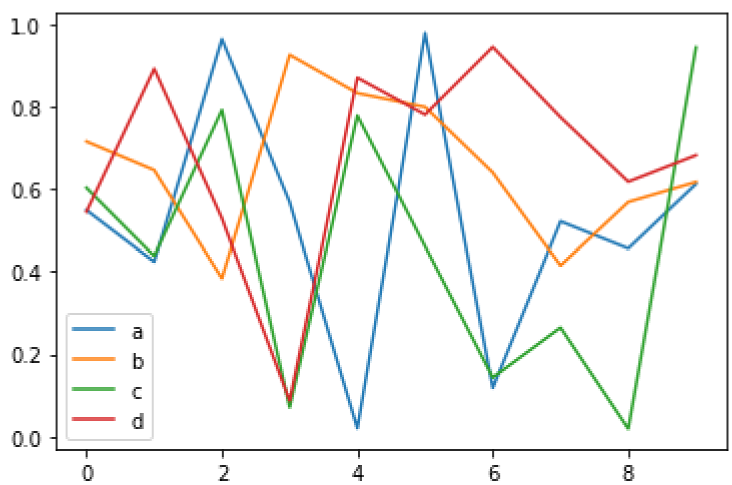
	a	b	c	d
a	0.662125	0.641717	0.211680	0.953232
b	0.254342	0.414241	0.863122	0.357922
c	0.860529	0.111735	0.285410	0.258927
d	0.415626	0.416776	0.634390	0.570526

In [38]:
```
np.random.seed(seed=0)
x=np.random.rand(10,4)
x
df=pd.DataFrame(x,columns=['a','b','c','d'])
df
```

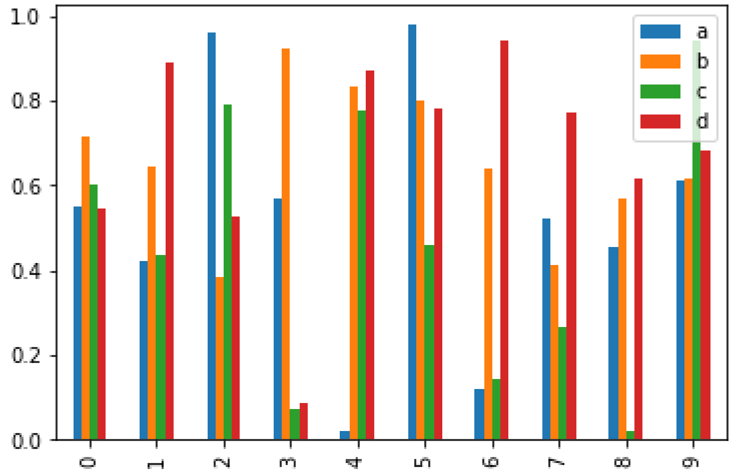
Out[38]:

	a	b	c	d
0	0.548814	0.715189	0.602763	0.544883
1	0.423655	0.645894	0.437587	0.891773
2	0.963663	0.383442	0.791725	0.528895
3	0.568045	0.925597	0.071036	0.087129
4	0.020218	0.832620	0.778157	0.870012
5	0.978618	0.799159	0.461479	0.780529
6	0.118274	0.639921	0.143353	0.944669
7	0.521848	0.414662	0.264556	0.774234
8	0.456150	0.568434	0.018790	0.617635
9	0.612096	0.616934	0.943748	0.681820

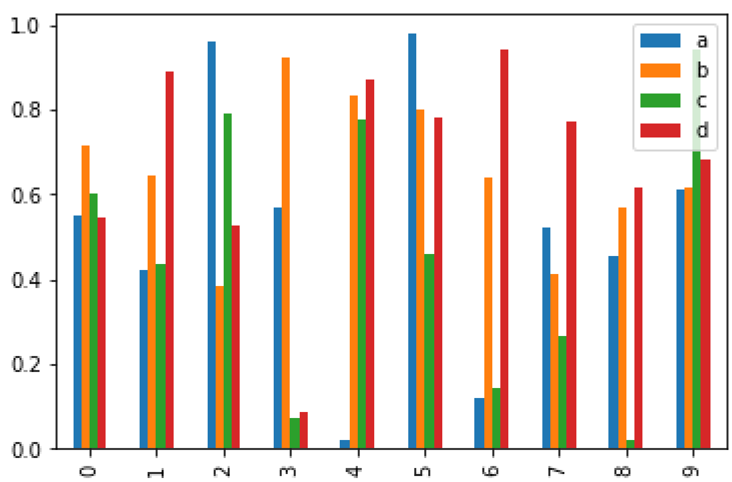
In [39]:
```
df.plot();
```



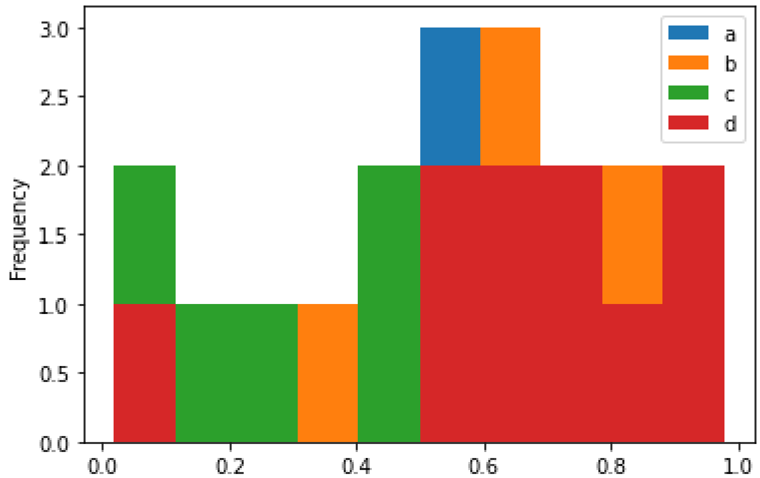
In [40]:
```
df.plot(kind='bar');
```



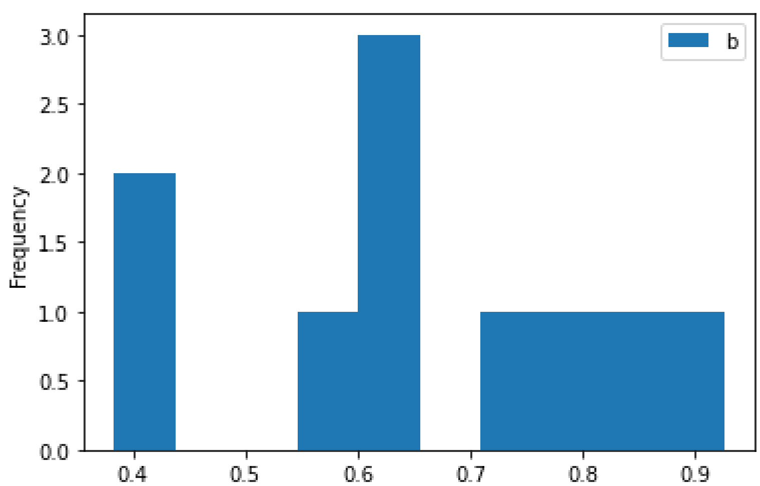
In [41]:
```
df.plot.bar();
```



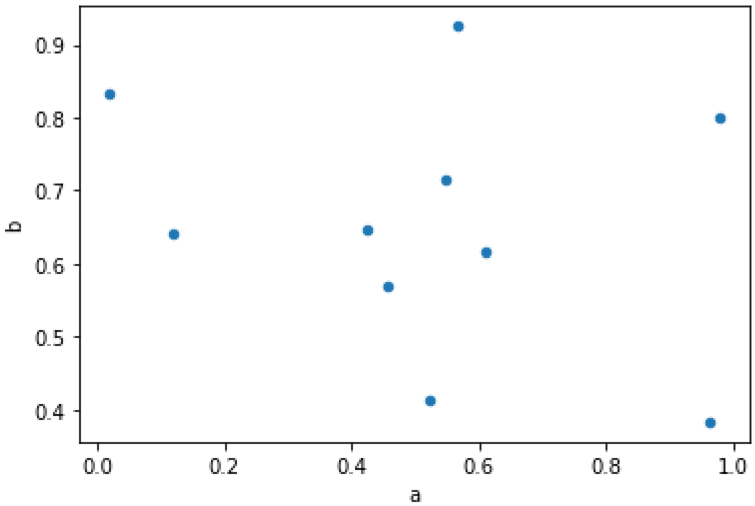
In [42]:
```
df.plot.hist();
```



In [43]:
```
df.plot.hist(x='a',y='b');
```



In [44]:
```
df.plot.scatter(x='a',y='b'); # scatter can only be plotted by giving x, y coordi
```

