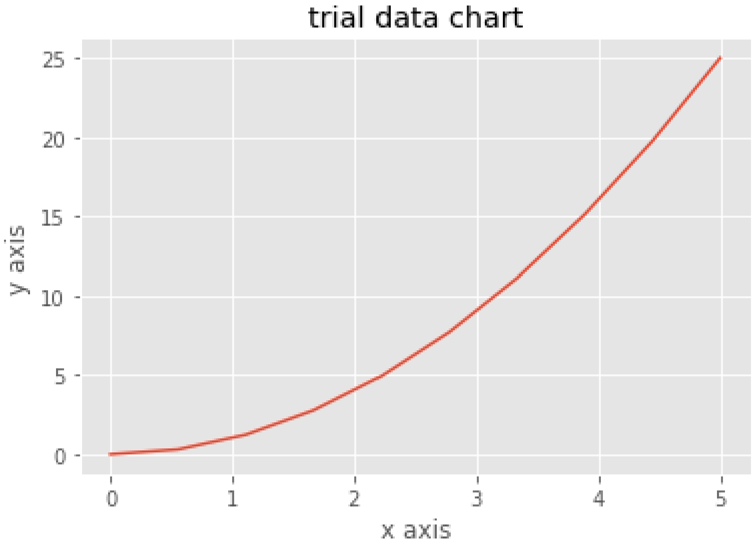
WELCOM TO MATPLOTLIB TUTORIAL

```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
        import numpy as np
        from matplotlib import style
        style.use('ggplot')                           ### diffrent styles of plotting
```

FUNCTIONAL PLOT

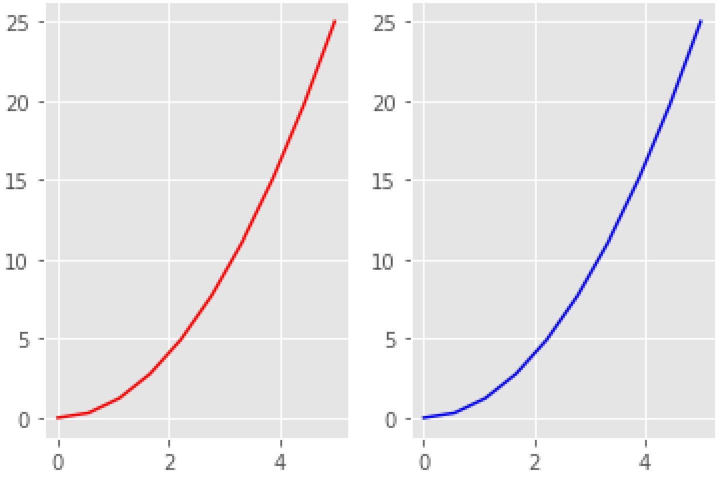
```
In [2]: x1 = np.linspace(0,5,10)                     ### creating data
        y1 = x1**2
        plt.plot(x1,y1)                              ### plotting x and y
        plt.title("trial data chart")                ### giving title                 [TITLE]
        plt.xlabel("x axis")                         ### adding label to x and y      [X,Y,LABEL]
        plt.ylabel("y axis")
        plt.show()                                   ### shows the plot (requierd for other IDEs)
```



MULTIPLE PLOT

In [3]:
```
plt.subplot(1,2,1)
plt.plot(x1,y1,'r')
plt.subplot(1,2,2)                                  ### add colour : type 'color'
plt.plot(x1,y1,'b')
```

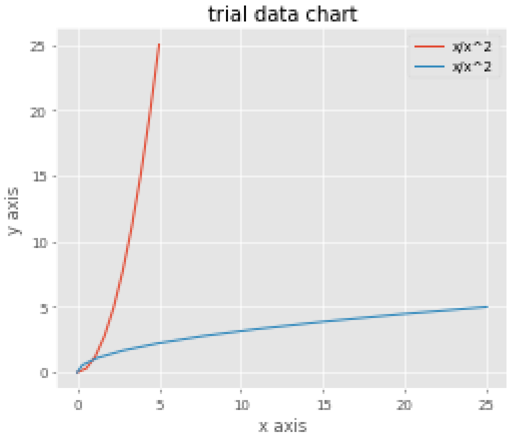
Out[3]:
```
[<matplotlib.lines.Line2D at 0x8d4cac0>]
```



USING FIGURE OBJECTS

```
In [4]: style.use('ggplot')
        fig1 = plt.figure(figsize=(5,4),dpi=50)                    ### plots figure, dpi = size of chart    [FIGURE]
        axes1 = fig1.add_axes([0.1,0.1,0.9,0.9])                    ### adds axes x and y
        axes1.set_xlabel("x axis")
        axes1.set_ylabel("y axis")
        axes1.set_title("trial data chart")

        axes1.plot(x1,y1,label="x/x^2")
        axes1.plot(y1,x1,label="x/x^2")

        axes1.legend(loc=0)                                         ### adds legend ,loc = location           [LEGEND]
```

```
Out[4]: <matplotlib.legend.Legend at 0x8dc1eb0>
```

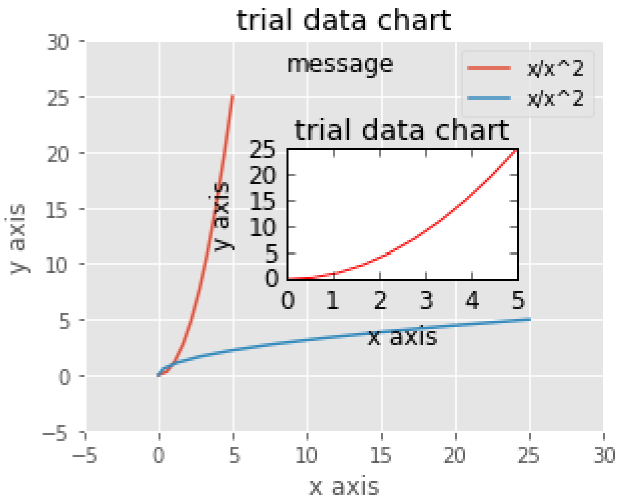


In [5]:

```
style.use('ggplot')
fig1 = plt.figure(figsize=(4,3))               ### plots figure, dpi = size of chart   [FIGURE]
axes1 = fig1.add_axes([0.1,0.1,0.9,0.9])               ### adds axes x and y
axes1.set_xlabel("x axis")
axes1.set_ylabel("y axis")
axes1.set_title("trial data chart")

axes1.plot(x1,y1,label="x/x^2")
axes1.plot(y1,x1,label="x/x^2")

axes1.legend(loc=0)

style.use('classic')                                        ### adding another chart
axes2 = fig1.add_axes([0.45,0.45,0.4,0.3])
axes2.set_xlabel("x axis")
axes2.set_ylabel("y axis")
axes2.set_title("trial data chart")
axes2.plot(x1,y1,'r')

axes2.text(0,40,'message')                                 ### type a msg in chart     [TEXT]
```

Out[5]:

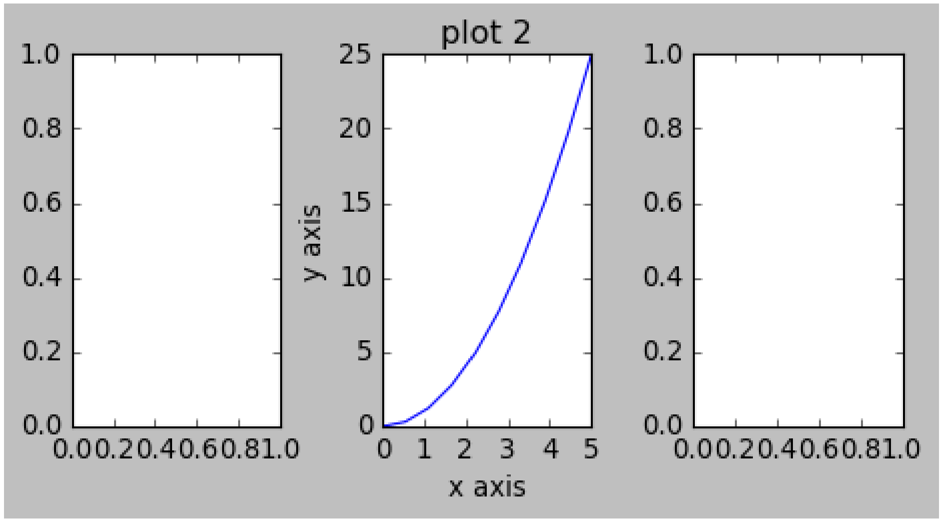
```
Text(0, 40, 'message')
```



SUBPLOTS

```
In [6]: fig2,axes2 = plt.subplots(figsize=(6,3),nrows=1,ncols=3)     ### plotting subplots    [SUBPLOTS]
        plt.tight_layout()                                           ### gives space betn subplots    [TIGHT_LAYOUT]
        axes2[1].set_title('plot 2')
        axes2[1].set_xlabel('x axis')
        axes2[1].set_ylabel('y axis')
        axes2[1].plot(x1,y1)
```

```
Out[6]: [<matplotlib.lines.Line2D at 0x8f85f10>]
```



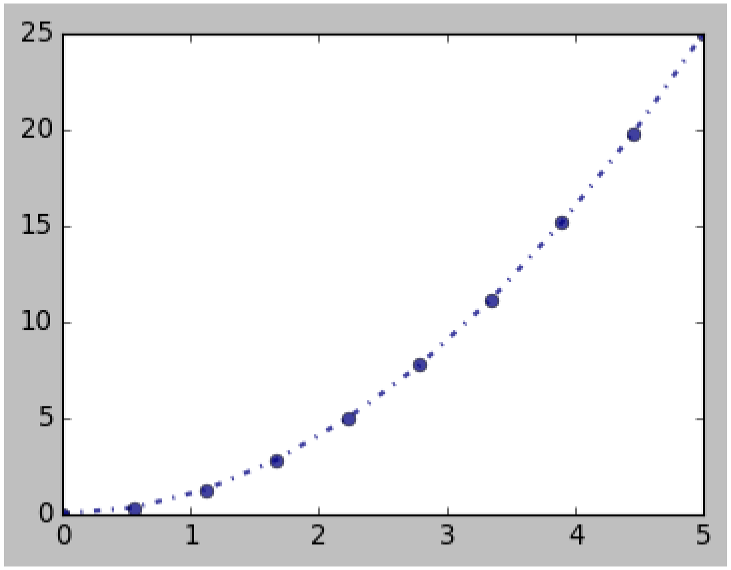
APPEARANCE OPTION

In [7]:

```
fig3 = plt.figure(figsize=(4,3))
axes3 = fig3.add_axes([0,0,1,1])
axes3.plot(x1,y1,color='navy',alpha=0.75,lw=2,ls='-.',marker='o')

### color = color  ,aplha = ? ,  lw = line width , ls = line style   marker = point where y and x intercect
```

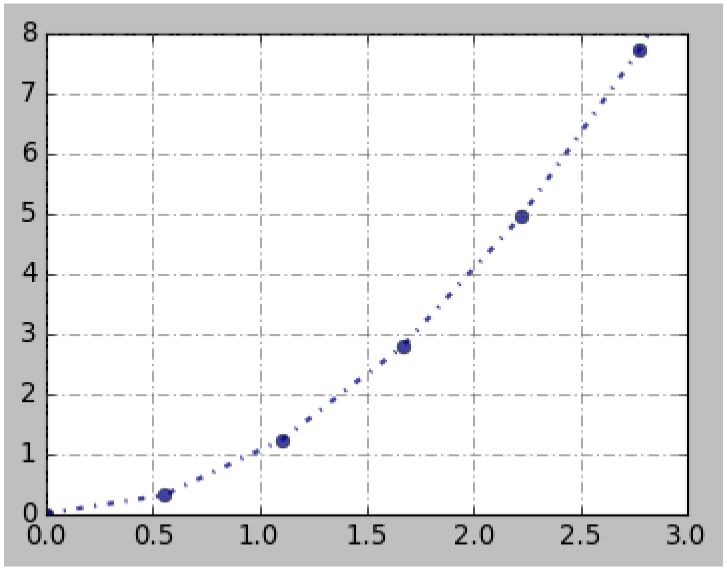
Out[7]:

```
[<matplotlib.lines.Line2D at 0x8e4f970>]
```



In [8]:

```
fig3 = plt.figure(figsize=(4,3))
axes3 = fig3.add_axes([0,0,1,1])
axes3.plot(x1,y1,color='navy',alpha=0.75,lw=2,ls='-.',marker='o')

axes3.set_xlim([0,3])                                     ### x limit for zooming in chart     [XLIM]
axes3.set_ylim([0,8])                                     ### y limit for zooming in chart     [YLIM]
axes3.grid(True,color='0.4',dashes=(5,2,1,2))
### adding background dashes in chart : color= ...% black , dashes = spaces
```



SAVING VISUALIZAION TO A FILE

In [9]:
```
fig3.savefig('3rd_plot.pdf')            ### plot has been saved
```

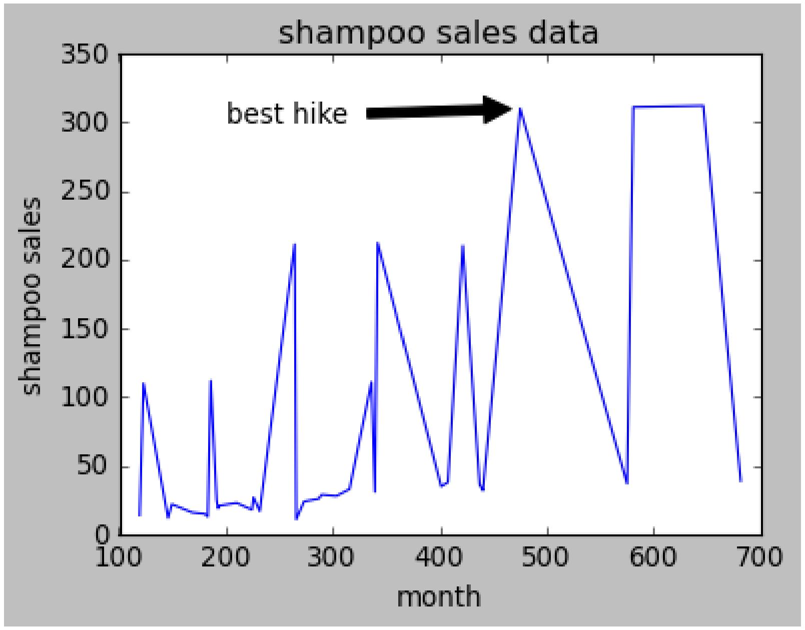
WORKING WITH PANDAS DATAFRAME

In [10]:
```
new_df = pd.read_csv(r"D:\harsh work\data_science\datasets\shampoo_1.csv")
new_df = new_df.sort_values(by='Sales')         ### sort data set by incresing order
new_df.head(10)
```

Out[10]:
```
    month  Sales
 3     14  119.3
 9    110  122.9
 1     12  145.9
13     22  149.5
 5     16  168.5
 4     15  180.3
 2     13  183.1
11    112  185.9
16     25  191.4
 8     19  192.8
```

```
In [11]: np_arr = new_df.values                                     ### conert data into array
         x2 = np_arr[:,0]                                          ### define dataset
         y2 = np_arr[:,1]
         fig4 = plt.figure(figsize=(4,3))
         axes4 = fig4.add_axes([0,0,1,1])

         axes4.set_xlabel("month")
         axes4.set_ylabel("shampoo sales")
         axes4.set_title("shampoo sales data")
         axes4.plot(y2,x2)

         axes4.annotate('best hike',xy=(480,310),xytext=(200,300),
                       arrowprops=dict(facecolor='black',shrink=0.08))
         ### "" : text , xy = coridnates of point , xytext = cordinates of text , arrowprops = color,shrink
```

```
Out[11]: Text(200, 300, 'best hike')
```



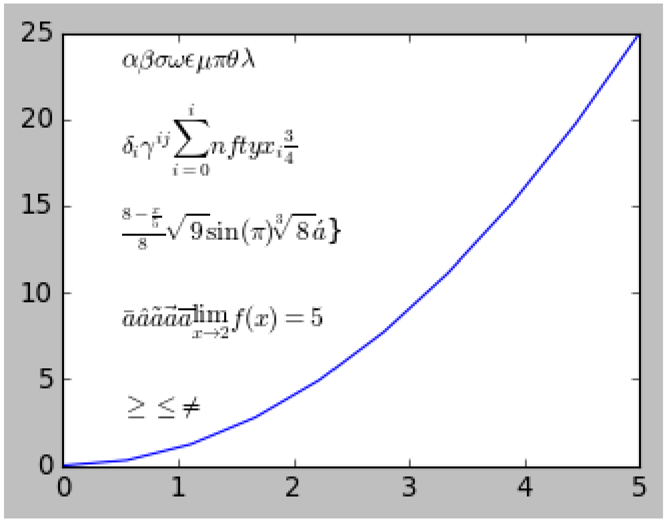
TEX MARKUP

In [12]:

```
fig5 = plt.figure(figsize=(4,3))
axes5 = fig5.add_axes([0.1,0.1,0.9,0.9])

axes5.text(0.5,23,              ### text location
           r'$\alpha \beta \sigma \omega \epsilon \mu \pi \theta \lambda$')
                                ### ading symbols in chart = r'%\(symbol_name)'         [1]
axes5.text(0.5,18,
           r'$\delta_i \gamma^{ij} \sum_{i=0}^infty x_i \frac{3}{4}$')
                                ### {} brackets for value                               [2]
axes5.text(0.5,13,
           r'$\frac{8 - \frac{x}{5}}{8} \sqrt{9} \sin(\pi) \sqrt[3]{8}\acute a$}')
                               ### frac = fraction , sqrt = square root                 [3]
axes5.text(0.5,8,
           r'$\bar a \hat a \tilde a \vec a \overline {a} \lim_{x\to 2} f(x)=5$')
                               ### x\to 2 = x tends to 2                                [4]
axes5.text(0.5,3,
           r'$\geq \leq \ne$')
                               ### geq = greter or equal , leq = less or equal , ne = not equal  [5]
axes5.plot(x1,y1)
```

Out[12]:

```
[<matplotlib.lines.Line2D at 0x9159940>]
```



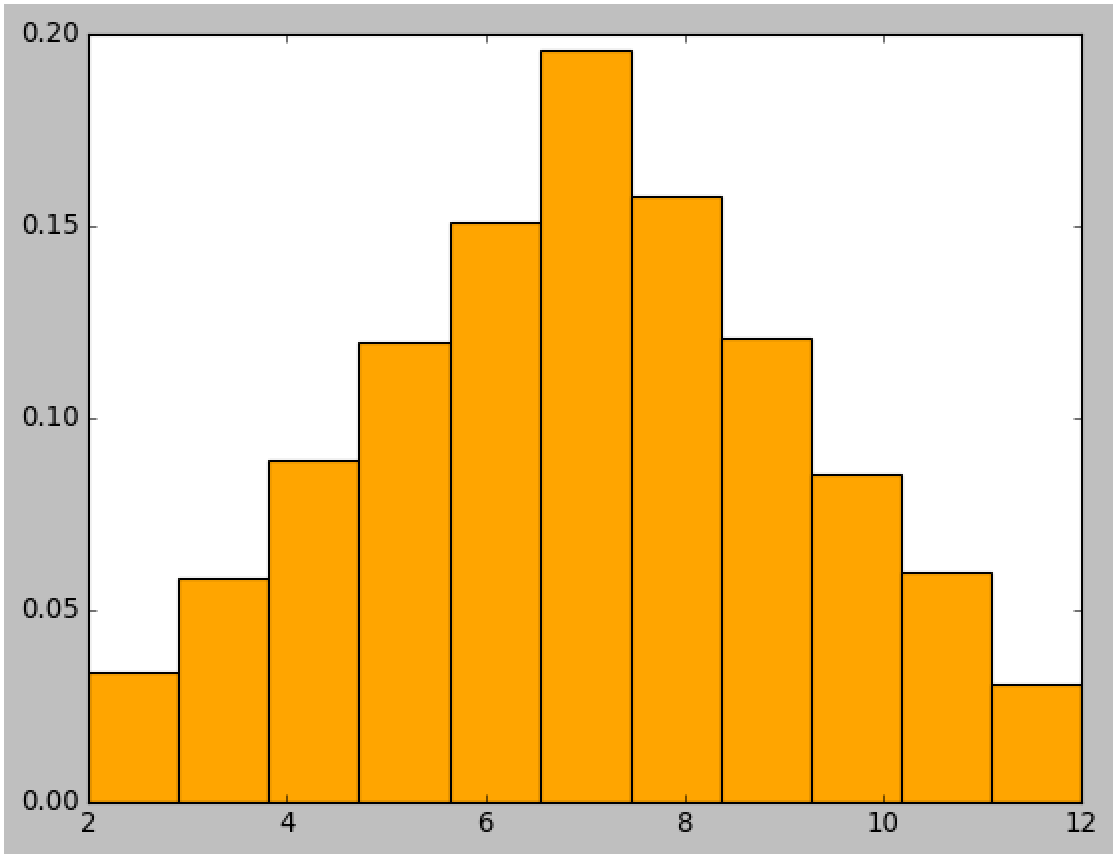
HISTOGRAMS

In [13]:

```
arr1 = np.random.randint(1,7,7000)
arr2 = np.random.randint(1,7,7000)
arr3 = arr1 + arr2

plt.hist(arr3,bins=11,density=True,stacked=True,color='orange')          ### plotting a histogram   [HIST]
#   bins = number of bars , cumilative - TRUE = cumulative distrubution , orientioan = horizontal
```

Out[13]:

```
(array([0.03378571, 0.05798571, 0.08894286, 0.11942857, 0.15101429,
        0.19548571, 0.15777143, 0.12037143, 0.08517143, 0.05971429,
        0.03032857]),
 array([ 2.        ,  2.90909091,  3.81818182,  4.72727273,  5.63636364,
         6.54545455,  7.45454545,  8.36363636,  9.27272727, 10.18181818,
        11.09090909, 12.        ]),
 <a list of 11 Patch objects>)
```



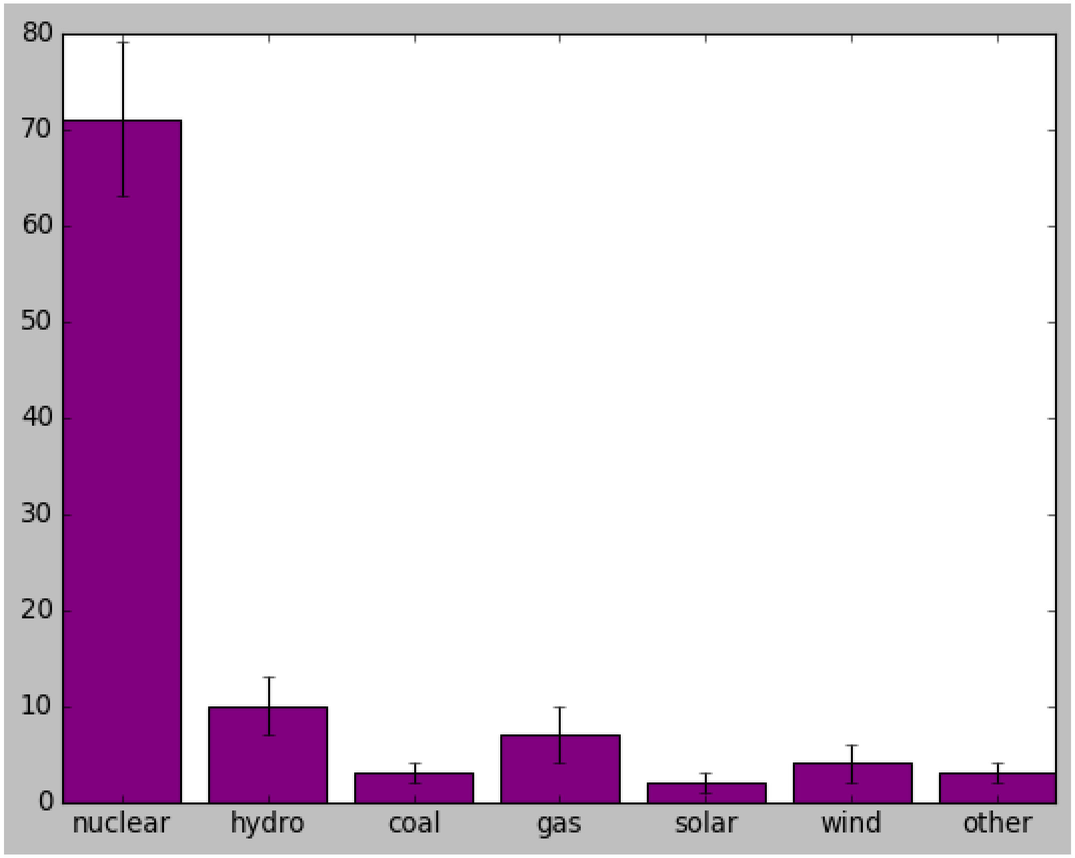
BAR CHART

In [14]:

```
x = ['nuclear','hydro','coal','gas','solar','wind','other']
per_1 = [71,10,3,7,2,4,3]
verriance = [8,3,1,3,1,2,1]                           ### plot a bar chart        [BAR]
plt.bar(x,per_1,yerr=verriance,color='purple')        ### yerr = show line of verriance
```

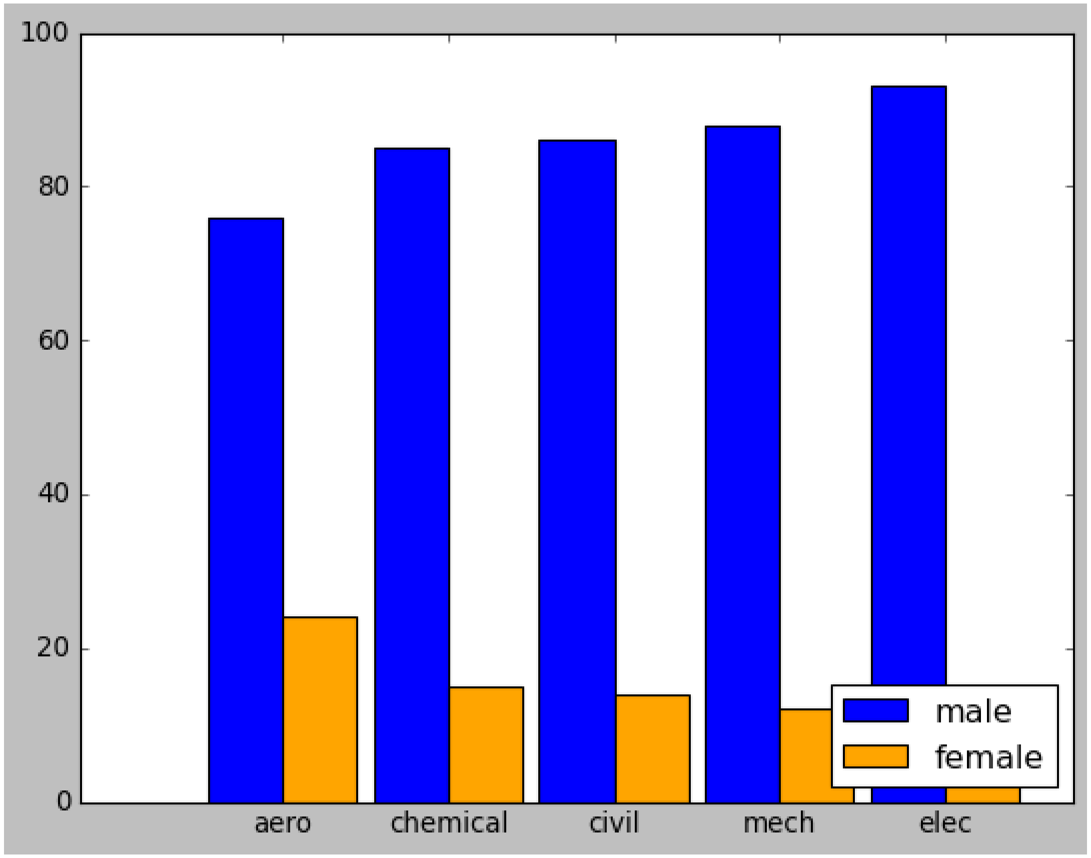
Out[14]:

```
<BarContainer object of 7 artists>
```



```
In [15]: m_eng = (76,85,86,88,93)
         f_eng = (24,15,14,12,7)
         spc = np.arange(5)
         plt.bar(spc,m_eng,width=0.45,label='male',edgecolor='k')
         plt.bar(spc+0.45,f_eng,width=0.45,label='female',edgecolor='k',color='orange')
         plt.xticks(spc + 0.45/2 , ('aero','chemical','civil','mech','elec',))
         plt.legend(loc='lower right')
```

```
Out[15]: <matplotlib.legend.Legend at 0x90c9d30>
```

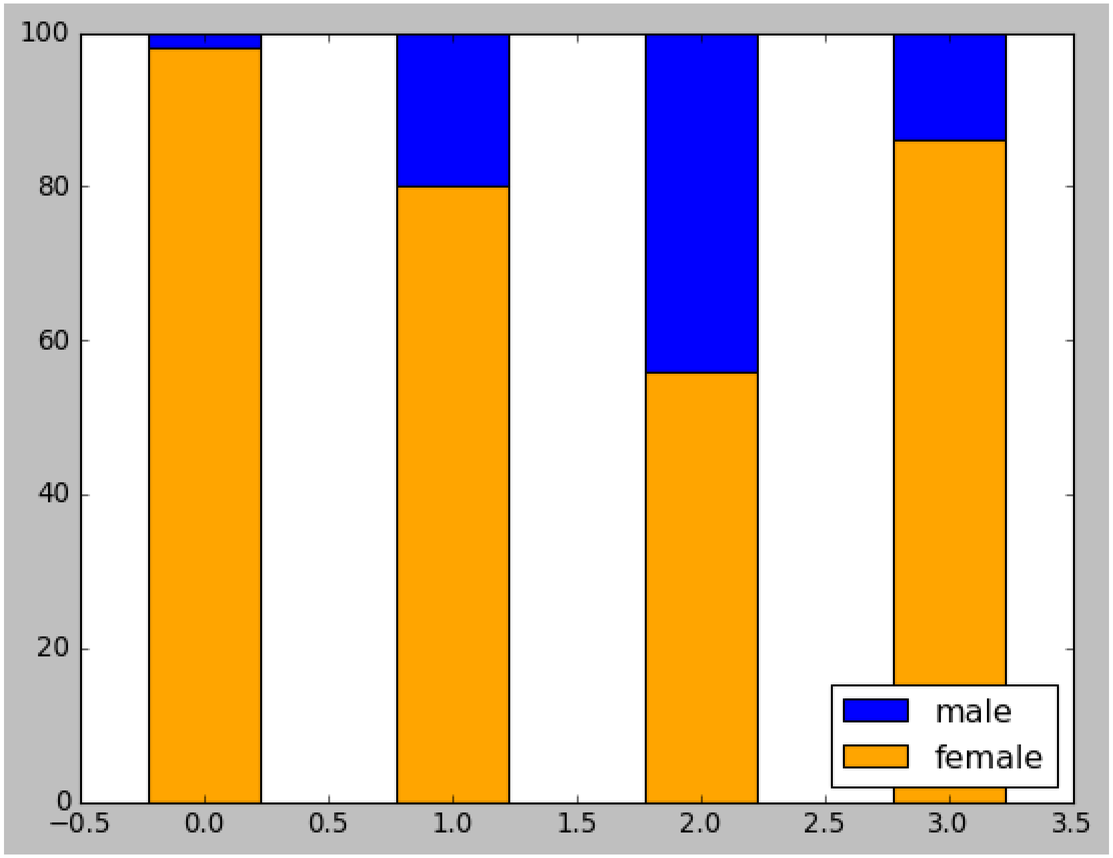


In [16]:

```
t_type = ['kind','elem','sec','spec']
m_tech = np.array([2,20,44,14])
f_tech = np.array([98,80,56,86])
ind = [x for x , _ in enumerate(t_type)]                       ### giving an index
plt.bar(ind,m_tech,width=0.45,label='male',bottom=f_tech)   ### bottom = f_tech will be at bottom of graph
plt.bar(ind,f_tech,width=0.45,label='female',color='orange')
plt.legend(loc='lower right')
```

Out[16]:

```
<matplotlib.legend.Legend at 0x9229df0>
```



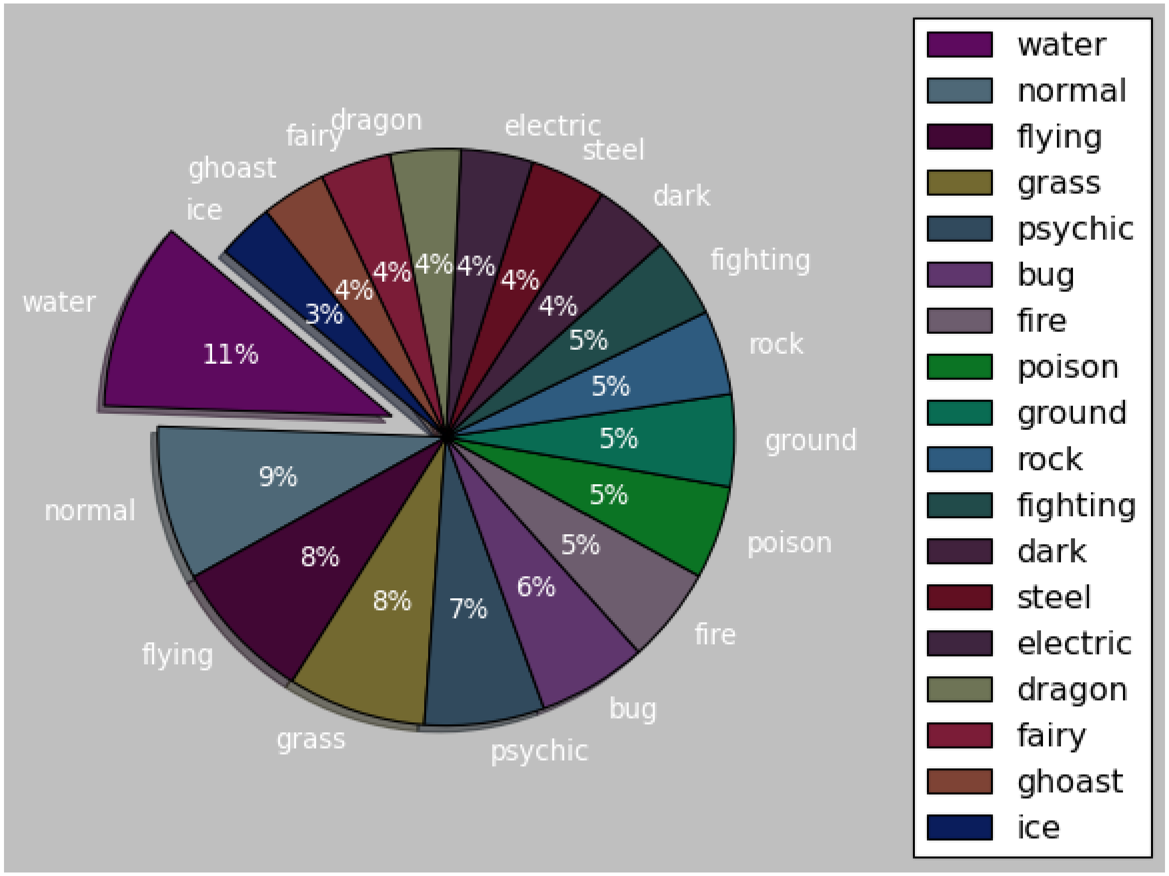
PIE CHART

In [17]:

```
import random
fig6 = plt.figure(figsize=(8,5))
axes6 = fig6.add_axes([0.1,0.1,0.9,0.9])
types = ['water','normal','flying','grass','psychic','bug','fire','poison',
        'ground','rock','fighting','dark','steel','electric','dragon','fairy',
         'ghoast','ice']
poke_num = [133,109,101,98,85,77,68,66,65,60,57,54,53,51,50,50,46,40]
### creating data of types of pokemon and their numbers
colors = []
for i in range(18):                                    ### adding 18 colors
    rgb = random.uniform(0,.5),random.uniform(0,.5),random.uniform(0,.5)
    colors.append(rgb)

explode = [0] * 18
explode[0] = 0.2
wedges , texts , autotexts = plt.pie(poke_num,explode=explode,labels=types,colors=colors,
                                   autopct='%1.0f%%',shadow=True,startangle=140,textprops=dict(color="w"))
plt.legend(wedges,types,loc='right',bbox_to_anchor=(1,0,0.5,1))
```

Out[17]:

```
<matplotlib.legend.Legend at 0xa396490>
```


water
11%
normal
9%
flying
8%
grass
8%
psychic
7%
bug
6%
fire
5%
poison
5%
ground
5%
rock
5%
fighting
5%
dark
4%
steel
4%
electric
4%
dragon
4%
fairy
4%
ghoast
4%
ice
3%
water
normal
flying
grass
psychic
bug
fire
poison
ground
rock
fighting
dark
steel
electric
dragon
fairy
ghoast
ice

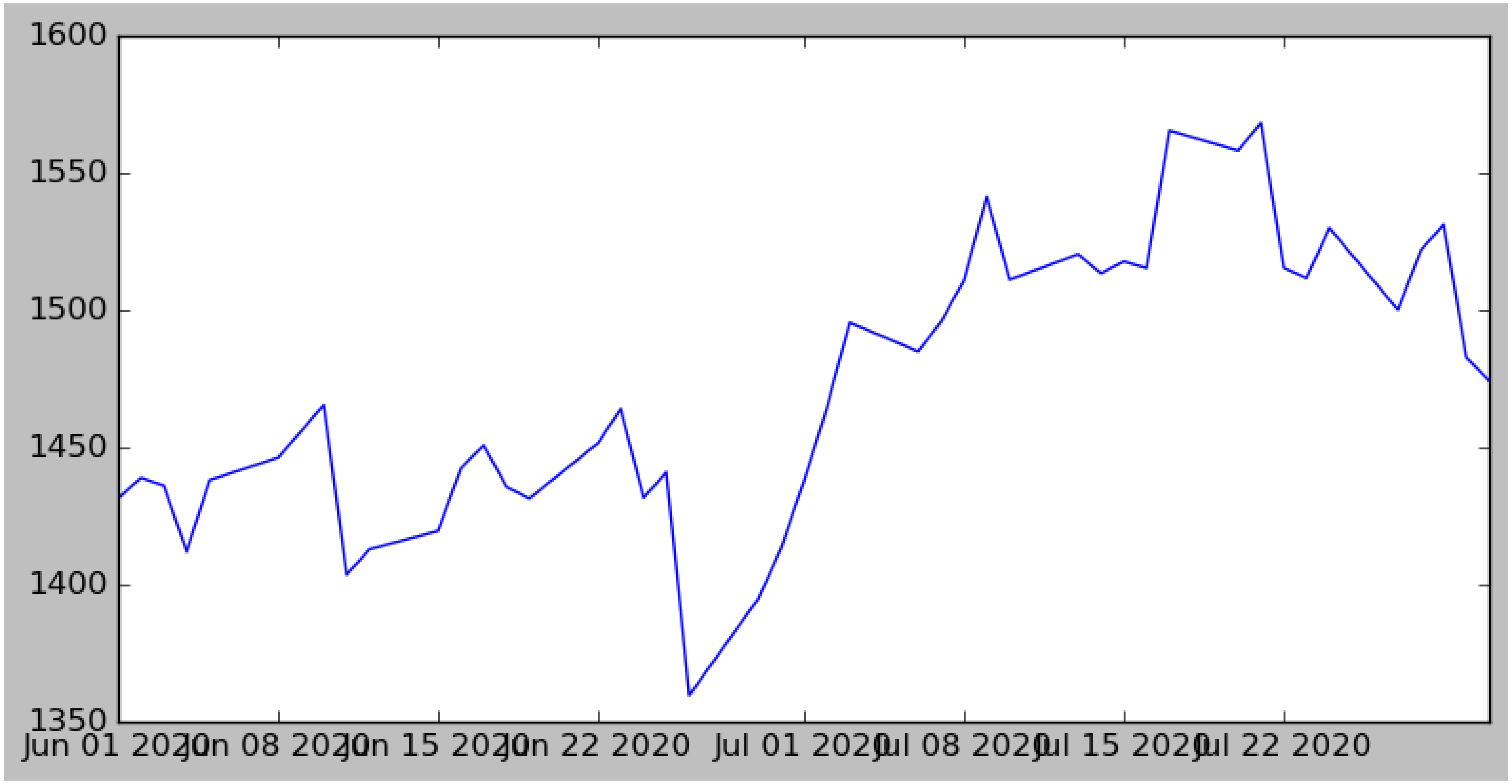
TIME SERIES

In [18]:

```
import datetime
goog_data = pd.read_csv('D:\harsh work\data_science\datasets\GOOG (2).csv')
goog_data_np = goog_data.to_numpy()
goog_cp = goog_data_np[:,4]                   # Get array of prices in 5th column
goog_cp

date_arr = pd.bdate_range(start='06/01/2020', end='08/02/2020',
                          freq='C')
### defining rage for dataser              [BDATE_RANGE]

date_arr_np = date_arr.to_numpy()

fig_7 = plt.figure(figsize=(8,4),dpi=100)
axes_7 = fig_7.add_axes([0.1,0.1,0.9,0.9])
plt.plot(date_arr_np, goog_cp)
```

Out[18]:

```
[<matplotlib.lines.Line2D at 0xa47ceb0>]
```



In [19]:
```
goog_data.tail()
```

Out[19]:

	Date	Open	High	Low	Close	Adj Close	Volume
40	28/07/2020	1525.180054	1526.479980	1497.660034	1500.339966	1500.339966	1702200
41	29/07/2020	1506.319946	1531.251953	1501.329956	1522.020020	1522.020020	1106500
42	30/07/2020	1497.000000	1537.869995	1492.219971	1531.449951	1531.449951	1671400
43	31/07/2020	1505.010010	1508.949951	1454.030029	1482.959961	1482.959961	3439900
44	03/08/2020	1486.640015	1490.469971	1465.640015	1474.449951	1474.449951	2330200

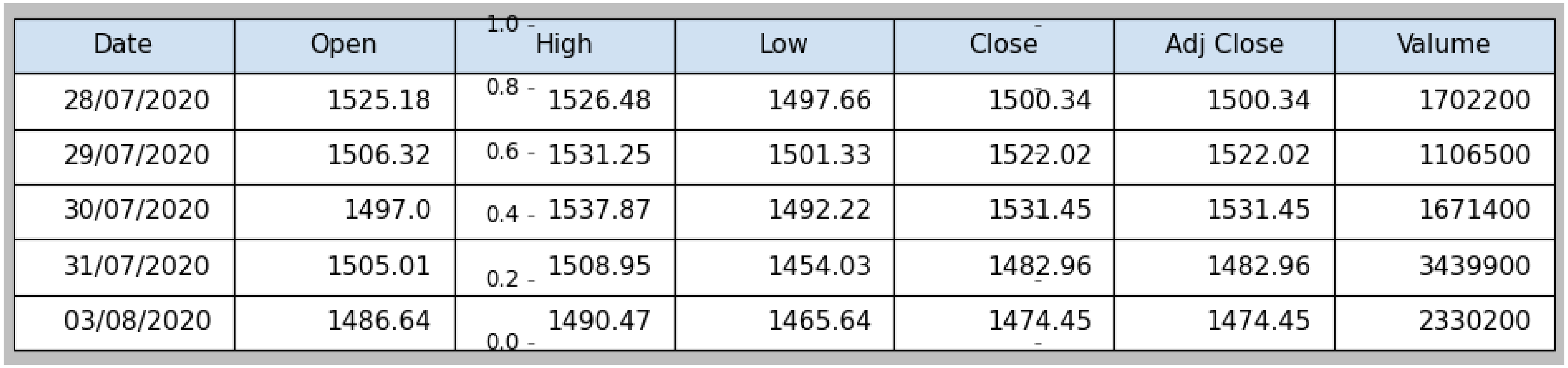
In [ ]:
```

```

TABLE

In [20]:

```
goog_data['Open'] = pd.Series([round(val,2)for val in goog_data['Open']],
                              index = goog_data.index)
goog_data['High'] = pd.Series([round(val,2)for val in goog_data['High']],
                              index = goog_data.index)
goog_data['Low'] = pd.Series([round(val,2)for val in goog_data['Low']],
                              index = goog_data.index)
goog_data['Close'] = pd.Series([round(val,2)for val in goog_data['Close']],
                              index = goog_data.index)
goog_data['Adj Close'] = pd.Series([round(val,2)for val in goog_data['Adj Close']],
                              index = goog_data.index)
### PLOTTING FIVE COLOUMNS FOR TABLE

stk_data = goog_data[-5:]                           ### ONLY TAKE LAST FIVE ROWS
col_head = ('Date','Open','High','Low','Close','Adj Close','Valume')    ### SETTING HEADERS
stk_data_np = stk_data.to_numpy()
plt.figure(linewidth=2,tight_layout={'pad':0.5},figsize=(5,3))
### PLOTTING TABLE

axes8 = plt.gca()
axes8.get_xaxis().set_visible(False)
axes8.get_xaxis().set_visible(False)
plt.box(on=None)                                    ### HIDING A BOX

colors = plt.cm.Blues(np.full(len(col_head),0.2))         ### COLORING THE HEADERS
the_table = plt.table(cellText=stk_data_np,loc='center',colLabels=col_head,
                      colColours=colors)            ### PLOTTING A TABLE      [TABLE]
### CELLtEXT = CONTENT OF TABLE , COLLABELS = PLOTTING HEADERS , COLcLOURS = PLOTTNG HEADERS COLOR

the_table.set_fontsize(14)                     ### FITTING THE FONT SIZE     [SET_FONTSIZE]
the_table.scale(3,2.5)                         ### SETTING THE SIZE OF TABLE      [SCALE]
```

```
D:\anaconda\envs\tf\lib\site-packages\IPython\core\pylabtools.py:132: UserWarning: Tight layout not applied. T
he left and right margins cannot be made large enough to accommodate all axes decorations.
  fig.canvas.print_figure(bytes_io, **kw)
```



Date	Open	High	Low	Close	Adj Close	Valume
28/07/2020	1525.18	1526.48	1497.66	1500.34	1500.34	1702200
29/07/2020	1506.32	1531.25	1501.33	1522.02	1522.02	1106500
30/07/2020	1497.0	1537.87	1492.22	1531.45	1531.45	1671400
31/07/2020	1505.01	1508.95	1454.03	1482.96	1482.96	3439900
03/08/2020	1486.64	1490.47	1465.64	1474.45	1474.45	2330200

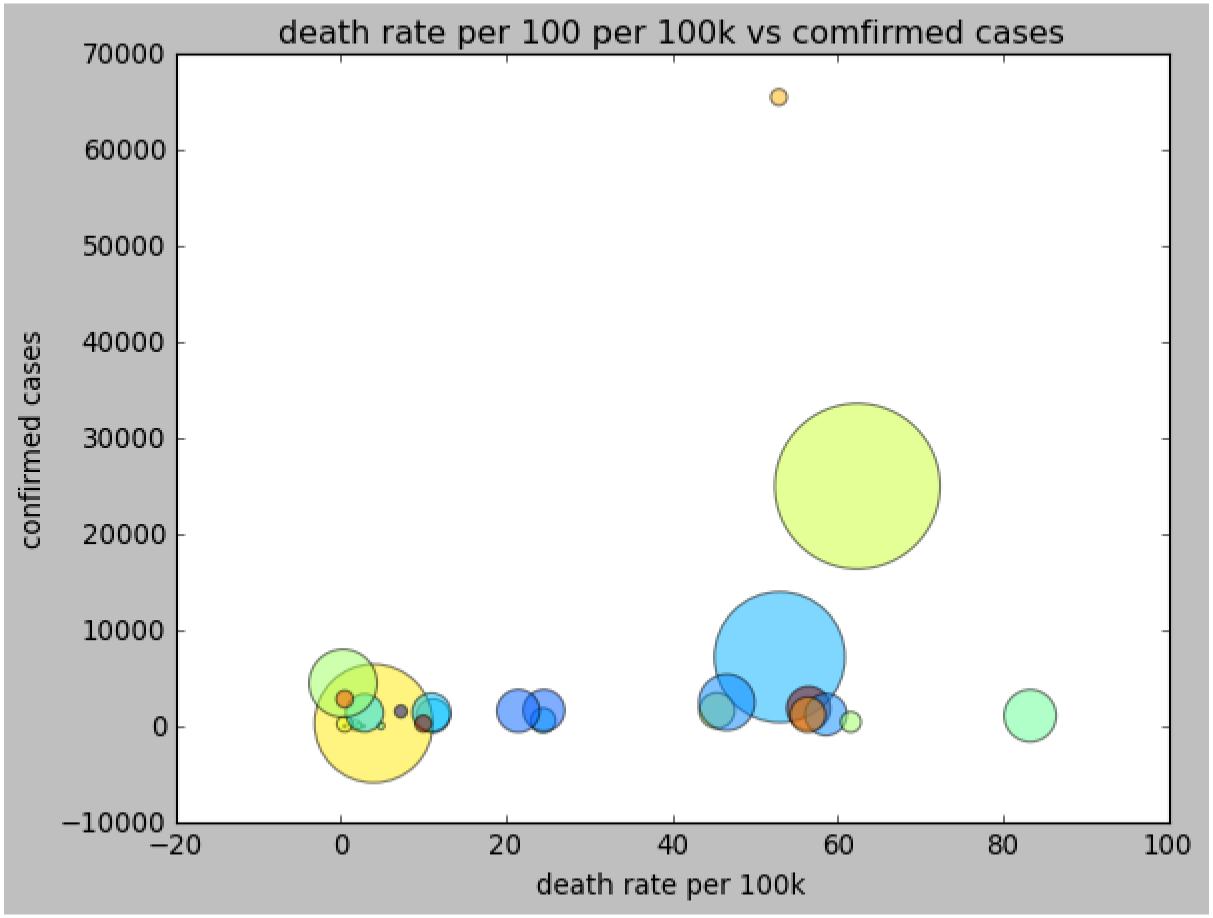
1.0
0.8
0.6
0.4
0.2
0.0

SCATTER PLOT

In [21]:

```
#country
cnt_arr = np.array(['austrelia','brazil','canada','chile','framce','germany','greec','iceland','india',
                    'iran','italy','mexico','nw zeland','nigeria','norway','pakistan','peru','russia',
                   'saudi areba','singapore','south africa','spain','sweden','turky','UK','US'])

#death rate per 100k coronavieus
dr_arr = np.array([1.8,53,24.5,56.5,45.4,11.2,2.2,2.8,4,24.6,58.6,
                  46.6,.5,.5,4.9,2.9,83.3,.3,11,10,.5,21.5,61.6,
                  56.4,7.3,62.4,52.9])

#daily confirmd cases
test_arr = np.array([110,7197,600,1862,1636,1103,35,10,295,1658,
                    1226,2490,8,243,48,1395,1101,4474,1443,280,
                    2830,1602,447,1205,1546,24988,65465])

#dot size confirmed cases
cc_arr = np.array([24236,3456652,125408,390037,256534,229706,7684,
                  2035,2836525,350279,355378,637011,1645,50488,
                  10162,290445,549321,935066,302656,56031,
                  59606,370867,85411,253108,32008,5529824,54654])
```

```
In [22]: cc_arr_sm = cc_arr / 1000
         color_arr = np.random.rand(27)                                 ### creating 27 random colors
         plt.title('death rate per 100 per 100k vs comfirmed cases')
         plt.xlabel('death rate per 100k')
         plt.ylabel('confirmed cases')
         plt.scatter(dr_arr,test_arr,s=cc_arr_sm,c=color_arr,alpha=0.5)      ### pltting scatter    [SCATTER]
         ###   s = dot size , c = color of dots , alpha = shade of color

Out[22]: <matplotlib.collections.PathCollection at 0xa56e070>
```



3D PLOTS

In [23]:

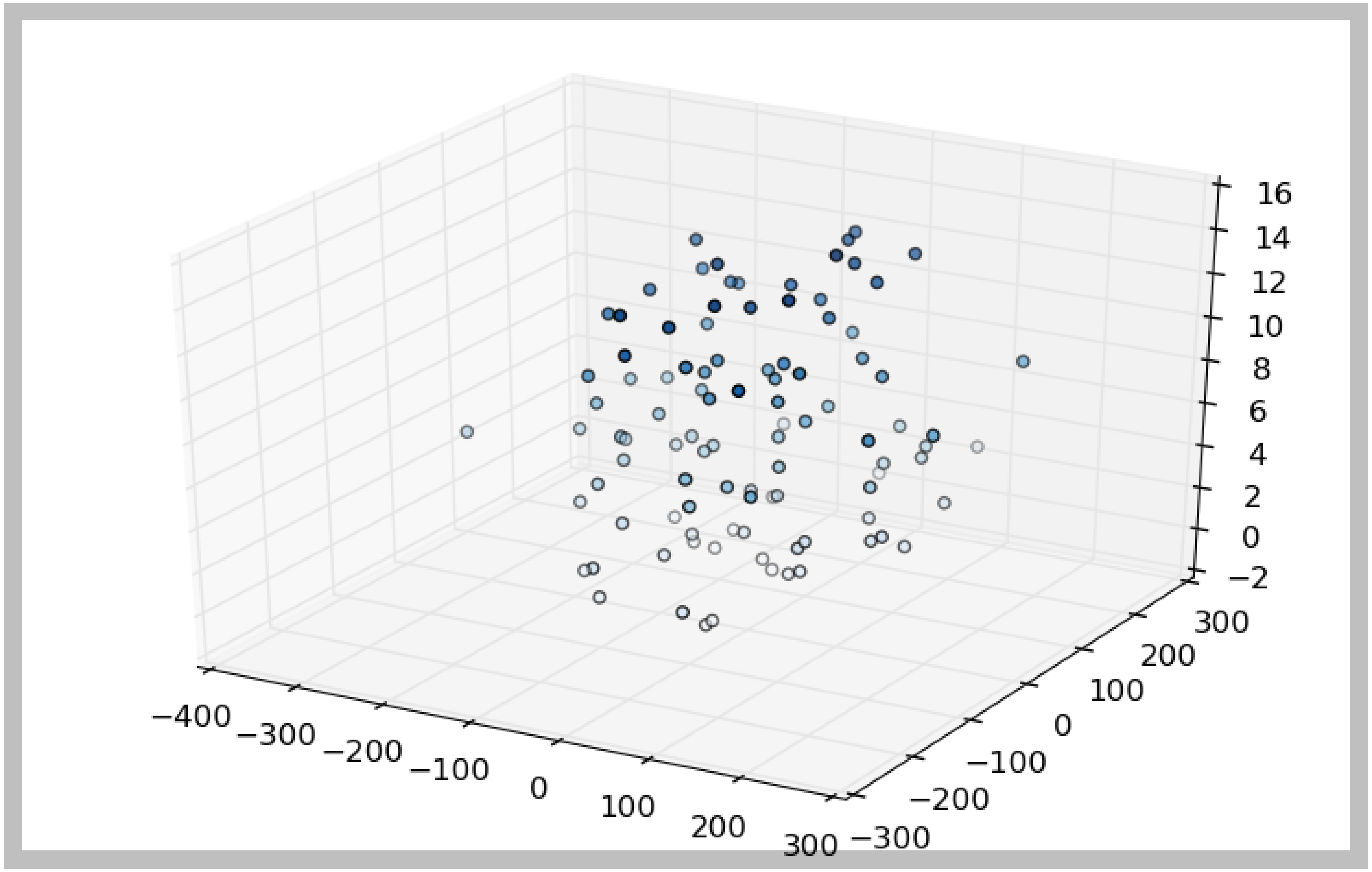
```
from mpl_toolkits import mplot3d                             ### importing mplot3d
```

In [24]:

```
fig9 = plt.figure(figsize=(8,5),dpi=100)
axes9 = fig9.add_axes([0.1,0.1,0.9,0.9],projection='3d')          ### projection = 3d

z_3 = 15 * np.random.random(100)
x_3 = np.size(z_3) * np.random.randn(100)                          ### creatig dataset
y_3 = np.size(z_3) * np.random.randn(100)
axes9.scatter3D(x_3,y_3,z_3,c=z_3,cmap='Blues')                    ### plotting 3d plot       [SCATTER3D]
```

Out[24]:

```
<mpl_toolkits.mplot3d.art3d.Path3DCollection at 0xa7dda30>
```



FINANCE

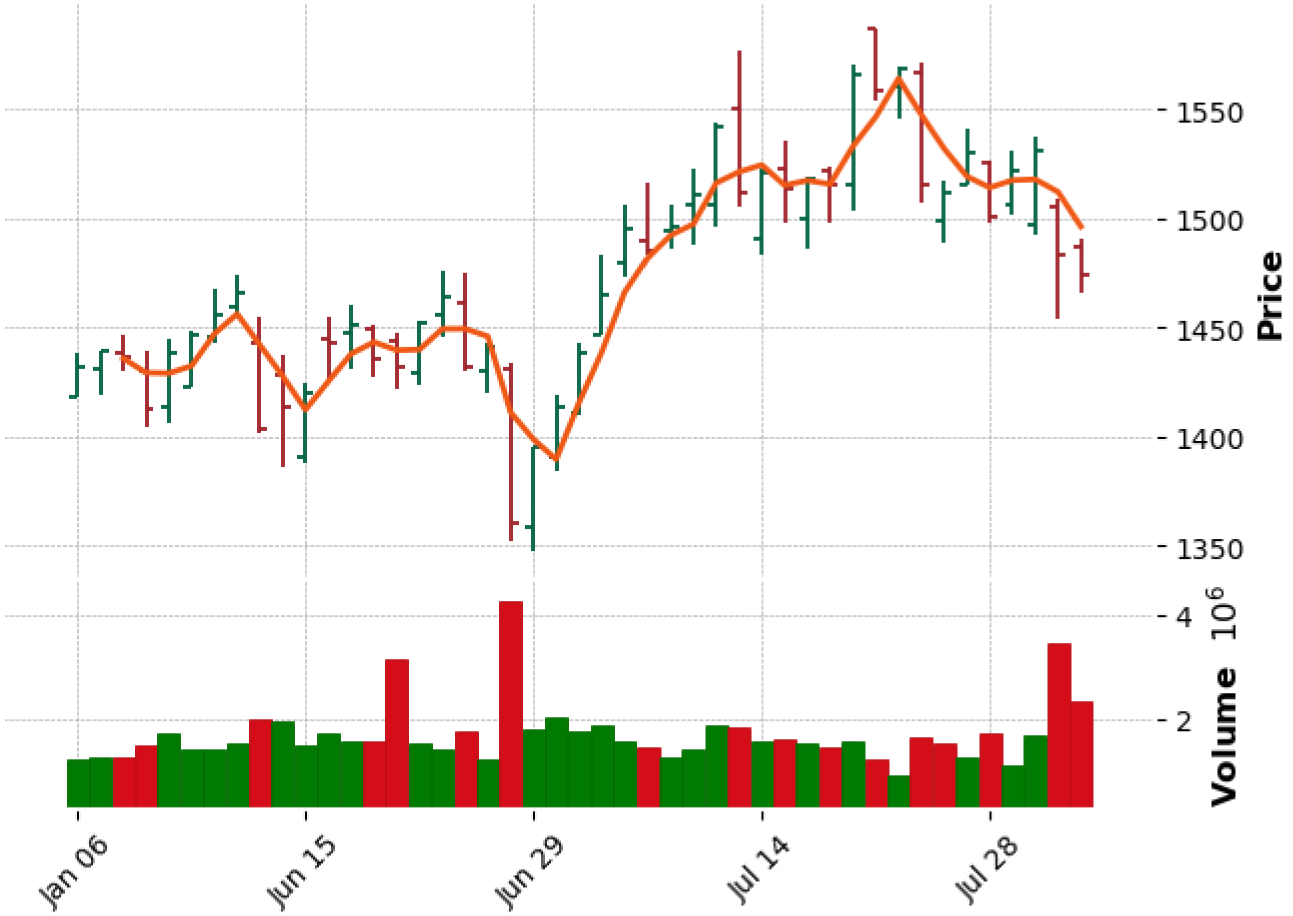
In [25]:

```
import mplfinance as mpf                    ### importng mph

goog_df = pd.read_csv("D:\harsh work\data_science\datasets\GOOG (2).csv",index_col=0,parse_dates=True)
goog_df.head(2)
```

Out[25]:

	Open	High	Low	Close	Adj Close	Volume
Date						
2020-01-06	1418.390015	1437.959961	1418.000000	1431.819946	1431.819946	1217100
2020-02-06	1430.550049	1439.609985	1418.829956	1439.219971	1439.219971	1278100

```
In [26]: goog_df.index.name = 'Date'                                  ### index name is date
         mpf.plot(goog_df,type='ohlc',style='charles',mav=(3,),volume=True)      ### pltting mph     [MPH.PLOT]
         #type =  (candel ,ohlc,line) ,style = (mike,charles,binance,blueskies,nightclous), mav = moving avreage ,
         #volume = volumed graph
```



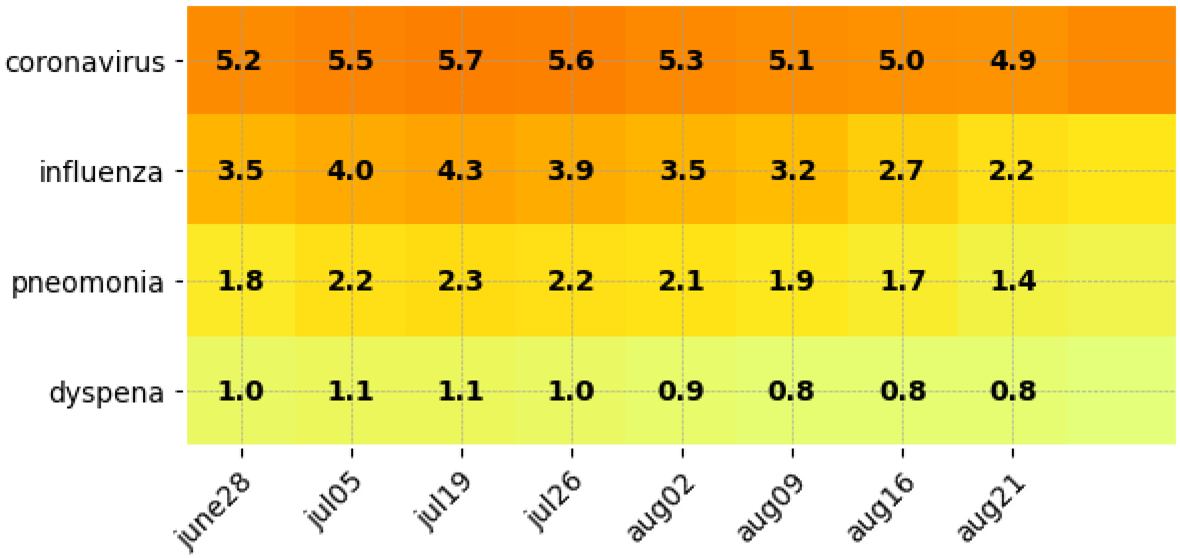
HEATMAPS

In [27]:

```
symptoms = ['coronavirus','influenza','pneomonia','dyspena']
dates = ['june28','jul05','jul19','jul26','aug02','aug09','aug16','aug21']
sympe_per = np.array([[5.2,5.5,5.7,5.6,5.3,5.1,5.0,4.9,5.3],
                     [3.5,4.0,4.3,3.9,3.5,3.2,2.7,2.2,2.0],
                     [1.8,2.2,2.3,2.2,2.1,1.9,1.7,1.4,1.3],
                     [1.0,1.1,1.1,1.0,0.9,0.8,0.8,0.8,0.7]])          ### creating data set
```

In [28]:

```
fig10,axes10 = plt.subplots()                               ### plot a subplot
im = axes10.imshow(sympe_per,cmap='Wistia')                 ###
axes10.set_xticks(np.arange(len(dates)))                    ### makes axes and gives name
axes10.set_yticks(np.arange(len(symptoms)))
axes10.set_xticklabels(dates)
axes10.set_yticklabels(symptoms)

plt.setp(axes10.get_xticklabels(),rotation=45,ha="right",rotation_mode="anchor")
### rotate the name to 45 degrees
for i in range(len(symptoms)):
    for j in range(len(dates)):                               ### adding numbers in boxes
        text = axes10.text(j,i,sympe_per[i,j],
                           ha='center',va='center',color='k',fontweight='bold')
        ### ha,va = adjusting location of text , color = bklack , fontweight = size of font
```



END

In [ ]: