In [6]:
```
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
def find_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]<low_val)| (ds[col]>high_val)]

    return ds
find_outliers(dataset,'acceleration')
```

```
Low: 8.800000000000008 High: 22.199999999999992
```

Out[6]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
7	14.0	8	440.0	215	4312	8.5	70	1	plymouth fury iii
9	15.0	8	390.0	190	3850	8.5	70	1	amc ambassador dpl
11	14.0	8	340.0	160	3609	8.0	70	1	plymouth 'cuda 340
59	23.0	4	97.0	54	2254	23.5	72	2	volkswagen type 3
195	29.0	4	85.0	52	2035	22.2	76	1	chevrolet chevette
299	27.2	4	141.0	71	3190	24.8	79	2	peugeot 504
300	23.9	8	260.0	90	3420	22.2	79	1	oldsmobile cutlass salon brougham
326	43.4	4	90.0	48	2335	23.7	80	2	vw dasher (diesel)
394	44.0	4	97.0	52	2130	24.6	82	2	vw pickup

In [7]:
```
find_outliers(dataset,'mpg')
```

```
Low: 0.25 High: 46.25
```

Out[7]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
322	46.6	4	86.0	65	2110	17.9	80	3	mazda glc

In [9]:
```
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
def remove_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]>=low_val) & (ds[col]<=high_val)]

    return ds
remove_outliers(dataset,'acceleration')
```

```
Low: 8.800000000000008 High: 22.199999999999992
```

Out[9]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
392	27.0	4	151.0	90	2950	17.3	82	1	chevrolet camaro
393	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl
395	32.0	4	135.0	84	2295	11.6	82	1	dodge rampage
396	28.0	4	120.0	79	2625	18.6	82	1	ford ranger
397	31.0	4	119.0	82	2720	19.4	82	1	chevy s-10

389 rows × 9 columns

In [10]:

```
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
def remove_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]>=low_val) & (ds[col]<=high_val)]

    return ds
remove_outliers(dataset,'mpg')
```

```
Low: 0.25 High: 46.25
```

Out[10]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
393	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl
394	44.0	4	97.0	52	2130	24.6	82	2	vw pickup
395	32.0	4	135.0	84	2295	11.6	82	1	dodge rampage
396	28.0	4	120.0	79	2625	18.6	82	1	ford ranger
397	31.0	4	119.0	82	2720	19.4	82	1	chevy s-10

397 rows × 9 columns

drop down dupliacte

In [24]:

```
import pandas as pd
data = {"A":["TeamA","TeamB","TeamB","TeamC","TeamA"],
        "B":[50,40,40,30,50],
        "C":[True,False,False,False,True]}
df = pd.DataFrame(data)
print(df)
```

```
       A   B      C
0  TeamA  50   True
1  TeamB  40  False
2  TeamB  40  False
3  TeamC  30  False
4  TeamA  50   True
```

In [25]:

```
dups = df.duplicated()
print(dups)
```

```
0    False
1    False
2     True
3    False
4     True
dtype: bool
```

In [26]:
```
df=df.drop_duplicates()
print(df)
```

```
       A   B      C
0  TeamA  50   True
1  TeamB  40  False
3  TeamC  30  False
```

In [27]:
```
df=df.reset_index(drop=True)
print(df)
```

```
       A   B      C
0  TeamA  50   True
1  TeamB  40  False
2  TeamC  30  False
```

In [5]:
```
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
dataset['horsepower']=pd.to_numeric(dataset['horsepower'],errors='coerce')
dataset.isna().sum()
```

Out[5]:
```
mpg             0
cylinders       0
displacement    0
horsepower      6
weight          0
acceleration    0
model year      0
origin          0
car name        0
dtype: int64
```

In [1]:
```
import pandas as pd
dataset=pd.read_csv('bollywood.csv')
print(dataset)
```

```
                                    movie              lead
0                 Uri: The Surgical Strike     Vicky Kaushal
1                            Battalion 609       Vicky Ahuja
2     The Accidental Prime Minister (film)       Anupam Kher
3                          Why Cheat India     Emraan Hashmi
4                          Evening Shadows  Mona Ambegaonkar
...                                    ...               ...
1495                 Hum Tumhare Hain Sanam    Shah Rukh Khan
1496                     Aankhen (2002 film)  Amitabh Bachchan
1497                        Saathiya (film)      Vivek Oberoi
1498                         Company (film)        Ajay Devgn
1499                   Awara Paagal Deewana      Akshay Kumar

[1500 rows x 2 columns]
```

In [3]:
```
dataset[dataset['lead']=='Vicky Kaushal']
```

Out[3]:
```
<bound method DataFrame.count of                       movie           lead
0   Uri: The Surgical Strike  Vicky Kaushal
86      Love per Square Foot  Vicky Kaushal>
```

In [6]:
```
dataset[dataset['lead']=='Vicky Kaushal'].count()
```

Out[6]:
```
movie    2
lead     2
dtype: int64
```

In [5]:
```
dataset[dataset['lead']=='Vicky Kaushal'].shape
```

Out[5]:
```
(2, 2)
```

In [7]:
```
dataset[dataset['lead']=='Vicky Kaushal'].shape[0]
```

Out[7]:
```
2
```

In [9]:
```
dataset[dataset['lead']=='Amitabh Bachchan'].count()
```

Out[9]:
```
movie    45
lead     45
dtype: int64
```

In [10]:
```
dataset[dataset['lead']=='Amitabh Bachchan'].shape[0]
```

Out[10]:
```
45
```

In [16]:
```
dataset[dataset['movie']=='Kaante']['lead']
```

Out[16]:
```
1494    Amitabh Bachchan
Name: lead, dtype: object
```

In [19]:
```
import pandas as pd
dataset=pd.read_csv('movies.csv')
print(dataset.info)
```

```
<bound method DataFrame.info of                                          title_x     imdb_i
d  \
0                  Uri: The Surgical Strike   tt8291224
1                             Battalion 609   tt9472208
2     The Accidental Prime Minister (film)   tt6986710
3                           Why Cheat India   tt8108208
4                           Evening Shadows   tt6028796
...                                     ...         ...
1624                   Tera Mera Saath Rahen   tt0301250
1625                    Yeh Zindagi Ka Safar   tt0298607
1626                         Sabse Bada Sukh   tt0069204
1627                                   Daaka  tt10833860
1628                                Humsafar   tt2403201

                                            poster_path  \
0     https://upload.wikimedia.org/wikipedia/en/thum...
1                                                   NaN
2     https://upload.wikimedia.org/wikipedia/en/thum...
3     https://upload.wikimedia.org/wikipedia/en/thum...
4                                                   NaN
...                                                 ...
1624  https://upload.wikimedia.org/wikipedia/en/2/2b...
1625  https://upload.wikimedia.org/wikipedia/en/thum...
1626                                                NaN
1627  https://upload.wikimedia.org/wikipedia/en/thum...
1628  https://upload.wikimedia.org/wikipedia/en/thum...

                                              wiki_link  \
0     https://en.wikipedia.org/wiki/Uri:_The_Surgica...
1           https://en.wikipedia.org/wiki/Battalion_609
2     https://en.wikipedia.org/wiki/The_Accidental_P...
3         https://en.wikipedia.org/wiki/Why_Cheat_India
4         https://en.wikipedia.org/wiki/Evening_Shadows
...                                                 ...
1624  https://en.wikipedia.org/wiki/Tera_Mera_Saath_...
1625  https://en.wikipedia.org/wiki/Yeh_Zindagi_Ka_S...
1626      https://en.wikipedia.org/wiki/Sabse_Bada_Sukh
1627                https://en.wikipedia.org/wiki/Daaka
1628             https://en.wikipedia.org/wiki/Humsafar

                          title_y               original_title  is_adult  \
0        Uri: The Surgical Strike     Uri: The Surgical Strike         0
1                   Battalion 609                Battalion 609         0
```

```
2     The Accidental Prime Minister  The Accidental Prime Minister         0
3                  Why Cheat India                Why Cheat India         0
4                  Evening Shadows                Evening Shadows         0
...                            ...                            ...       ...
1624          Tera Mera Saath Rahen          Tera Mera Saath Rahen         0
1625           Yeh Zindagi Ka Safar           Yeh Zindagi Ka Safar         0
1626                Sabse Bada Sukh                Sabse Bada Sukh         0
1627                          Daaka                          Daaka         0
1628                       Humsafar                       Humsafar         0

      year_of_release runtime           genres  imdb_rating  imdb_votes  \
0                2019     138  Action|Drama|War          8.4       35112   
1                2019     131               War          4.1          73   
2                2019     112   Biography|Drama          6.1        5549   
3                2019     121       Crime|Drama          6.0        1891   
4                2018     102             Drama          7.3         280   
...               ...     ...              ...          ...         ...   
1624             2001     148             Drama          4.9         278   
1625             2001     146             Drama          3.0         133   
1626             2018      \N      Comedy|Drama          6.1          13   
1627             2019     136            Action          7.4          38   
1628             2011      35     Drama|Romance          9.0        2968   

                                                  story  \
0     Divided over five chapters  the film chronicle...   
1     The story revolves around a cricket match betw...   
2     Based on the memoir by Indian policy analyst S...   
3     The movie focuses on existing malpractices in ...   
4     While gay rights and marriage equality has bee...   
...                                                 ...   
1624  Raj Dixit lives with his younger brother  Rahu...   
1625  Hindi pop-star  Sarina Devan  lives a wealthy ...   
1626  Village born Lalloo re-locates to Bombay  and ...   
1627  Shinda tries robbing a bank so he can be wealt...   
1628  Sara and Ashar are childhood friends who share...   

                                                summary tagline  \
0     Indian army special forces execute a covert op...     NaN   
1     The story of Battalion 609 revolves around a c...     NaN   
2     Explores Manmohan Singh's tenure as the Prime ...     NaN   
3     The movie focuses on existing malpractices in ...     NaN   
4     Under the 'Evening Shadows'  truth often plays...     NaN   
...                                                 ...     ...   
1624  A man is torn between his handicapped brother ...     NaN   
1625  A singer finds out she was adopted when the ed...     NaN   
1626  Village born Lalloo re-locates to Bombay  and ...     NaN   
1627  Shinda tries robbing a bank so he can be wealt...     NaN   
1628  Ashar and Khirad are forced to get married due...     NaN   

                                                 actors  \
0     Vicky Kaushal|Paresh Rawal|Mohit Raina|Yami Ga...   
1     Vicky Ahuja|Shoaib Ibrahim|Shrikant Kamat|Elen...   
2     Anupam Kher|Akshaye Khanna|Aahana Kumra|Atul S...   
3     Emraan Hashmi|Shreya Dhanwanthary|Snighdadeep ...   
4     Mona Ambegaonkar|Ananth Narayan Mahadevan|Deva...   
...                                                 ...   
1624  Ajay Devgn|Sonali Bendre|Namrata Shirodkar|Pre...   
1625  Ameesha Patel|Jimmy Sheirgill|Nafisa Ali|Gulsh...   
1626  Vijay Arora|Asrani|Rajni Bala|Kumud Damle|Utpa...   
1627                         Gippy Grewal|Zareen Khan|   
1628                                       Fawad Khan|   

            wins_nominations             release_date  
0                     4 wins     11 January 2019 (USA)  
1                        NaN   11 January 2019 (India)  
2                        NaN     11 January 2019 (USA)  
3                        NaN     18 January 2019 (USA)  
4     17 wins & 1 nomination   11 January 2019 (India)  
...                      ...                       ...  
```

```
1624                    NaN   7 November 2001 (India)
1625                    NaN  16 November 2001 (India)
1626                    NaN                       NaN
1627                    NaN     1 November 2019 (USA)
1628                    NaN     TV Series (2011–2012)

[1629 rows x 18 columns]>
```

In [51]:
```
dataset[dataset['year_of_release']==2019].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 75 entries, 0 to 1627
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   title_x           75 non-null     object 
 1   imdb_id           75 non-null     object 
 2   poster_path       70 non-null     object 
 3   wiki_link         75 non-null     object 
 4   title_y           75 non-null     object 
 5   original_title    75 non-null     object 
 6   is_adult          75 non-null     int64  
 7   year_of_release   75 non-null     int64  
 8   runtime           75 non-null     object 
 9   genres            75 non-null     object 
 10  imdb_rating       75 non-null     float64
 11  imdb_votes        75 non-null     int64  
 12  story             73 non-null     object 
 13  summary           75 non-null     object 
 14  tagline           10 non-null     object 
 15  actors            75 non-null     object 
 16  wins_nominations  6 non-null      object 
 17  release_date      62 non-null     object 
dtypes: float64(1), int64(3), object(14)
memory usage: 11.1+ KB
```

In [24]:
```
dataset[dataset["imdb_rating"]>=7].shape[0]
```

Out[24]:
```
361
```

In [ ]:
```
#list of the movies whoes imdb votes greather than 20000
```

In [46]:
```
story=dataset[dataset["imdb_votes"]>20000]
story[['title_x','story','wiki_link']].to_csv('mydata.csv',index=False)
```

In [45]:
```
dataset[(dataset['year_of_release']==2019) & (dataset['imdb_rating']>=7) &  (dataset
```

Out[45]:
```
3
```

In [52]:
```
release=dataset[dataset['year_of_release']==2019]
release=release.reset_index(drop=True)
j=0
for i in release['original_title']:
    print(i,":",release['release_date'][j])
    j=j+1
```

```
Uri: The Surgical Strike : 11 January 2019 (USA)
Battalion 609 : 11 January 2019 (India)
The Accidental Prime Minister : 11 January 2019 (USA)
Why Cheat India : 18 January 2019 (USA)
Fraud Saiyyan : 18 January 2019 (India)
Bombairiya : 18 January 2019 (India)
Manikarnika: The Queen of Jhansi : 25 January 2019 (USA)
Thackeray : 25 January 2019 (India)
Amavas : 8 February 2019 (India)
Gully Boy : 14 February 2019 (USA)
```

```
Hum chaar : 15 February 2019 (India)
Total Dhamaal : 22 February 2019 (India)
Sonchiriya : 1 March 2019 (India)
Badla : 8 March 2019 (India)
Photograph : 17 May 2019 (USA)
Risknamaa : 15 March 2019 (India)
22 Yards : 15 March 2019 (India)
Kesari : 21 March 2019 (USA)
Notebook : 29 March 2019 (USA)
Junglee : 29 March 2019 (USA)
Gone Kesh : 29 March 2019 (India)
Albert Pinto Ko Gussa Kyun Aata Hai? : 12 April 2019 (India)
The Tashkent Files : 12 April 2019 (India)
Kalank : 17 April 2019 (USA)
Setters : 3 May 2019 (India)
Student of the Year 2 : 10 May 2019 (USA)
PM Narendra Modi : 24 May 2019 (USA)
De De Pyaar De : 17 May 2019 (USA)
India's Most Wanted : 24 May 2019 (USA)
Khamoshi : 14 June 2019 (India)
Kabir Singh : 20 June 2019 (USA)
Article 15 : 28 June 2019 (USA)
One Day: Justice Delivered : 5 July 2019 (India)
Hume Tumse Pyaar Kitna : 5 July 2019 (India)
Super 30 : 12 July 2019 (USA)
Family of Thakurganj : 19 July 2019 (India)
Batla House : 15 August 2019 (USA)
Jhootha Kahin Ka : 19 July 2019 (India)
Judgementall Hai Kya : 26 July 2019 (USA)
Chicken Curry Law : 9 August 2019 (India)
Arjun Patiala : 26 July 2019 (USA)
Jabariya Jodi : 9 August 2019 (USA)
Pranaam : nan
The Sky Is Pink : 11 October 2019 (USA)
Mission Mangal : 15 August 2019 (USA)
Saaho : 30 August 2019 (USA)
Dream Girl : 13 September 2019 (USA)
Section 375 : 13 September 2019 (USA)
The Zoya Factor : 20 September 2019 (USA)
Pal Pal Dil Ke Paas : 20 September 2019 (USA)
Prassthanam : 20 September 2019 (USA)
P Se Pyaar F Se Faraar : 18 October 2019 (India)
Ghost : 18 October 2019 (India)
Bala : 7 November 2019 (USA)
#Yaaram : nan
Housefull 4 : 25 October 2019 (USA)
Saand Ki Aankh : 25 October 2019 (USA)
Made in China : 25 October 2019 (USA)
Ujda Chaman : 1 November 2019 (USA)
Bypass Road : 8 November 2019 (USA)
Satellite Shankar : 8 November 2019 (India)
Jhalki : nan
Marjaavaan : 15 November 2019 (USA)
Motichoor Chaknachoor : 15 November 2019 (USA)
Keep Safe Distance : nan
Pagalpanti : nan
Ramprasad Ki Tehrvi : nan
Yeh Saali Aashiqui : nan
Dil Bechara : nan
Pati Patni Aur Woh : nan
Commando 3 : nan
Mardaani 2 : nan
Dabangg 3 : nan
Good Newwz : nan
Daaka : 1 November 2019 (USA)
```

In [62]:
```python
import pandas as pd
dataset=pd.read_csv('kohli_ipl.csv').shape[0]
print(dataset)
```

```
215
```

In [64]:
```
import pandas as pd
dataset=pd.read_csv('kohli_ipl.csv',index_col='match_no',squeeze=True)
print(dataset)
```

```
match_no
1        1
2       23
3       13
4       12
5        1
        ..
211      0
212     20
213     73
214     25
215      7
Name: runs, Length: 215, dtype: int64
```

In [72]:
```
kohli=dataset.sort_values(ascending=True)
print(kohli.head(10))
```

```
match_no
87     0
211    0
207    0
206    0
91     0
93     0
8      0
130    0
135    0
106    1
Name: runs, dtype: int64
```

In [73]:
```
kohli=dataset.sort_values(ascending=False)
print(kohli.head(10))
```

```
match_no
128    113
126    109
123    108
120    100
164    100
82      99
81      93
145     92
178     90
160     84
Name: runs, dtype: int64
```

In [76]:
```
kohli[dataset.values==0].shape[0]
```

Out[76]:
```
9
```

In [80]:
```
kohli[(dataset.values>90) & (dataset.value<100)].shape[0]
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-80-e42605e4eaf6> in <module>
----> 1 kohli[(dataset.values>90) & (dataset.value<100)].shape[0]

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py in __getattr__(sel
f, name)
   5137             if self._info_axis._can_hold_identifiers_and_holds_name(name):
   5138                 return self[name]
-> 5139             return object.__getattribute__(self, name)
```

```
   5140
   5141     def __setattr__(self, name: str, value) -> None:

AttributeError: 'Series' object has no attribute 'value'
```

In [81]:
```
import pandas as pd
dataset=pd.read_csv('ipl-matches.csv')
print(dataset)
```

```
          ID        City        Date   Season  MatchNumber  \
0    1312200   Ahmedabad  2022-05-29     2022        Final
1    1312199   Ahmedabad  2022-05-27     2022  Qualifier 2
2    1312198     Kolkata  2022-05-25     2022   Eliminator
3    1312197     Kolkata  2022-05-24     2022  Qualifier 1
4    1304116      Mumbai  2022-05-22     2022           70
..       ...         ...         ...      ...          ...
945   335986     Kolkata  2008-04-20  2007/08            4
946   335985      Mumbai  2008-04-20  2007/08            5
947   335984       Delhi  2008-04-19  2007/08            3
948   335983  Chandigarh  2008-04-19  2007/08            2
949   335982   Bangalore  2008-04-18  2007/08            1

                           Team1                        Team2  \
0               Rajasthan Royals               Gujarat Titans
1    Royal Challengers Bangalore             Rajasthan Royals
2    Royal Challengers Bangalore        Lucknow Super Giants
3               Rajasthan Royals               Gujarat Titans
4            Sunrisers Hyderabad                 Punjab Kings
..                           ...                          ...
945        Kolkata Knight Riders              Deccan Chargers
946               Mumbai Indians  Royal Challengers Bangalore
947             Delhi Daredevils             Rajasthan Royals
948              Kings XI Punjab          Chennai Super Kings
949  Royal Challengers Bangalore        Kolkata Knight Riders

                                        Venue                   TossWinner  \
0             Narendra Modi Stadium, Ahmedabad             Rajasthan Royals
1             Narendra Modi Stadium, Ahmedabad             Rajasthan Royals
2                        Eden Gardens, Kolkata         Lucknow Super Giants
3                        Eden Gardens, Kolkata               Gujarat Titans
4                     Wankhede Stadium, Mumbai          Sunrisers Hyderabad
..                                        ...                          ...
945                              Eden Gardens              Deccan Chargers
946                          Wankhede Stadium               Mumbai Indians
947                          Feroz Shah Kotla             Rajasthan Royals
948  Punjab Cricket Association Stadium, Mohali          Chennai Super Kings
949                     M Chinnaswamy Stadium  Royal Challengers Bangalore

    TossDecision SuperOver                  WinningTeam    WonBy  Margin  \
0            bat         N               Gujarat Titans  Wickets     7.0
1          field         N             Rajasthan Royals  Wickets     7.0
2          field         N  Royal Challengers Bangalore     Runs    14.0
3          field         N               Gujarat Titans  Wickets     7.0
4            bat         N                 Punjab Kings  Wickets     5.0
..           ...       ...                          ...      ...     ...
945          bat         N        Kolkata Knight Riders  Wickets     5.0
946          bat         N  Royal Challengers Bangalore  Wickets     5.0
947          bat         N             Delhi Daredevils  Wickets     9.0
948          bat         N          Chennai Super Kings     Runs    33.0
949        field         N        Kolkata Knight Riders     Runs   140.0

    method Player_of_Match                                       Team1Players  \
0      NaN        HH Pandya  ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D ...
1      NaN       JC Buttler  ['V Kohli', 'F du Plessis', 'RM Patidar', 'GJ ...
2      NaN       RM Patidar  ['V Kohli', 'F du Plessis', 'RM Patidar', 'GJ ...
3      NaN        DA Miller  ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D ...
4      NaN    Harpreet Brar  ['PK Garg', 'Abhishek Sharma', 'RA Tripathi', ...
..     ...             ...                                                ...
945    NaN        DJ Hussey  ['WP Saha', 'BB McCullum', 'RT Ponting', 'SC G...
```

```
946    NaN      MV Boucher  ['L Ronchi', 'ST Jayasuriya', 'DJ Thornely', '...
947    NaN     MF Maharoof  ['G Gambhir', 'V Sehwag', 'S Dhawan', 'MK Tiwa...
948    NaN      MEK Hussey  ['K Goel', 'JR Hopes', 'KC Sangakkara', 'Yuvra...
949    NaN     BB McCullum  ['R Dravid', 'W Jaffer', 'V Kohli', 'JH Kallis...

                                         Team2Players       Umpire1  \
0    ['WP Saha', 'Shubman Gill', 'MS Wade', 'HH Pan...    CB Gaffaney
1    ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D ...    CB Gaffaney
2    ['Q de Kock', 'KL Rahul', 'M Vohra', 'DJ Hooda...  J Madanagopal
3    ['WP Saha', 'Shubman Gill', 'MS Wade', 'HH Pan...   BNJ Oxenford
4    ['JM Bairstow', 'S Dhawan', 'M Shahrukh Khan',...   AK Chaudhary
..                                                 ...            ...
945  ['AC Gilchrist', 'Y Venugopal Rao', 'VVS Laxma...      BF Bowden
946  ['S Chanderpaul', 'R Dravid', 'LRPL Taylor', '...       SJ Davis
947  ['T Kohli', 'YK Pathan', 'SR Watson', 'M Kaif'...      Aleem Dar
948  ['PA Patel', 'ML Hayden', 'MEK Hussey', 'MS Dh...      MR Benson
949  ['SC Ganguly', 'BB McCullum', 'RT Ponting', 'D...      Asad Rauf

            Umpire2
0      Nitin Menon
1      Nitin Menon
2         MA Gough
3        VK Sharma
4     NA Patwardhan
..              ...
945     K Hariharan
946       DJ Harper
947  GA Pratapkumar
948      SL Shastri
949     RE Koertzen

[950 rows x 20 columns]
```

In [96]:
```python
dataset[dataset['SuperOver']=="Y"].shape[0]
```
Out[96]:
```
14
```

In [95]:
```python
dataset[dataset['TossWinner']==dataset['WinningTeam']].shape[0]
```
Out[95]:
```
489
```

In [ ]:
```python
#how many matches won by csk at kolkata
```

In [94]:
```python
dataset[(dataset['WinningTeam']=='Chennai Super Kings') &(dataset['City']=='Kolkata'
```
Out[94]:
```
5
```

In [ ]:
```python
#how many matches hardik pandya is player of the match vs RR
```

In [103…
```python
dataset[(dataset['Player_of_Match']=='HH Pandya') &((dataset['Team1']=='Rajasthan Ro
```
Out[103…
```
2
```

In [ ]:
```python
#How many matches Gt won the toss and Elected the bat
```

In [108…
```python
dataset[(dataset['TossWinner']=='Gujarat Titans')&(dataset['TossDecision']=='bat')].
```
Out[108…
```
4
```

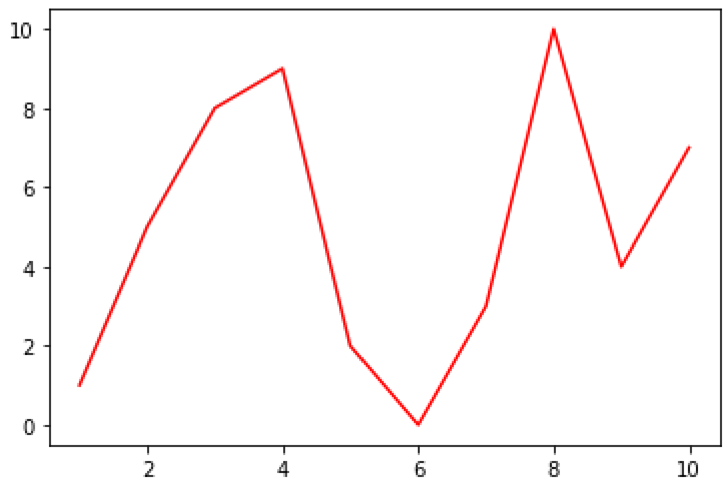
In [112…
```python
dataset[(dataset['WinningTeam']=='Gujarat Titans') & (dataset['SuperOver']=='Y')].sh
```
Out[112…
```
0
```

In [118…
```
dataset["WinningTeam"].value_counts()
```

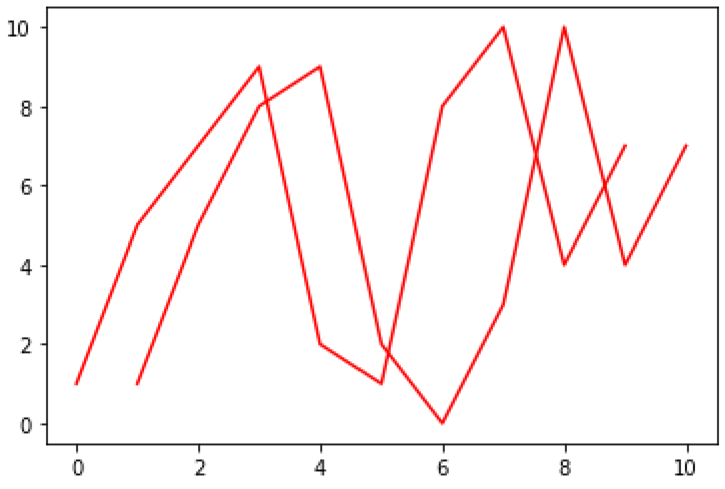
Out[118…
```
Mumbai Indians                 131
Chennai Super Kings            121
Kolkata Knight Riders          114
Royal Challengers Bangalore    109
Rajasthan Royals                96
Kings XI Punjab                 88
Sunrisers Hyderabad             75
Delhi Daredevils                67
Delhi Capitals                  36
Deccan Chargers                 29
Gujarat Lions                   13
Punjab Kings                    13
Pune Warriors                   12
Gujarat Titans                  12
Rising Pune Supergiant          10
Lucknow Super Giants             9
Kochi Tuskers Kerala             6
Rising Pune Supergiants          5
Name: WinningTeam, dtype: int64
```

ch:2 Data visulization with python

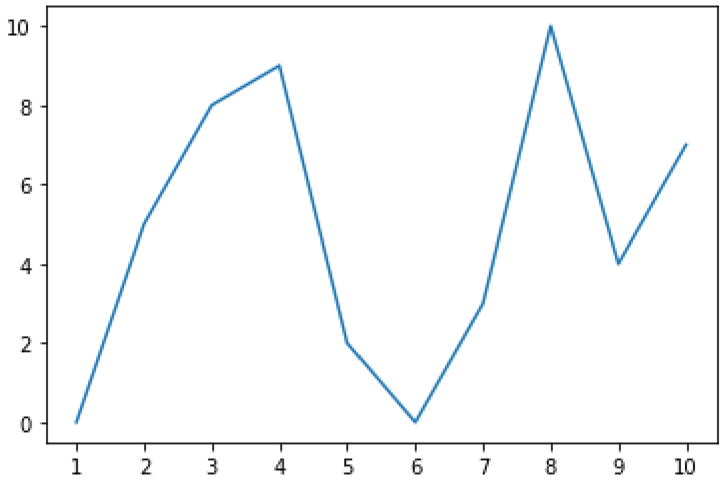
In [1]:
```
import matplotlib.pyplot as plt
value=[1,5,8,9,2,0,3,10,4,7]
x=range(1,11)
plt.plot(x,value,color='r')
plt.show()
```



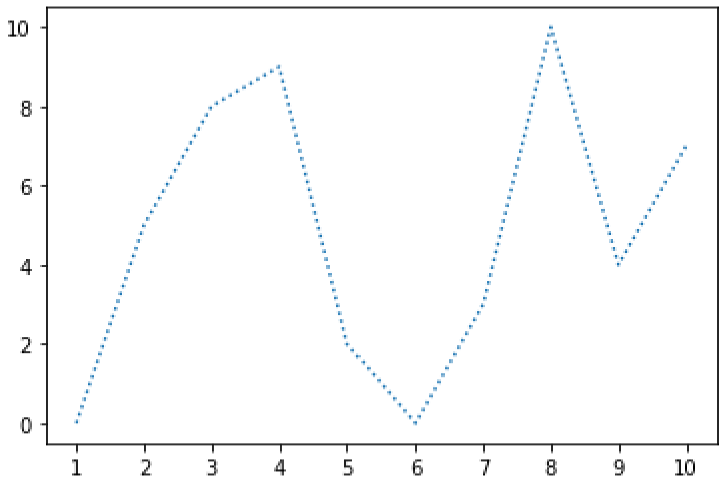
In [4]:
```
import matplotlib.pyplot as plt
value=[1,5,8,9,2,0,3,10,4,7]
value1=[1,5,7,9,2,1,8,10,4,7]
x=range(1,11)
plt.plot(x,value,value1,color='r')
plt.savefig('myplot.png',format='png')
plt.show()
```



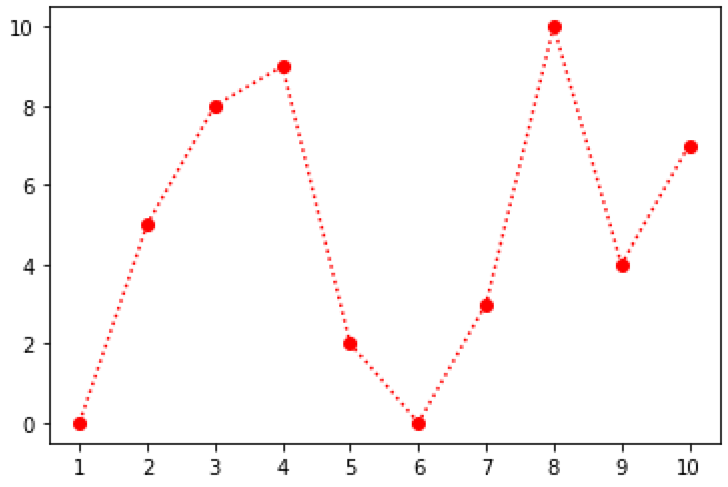
In [5]:
```python
import matplotlib.pyplot as plt
value=[0,5,8,9,2,0,3,10,4,7]
plt.xticks([1,2,3,4,5,6,7,8,9,10])
plt.xticks([0,1,2,3,4,5,6,7,8,9,10])
plt.plot(range(1,11),value)
plt.show()
```



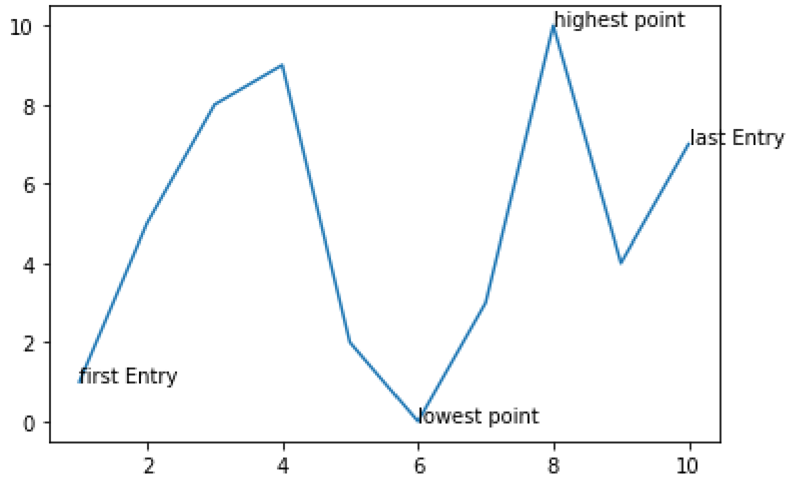
In [6]:
```python
import matplotlib.pyplot as plt
value=[0,5,8,9,2,0,3,10,4,7]
plt.xticks([1,2,3,4,5,6,7,8,9,10])
plt.xticks([0,1,2,3,4,5,6,7,8,9,10])
plt.plot(range(1,11),value,':')
plt.show()
```



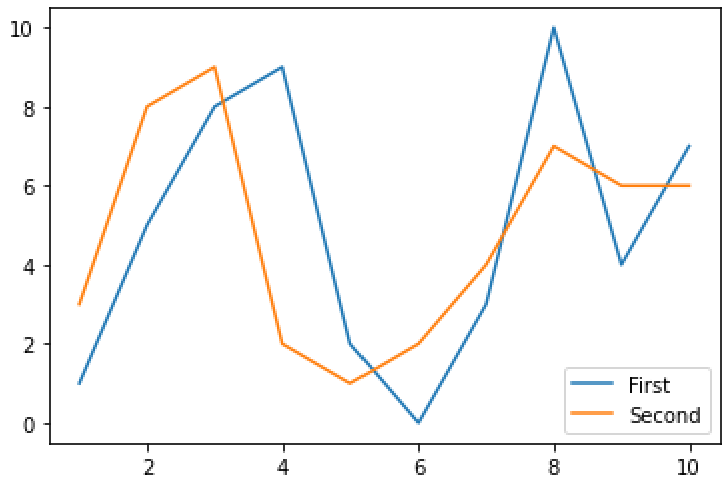
In [7]:
```
import matplotlib.pyplot as plt
value=[0,5,8,9,2,0,3,10,4,7]
plt.xticks([1,2,3,4,5,6,7,8,9,10])
plt.xticks([0,1,2,3,4,5,6,7,8,9,10])
plt.plot(range(1,11),value,'o:r')#style,marker,color
plt.show()
```



In [15]:
```
import matplotlib.pyplot as plt
value=[1,5,8,9,2,0,3,10,4,7]
x=range(1,11)
plt.annotate(text="first Entry",xy=[1,1])
plt.annotate(text="last Entry",xy=[10,7])
plt.annotate(text="lowest point",xy=[6,0])
plt.annotate(text="highest point",xy=[8,10])
plt.plot(x,value)
plt.show()
```

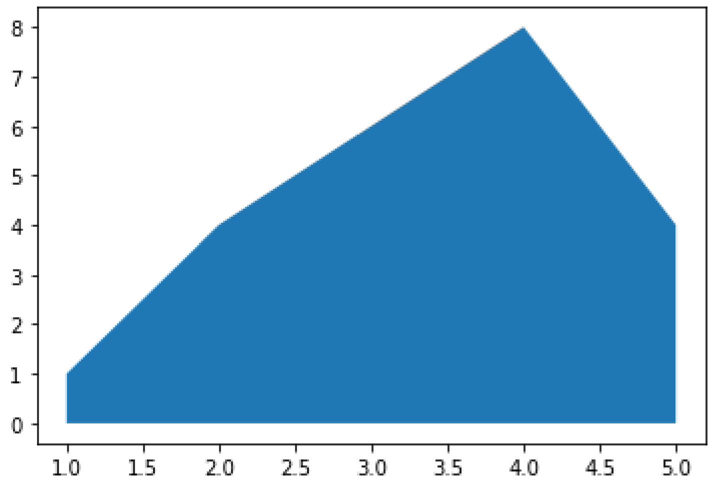


In [19]:
```
import matplotlib.pyplot as plt
y1=[1,5,8,9,2,0,3,10,4,7]
y2=[3,8,9,2,1,2,4,7,6,6]
plt.plot(range(1,11),y1,label='First')
plt.plot(range(1,11),y2,label='Second')
plt.legend(loc=0)
plt.show()
```

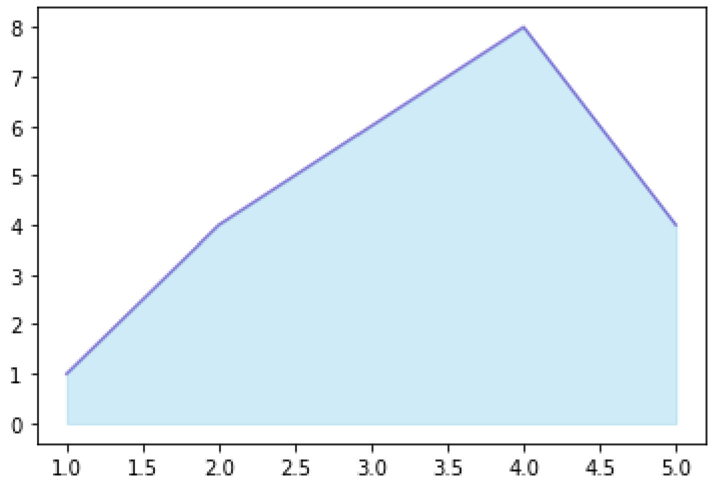


area plot

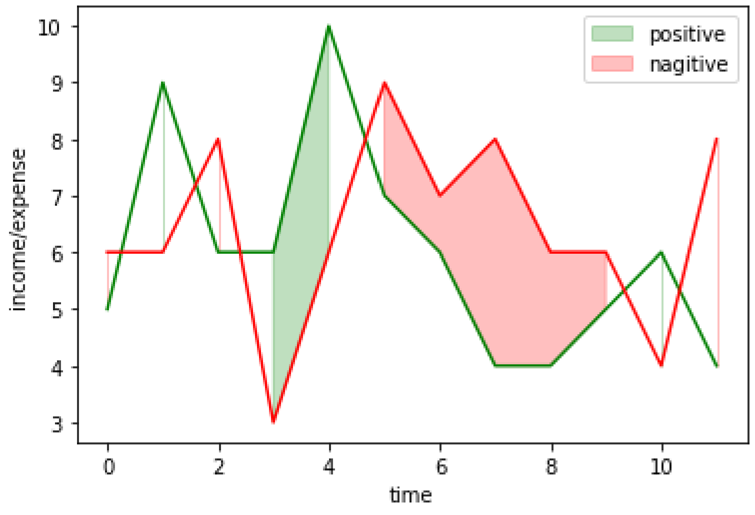
In [20]:
```
import matplotlib.pyplot as plt
x=range(1,6)
y=[1,4,6,8,4]
plt.fill_between(x,y)
plt.show()
```



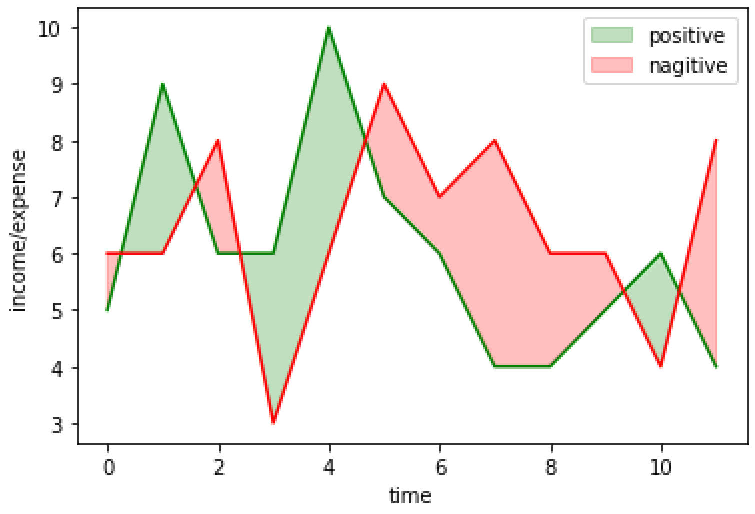
In [23]:
```
import matplotlib.pyplot as plt
x=range(1,6)
y=[1,4,6,8,4]
plt.fill_between(x,y,color='skyblue',alpha=0.4)
plt.plot(x,y,color='slateblue',alpha=0.8)
plt.show()
```



In [25]:
```
import matplotlib.pyplot as plt
import numpy as np
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])
plt.plot(time,income,color='green')
plt.plot(time,expense,color='red')
plt.fill_between(time,income,expense,where=(income>expense),color='green',alpha=0.25
plt.fill_between(time,income,expense,where=(income<=expense),color='red',alpha=0.25,
plt.xlabel('time')
plt.ylabel('income/expense')
plt.legend()
plt.show()
```

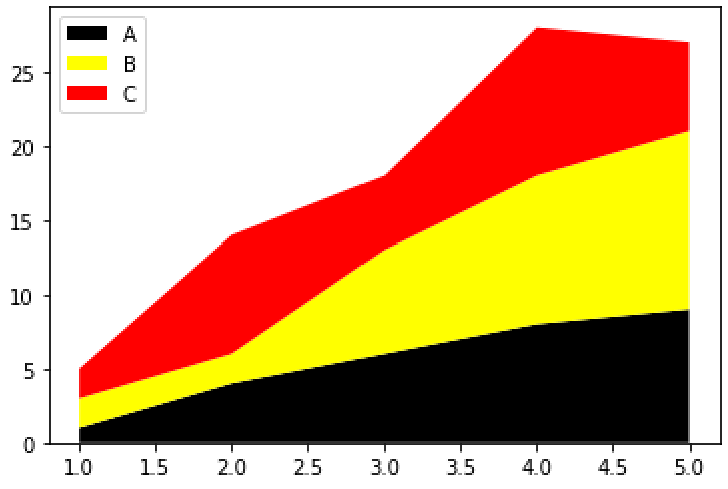


In [26]:
```
import matplotlib.pyplot as plt
import numpy as np
time=np.arange(12)
income=np.array([5,9,6,6,10,7,6,4,4,5,6,4])
expense=np.array([6,6,8,3,6,9,7,8,6,6,4,8])
plt.plot(time,income,color='green')
plt.plot(time,expense,color='red')
plt.fill_between(time,income,expense,where=(income>expense),color='green',alpha=0.25
plt.fill_between(time,income,expense,where=(income<=expense),color='red',alpha=0.25,
plt.xlabel('time')
plt.ylabel('income/expense')
plt.legend()
plt.show()
```

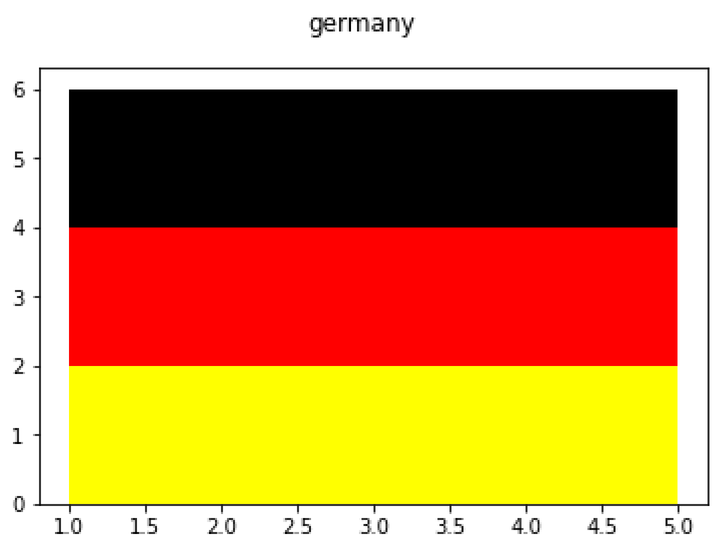


stacked area chart

In [32]:
```
import matplotlib.pyplot as plt
import numpy as np
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=['black','yellow','red'])
plt.legend(loc='upper left')
plt.show()
```

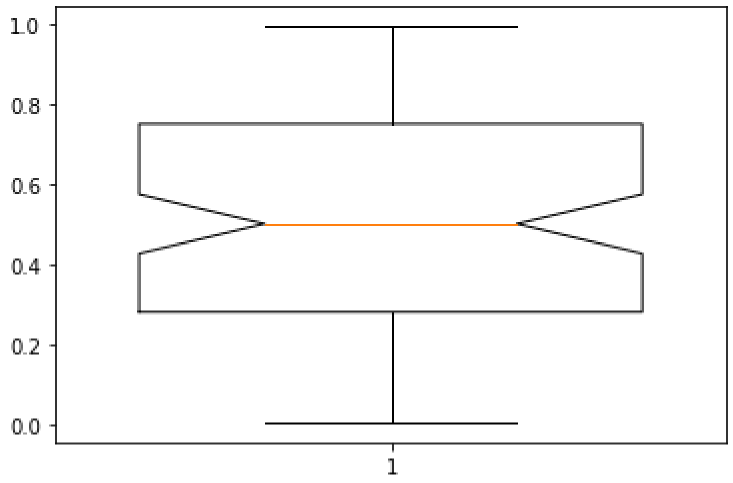


In [38]:
```
import matplotlib.pyplot as plt
import numpy as np
x=range(1,6)
y1=[2,2,2,2,2]
y2=[2,2,2,2,2]
y3=[2,2,2,2,2]

plt.stackplot(x,y1,y2,y3,labels=['A','B','C'],colors=['yellow','red','black'])
plt.suptitle('germany')

plt.show()
```

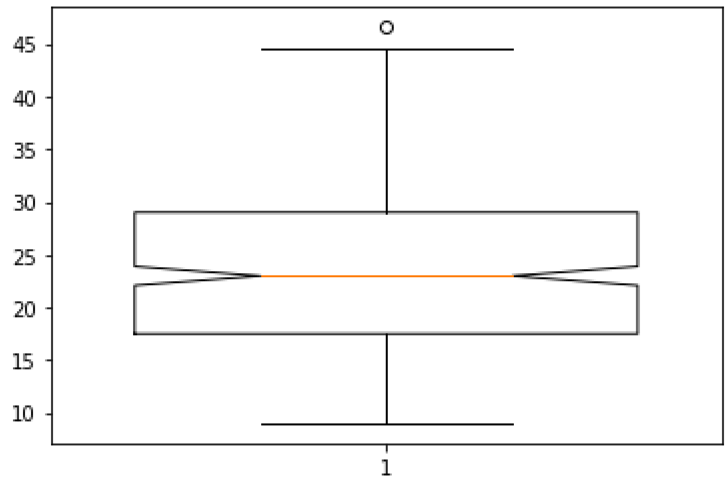


In [40]:
```
import matplotlib.pyplot as plt
import numpy as np
data=np.random.rand(100)
print(data)
plt.boxplot(data,widths=0.75,notch=True)
plt.show()
```

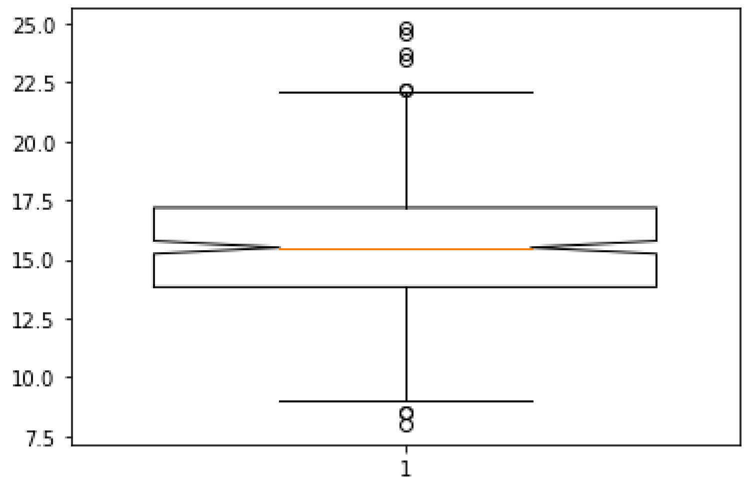
```
[0.3646308  0.60065764 0.98242585 0.18830324 0.99333838 0.5007181
 0.74764044 0.83589133 0.01001702 0.83079648 0.28643572 0.69292427
 0.28660488 0.59134539 0.45497199 0.86965032 0.49417785 0.46868141
 0.69650201 0.85669651 0.53509704 0.59187179 0.53681985 0.45339933
 0.11463333 0.47673287 0.84242136 0.55701133 0.88371532 0.09300833
 0.98658744 0.90414247 0.04357474 0.09378105 0.37706368 0.74153241
 0.02216007 0.28495799 0.18840828 0.55764622 0.28009012 0.80759773
 0.57682436 0.44653018 0.34421827 0.68000765 0.37207577 0.29211619
 0.81861166 0.38726978 0.36700268 0.55069575 0.95110265 0.43464233
 0.73779711 0.90323291 0.86430012 0.64432504 0.94727844 0.98149928
 0.01731691 0.90717738 0.38100296 0.45188167 0.2004734  0.12913297
 0.12466987 0.80977364 0.14084099 0.03334463 0.86876745 0.28191712
 0.0096459  0.75754442 0.33860226 0.77052121 0.17208638 0.18023365
 0.43718688 0.69612508 0.66609744 0.90441525 0.50012044 0.30038328
 0.81033879 0.5538235  0.37751706 0.07587139 0.0306985  0.27491384
 0.57766275 0.16727594 0.19606314 0.00550521 0.64764763 0.24233253
 0.63608717 0.58094779 0.7650044  0.69866742]
```



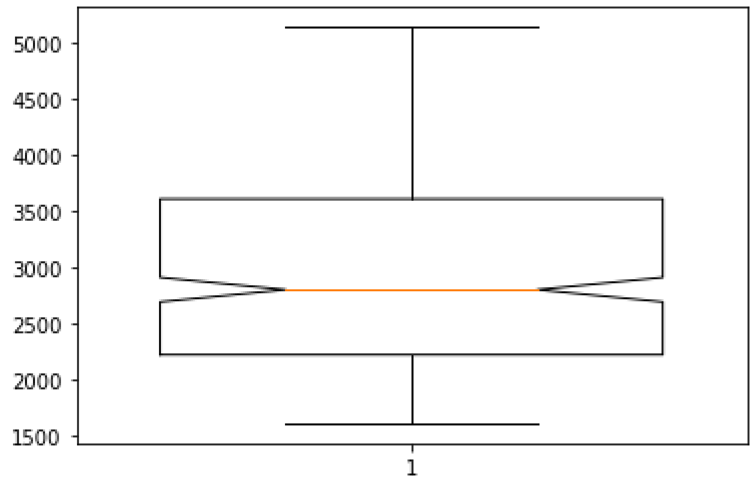
In [42]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
plt.boxplot(dataset['mpg'],widths=0.75,notch=True)
plt.show()
```



In [43]:
```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
plt.boxplot(dataset['acceleration'],widths=0.75,notch=True)
plt.show()
```

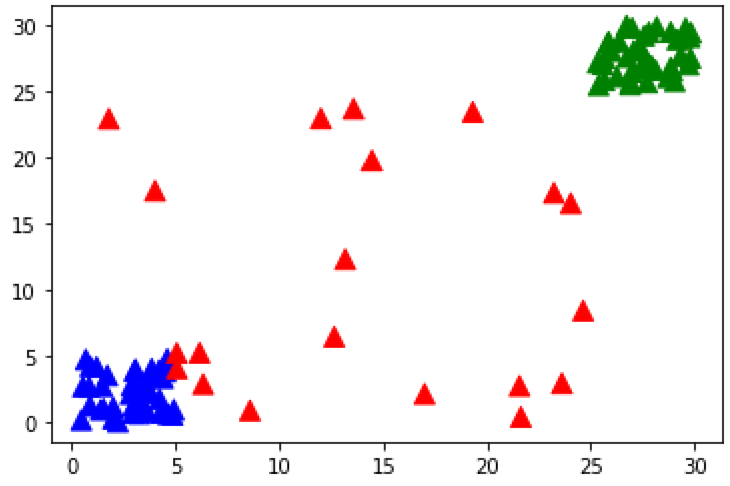


In [44]:
```python
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
dataset=pd.read_csv('auto-mpg.csv')
plt.boxplot(dataset['weight'],widths=0.75,notch=True)
plt.show()
```



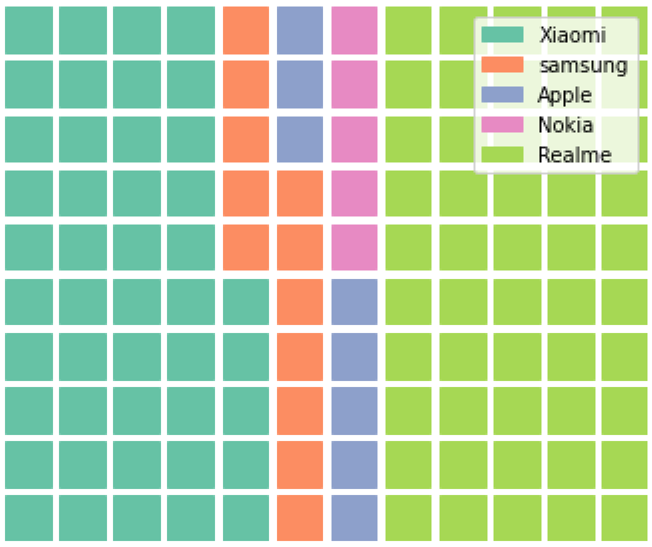
In [58]:
```
import matplotlib.pyplot as plt
import numpy as np
x1 = 5 * np.random.rand(40)
x2 = 5 * np.random.rand(40) + 25
x3 = 25 * np.random.rand(20)
x = np.concatenate((x1, x2, x3))
y1 = 5 * np.random.rand(40)
y2 = 5 * np.random.rand(40) + 25
y3 = 25 * np.random.rand(20)
color_array=['b']*40+['g']*40+['r']*20
y = np.concatenate((y1, y2, y3))
plt.scatter(x, y, s=100, marker='^', c=color_array)
print(color_array)
plt.show()
```

```
['b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b',
'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b', 'b',
'b', 'b', 'b', 'b', 'b', 'b', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g',
'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g',
'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'g', 'r', 'r', 'r', 'r', 'r',
'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r', 'r']
```



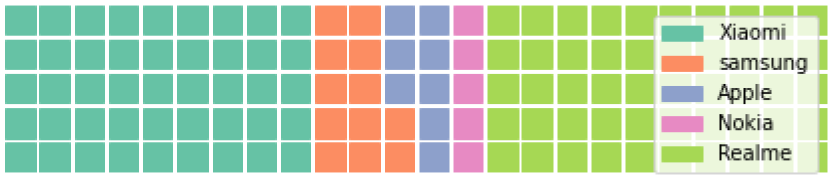
waffle chart

In [66]:
```
import pandas as pd
import matplotlib.pyplot as plt
from pywaffle import Waffle
data={'phone':['Xiaomi','samsung','Apple','Nokia','Realme'],'Stock':[45,12,8,5,50]}
df=pd.DataFrame(data)
fig=plt.figure(FigureClass=Waffle,rows=10,values=df.Stock,labels=list(df.phone))
plt.show()
```



In [ ]:

In [69]:
```
import pandas as pd
import matplotlib.pyplot as plt
from pywaffle import Waffle
data={'phone':['Xiaomi','samsung','Apple','Nokia','Realme'],'Stock':[45,12,8,5,50]}
df=pd.DataFrame(data)
fig=plt.figure(FigureClass=Waffle,rows=5,values=df.Stock,labels=list(df.phone))
plt.show()
```



In [85]:
```
from wordcloud import WordCloud,STOPWORDS
alice_noval=open('alice_in_wonderland.txt','r')
alice_noval=alice_noval.read()
stopwords=set(STOPWORDS)
print(stopwords)
print(len(stopwords))
```

```
{'should', 'few', 'herself', "here's", "couldn't", "he'd", 'shall', "wouldn't", 'fro
m', 'further', 'him', "why's", 'whom', 'by', "how's", "shan't", "we've", 'only', 'ge
t', 'over', 'my', "they'll", 'nor', 'otherwise', 'such', 'but', 'having', 'that', "yo
u'd", "haven't", 'doing', 'same', "there's", "i've", 'am', 'they', "they'd", 'yoursel
ves', 'an', 'myself', 'she', 'else', "what's", 'why', 'some', "we'll", 'if', 'i', 'do
es', "who's", "didn't", 'r', "where's", "doesn't", 'did', 'ourselves', 'is', 'canno
t', 'under', 'both', 'this', 'can', 'therefore', 'a', 'with', 'ours', 'at', 'which',
'com', 'been', "shouldn't", 'more', 'hers', 'just', 'out', 'these', 'and', 'being',
'itself', 'are', 'you', 'while', 'no', 'of', 'where', "won't", 'be', "i'll", "isn't",
'again', 'all', "she's", 'down', "i'm", 'into', 'through', "wasn't", 'ought', "you'v
e", 'before', 'as', "it's", 'our', 'their', 'then', "she'd", 'when', 'what', 'not',
"you'll", 'was', 'or', 'so', 'also', "mustn't", 'the', "they're", 'against', "we'd",
'since', 'however', "can't", 'off', 'other', 'once', 'below', 'how', "he'll", 'betwee
n', "let's", 'themselves', 'those', 'because', 'each', 'ever', 'to', 'his', 'www', 'o
wn', "hasn't", 'up', 'very', "weren't", 'do', 'until', 'too', 'after', "we're", 'thei
rs', 'for', 'above', 'could', 'them', "they've", "he's", 'here', 'any', "i'd", 'abou
t', "aren't", 'who', 'your', 'http', 'me', 'yourself', 'have', 'in', 'himself', 'he
r', 'on', 'during', "that's", 'we', 'than', 'yours', "don't", 'k', 'there', 'would',
"when's", 'like', 'hence', 'has', 'were', "you're", 'its', 'had', 'he', 'most', "sh
e'll", "hadn't", 'it'}
192
```

Folium Map

In [2]:
```python
import folium
world_map=folium.Map()
world_map
```

Out[2]: Make this Notebook Trusted to load map: File -> Trust Notebook

+

−

🇺🇦 Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [6]:
```python
world_map=folium.Map(
location=[56.130,-106.35],
zoom_start=3)
world_map
```

Out[6]: Make this Notebook Trusted to load map: File -> Trust Notebook

+

−

🇺🇦 Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [8]:
```python
world_map=folium.Map(
location=[37.7749,-122.4194],
zoom_start=3,title='stamentoner')
world_map
```

Out[8]: Make this Notebook Trusted to load map: File -> Trust Notebook

+
−

Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [13]:
```
import pandas as pd
fs=pd.read_csv('Police_Department_Incidents_-_Previous_Year__2016_.csv')
fs.head()
```

Out[13]:

	IncidntNum	Category	Descript	DayOfWeek	Date	Time	PdDistrict	Resolution	
0	120058272	WEAPON LAWS	POSS OF PROHIBITED WEAPON	Friday	01/29/2016 12:00:00 AM	11:00	SOUTHERN	ARREST, BOOKED	8
1	120058272	WEAPON LAWS	FIREARM, LOADED, IN VEHICLE, POSSESSION OR USE	Friday	01/29/2016 12:00:00 AM	11:00	SOUTHERN	ARREST, BOOKED	8
2	141059263	WARRANTS	WARRANT ARREST	Monday	04/25/2016 12:00:00 AM	14:59	BAYVIEW	ARREST, BOOKED	I S
3	160013662	NON-CRIMINAL	LOST PROPERTY	Tuesday	01/05/2016 12:00:00 AM	23:50	TENDERLOIN	NONE	J O
4	160002740	NON-CRIMINAL	LOST PROPERTY	Friday	01/01/2016 12:00:00 AM	00:30	MISSION	NONE	1 I

In [15]:
```
fs.shape
```

Out[15]:
```
(150500, 13)
```

In [21]:
```
fd=fs.iloc[0:100]
```

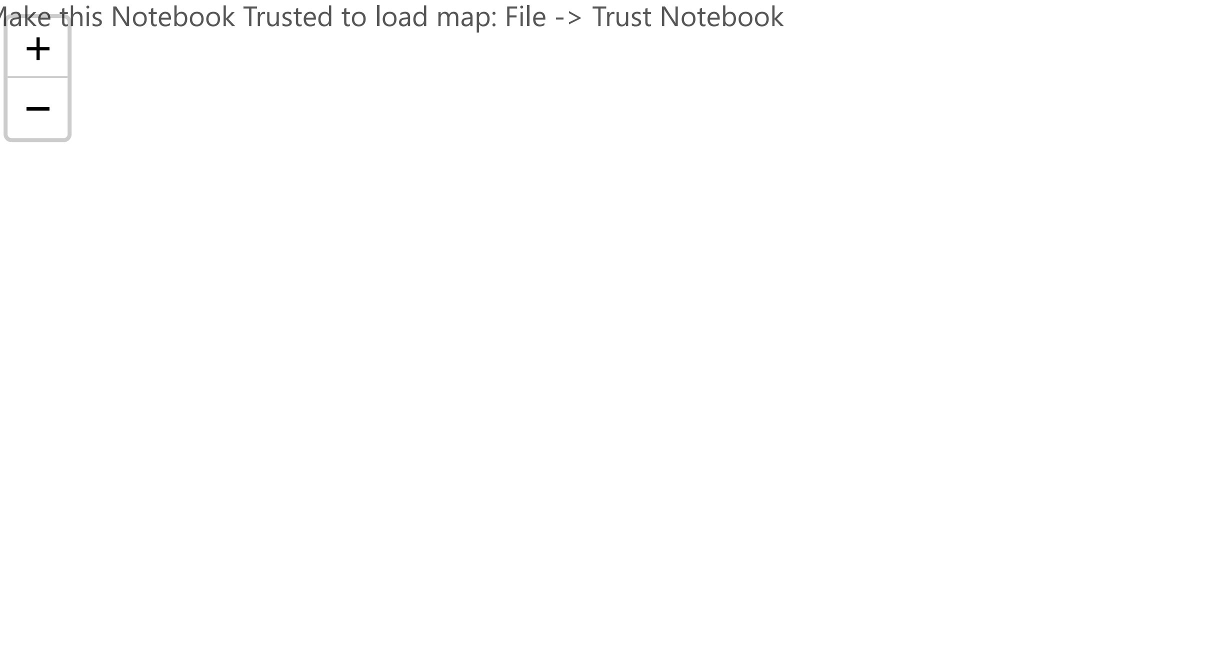
In [22]:
```
fd.shape
```

Out[22]:
```
(100, 13)
```

In [23]:
```
latitude=37.77
longitude=-122.42
```

In [25]:
```
import folium
sanfran_map=folium.Map(location=[latitude,longitude],zoom_start=120)
sanfran_map
# sanfran_map.save('sanfran.html') if you want html file
```

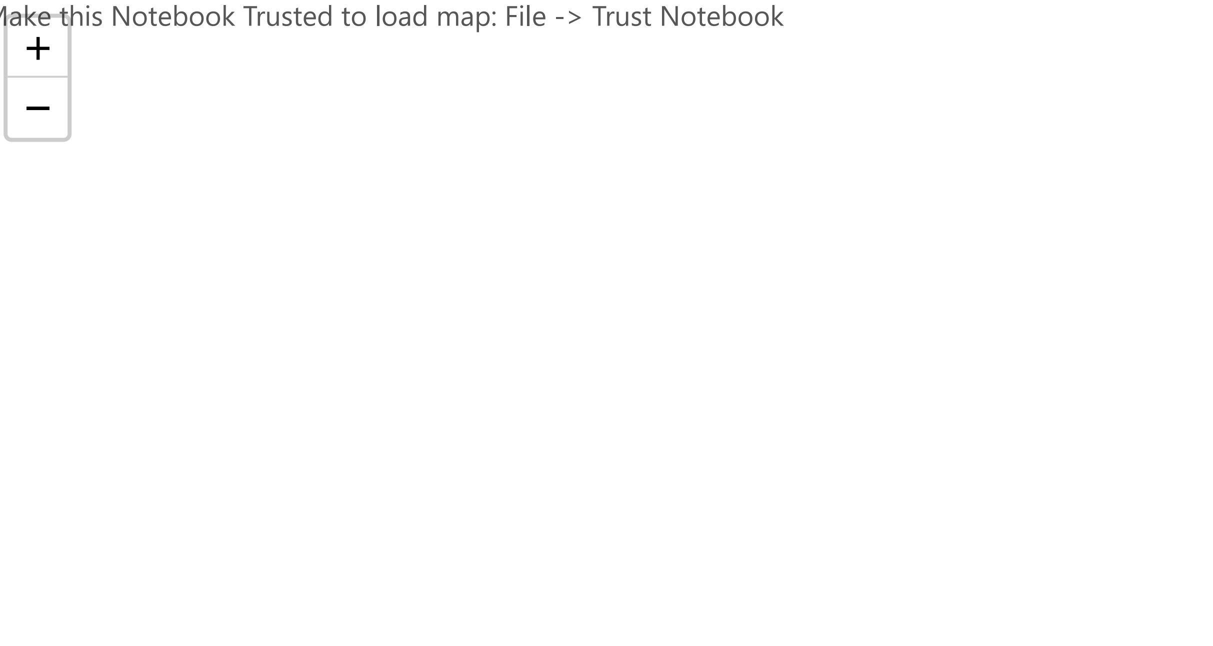
Out[25]: Make this Notebook Trusted to load map: File -> Trust Notebook



Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [40]:
```
incident=folium.map.FeatureGroup()
for latitude,longitude,in zip(fd.Y,fd.X):
    incident.add_child(
    folium.CircleMarker([latitude,longitude],radius=5,color='yellow',fill=True, fill
sanfran_map.add_child(incident)
```

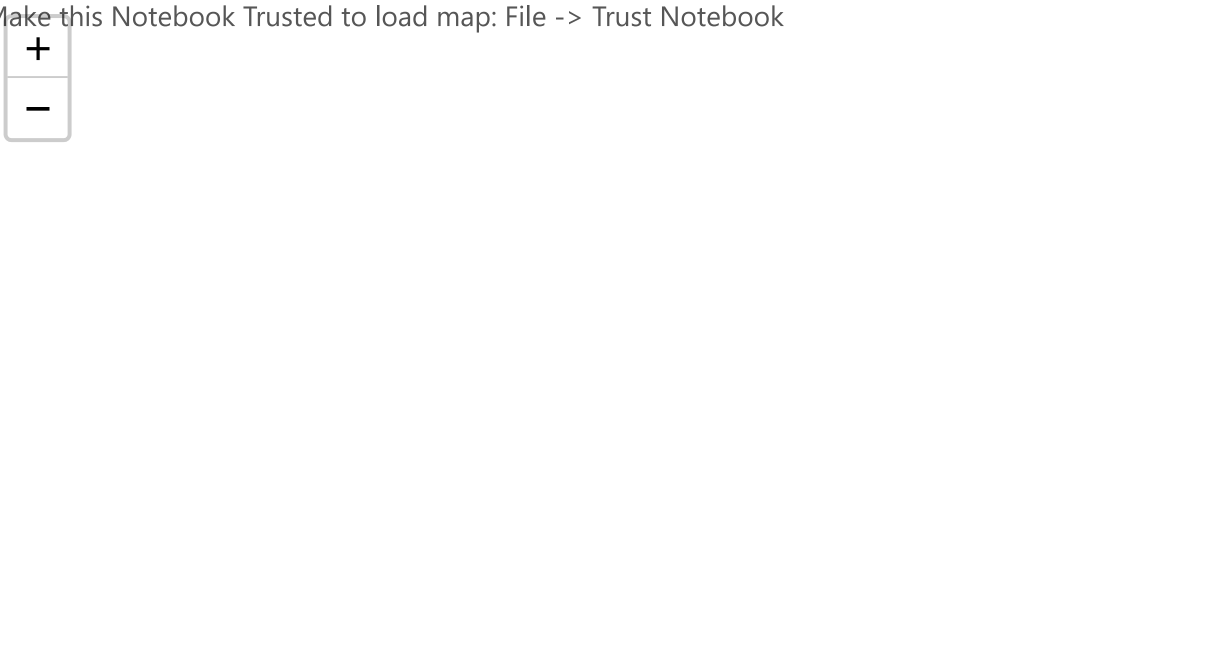
Out[40]: Make this Notebook Trusted to load map: File -> Trust Notebook



Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [41]:
```
incident=folium.map.FeatureGroup()
for latitude,longitude,labels in zip(fd.Y,fd.X,fs.Category):
    incident.add_child(
    folium.CircleMarker([latitude,longitude],radius=5,color='yellow',fill=True, fill
    folium.Marker([latitude,longitude],popup=labels).add_to(sanfran_map)
sanfran_map.add_child(incident)
```

Out[41]: Make this Notebook Trusted to load map: File -> Trust Notebook

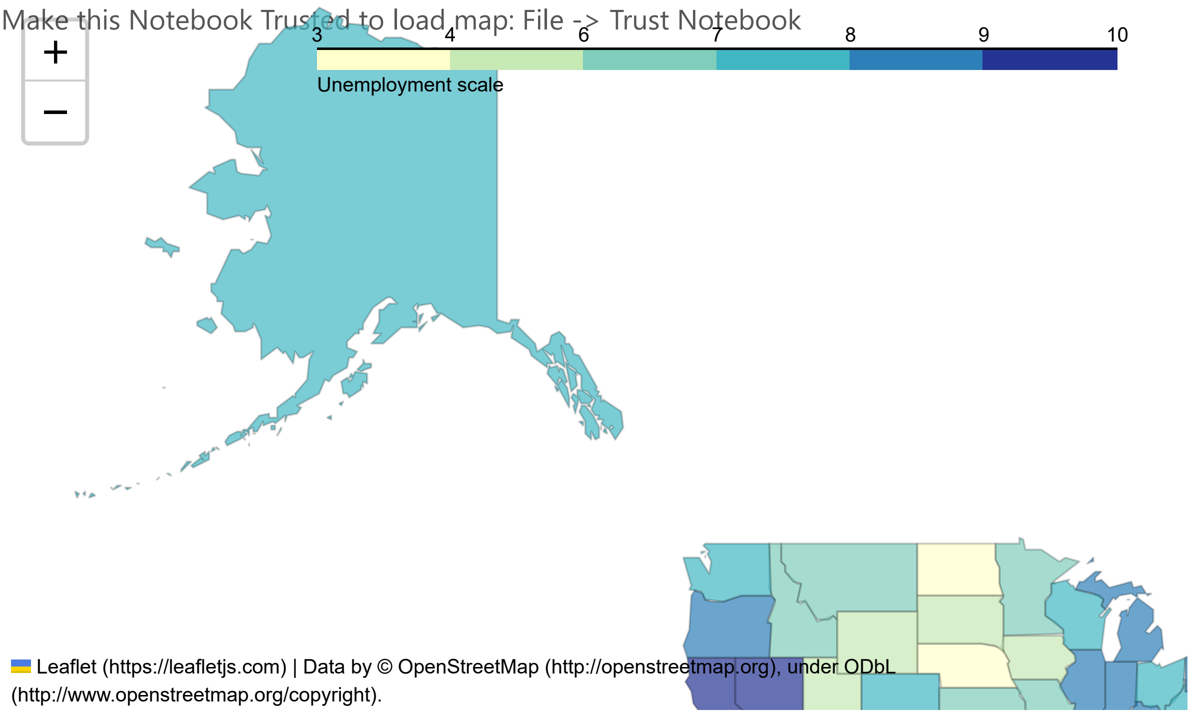


Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

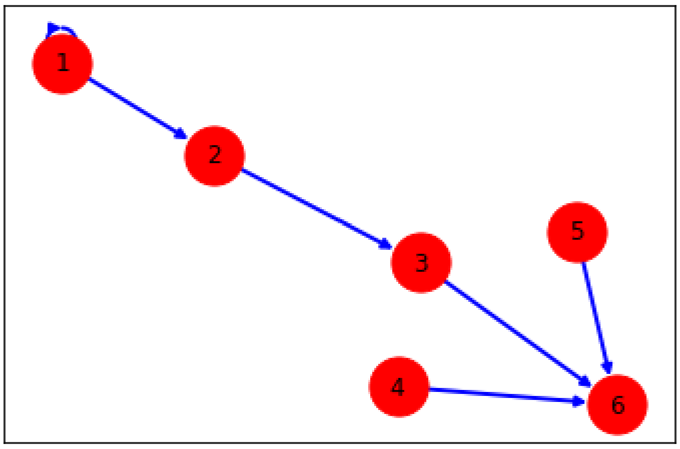
In [57]:
```
import pandas as pd
state_unemp=pd.read_csv('US_Unemployment_Oct2012.csv')
state_geo='us-states.json'
```

In [60]:
```
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',
    line_opacity=0.2,
    fill_opacity=0.7,
    legend_name='Unemployment scale'
    ).add_to(usa_state)

usa_state
```

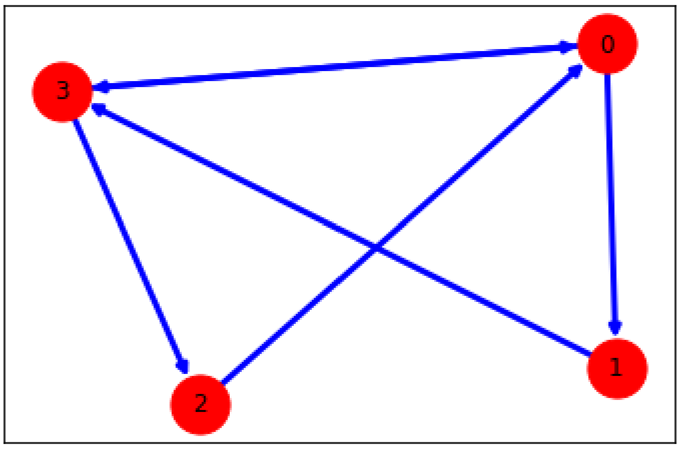
Out[60]: Make this Notebook Trusted to load map: File -> Trust Notebook


+
−
3
4
6
7
8
9
10
Unemployment scale
Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

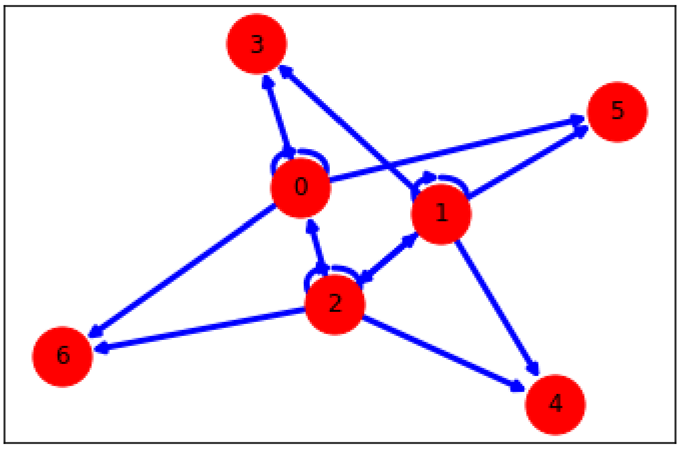
In [4]:
```
import networkx as nx
import matplotlib.pyplot as plt
G=nx.DiGraph()
G.add_node(1)
G.add_nodes_from([2,3])
G.add_nodes_from(range(4,7))

G.add_edge(1,2)
G.add_edge(1,1)
G.add_edges_from([(2,3),(3,6),(4,6),(5,6)])

nx.draw_networkx(G,node_size=850,node_color='red',width=2,edge_color='blue')
plt.show()
```



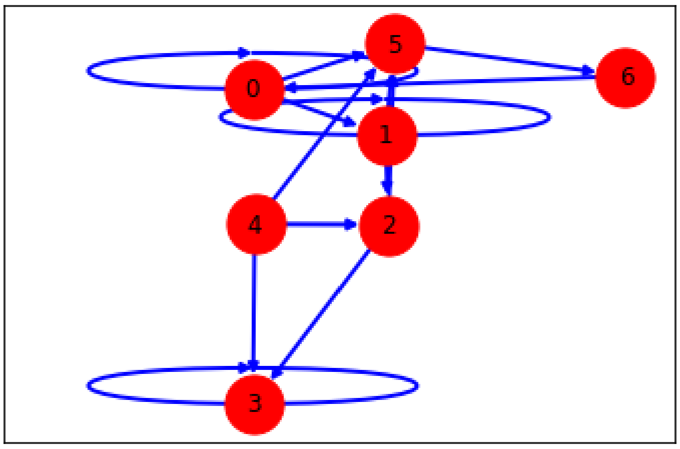
In [10]:
```
import matplotlib.pyplot as plt
import networkx as nx

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=850,node_color='red',edge_color='blue',width=3)
```



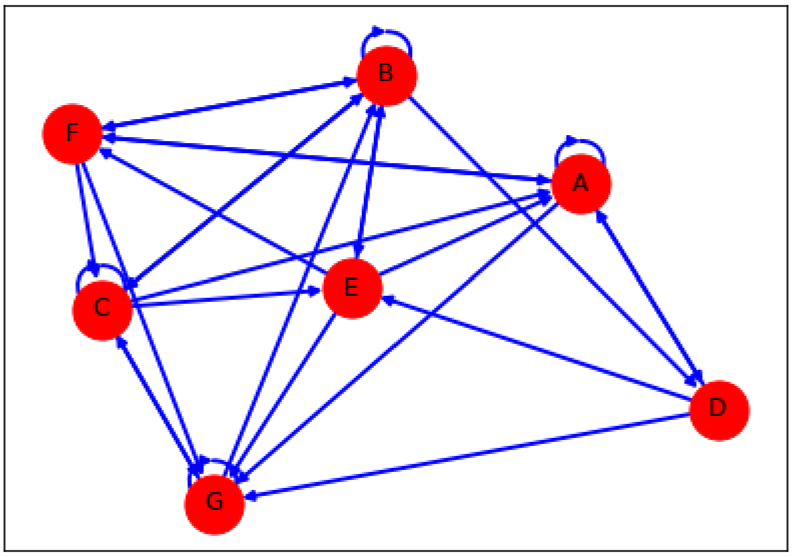
In [14]:
```
import matplotlib.pyplot as plt
import networkx as nx

G=nx.DiGraph()
G.add_nodes_from([0,1,2])
L=[[1,0,0,1,0,1,1],[0,1,1,1,1,1,0],[1,1,1,0,1,0,1]]
for i in range(3):
    for j in range(7):
        if L[i][j]==1:
            G.add_edge(i,j)
nx.draw_networkx(G,node_size=850,node_color='red',edge_color='blue',width=3)
plt.show()
```



In [51]:
```
import matplotlib.pyplot as plt
import networkx as nx

G=nx.DiGraph()
G.add_nodes_from([0,1,2,4,5,6])
G.add_edges_from([(0,0),(0,1),(1,1),(1,2),(2,3),(2,5),(3,3),(4,3),(4,2),(4,5),(5,6),
nx.draw_networkx(G,node_size=850,node_color='red',edge_color='blue',width=2)
plt.show()
```



In [54]:
```
import matplotlib.pyplot as plt
import networkx as nx
plt.figure(figsize=[7,5])
G=nx.DiGraph()
L={"A":[1,0,0,1,0,1,1],"B":[0,1,1,1,1,1,0],"C":[1,1,1,0,1,0,1],"D":[1,0,0,0,1,0,1],"
G.add_nodes_from(L.keys())
for i,j in enumerate(L):
    for k,l in enumerate(L[j]):
        if l:
            G.add_edge(j,list(L.keys())[k])

nx.draw_networkx(G,node_size=850,node_color="red",edge_color="blue",width=2)
plt.show()
```



ch-3 Regular Expression

In [59]:
```
import re
txt=" The rain in spain"
x=re.findall("ai",txt)
print(x)
```

```
['ai', 'ai']
```

In [62]:
```
import re
txt=" The rain in spain"
x=re.findall('[arn]',txt)
print(x)
```

```
['r', 'a', 'n', 'n', 'a', 'n']
```

In [63]:
```
import re
txt=" The rain in spain"
x=re.findall('[^arn]',txt)
print(x)
```

```
[' ', 'T', 'h', 'e', ' ', 'i', ' ', 'i', ' ', 's', 'p', 'i']
```

In [64]:
```
import re
txt=" The rain in spain"
x=re.findall('paris',txt)
print(x)
```

```
[]
```

In [66]:
```
import re
txt=" The rain in spain"
x=re.findall('[a-m]',txt)
print(x)
```

```
['h', 'e', 'a', 'i', 'i', 'a', 'i']
```

In [70]:
```
import re
txt=" that will be 100 dollars"
x=re.findall('\d',txt)
print(x)
```

```
['1', '0', '0']
```

In [71]:
```
import re
txt=" that will be 100 dollars"
x=re.findall('\d+',txt)
print(x)
```

```
['100']
```

In [72]:
```
import re
txt=" that will be 100 dollars 1001 100"
x=re.findall('\d+',txt)
print(x)
```

```
['100', '1001', '100']
```

In [73]:
```
import re
txt=" that will be 100 dollars 12xy3z"
x=re.findall('\d+',txt)
print(x)
```

```
['100', '12', '3']
```

In [75]:
```
import re
txt="hello planet helo"
x=re.findall('he..o',txt)
print(x)
```

```
['hello']
```

In [76]:
```
import re
txt="hello planet helo"
x=re.findall('^h.+\s',txt)
print(x)
```

```
['hello planet ']
```

In [77]:
```
import re
txt="hello planet helo"
```

```
x=re.findall('planet$',txt)
print(x)
```

```
[]
```

In [78]:
```
import re
txt="hello planet"
x=re.findall('planet$',txt)
print(x)
```

```
['planet']
```

In [82]:
```
import re
txt="hello hellar"
x=re.findall('hello.*o',txt)
print(x)
```

```
[]
```

In [83]:
```
import re
txt="hello hellaro"
x=re.findall('hello.*o',txt)
print(x)
```

```
['hello hellaro']
```

In [85]:
```
import re
txt="hello hellar"
x=re.findall('hello.+o',txt)
print(x)
```

```
[]
```

In [86]:
```
import re
txt="hello hellaro"
x=re.findall('hello.+o',txt)
print(x)
```

```
['hello hellaro']
```

In [88]:
```
import re
txt="hello hellaro"
x=re.findall('he.?o',txt)
print(x)
```

```
['hello']
```

In [89]:
```
import re
txt="hello hellaro"
x=re.findall('he.{2}o',txt)
print(x)
```

```
['hello']
```

In [90]:
```
import re
txt="The raib in spain falls in the plain"
x=re.findall('spain|plain',txt)
print(x)
```

```
['spain', 'plain']
```

In [91]:
```
import re
txt="The raib in spain falls in the plain"
x=re.findall('spain|paris',txt)
print(x)
```

```
['spain']
```

In [93]:
```
import re
txt="The raib in spain plain123"
x=re.findall('\D',txt)
print(x)
```

```
['T', 'h', 'e', ' ', 'r', 'a', 'i', 'b', ' ', 'i', 'n', ' ', 's', 'p', 'a', 'i', 'n',
' ', 'p', 'l', 'a', 'i', 'n']
```

In [94]:
```
import re
txt="The raib in spain plain123"
x=re.findall('\s',txt)
print(x)
```

```
[' ', ' ', ' ', ' ']
```

In [95]:
```
import re
txt="The raib in spain plain123"
x=re.findall('\S',txt)
print(x)
```

```
['T', 'h', 'e', 'r', 'a', 'i', 'b', 'i', 'n', 's', 'p', 'a', 'i', 'n', 'p', 'l', 'a',
'i', 'n', '1', '2', '3']
```

In [96]:
```
import re
txt="The raib in spain plain123"
x=re.findall('\w',txt)
print(x)
```

```
['T', 'h', 'e', 'r', 'a', 'i', 'b', 'i', 'n', 's', 'p', 'a', 'i', 'n', 'p', 'l', 'a',
'i', 'n', '1', '2', '3']
```

In [97]:
```
import re
txt="The raib in spain plain123"
x=re.findall('\W',txt)
print(x)
```

```
[' ', ' ', ' ', ' ']
```

In [98]:
```
import re
txt="8 times before 11:45AM"
x=re.findall('[0-9]',txt)
print(x)
```

```
['8', '1', '1', '4', '5']
```

In [104…
```
import re
txt="8 times before 11:45 AM"
x=re.findall('[0-5][0-9]',txt)
print(x)
```

```
['11', '45']
```

In [107…
```
import re
txt="08 times before 11:45 AM"
x=re.findall('[0-5][0-9]\d+',txt)
print(x)
```

```
['08', '11', '45']
```

In [109…
```
import re
txt="08 times before 11:45 AM"
x=re.findall('[\d+]',txt)
print(x)
```

```
['0', '8', '1', '1', '4', '5']
```

In [112…

```
import re
txt="The raibn in spain"
x=re.search("\s",txt)
print(x)
```

```
<re.Match object; span=(3, 4), match=' '>
```

In [113…

```
import re
txt="The raibn in spain"
x=re.search("\s",txt)
print(x.span())
```

```
(3, 4)
```

In [117…

```
import re
txt="The raibn in spain"
x=re.search("\s",txt)
print(x.span())
print(x.start())
print(x.end())
```

```
(3, 4)
3
4
```

In [118…

```
import re
txt="The    raibn in spain"
x=re.search("\s",txt)
print(x.span())
print(x.start())
print(x.end())
```

```
(3, 4)
3
4
```

In [119…

```
import re
txt="The    raibn in spain"
x=re.search("\s+",txt)
print(x.span())
print(x.start())
print(x.end())
```

```
(3, 7)
3
7
```

In [120…

```
import re
txt="The rain in spain"
x=re.search("rain",txt)
print(x)
```

```
<re.Match object; span=(4, 8), match='rain'>
```

In [121…

```
import re
txt="The rain in spain"
x=re.search("paris",txt)
print(x)
```

```
None
```

In [123…

```
import re
txt="No 7756spain 123"
x=re.search("\d+",txt)
print(x)
```

```
<re.Match object; span=(3, 7), match='7756'>
```

In [124…
```
import re
txt="No 7756spain 123"
x=re.search("\d",txt)
print(x)
```

```
<re.Match object; span=(3, 4), match='7'>
```

In [125…
```
import re
txt="python is fun"
x=re.search("^python",txt)
print(x)
```

```
<re.Match object; span=(0, 6), match='python'>
```

In [126…
```
import re
txt="python is fun"
x=re.search("^java",txt)
print(x)
```

```
None
```

In [127…
```
import re
txt="python is fun"
x=re.search("^p",txt)
print(x)
```

```
<re.Match object; span=(0, 1), match='p'>
```

In [129…
```
import re
txt="The rain in spain"
x=re.split("\s",txt)
print(x)
```

```
['The', 'rain', 'in', 'spain']
```

In [4]:
```
import re
txt='The_quick_brow@forjump#over$the$lazy&dog'
pattern='[a-zA-z]+'
x=re.split(pattern,txt)
print(x)
```

```
['', '@', '#', '$', '$', '&', '']
```

In [5]:
```
import re
txt='The quick brown for jumps over the lazy dog'
pattern='\s+\w+\s'
x=re.split(pattern,txt)
print(x)
```

```
['The', 'brown', 'jumps', 'the', 'dog']
```

In [6]:
```
import re
txt='The quick brown for jumps over the lazy dog'
pattern='\s[a-z]+\s'
x=re.split(pattern,txt)
print(x)
```

```
['The', 'brown', 'jumps', 'the', 'dog']
```

In [14]:
```
import re
txt='Twelve:8 Eighty Nine:9.'
pattern='\d'
x=re.split(pattern,txt)
print(x)
```

```
['Twelve:', ' Eighty Nine:', '.']
```

In [10]:
```
import re
txt='Twelve:8 Eighty Nine:9'
pattern='\d'
x=re.split(pattern,txt)
print(x)
```

```
['Twelve:', ' Eighty Nine:', '']
```

In [12]:
```
import re
txt='Twelve:8 Eighty Nine:89.'
pattern='\d'
x=re.split(pattern,txt)
print(x)
```

```
['Twelve:', ' Eighty Nine:', '', '.']
```

In [13]:
```
import re
txt='Twelve:8 Eighty Nine:89.'
pattern='\d+'
x=re.split(pattern,txt)
print(x)
```

```
['Twelve:', ' Eighty Nine:', '.']
```

In [25]:
```
import re
txt='The rain in ahmedabad earth'
pattern='[a|e]'
x=re.findall(pattern,txt)
print(x)
```

```
['e', 'a', 'a', 'e', 'a', 'a', 'e', 'a']
```

In [29]:
```
import re
txt='The rain in ahmedabad earth'
x=re.split('\s',txt)
print(x)
for i in x:
    if i[0]=='a' or i[0]=='e':
        print(i)
```

```
['The', 'rain', 'in', 'ahmedabad', 'earth']
ahmedabad
earth
```

In [30]:
```
import re
txt='The Rain in spain'
x=re.sub('\s','9',txt)
print(x)
```

```
The9Rain9in9spain
```

In [31]:
```
import re
txt='The Rain in spain'
x=re.sub('\s','9',txt,2)
print(x)
```

```
The9Rain9in spain
```

In [32]:
```
import re
txt='The Rain in spain'
x=re.sub('\s','9',txt,1)
print(x)
```

```
The9Rain in spain
```

In [34]:
```
import re
txt='The Rain in      spain   '
```

```
x=re.sub('\s','9',txt)
print(x)
```

```
The9Rain9in999999spain999
```

In [35]:
```
import re
txt='The Rain in      spain   '
x=re.sub('\s+',' ',txt)
print(x)
```

```
The Rain in spain
```

In [ ]:

In [43]:
```
import re
txt='96870000000 850250220250 1234568910 6354002000 63250505500 8200110220 120304050
x=re.split('\s',txt)
print(x)
for i in x:
    if len(i)==10 and re.findall("[6-9][0-9]{9}",i):
        print("valid number is: ",i)
```

```
['96870000000', '850250220250', '1234568910', '6354002000', '63250505500', '820011022
0', '1203040506', '9825000320450']
valid number is:  6354002000
valid number is:  8200110220
```

In [63]:
```
import re
url='http://www.washingtonpost.com/news/football-inslder/wp/2016/09/02/odell-backham
x=re.findall('\d{4}[/]\d{2}[/]\d{2}',url)
print(x[0])
```

```
2016/09/02
```

In [67]:
```
import re
url='http://www.washingtonpost.com/news/football-inslder/wp/2016/09/02/odell-backham
x=re.findall('[^/]\w+',url)
print(x[1]+x[2]+x[3])
```

```
www.washingtonpost.com
```

In [71]:
```
import re
url='http://www.washingtonpost.com/news/football-inslder/wp/2016/09/02/odell-backham
x=re.findall('\w{3}[.]\w+[.]\w{3}',url)
print(x[0])
```

```
www.washingtonpost.com
```

In [82]:
```
import re
txt='my email id is abc abc.def@gmail.com'
x=re.findall("\w[.]*\w+[@]\w+[.]\w{3}",txt)
print(x[0])
```

```
c.def@gmail.com
```

In [94]:
```
import re
txt='vishal21@gmail.com aryan01gmail.com jigs3@yahoo.com aakash3@gmail.com'
x=re.split('\s',txt)
print(x)

for i in x:
    if i!=10 or i=='@':
        print("Valid email")
    else:
        print("invalid email")
```

```
['vishal21@gmail.com', 'aryan01gmail.com', 'jigs3@yahoo.com', 'aakash3@gmail.com']
Valid email
Valid email
Valid email
Valid email
```

In [105…

```python
import re
txt='vishal@gmail.com aryan@gmail.com jigs3@yahoo.com aakash3@gmail.com '
x=re.findall("\[a-zA-Z]+[.]*\w+[@]\w+[.]\w{3}",txt)
print(x)
```

```
[]
```

In [107…

```python
import pandas as pd
data=pd.read_csv("car data.csv")
data.head()
```

Out[107…

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	C
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	

In [108…

```python
import pandas as pd
data=pd.read_csv("car data.csv")
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Car_Name       301 non-null    object 
 1   Year           301 non-null    int64  
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Kms_Driven     301 non-null    int64  
 5   Fuel_Type      301 non-null    object 
 6   Seller_Type    301 non-null    object 
 7   Transmission   301 non-null    object 
 8   Owner          301 non-null    int64  
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

In [109…

```python
import pandas as pd
data=pd.read_csv("car data.csv")
data.describe()
```

Out[109…

	Year	Selling_Price	Present_Price	Kms_Driven	Owner
count	301.000000	301.000000	301.000000	301.000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980	0.043189
std	2.891554	5.082812	8.644115	38886.883882	0.247915
min	2003.000000	0.100000	0.320000	500.000000	0.000000
25%	2012.000000	0.900000	1.200000	15000.000000	0.000000
50%	2014.000000	3.600000	6.400000	32000.000000	0.000000

	Year	Selling_Price	Present_Price	Kms_Driven	Owner
75%	2016.000000	6.000000	9.900000	48767.000000	0.000000
max	2018.000000	35.000000	92.600000	500000.000000	3.000000

In [112…

```
import pandas as pd
dataset=pd.read_csv('car data.csv')
def find_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]<low_val)| (ds[col]>high_val)]

    return ds
find_outliers(dataset,'Selling_Price')
```

```
Low: -6.749999999999999 High: 13.649999999999999
```

Out[112…

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission
50	fortuner	2012	14.90	30.61	104707	Diesel	Dealer	Automatic
51	fortuner	2015	23.00	30.61	40000	Diesel	Dealer	Automatic
52	innova	2017	18.00	19.77	15000	Diesel	Dealer	Automatic
53	fortuner	2013	16.00	30.61	135000	Diesel	Individual	Automatic
59	fortuner	2014	19.99	35.96	41000	Diesel	Dealer	Automatic
62	fortuner	2014	18.75	35.96	78000	Diesel	Dealer	Automatic
63	fortuner	2015	23.50	35.96	47000	Diesel	Dealer	Automatic
64	fortuner	2017	33.00	36.23	6000	Diesel	Dealer	Automatic
66	innova	2017	19.75	23.15	11000	Petrol	Dealer	Automatic
69	corolla altis	2016	14.25	20.91	12000	Petrol	Dealer	Manual
79	fortuner	2012	14.50	30.61	89000	Diesel	Dealer	Automatic
80	corolla altis	2016	14.73	14.89	23000	Diesel	Dealer	Manual
82	innova	2017	23.00	25.39	15000	Diesel	Dealer	Automatic
86	land cruiser	2010	35.00	92.60	78000	Diesel	Dealer	Manual
93	fortuner	2015	23.00	30.61	40000	Diesel	Dealer	Automatic
96	innova	2016	20.75	25.39	29000	Diesel	Dealer	Automatic
97	corolla altis	2017	17.00	18.64	8700	Petrol	Dealer	Manual

In [ ]:

```
import pandas as pd
dataset=pd.read_csv('car data.csv')
def find_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]<low_val)| (ds[col]>high_val)]

    return ds
find_outliers(dataset,'Selling_Price')
```

In [113…

```
import pandas as pd
dataset=pd.read_csv('car data.csv')
def find_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]<low_val)| (ds[col]>high_val)]

    return ds
find_outliers(dataset,'Year')
```

```
Low: 2006.0 High: 2022.0
```

Out[113…

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission
37	800	2003	0.35	2.28	127000	Petrol	Individual	Manual
39	sx4	2003	2.25	7.98	62000	Petrol	Dealer	Manual
54	innova	2005	2.75	10.21	90000	Petrol	Individual	Manual
77	corolla	2004	1.50	12.35	135154	Petrol	Dealer	Automatic
84	innova	2005	3.49	13.46	197176	Diesel	Dealer	Manual
92	innova	2005	3.51	13.70	75000	Petrol	Dealer	Manual
189	Hero Super Splendor	2005	0.20	0.57	55000	Petrol	Individual	Manual

In [114…

```
import pandas as pd
dataset=pd.read_csv('car data.csv')
def find_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]<low_val)| (ds[col]>high_val)]

    return ds
find_outliers(dataset,'Present_Price')
```

```
Low: -11.850000000000001 High: 22.950000000000003
```

Out[114…

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission
50	fortuner	2012	14.90	30.61	104707	Diesel	Dealer	Automatic
51	fortuner	2015	23.00	30.61	40000	Diesel	Dealer	Automatic
53	fortuner	2013	16.00	30.61	135000	Diesel	Individual	Automatic

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission
59	fortuner	2014	19.99	35.96	41000	Diesel	Dealer	Automatic
62	fortuner	2014	18.75	35.96	78000	Diesel	Dealer	Automatic
63	fortuner	2015	23.50	35.96	47000	Diesel	Dealer	Automatic
64	fortuner	2017	33.00	36.23	6000	Diesel	Dealer	Automatic
66	innova	2017	19.75	23.15	11000	Petrol	Dealer	Automatic
79	fortuner	2012	14.50	30.61	89000	Diesel	Dealer	Automatic
82	innova	2017	23.00	25.39	15000	Diesel	Dealer	Automatic
85	camry	2006	2.50	23.73	142000	Petrol	Individual	Automatic
86	land cruiser	2010	35.00	92.60	78000	Diesel	Dealer	Manual
93	fortuner	2015	23.00	30.61	40000	Diesel	Dealer	Automatic
96	innova	2016	20.75	25.39	29000	Diesel	Dealer	Automatic

In [115…

```python
import pandas as pd
dataset=pd.read_csv('car data.csv')
def find_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]<low_val)| (ds[col]>high_val)]

    return ds
find_outliers(dataset,'Kms_Driven')
```

```
Low: -35650.5 High: 99417.5
```

Out[115…

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission
37	800	2003	0.35	2.28	127000	Petrol	Individual	Manual
50	fortuner	2012	14.90	30.61	104707	Diesel	Dealer	Automatic
53	fortuner	2013	16.00	30.61	135000	Diesel	Individual	Automatic
77	corolla	2004	1.50	12.35	135154	Petrol	Dealer	Automatic
84	innova	2005	3.49	13.46	197176	Diesel	Dealer	Manual
85	camry	2006	2.50	23.73	142000	Petrol	Individual	Automatic
179	Honda Karizma	2010	0.31	1.05	213000	Petrol	Individual	Manual
196	Activa 3g	2008	0.17	0.52	500000	Petrol	Individual	Automatic

In [121…

```python
import pandas as pd
dataset=pd.read_csv('car data.csv')
def find_outliers(ds,col):
    quart1=ds[col].quantile(0.25)
    quart3=ds[col].quantile(0.75)
    IQR=quart3-quart1
    low_val=quart1-1.5*IQR
    high_val=quart3+1.5*IQR
    print("Low:",low_val,"High:",high_val)
    ds=ds.loc[(ds[col]<low_val)| (ds[col]>high_val)]

    return ds
find_outliers(dataset,'Owner')
```

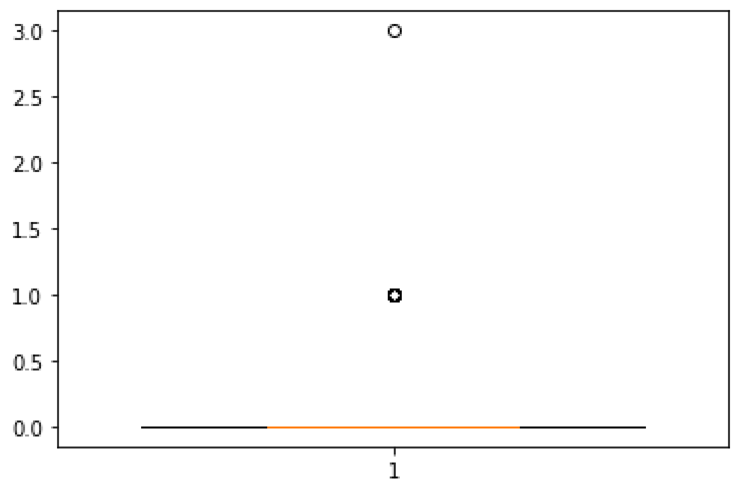
```
Low: 0.0 High: 0.0
```

Out[121…

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission
58	etios g	2014	4.10	6.80	39485	Petrol	Dealer	Manual
85	camry	2006	2.50	23.73	142000	Petrol	Individual	Automatic
106	Hyosung GT250R	2014	1.35	3.45	16500	Petrol	Individual	Manual
184	Bajaj Pulsar 150	2008	0.25	0.75	26000	Petrol	Individual	Manual
191	Bajaj Discover 125	2012	0.20	0.57	25000	Petrol	Individual	Manual
192	Hero Hunk	2007	0.20	0.75	49000	Petrol	Individual	Manual
193	Hero Ignitor Disc	2013	0.20	0.65	24000	Petrol	Individual	Manual
198	Bajaj Discover 125	2011	0.15	0.57	35000	Petrol	Individual	Manual
201	i20	2010	3.25	6.79	58000	Diesel	Dealer	Manual
205	grand i10	2016	5.25	5.70	3493	Petrol	Dealer	Manual
241	xcent	2015	4.75	7.13	35866	Petrol	Dealer	Manual

In [123…

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
dataset=pd.read_csv('car data.csv')
plt.boxplot(dataset['Owner'],widths=0.75,notch=True)
plt.show()
```



In [124…
```
import pandas as pd
data=pd.read_csv("car data.csv")
data.corr()
```

Out[124…

	Year	Selling_Price	Present_Price	Kms_Driven	Owner
Year	1.000000	0.236141	-0.047584	-0.524342	-0.182104
Selling_Price	0.236141	1.000000	0.878983	0.029187	-0.088344
Present_Price	-0.047584	0.878983	1.000000	0.203647	0.008057
Kms_Driven	-0.524342	0.029187	0.203647	1.000000	0.089216
Owner	-0.182104	-0.088344	0.008057	0.089216	1.000000

In [126…
```
import pandas as pd
data=pd.read_csv("car data.csv")
data.isna().sum()
```

Out[126…
```
Car_Name         0
Year             0
Selling_Price    0
Present_Price    0
Kms_Driven       0
Fuel_Type        0
Seller_Type      0
Transmission     0
Owner            0
dtype: int64
```