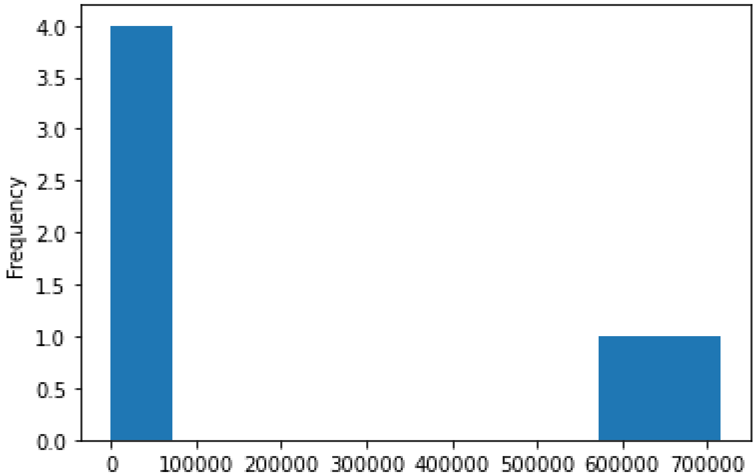


In [45]:
```
data
```

Out[45]:

	year	category	variable	units	magnitude	source	data_value	flag	added_colu
0	2007	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	43.1	R	
1	2007	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	NaN	F	
2	2007	Fisheries and aquaculture	Contribution to marine economy earnings	Proportion	Actual	LEED	42.7	R	
3	2007	Fisheries and aquaculture	Contribution to total GDP	Proportion	Actual	Environmental Accounts	NaN	F	
4	2007	Fisheries and aquaculture	GDP	Dollars	Thousands	Environmental Accounts	715722.0	F	
5	2007	Fisheries and aquaculture	Gross earnings	Dollars	Thousands	LEED	582377.0	F	
6	2007	Fisheries and aquaculture	Wage and salary earners	Number	Actual	LEED	NaN	F	
7	2008	Fisheries and aquaculture	Cont. to ME Wage and salary earners	Proportion	Actual	LEED	39.9	R	
8	2008	Fisheries and aquaculture	Contribution to marine economy GDP	Proportion	Actual	Environmental Accounts	14.2	F	

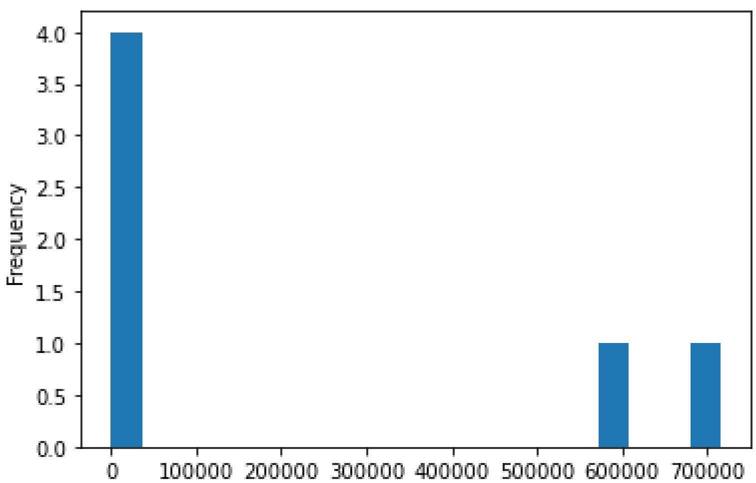
In [46]:
```
data['data_value'].plot.hist();
```



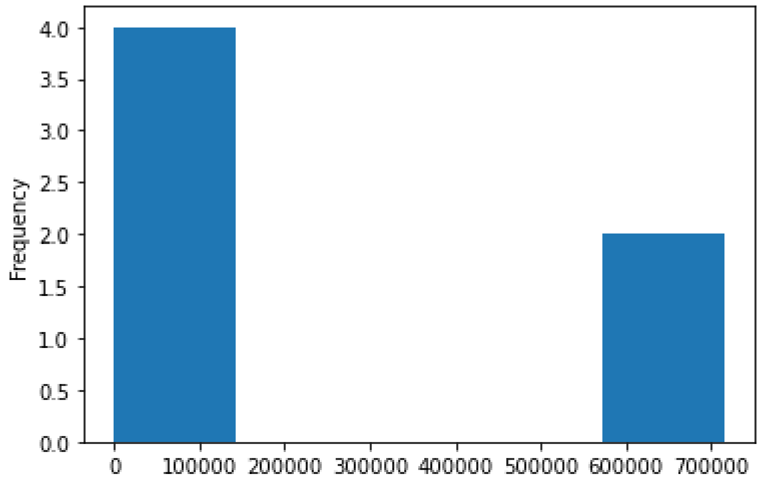
In [47]:
```
Outliers:
data shown out of normal distribution.  This donot represent data. They can mess
```

```
  File "<ipython-input-47-e90ef8362d6e>", line 1
    Outliers:
             ^
SyntaxError: invalid syntax
```

In [48]:
```
data['data_value'].plot.hist(bins=20);
```



In [49]:
```
data['data_value'].plot.hist(bins=5);
```



Pyplot Vs Object Oriented Method Plot

In [50]:
```
heart_disease=pd.read_csv('heart_failure_clinical_records_dataset.csv')
heart_disease
```

Out[50]:

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	p
0	75.0	0	582	0	20	1	26
1	55.0	0	7861	0	38	0	26
2	65.0	0	146	0	20	0	16
3	50.0	1	111	0	20	0	21
4	65.0	1	160	1	20	0	32
...	...	...	...	...	...	...	
294	62.0	0	61	1	38	1	15
295	55.0	0	1820	0	38	0	27
296	45.0	0	2060	1	60	0	74
297	45.0	0	2413	0	38	0	14
298	50.0	0	196	0	45	0	39

299 rows × 13 columns

Pyplot Method

In [51]:
```
over_forty = heart_disease[heart_disease['age']>40]
over_forty
```

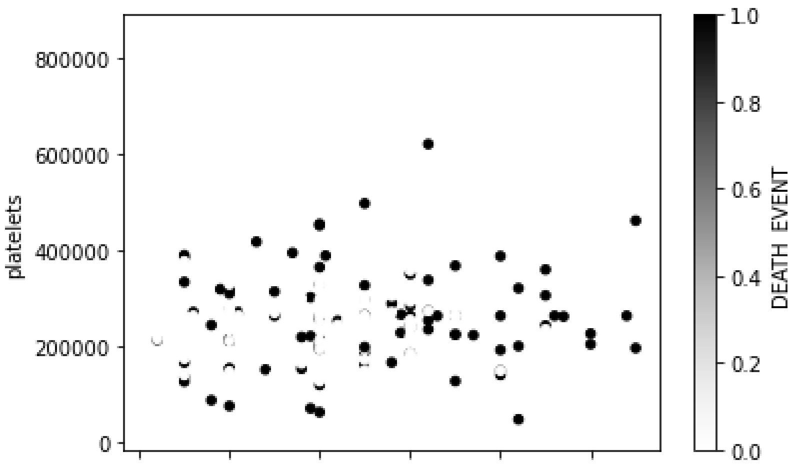
Out[51]:

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	p
0	75.0	0	582	0	20	1	26
1	55.0	0	7861	0	38	0	26
2	65.0	0	146	0	20	0	16
3	50.0	1	111	0	20	0	21
4	65.0	1	160	1	20	0	32
...	...	...	...	...	...	...	
294	62.0	0	61	1	38	1	15
295	55.0	0	1820	0	38	0	27
296	45.0	0	2060	1	60	0	74
297	45.0	0	2413	0	38	0	14
298	50.0	0	196	0	45	0	39

292 rows × 13 columns

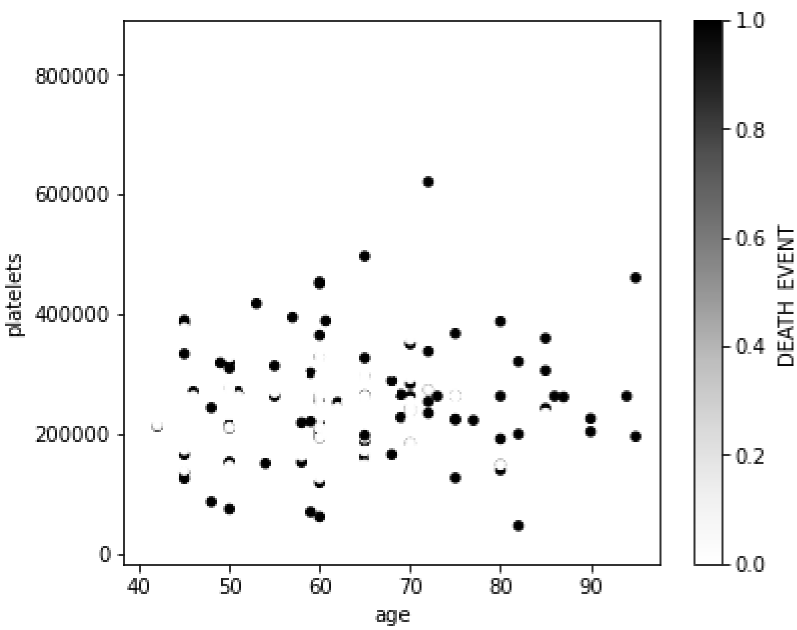
Object Oriented Method

In [52]:
```python
over_forty.plot(kind='scatter', x='age',
                 y='platelets',
               c = 'DEATH_EVENT');
```



In scatter plot there is third parameter which does color-coding on the basis of suspectibility. over_40.plot(kind=scatter, x=age,y=chol, c =target) c is showing suspectibility of disease wrt occurence.

In [53]:
```python
figure, ax=plt.subplots(figsize=(6,5))
over_forty.plot(kind='scatter', x='age',
                 y='platelets',
               c = 'DEATH_EVENT',ax=ax);
```



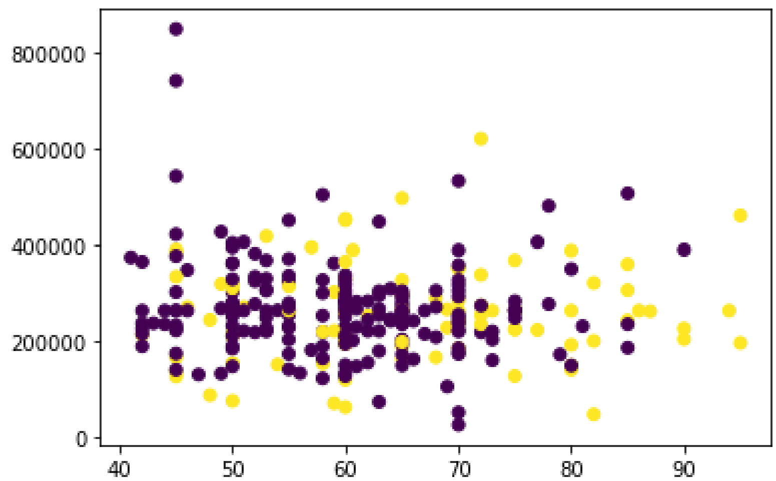
If ponder upon the graph, it can be seen that x axis is more defined. Therefore OO is better than py.

ax is Base class for subplots, which are :class: `Axes` instances with additional methods to facilitate generating and manipulating a set of :class: `Axes` within a figure.

In other words, assigned data to to axes and to suspectibility variable is given to ax which after that save and plot it.

In [54]:
```
#Creating figure
figure, sub_plot=plt.subplots(figsize=(6,4))
#Plotting Data
sub_plot.scatter(x=over_forty['age'],
          y=over_forty['platelets'],
          c= over_forty['DEATH_EVENT'], )
```

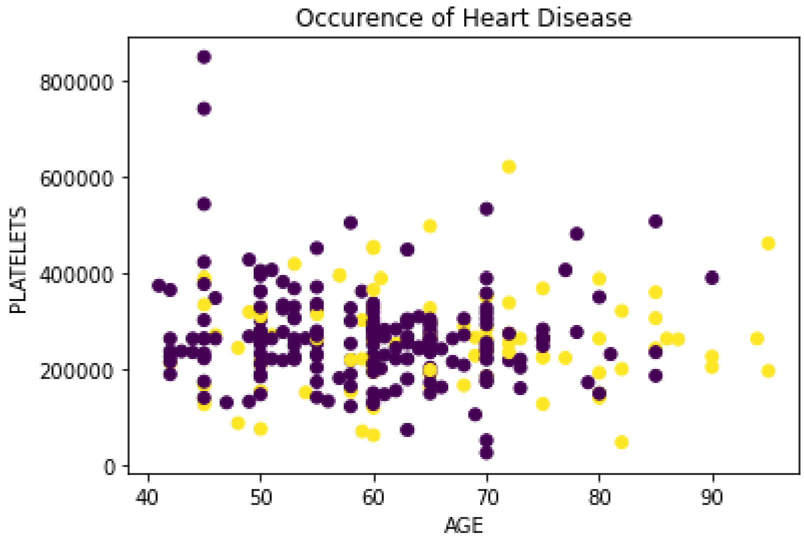
Out[54]:
```
<matplotlib.collections.PathCollection at 0xafce790>
```



above graph DOES NOT HAVE NAME Of AXIS, IT WOULD BE GIVEN MANUALLY in this method of "ax.scatter" while in previous method column name is only given which not only plot data but also gives name to axis and to label c.

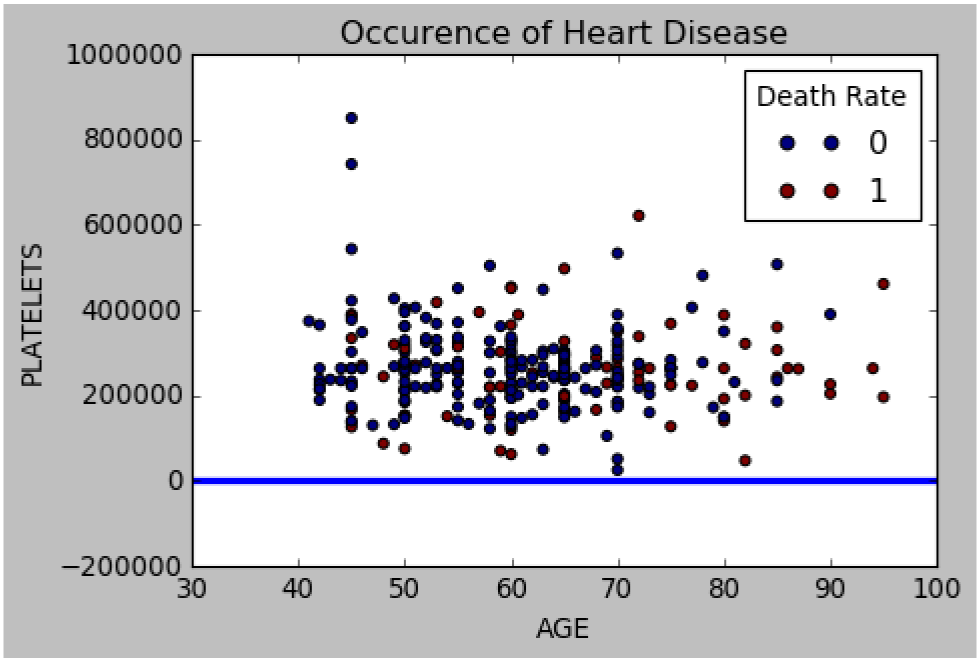
Customization

In [55]:
```
#Creating figure
figure, ax=plt.subplots(figsize=(6,4))# here variable must be ax.
#Plotting Data
sub_plot=ax.scatter(x=over_forty['age'],
          y=over_forty['platelets'],
          c= over_forty['DEATH_EVENT'], ) # IN THIS FRAMEWORK IT DOES NOT HAVE GI
#Customization
ax.set(title = 'Occurence of Heart Disease', xlabel='AGE', ylabel='PLATELETS' );
```



Setting Legend

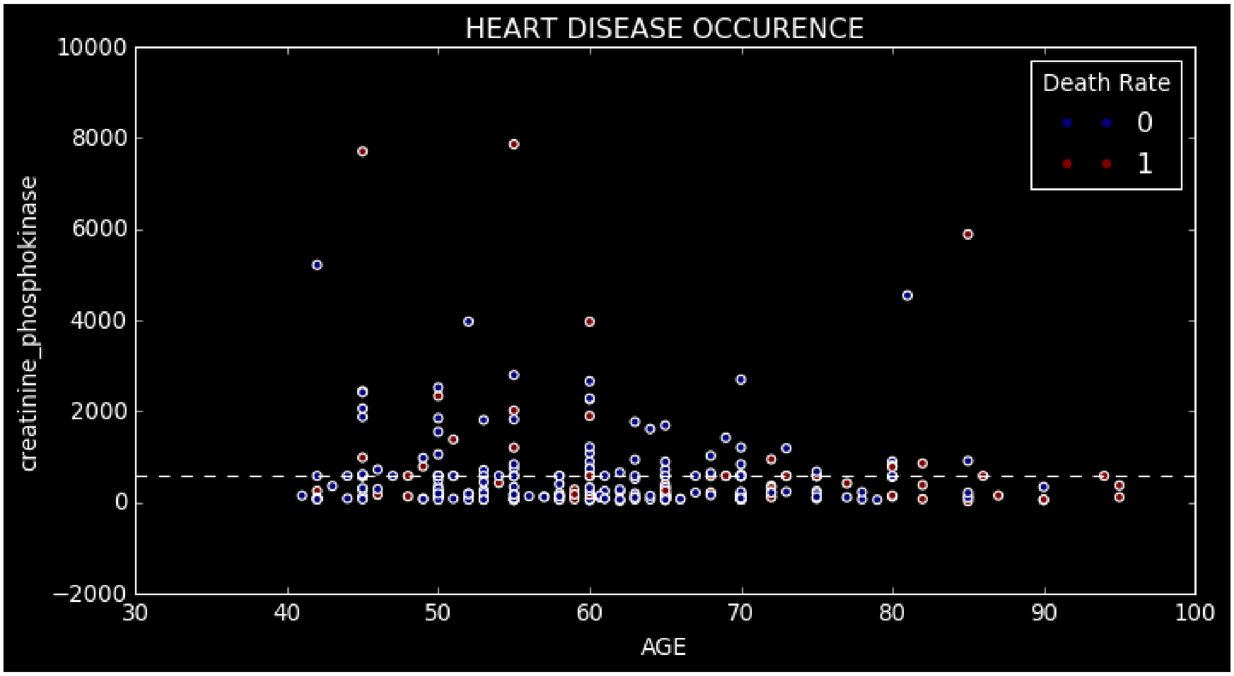
In [161]:
```
#Creating figure
figure, ax=plt.subplots(figsize=(6,4))
#Plotting Data
sub_plot=ax.scatter(x=over_forty['age'],
          y=over_forty['platelets'],
          c= over_forty['DEATH_EVENT'], )
#Customization
ax.set(title = 'Occurence of Heart Disease', xlabel='AGE', ylabel='PLATELETS' );
#setting Legend
ax.legend(*sub_plot.legend_elements(), title='Death Rate'); #must place *.
#Meanline: to show mean value of any column on y-axis.
ax.axhline(y=over_forty['creatinine_phosphokinase'].mean(), linestyle='-', linewi
```



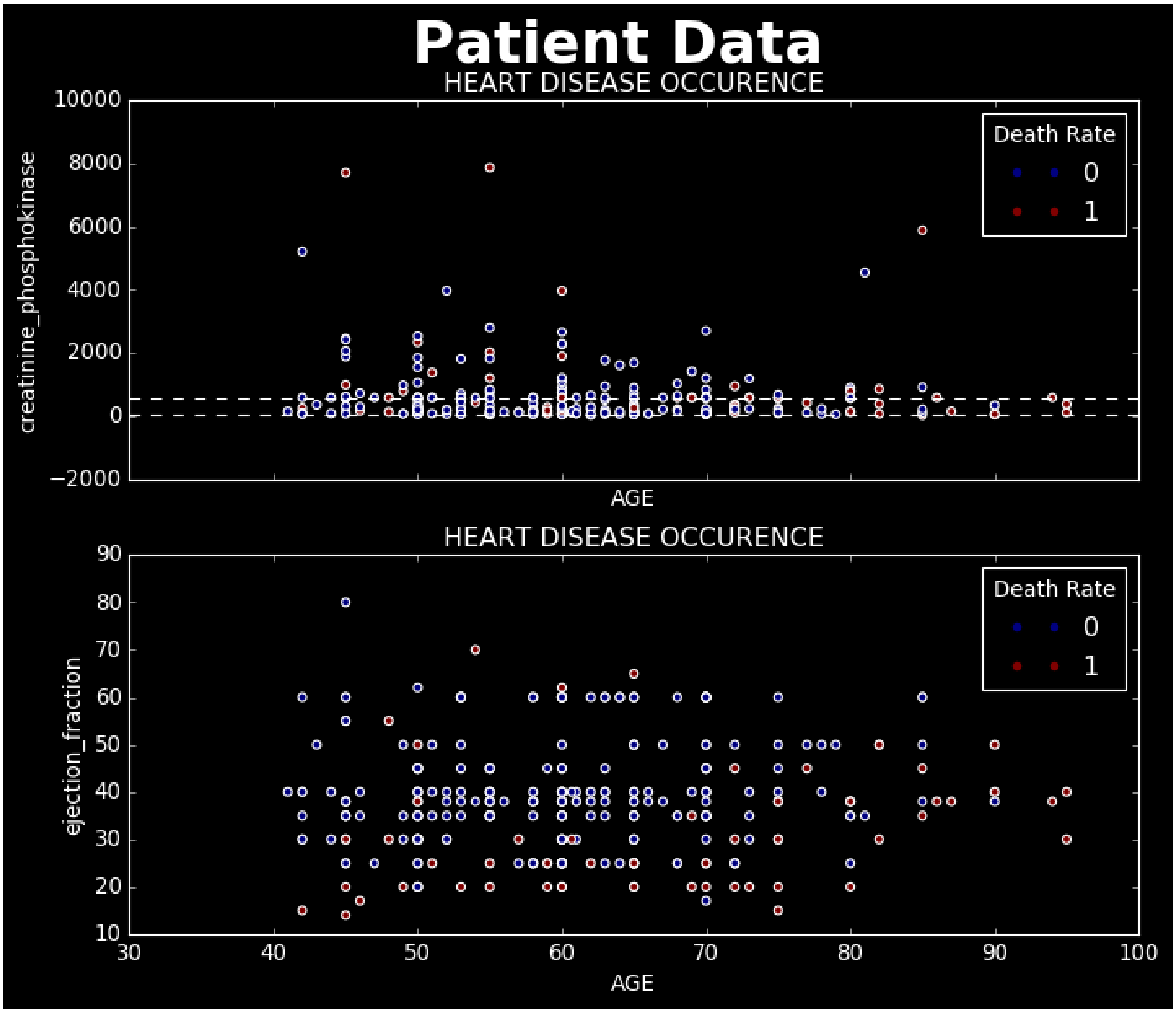
In last line of above code *subplot.legendelement is directing the legend to sub_plot to get given values to c, which is called legend, DEATH- EVENT IS GIVEN.

In [57]:
```
#ejection_fraction  high_blood_pressure platelets   serum_creatinine    serum_soc
```

In [169]:
```
figure, plot1=plt.subplots(figsize=(10,5))
plotting=plot1.scatter(x=over_forty['age'], y=over_forty['creatinine_phosphokinas
plot1.set(title='HEART DISEASE OCCURENCE', xlabel='AGE', ylabel='creatinine_phosp
plot1.legend(*plotting.legend_elements(), title= 'Death Rate' )
plot1.axhline(y = over_forty['creatinine_phosphokinase'].mean(),linestyle='--')
plt.style.use('dark_background')
```



In [185]:
```
#creating plot area
figure, (plot1,plot2)=plt.subplots(nrows=2, ncols=1, figsize=(10,8), sharex=True)

#ADDING DATA TO PLOTS
plotting1=plot1.scatter(x=over_forty['age'], y=over_forty['creatinine_phosphokina
plotting2=plot2.scatter(x=over_forty['age'], y=over_forty['ejection_fraction'], c

#CUSTOMIZATION
plot1.set(title='HEART DISEASE OCCURENCE', xlabel='AGE', ylabel='creatinine_phosp
plot2.set(title='HEART DISEASE OCCURENCE', xlabel='AGE', ylabel='ejection_fractic

#Adding Legend
plot1.legend(*plotting1.legend_elements(), title= 'Death Rate' )
plot2.legend(*plotting2.legend_elements(), title = 'Death Rate')

#Meanline
plot1.axhline(y = over_forty['creatinine_phosphokinase'].mean(),linestyle='--')
plot1.axhline(y = over_forty['ejection_fraction'].mean(),linestyle='--')

#Giving style to fig
plt.style.use('dark_background')

#Naming the fig
figure.suptitle('Patient Data', fontsize=32, fontweight='bold')

#Saving the fig
figure.savefig('patient_data_report.png')
```



In [174]:
```
over_forty
```

Out[174]:

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	p
0	75.0	0	582	0	20	1	26
1	55.0	0	7861	0	38	0	26
2	65.0	0	146	0	20	0	16
3	50.0	1	111	0	20	0	21
4	65.0	1	160	1	20	0	32
...	...	...	...	...	...	...	
294	62.0	0	61	1	38	1	15
295	55.0	0	1820	0	38	0	27
296	45.0	0	2060	1	60	0	74
297	45.0	0	2413	0	38	0	14
298	50.0	0	196	0	45	0	39

292 rows × 13 columns

In [ ]: