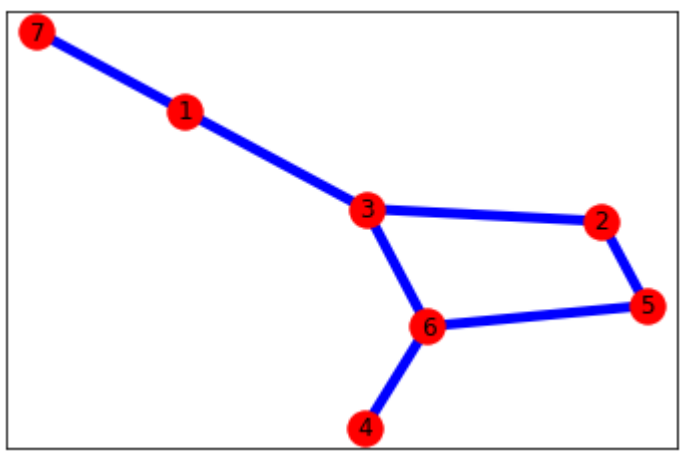
In [1]:
```
!pip install wordcloud
```

```
Collecting wordcloud
  Downloading wordcloud-1.8.2.2-cp38-cp38-win_amd64.whl (152 kB)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-packages (from wordcloud) (3.3.2)
Requirement already satisfied: pillow in c:\programdata\anaconda3\lib\site-packages (from wordcloud) (8.0.1)
Requirement already satisfied: numpy>=1.6.1 in c:\programdata\anaconda3\lib\site-packages (from wordcloud) (1.19.2)
Requirement already satisfied: python-dateutil>=2.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->wor
dcloud) (2.8.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->wordcl
oud) (1.3.0)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->wordcloud)
(0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\programdata\anaconda3\lib\site-packages
(from matplotlib->wordcloud) (2.4.7)
Requirement already satisfied: certifi>=2020.06.20 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->word
cloud) (2020.6.20)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.1->matpl
otlib->wordcloud) (1.15.0)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.8.2.2
```

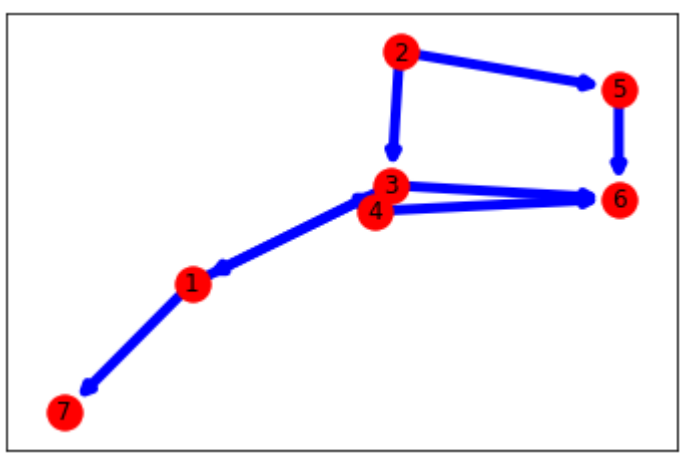
NetworkX Library

Graph is a data structure.It is used to show relationships. Eg Facebook Name and friends, routers, roads (this many paths) Nodes (vertices) edge

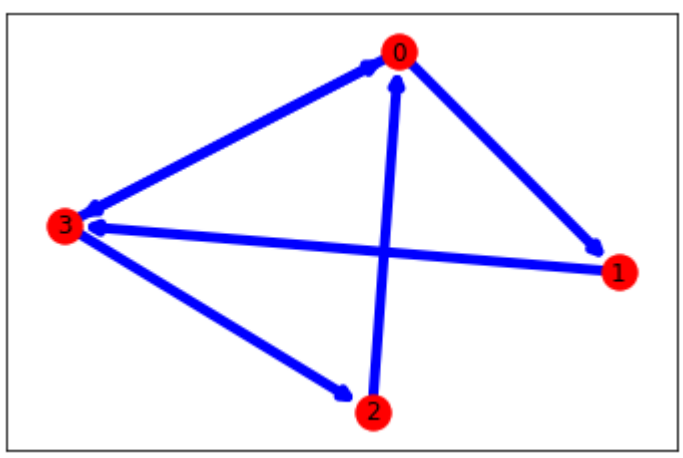
In [11]:
```
#undirected graphs
import networkx as nx
import matplotlib.pyplot as plt
G=nx.Graph()
G.add_node(1)
G.add_nodes_from([2,3]) #to take nodes from list
G.add_nodes_from(range(4,7)) #to take nodes in range 7 not included
G.add_edge(1,3)
G.add_edge(2,5)
G.add_edge(1,7)
G.add_edge(3,3)
G.add_edges_from([(2,3),(3,6),(4,6),(5,6)]) #to draw edges in list
nx.draw_networkx(G,node_size=300,node_color='red',edge_color='blue',width=5)
plt.figsize=[150,150]
plt.show()

```



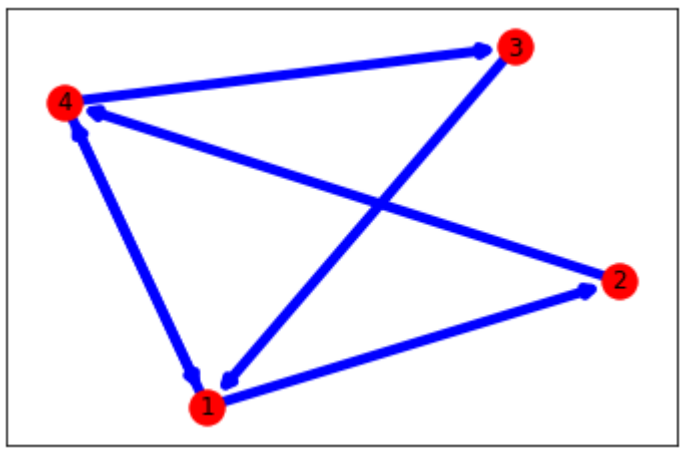
In [13]:
```
#Directed edges graph
import networkx as nx
import matplotlib.pyplot as plt
G=nx.DiGraph()
G.add_node(1)
G.add_nodes_from([2,3]) #to take nodes from list
G.add_nodes_from(range(4,7)) #to take nodes in range 7 not included
G.add_edge(1,3)
G.add_edge(3,1)
G.add_edge(2,5)
G.add_edge(1,7)
G.add_edge(3,3)
G.add_edges_from([(2,3),(3,6),(4,6),(5,6)]) #to draw edges in list
nx.draw_networkx(G,node_size=300,node_color='red',edge_color='blue',width=5)
plt.figsize=[150,150]
plt.show()
```



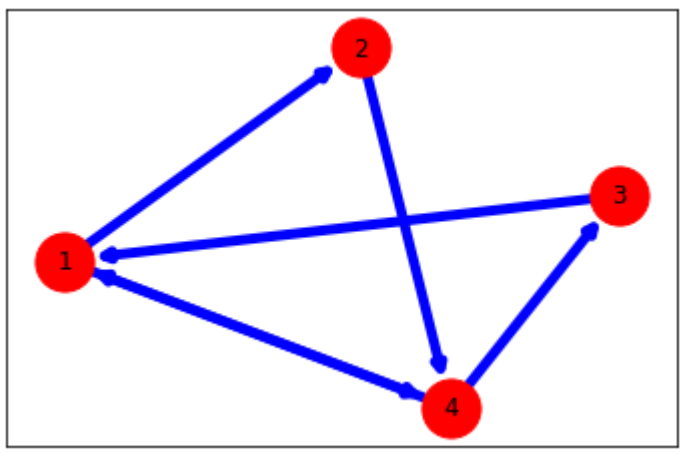
In [14]:
```
#Creating graph from adjacency matrix
#Directed edges graph
import networkx as nx
import matplotlib.pyplot as plt
G=nx.DiGraph()
G.add_nodes_from(range(4))
L=[[0,1,0,1],[0,0,0,1],[1,0,0,0],[1,0,1,0]]
for i in range(4):
    for j in range(4):
        if L[i][j]==1:
            G.add_edge(i,j)
nx.draw_networkx(G,node_size=300,node_color='red',edge_color='blue',width=5)
plt.figsize=[150,150]
plt.show()


```



In [17]:
```
#Creating graph from adjacency matrix
#Directed edges graph
import networkx as nx
import matplotlib.pyplot as plt
G=nx.DiGraph()
G.add_nodes_from(range(1,5))
L=[[0,1,0,1],[0,0,0,1],[1,0,0,0],[1,0,1,0]]
for i in range(1,5):
    for j in range(1,5):
        if L[i-1][j-1]==1:
            G.add_edge(i,j)
nx.draw_networkx(G,node_size=300,node_color='red',edge_color='blue',width=5)
plt.figsize=[150,150]
plt.show()
```



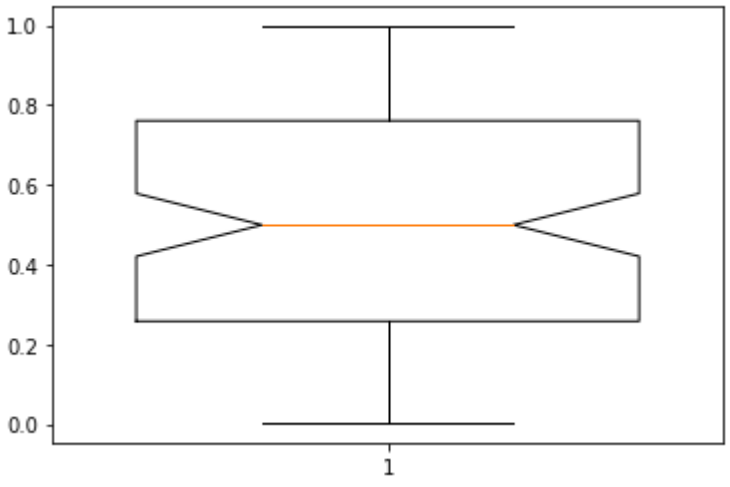
In [26]:
```
#Creating graph from adjacency matrix
#Directed edges graph
import networkx as nx
import matplotlib.pyplot as plt
G=nx.DiGraph()
G.add_nodes_from(range(1,5))
L=[[0,1,0,1],[0,0,0,1],[1,0,0,0],[1,0,1,0]]
for i in range(1,5):
    for j in range(1,5):
        if L[i-1][j-1]==1:
            G.add_edge(i,j)
nx.draw_networkx(G,node_size=850,node_color='red',edge_color='blue',width=5)
plt.show()
```



Box Plot

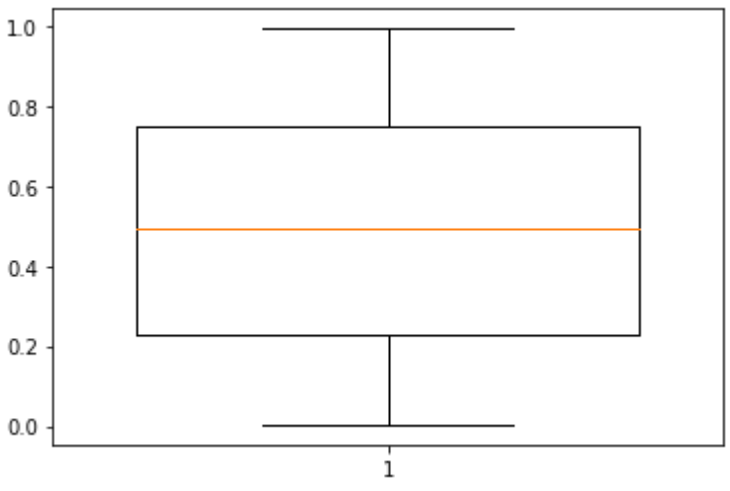
In [30]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
data=np.random.rand(100)
print(data)
plt.boxplot(data,widths=0.75,notch=True)
plt.show()
```

```
[0.19587193 0.33024785 0.59060691 0.04236225 0.99677956 0.42992894
 0.41972571 0.48150536 0.43337453 0.46894457 0.45117453 0.73819281
 0.87866061 0.71508082 0.20478861 0.75878633 0.88479059 0.79734686
 0.15907678 0.98850057 0.59795628 0.81377052 0.968925   0.86874806
 0.20523662 0.67232838 0.74754315 0.87123192 0.89082637 0.52513697
 0.42395384 0.61448532 0.11433618 0.12991207 0.8545214  0.36815924
 0.17232991 0.25300328 0.56307971 0.44676789 0.27151793 0.20029932
 0.77980191 0.87691477 0.55342    0.12934368 0.44899172 0.70684579
 0.70055347 0.76634792 0.32785517 0.5857054  0.57740545 0.39341813
 0.51706552 0.22679153 0.53159681 0.90285295 0.49519293 0.33728122
 0.09655309 0.24198318 0.31269364 0.74301925 0.25980094 0.6789896
 0.89194518 0.61540141 0.06993671 0.0658349  0.18222315 0.75737475
 0.62344797 0.79493436 0.26640798 0.89475245 0.50366709 0.12824438
 0.85106241 0.21479843 0.70754348 0.10479756 0.00369029 0.04492854
 0.4334735  0.81584484 0.72374187 0.15242209 0.91454146 0.94746979
 0.43153164 0.38134495 0.94426302 0.9669149  0.97949214 0.37730551
 0.44626026 0.01735724 0.45898665 0.02599344]
```



In [32]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
data=np.random.rand(100)
print(data)
plt.boxplot(data,widths=0.75,notch=False)
plt.show()
```

```
[0.0955448  0.84465516 0.85589685 0.35780966 0.64075912 0.71383076
 0.50719031 0.07838822 0.4639324  0.32215492 0.62284858 0.27356936
 0.4697394  0.43483209 0.87859011 0.22063502 0.94004306 0.79006354
 0.29403209 0.99380664 0.06758588 0.21780085 0.23324997 0.44707076
 0.78080737 0.14328862 0.16464221 0.3731882  0.56884569 0.33721647
 0.50870924 0.9434136  0.26579865 0.05582965 0.20171255 0.65329756
 0.38843897 0.92178247 0.20168028 0.88355061 0.66953692 0.10434104
 0.24724122 0.15343322 0.62787844 0.66825599 0.77326616 0.7011085
 0.8587138  0.49739781 0.53139411 0.77366055 0.67287932 0.68669083
 0.21859197 0.01944586 0.76119395 0.54863795 0.99099656 0.93836375
 0.68415611 0.67454496 0.8829899  0.27195312 0.67943155 0.15801362
 0.54027603 0.09710567 0.8274521  0.80603869 0.6094487  0.42272219
 0.42549433 0.00293983 0.05909565 0.35158457 0.18211368 0.4274285
 0.41825982 0.28517995 0.57010641 0.31069368 0.70295952 0.20677163
 0.86980627 0.39171172 0.90885018 0.54817239 0.1067077  0.74228572
 0.15382679 0.91202592 0.04040793 0.93117064 0.78688693 0.85717159
 0.08355916 0.21507049 0.48900297 0.25690742]
```

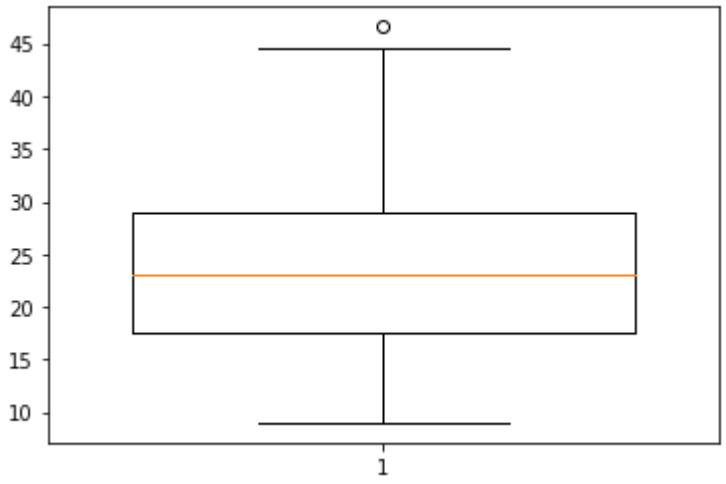


In [35]:
```
#To describe all statistics
data1=pd.DataFrame(data)
data1.describe()
```

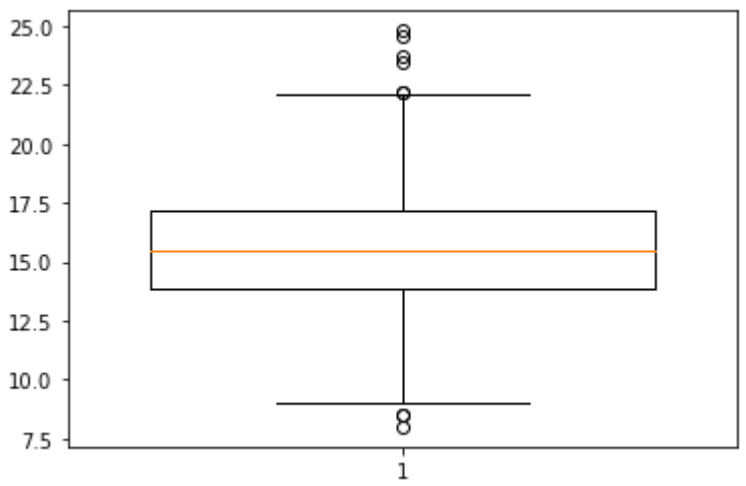
Out[35]:

	0
count	100.000000
mean	0.494896
std	0.287786
min	0.002940
25%	0.230096
50%	0.493200
75%	0.747013
max	0.993807

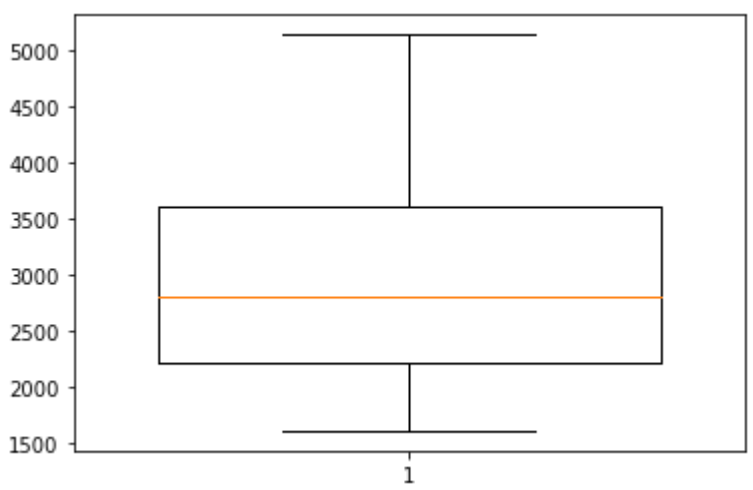
In [39]:
```
#To describe outliers in boxplot for mpg
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
plt.boxplot(dataset['mpg'],widths=0.75)
plt.show()
```



In [40]:
```
#To describe outliers in boxplot for acceleration
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
plt.boxplot(dataset['acceleration'],widths=0.75)
plt.show()
```



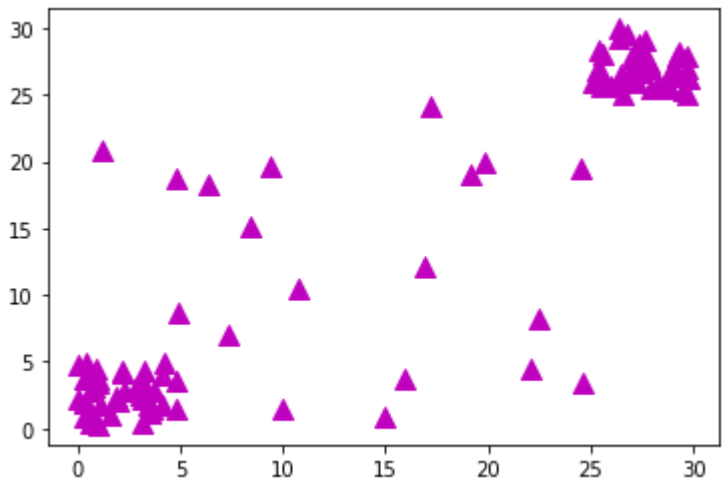
In [41]:
```
#To describe outliers in boxplot for weight
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
plt.boxplot(dataset['weight'],widths=0.75)
plt.show()
```



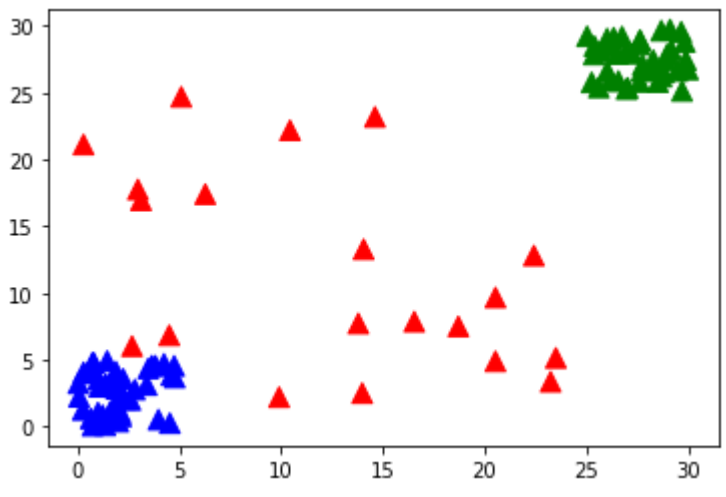
Scatter Plots

Creating groups

In [45]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
x1=5*np.random.rand(40)
x2=5*np.random.rand(40)+25
x3=25*np.random.rand(20)
x=np.concatenate((x1,x2,x3))
y1=5*np.random.rand(40)
y2=5*np.random.rand(40)+25
y3=25*np.random.rand(20)
y=np.concatenate((y1,y2,y3))
plt.scatter(x,y,s=[100],marker="^",c='m') #s for size, color=magneta
plt.show()
```



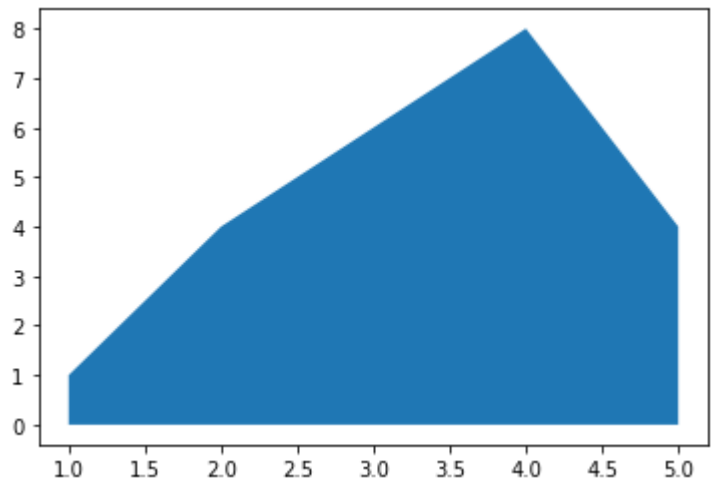
In [48]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
x1=5*np.random.rand(40)
x2=5*np.random.rand(40)+25
x3=25*np.random.rand(20)
x=np.concatenate((x1,x2,x3))
y1=5*np.random.rand(40)
y2=5*np.random.rand(40)+25
y3=25*np.random.rand(20)
y=np.concatenate((y1,y2,y3))
color_array=['b']*40+['g']*40+['r']*20
plt.scatter(x,y,s=[100],marker="^",c=color_array) #s for size, color=magneta
plt.show()
```



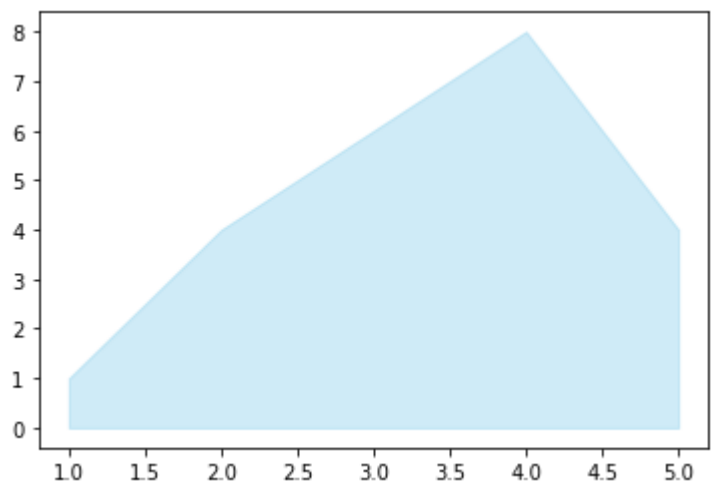
Area Plot (Area Charts)

Combination of line and bar graph For more than one y axis like income, expense and on x axis monthwise

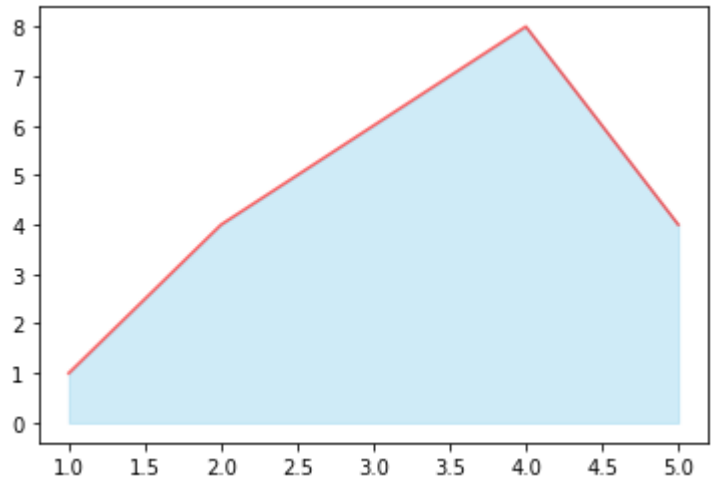
In [49]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
x=range(1,6)
y=[1,4,6,8,4]
plt.fill_between(x,y)
plt.show()
#ideally time or month on x axis is usually taken
```



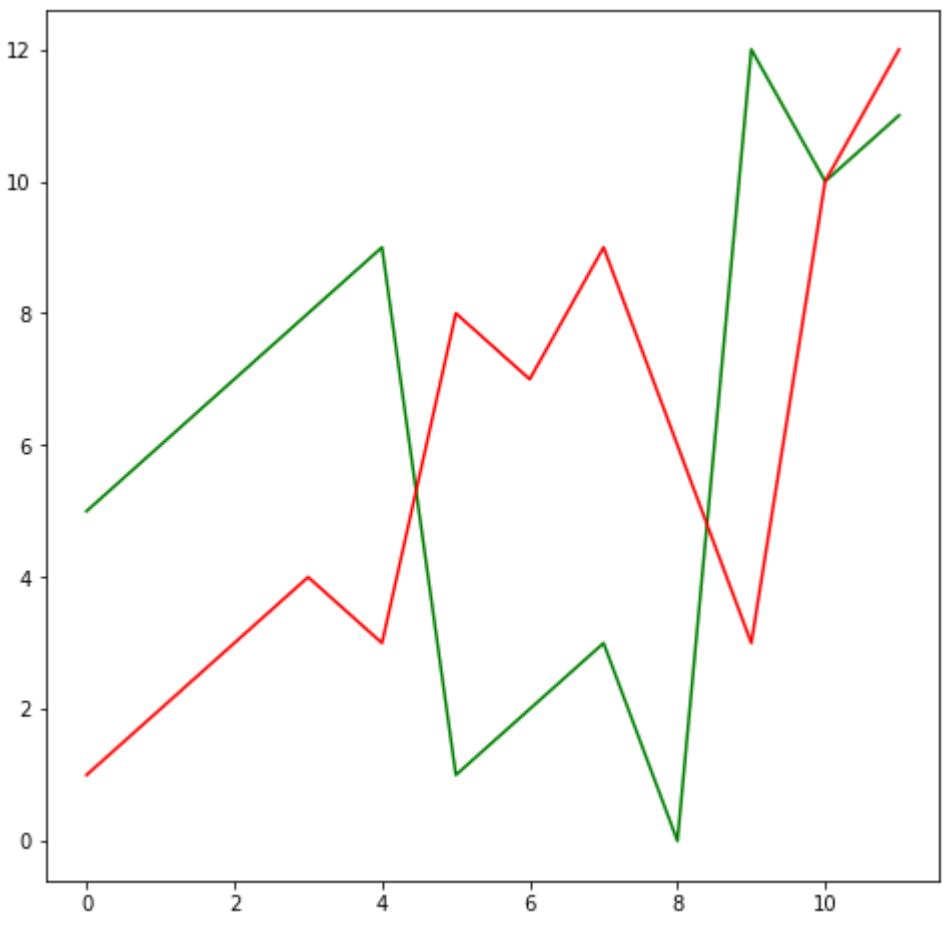
In [50]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
x=range(1,6)
y=[1,4,6,8,4]
plt.fill_between(x,y,color='skyblue',alpha=0.4)
plt.show()
```



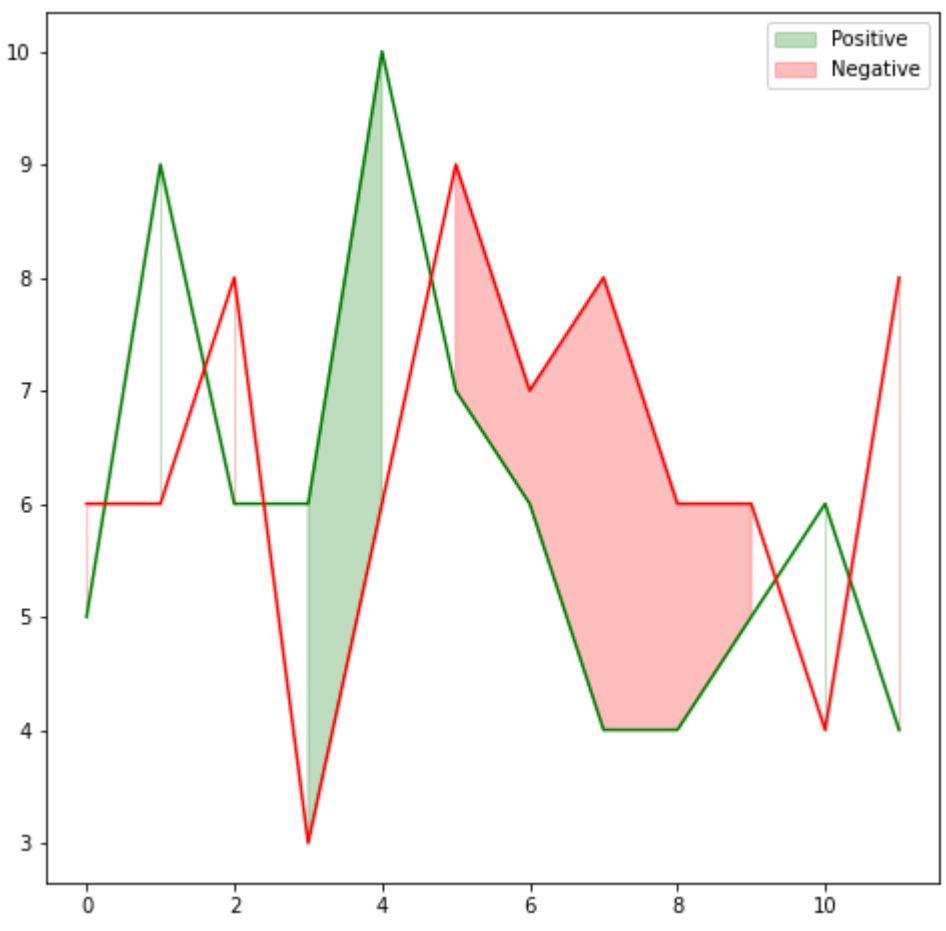
In [51]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
x=range(1,6)
y=[1,4,6,8,4]
plt.fill_between(x,y,color='skyblue',alpha=0.4)
#To plot line in fillcolor
plt.plot(x,y,color='red',alpha=0.6)
plt.show()
```



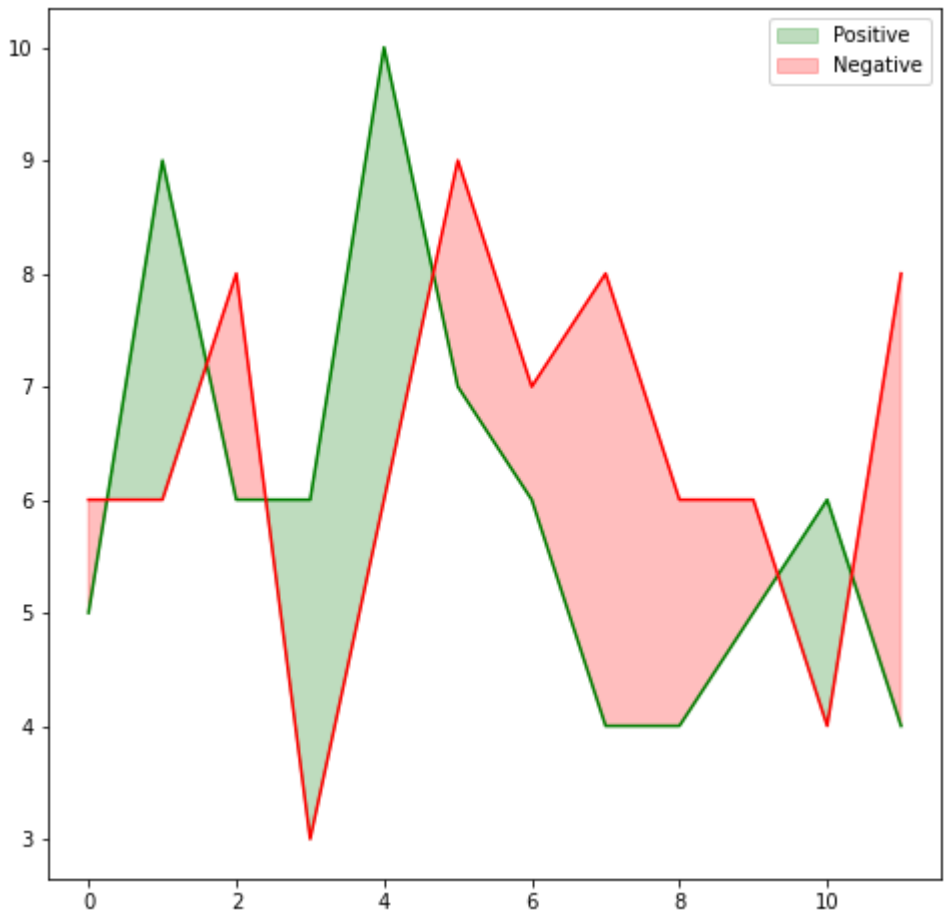
In [55]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
time=np.arange(12)
income=np.array([5,6,7,8,9,1,2,3,0,12,10,11])
expense=np.array([1,2,3,4,3,8,7,9,6,3,10,12])
fig,ax=plt.subplots(figsize=(8,8))
ax.plot(time,income,color='green')
ax.plot(time,expense,color='red')


plt.show()
```



In [70]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
time=np.arange(12)
income=np.array([5,9,6,6,10,7,6,4,4,5,6,4])
expense=np.array([6,6,8,3,6,9,7,8,6,6,4,8])
fig,ax=plt.subplots(figsize=(8,8))
ax.plot(time,income,color='green')
ax.plot(time,expense,color='red')
ax.fill_between(time,income,expense,where=(income>expense),color="green",alpha=0.25,label="Positive",
                interpolate=False)
#Default interpolate False
ax.fill_between(time,income,expense,where=(income<=expense),color="red",alpha=0.25,label="Negative",
                interpolate=False)
ax.legend()
plt.show()
```



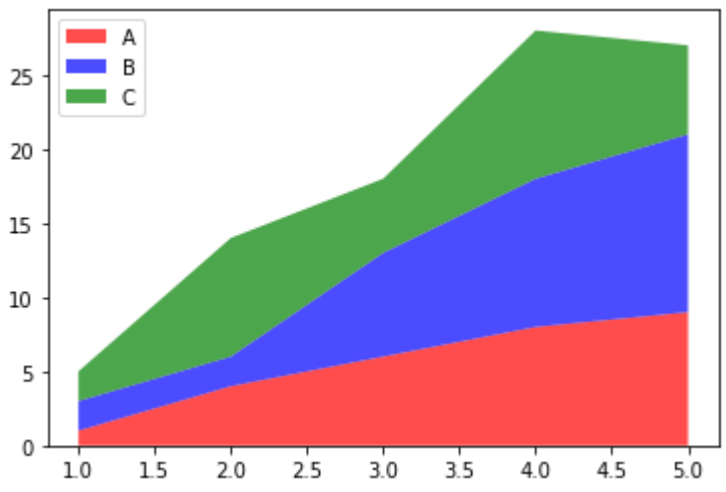
In [69]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
time=np.arange(12)
income=np.array([5,9,6,6,10,7,6,4,4,5,6,4])
expense=np.array([6,6,8,3,6,9,7,8,6,6,4,8])
fig,ax=plt.subplots(figsize=(8,8))
ax.plot(time,income,color='green')
ax.plot(time,expense,color='red')
ax.fill_between(time,income,expense,where=(income>expense),color="green",alpha=0.25,label="Positive",
                interpolate=True)
#Default interpolate False
ax.fill_between(time,income,expense,where=(income<=expense),color="red",alpha=0.25,label="Negative",
                interpolate=True)
ax.legend()
plt.show()
#Why interpolate?
```



Stacked Area Charts

to compare 3 variables along with one

In [77]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
x=range(1,6)
y1=[1,4,6,8,9]
y2=[2,2,7,10,12]
y3=[2,8,5,10,6]
plt.stackplot(x,y1,y2,y3,labels=['A','B','C'],colors=['red','blue','green'],alpha=0.7)
#alpha to make transparent
plt.legend(loc='upper left')
plt.show()

```



Advanced Plots

Word Clouds

alice_in_wonderland.txt

In [78]:
```
from wordcloud import WordCloud,STOPWORDS
stopwords=set(STOPWORDS)
print(stopwords)
#stopwords will not give any meaning it is only used for joining as suppose "the" we are using much so it
#it should be stopped and removed from respective textfile
```

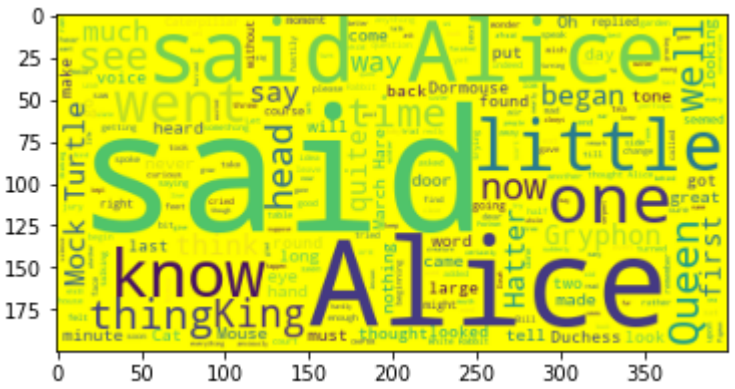
```
{"we're", 'otherwise', 'myself', 'a', "weren't", "can't", 'those', 'more', 'did', 'same', "that's", "when's", 'could',
'or', 'have', 'once', 'too', 'while', 'some', 'before', 'do', 'down', 'each', "mustn't", "i'll", 'of', 'such', 'their
s', 'so', 'this', 'few', "hadn't", 'all', 'my', 'yourselves', 'hers', 'himself', "there's", "aren't", 'in', 'are', "w
e'll", "haven't", 'www', "you've", 'had', 'you', 'after', 'me', "she'll", 'like', "he'd", 'were', 'her', 'him', 'bein
g', 'nor', 'how', 'through', "couldn't", "you'd", "where's", 'who', 'herself', 'and', "wouldn't", 'itself', 'has', 'aga
in', 'until', "they'd", 'off', "he'll", 'can', 'was', "let's", 'we', "didn't", 'but', 'these', "i'm", 'his', 'over', "w
e've", "it's", 'she', 'having', "shouldn't", "why's", 'than', 'above', "here's", 'cannot', "don't", 'where', "you're",
'if', "how's", 'be', 'them', 'between', "he's", "she'd", 'when', 'ours', 'it', 'then', 'any', 'that', 'both', 'below',
'yourself', "she's", "i'd", 'own', 'by', 'does', 'ought', 'is', 'out', "they've", 'there', 'what', 'am', 'which', "wo
n't", 'i', "we'd", "they're", "what's", 'from', 'to', 'however', 'k', 'themselves', 'because', 'no', 'get', 'an', 'duri
ng', 'ourselves', 'your', 'he', 'else', 'under', 'against', 'doing', 'ever', 'not', 'very', 'yours', 'as', 'been', "is
n't", "doesn't", 'most', 'into', 'since', "shan't", 'com', 'the', 'r', 'here', "they'll", 'they', 'with', "wasn't", 'fu
rther', 'also', 'about', 'its', 'hence', 'just', 'on', 'only', 'for', 'up', 'why', 'would', 'shall', "i've", 'other',
'at', 'our', 'their', "you'll", 'therefore', 'whom', 'http', "hasn't", "who's", 'should'}
```

In [79]:
```
f=open('alice_in_wonderland.txt','r')
alice=f.read()
alice_wc=WordCloud(background_color='yellow',max_words=2000,stopwords=set(STOPWORDS))
#max_words to limit the words
alice_wc.generate(alice)
```

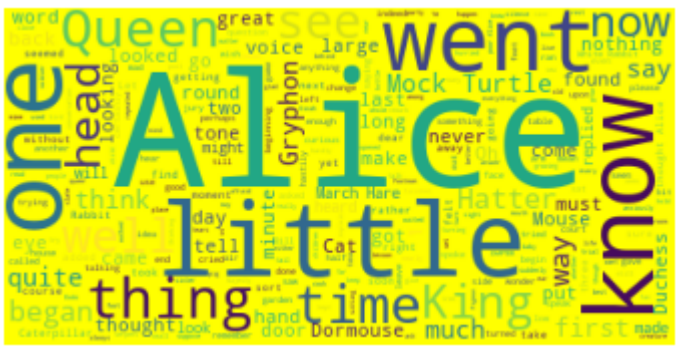
Out[79]: <wordcloud.wordcloud.WordCloud at 0x1682747f430>

In [80]:
```
#im_show()-image show
plt.imshow(alice_wc)
#to remove axis
plt.axis('off')
plt.show()
```

In [81]:
```
#im_show()-image show
plt.imshow(alice_wc)
#to remove axis
plt.axis('on')
plt.show()
```



In [91]:
```
f=open('alice_in_wonderland.txt','r')
alice=f.read()
stopwords=set(STOPWORDS)
alice_wc=WordCloud(background_color='yellow',max_words=2000,stopwords=stopwords)
stopwords.add('said')
alice_wc.generate(alice)
#im_show()-image show
plt.imshow(alice_wc)
#to remove axis
plt.axis('off')
plt.show()
```



Regression Plots

In [4]:
```
import seaborn as sns
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
dataset.corr()
#regplot() is regression plot
sns.regplot(x='mpg',y=displacement,data=dataset)
```

```
---------------------------------------------------------------------------
ImportError                               Traceback (most recent call last)
<ipython-input-4-a3084229f52d> in <module>
----> 1 import seaborn as sns
      2 import pandas as pd
      3 dataset=pd.read_csv('auto-mpg.csv')
      4 dataset.corr()
      5 #regplot() is regression plot

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\__init__.py in <module>
      1 # Import seaborn objects
----> 2 from .rcmod import *  # noqa: F401,F403
      3 from .utils import *  # noqa: F401,F403
      4 from .palettes import *  # noqa: F401,F403
      5 from .relational import *  # noqa: F401,F403

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\rcmod.py in <module>
      5 import matplotlib as mpl
      6 from cycler import cycler
```

In [2]:
```
!pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\programdata\anaconda3\lib\site-packages (0.11.0)
Requirement already satisfied: pandas>=0.23 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (1.1.3)
Requirement already satisfied: numpy>=1.15 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (1.19.2)
Requirement already satisfied: matplotlib>=2.2 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (3.3.2)
Requirement already satisfied: scipy>=1.0 in c:\programdata\anaconda3\lib\site-packages (from seaborn) (1.5.2)
Requirement already satisfied: pytz>=2017.2 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.23->seaborn)
(2020.1)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.23-
>seaborn) (2.8.1)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seabo
rn) (8.0.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->s
eaborn) (1.3.0)
Requirement already satisfied: certifi>=2020.06.20 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2-
>seaborn) (2020.6.20)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>=2.2->seabor
n) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\programdata\anaconda3\lib\site-packages
(from matplotlib>=2.2->seaborn) (2.4.7)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.7.3->pan
das>=0.23->seaborn) (1.15.0)
```

In [10]:
```
import seaborn as sns
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')

#regplot() is regression plot
sns.regplot(x='mpg',y='displacement',data=dataset)
```

```
---------------------------------------------------------------------------
ImportError                               Traceback (most recent call last)
<ipython-input-10-17174c7d20cc> in <module>
----> 1 import seaborn as sns
      2 import pandas as pd
      3 dataset=pd.read_csv('auto-mpg.csv')
      4
      5 #regplot() is regression plot

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\__init__.py in <module>
      1 # Import seaborn objects
----> 2 from .rcmod import *  # noqa: F401,F403
      3 from .utils import *  # noqa: F401,F403
      4 from .palettes import *  # noqa: F401,F403
      5 from .relational import *  # noqa: F401,F403

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\rcmod.py in <module>
      5 import matplotlib as mpl
      6 from cycler import cycler
```

In [ ]:
```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
plt.figure(figsize=(15,10))
dataset=pd.read_csv('auto-mpg.csv')
dataset.corr()
#regplot() is regression plot
ax=sns.regplot(x='mpg',y='displacement',data=dataset,color='orange',marker="*",scatter_kws={'s':200})
#to enter the marker size
ax.set(xlabel='mpg',ylabel='displacement')
ax.set_title('mpg vs displacement')
```

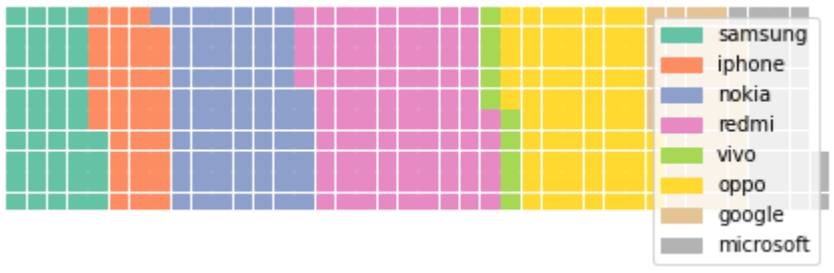
In [11]:
```
!pip install pywaffle
```

```
Collecting pywaffle
  Downloading pywaffle-1.1.0-py2.py3-none-any.whl (30 kB)
Collecting fontawesomefree
  Downloading fontawesomefree-6.2.1-py3-none-any.whl (25.1 MB)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-packages (from pywaffle) (3.3.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\programdata\anaconda3\lib\site-packages
(from matplotlib->pywaffle) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->pywaff
le) (1.3.0)
Requirement already satisfied: numpy>=1.15 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->pywaffle)
(1.19.2)
Requirement already satisfied: python-dateutil>=2.1 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->pyw
affle) (2.8.1)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->pywaffle)
(8.0.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->pywaffle)
(0.10.0)
Requirement already satisfied: certifi>=2020.06.20 in c:\programdata\anaconda3\lib\site-packages (from matplotlib->pywa
ffle) (2020.6.20)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.1->matpl
otlib->pywaffle) (1.15.0)
Installing collected packages: fontawesomefree, pywaffle
Successfully installed fontawesomefree-6.2.1 pywaffle-1.1.0
```

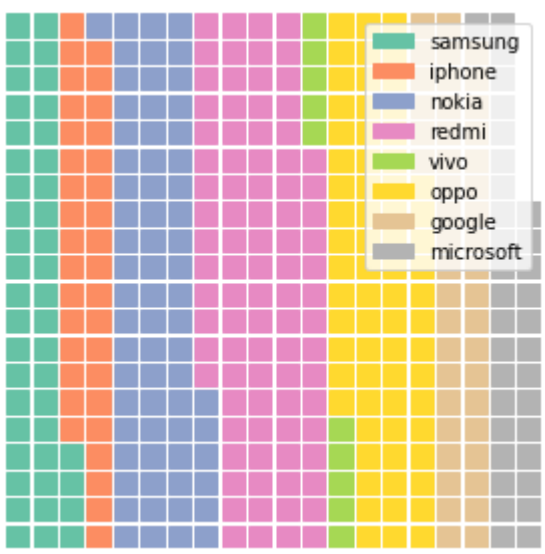
Waffle Chart

Eg. Unemplyment rate,

In [16]:
```
import pandas as pd
import matplotlib.pyplot as plt
from pywaffle import Waffle
data={'phone':['samsung','iphone','nokia','redmi','vivo','oppo','google','microsoft'],
      'stock':[44,35,67,89,10,69,45,34]}
df=pd.DataFrame(data)
df.head() #It prints first five rows
fig=plt.figure(FigureClass=Waffle,rows=10,values=df.stock,labels=list(df.phone))
```



In [17]:
```
import pandas as pd
import matplotlib.pyplot as plt
from pywaffle import Waffle
data={'phone':['samsung','iphone','nokia','redmi','vivo','oppo','google','microsoft'],
      'stock':[44,35,67,89,10,69,45,34]}
df=pd.DataFrame(data)
df.head() #It prints first five rows
fig=plt.figure(FigureClass=Waffle,rows=20,values=df.stock,labels=list(df.phone))
#rows variation
```



Heat maps

In [20]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
#to plot a matrix
data=np.random.randint(low=1,high=100,size=(10,10))
print(data)
hm=sns.heatmap(data=da)
```

```
---------------------------------------------------------------------------
ImportError                               Traceback (most recent call last)
<ipython-input-20-e2c0485b6a9d> in <module>
      2 import matplotlib.pyplot as plt
      3 import numpy as np
----> 4 import seaborn as sns
      5 #to plot a matrix
      6 data=np.random.randint(low=1,high=100,size=(10,10))

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\__init__.py in <module>
      1 # Import seaborn objects
----> 2 from .rcmod import *  # noqa: F401,F403
      3 from .utils import *  # noqa: F401,F403
      4 from .palettes import *  # noqa: F401,F403
      5 from .relational import *  # noqa: F401,F403

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\rcmod.py in <module>
      5 import matplotlib as mpl
      6 from cycler import cycler
```

In [21]:
```
!pip install folium

```

```
Collecting folium
  Downloading folium-0.14.0-py2.py3-none-any.whl (102 kB)
Requirement already satisfied: jinja2>=2.9 in c:\programdata\anaconda3\lib\site-packages (from folium) (2.11.2)
Collecting branca>=0.6.0
  Downloading branca-0.6.0-py3-none-any.whl (24 kB)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from folium) (1.19.2)
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (from folium) (2.24.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\programdata\anaconda3\lib\site-packages (from jinja2>=2.9->foliu
m) (1.1.1)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests->folium)
(2020.6.20)
Requirement already satisfied: chardet<4,>=3.0.2 in c:\programdata\anaconda3\lib\site-packages (from requests->folium)
(3.0.4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (f
rom requests->folium) (1.25.11)
Requirement already satisfied: idna<3,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests->folium) (2.1
0)
Installing collected packages: branca, folium
Successfully installed branca-0.6.0 folium-0.14.0
```

geospatial data

latitude and longitudinal

In [22]:
```
import folium
world_map=folium.Map()
world_map
```

Out[22]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [30]:
```
import folium
world_map=folium.Map(location=[20.5937,78.9629],zoom_start=8)
#North positive value, South positive value, east positive value, west negative value
world_map
```

Out[30]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [28]:
```
import folium
world_map=folium.Map(location=[22.9900,72.4867])
#North positive value, South positive value, east positive value, west negative value
world_map
```

Out[28]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [31]:
```
import folium
world_map=folium.Map(location=[20.5937,78.9629],zoom_start=8,tiles='Stamen Toner')
#tiles are used for copy black and white
#North positive value, South positive value, east positive value, west negative value
world_map
```

Out[31]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [32]:
```
import folium
world_map=folium.Map(location=[20.5937,78.9629],zoom_start=8,tiles='Stamen Terrain')
#tiles are used for copy black and white
#North positive value, South positive value, east positive value, west negative value
world_map
```

Out[32]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [40]:
```
#police file for san fransico
import folium
import pandas as pd
df_incidents=pd.read_csv('Police_Department_Incidents_-_Previous_Year__2016_.csv')
print(df_incidents.head())
df_incidents.shape
df=df_incidents.iloc[0:100,:] #passed rows (0 to 100) and columns (all 13)
df.shape
sanfran_map=folium.Map(location=[37.77,-122.44],zoom_start=12)
incidents=folium.map.FeatureGroup()
for lat,lon in zip(df.Y,df.X):
    #zip can be done using df.X and df.Y if both have some number of datas
    #to create a child node
    incidents.add_child(folium.features.CircleMarker([lat,lon],
                                                     radius=5,color='red',
                                                     fill=True,fill_color='blue',
                                                     fill_opacity=0.8,))

for lat,lon,label in zip(df.Y,df.X,df.Category):
    folium.Marker([lat,lon],popup=label).add_to(sanfran_map)
    #to show circle with filled color
#iloc in folium for limiting data
sanfran_map.add_child(incidents)
```

```
   IncidntNum      Category                                   Descript  \
0   120058272   WEAPON LAWS                     POSS OF PROHIBITED WEAPON
1   120058272   WEAPON LAWS  FIREARM, LOADED, IN VEHICLE, POSSESSION OR USE
2   141059263      WARRANTS                                WARRANT ARREST
3   160013662  NON-CRIMINAL                                 LOST PROPERTY
4   160002740  NON-CRIMINAL                                 LOST PROPERTY

  DayOfWeek                  Date   Time  PdDistrict      Resolution  \
0    Friday  01/29/2016 12:00:00 AM  11:00    SOUTHERN  ARREST, BOOKED
1    Friday  01/29/2016 12:00:00 AM  11:00    SOUTHERN  ARREST, BOOKED
2    Monday  04/25/2016 12:00:00 AM  14:59     BAYVIEW  ARREST, BOOKED
3   Tuesday  01/05/2016 12:00:00 AM  23:50  TENDERLOIN            NONE
4    Friday  01/01/2016 12:00:00 AM  00:30     MISSION            NONE

                Address           X          Y  \
0  800 Block of BRYANT ST -122.403405  37.775421
1  800 Block of BRYANT ST -122.403405  37.775421
2   KEITH ST / SHAFTER AV -122.388856  37.729981
3  JONES ST / OFARRELL ST -122.412971  37.785788
```

Choropleth Maps

to make such type of heat maps in folium (world map)

In [46]:
```
#unemployement file in US JSON
import folium
import pandas as pd
state_unemp=pd.read_csv('US_Unemployment_Oct2012.csv')
state_geo='us-states.json'
usa_state=folium.Map(location=[48,-102],zoom_start=3)
folium.Choropleth(geo_data=state_geo,name='choropleth',data=state_unemp,
                  columns=['State','Unemployment'],key_on='feature.id',
                  fill_color='YlGnBu',fill_opacity=0.7,line_opacity=0.2,
                  legend_name='Unemployment scale').add_to(usa_state)
usa_state

#to match from json file feature and short name is given in id
#to give location through json file

```

Out[46]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [ ]:
```

```