Python Intro

JUPYTER NOTEBOOK

Python 3

In [ ]:

```
# Python is one of the most popular general-purpose programming languages in modern times.it was developed in 1990s
# The term “general-purpose” simply means that Python can be used for a variety of applications and does not focus on any one aspect of programming.

# Python is high level programming language
# unlike C++ , we don't need to compile then execute , we can run python code directly

# Applications of Python :
# 1. Web Applications
# 2. AI/ML
# 3. Data Science
# 4. OS development and many more ...

# Interesting fact : why python is called as python?
# The interesting fact is that the programming language Python was implemented by Guido van Rossum.
# Whilst Guido van Rossum was implementing Python, he was also reading the published scripts from Monty Python’s Flying Circus.
# Monty Python’s Flying Circus is a BBC Comedy TV series from the year 1969+. It is a highly viewed TV series and is rated 8.8 in IMDB.
```

NASA

Sklearn

$A = [1, 2, 3, 6, 0]$

$A' = [0, 1, 2, 3, 6]$

25

Pandas Intro

Panel data

Tim Peter

Python data analysis Tool

Cell

#

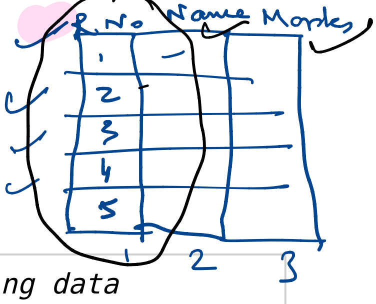
In [7]:

```
# first install the pacakge using following command
!pip install pandas
# pip is preferred installer program
#as i have already downloaded this package output shows requirement already satisfied
#similarly you can install any libraries available in python
```

```
Requirement already satisfied: pandas in ./.local/lib/python3.8/site-packages (1.2.4)
Requirement already satisfied: numpy>=1.16.5 in ./.local/lib/python3.8/site-packages (from pandas) (1.20.2)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/lib/python3/dist-packages (from pandas) (2.8.1)
Requirement already satisfied: pytz>=2017.3 in /usr/lib/python3/dist-packages (from pandas) (2020.1)
```

In [2]:

```
# first we have to import the required library in our jupyter notebook
# for plotiing , charts, bar graph and histograms we also need to import matplotlib package
import pandas as pd
import matplotlib.pyplot as plt
```



Data Structures

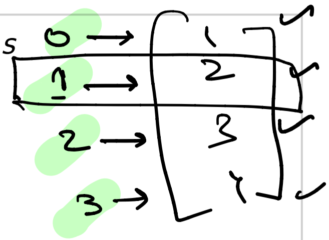
In [168]:

```
# there are two types of data structures mostly used in organising data

# 1. Series : Single column data of nD array
# 2. Data Frame: is like table that store data similar to spreadsheet using mult
iple columns and rows
# 3.  DataFrame is a container for Series, and Series is a container for scalars
```

Series

In [7]:

```
# To create series we can use the following syntax or operations
series1= pd.Series([1,2,3,4])

#to print this series we use standard print command
print(series1)
p= series1.mean()
print(p)
# you can see series is created successfully where left hand side column represe
nt index of particular element
#dtype is data type
```



```
0    1
1    2
2    3
3    4
dtype: int64
2.5
```

$$\frac{1+2+3+4}{4} = 2.5$$

In [42]:

```
# lets create some mores series but little bit complex type
# we can name the series just by writing ''.'' after series variable
srs= pd.Series([11.9,35,14,20,33], index=['china','india','germany','USA','Russi
a'])
srs.name= "Growth Rate"
srs.index.name= "Country"
print(srs)
```

```
Country
china      11.9
india      35.0
germany    14.0
USA        20.0
Russia     33.0
Name: Growth Rate, dtype: float64
```

0
1
2
3
4

In [43]:

```
# we can fetch element by using following command
print(srs[3])
print(srs['india'])
print(srs[['india','Russia']])
```

```
20.0
35.0
Country
india     35.0
Russia    33.0
Name: Growth Rate, dtype: float64
```

In [140]:

```
#how to get subset of a series : just use the following syntax
srs[1:]
```

Out[140]:

```
Country
india          35.0
germany        14.0
USA            20.0
Russia         33.0
Name: Growth Rate, dtype: float64
```

In [44]:

```
# we can also drop the elemnents by using .drop commands
series2= srs.drop('china')
print(series2)
```

```
Country
india          35.0
germany        14.0
USA            20.0
Russia         33.0
Name: Growth Rate, dtype: float64
```

Data Frame

In [8]:

```
# lets create a data frame in python with the help of pandas
df= pd.DataFrame({"Roll No": [1,2,5,8,10,12],"Name":['Amit','Anil','Shashwat','Raushan','Hitesh','Atul'],"Marks":[101,102,95,96,97,99]})
print(df)
#we can add as many as columns
# we can get info of our data by writing
df.info()
# we can calculate mean/median just by writing this short commands
df.mean()
```

```
   Roll No      Name  Marks
0        1      Amit    101
1        2      Anil    102
2        5  Shashwat     95
3        8   Raushan     96
4       10    Hitesh     97
5       12      Atul     99
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 3 columns):
 #   Column   Non-Null Count  Dtype 
---  ------   --------------  ----- 
 0   Roll No  6 non-null      int64 
 1   Name     6 non-null      object
 2   Marks    6 non-null      int64 
dtypes: int64(2), object(1)
memory usage: 272.0+ bytes
```

Out[8]:

```
Roll No     6.333333
Marks      98.333333
dtype: float64
```

In [9]:

```
# similarly for dataframe you can get subset using :
df[1:]
```

Out[9]:

	Roll No	Name	Marks
1	2	Anil	102
2	5	Shashwat	95
3	8	Raushan	96
4	10	Hitesh	97
5	12	Atul	99

Unlike C++ ✓

85.30

In [10]:

```
# we can also sort the values by :
df.sort_values(by='Marks')
```

Out[10]:

	Roll No	Name	Marks
2	5	Shashwat	95
3	8	Raushan	96
4	10	Hitesh	97
5	12	Atul	99
0	1	Amit	101
1	2	Anil	102

In [11]:

```
df.sort_values(by='Name')
```

Out[11]:

	Roll No	Name	Marks
0	1	Amit	101
1	2	Anil	102
5	12	Atul	99
4	10	Hitesh	97
3	8	Raushan	96
2	5	Shashwat	95

90%

In [12]:

```
df.sort_values(by='Roll No')
```

Out[12]:

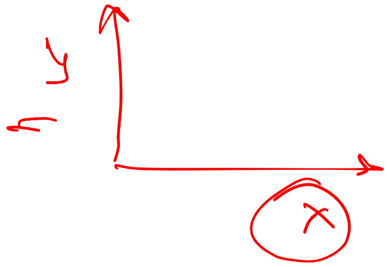
	Roll No	Name	Marks
0	1	Amit	101
1	2	Anil	102
2	5	Shashwat	95
3	8	Raushan	96
4	10	Hitesh	97
5	12	Atul	99

How to play with data sets in csv/xlsx/txt/tsv

In [174]:

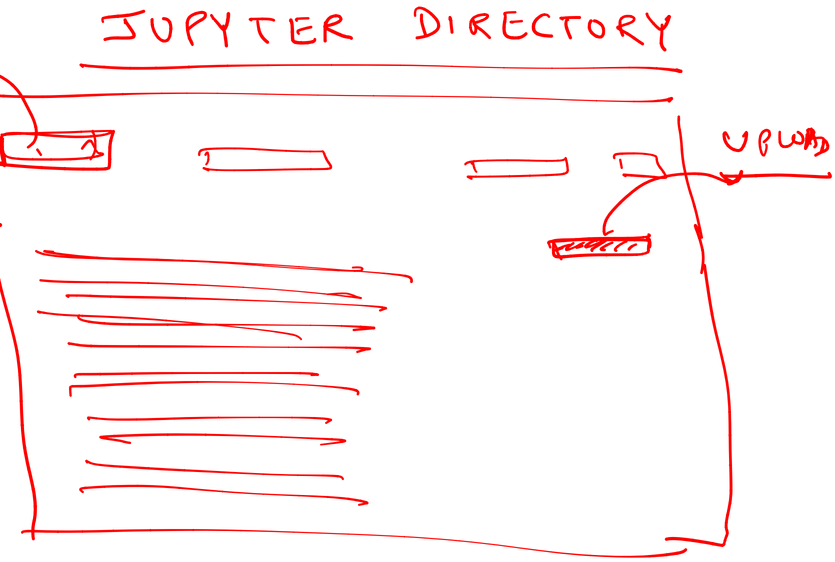
```
# now i am going to show you how you can read csv /tsv or any kind of data sets
 in pandas
# i have already uploaded some datasets in my library so i am just importing in
 notebook
# first create a data frame
df= pd.read_csv('MySubscribers.csv')
#for excel just put read_excel


print(df)
df.info()
x=df['Date']
y= df['Number of Subscribers']
#you can create scatter plot using following commands
plt.scatter(x,y)
```

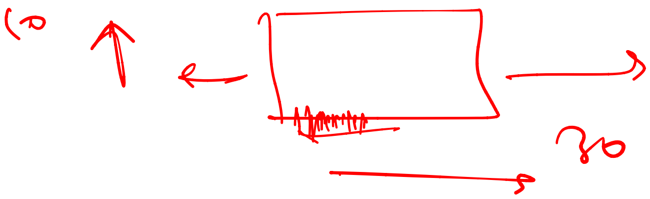
read_txt



open

plt.figure(figsize=(30,10))

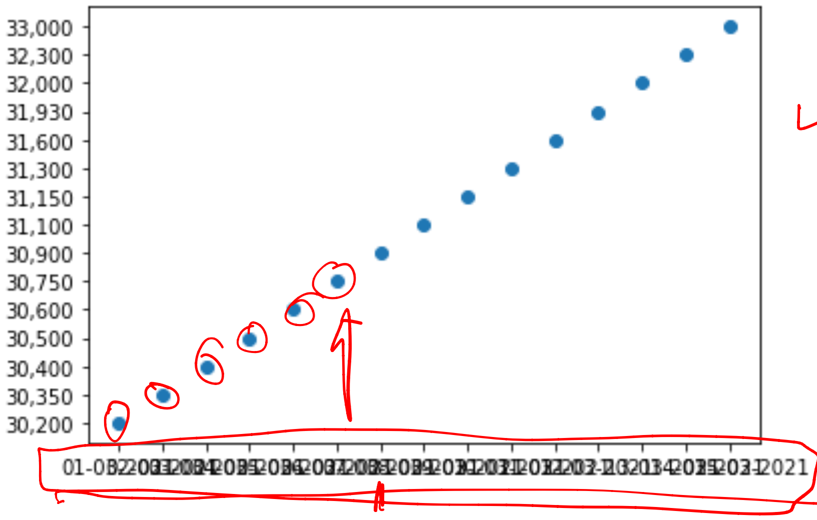




```
          Date Number of Subscribers
0   01-03-2021                30,200
1   02-03-2021                30,350
2   03-03-2021                30,400
3   04-03-2021                30,500
4   05-03-2021                30,600
5   06-03-2021                30,750
6   07-03-2021                30,900
7   08-03-2021                31,100
8   09-03-2021                31,150
9   10-03-2021                31,300
10  11-03-2021                31,600
11  12-03-2021                31,930
12  13-03-2021                32,000
13  14-03-2021                32,300
14  15-03-2021                33,000
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 2 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   Date                   15 non-null     object
 1   Number of Subscribers  15 non-null     object
dtypes: object(2)
memory usage: 368.0+ bytes
```

```
Out[174]:
```

```
<matplotlib.collections.PathCollection at 0x7f8bb0b088e0>
```

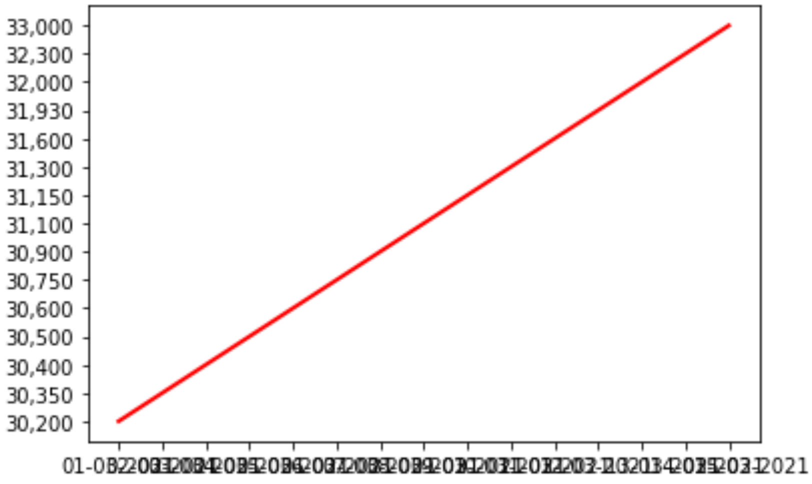


In [175]:

```
#for linear plot or graph you have to use
plt.plot(x,y,'r',lw=2)
```

Out[175]:

```
[<matplotlib.lines.Line2D at 0x7f8bb0d9c160>]
```



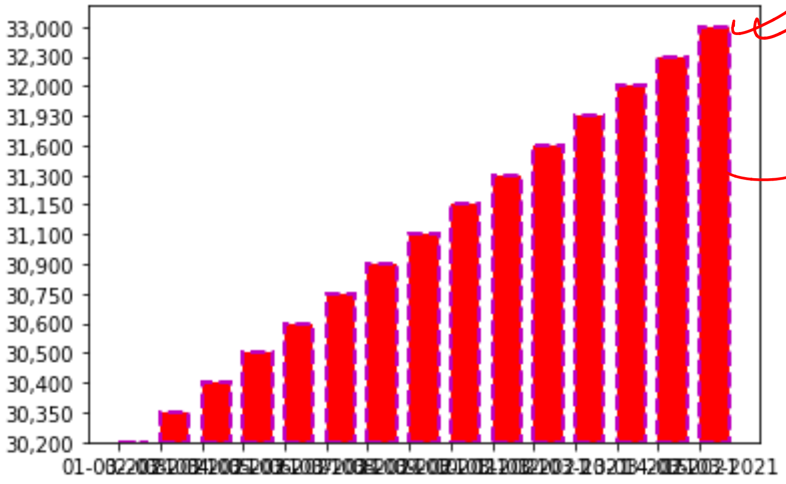
Plt. bar(x,y)

In [176]:

```
#for bar graph use the following
plt.bar(x,y,width=0.7,color="r",align ="edge",edgecolor="m",linewidth=2,linestyle="--")
#for plotting horizontally just put h after bar
```

Out[176]:

```
<BarContainer object of 15 artists>
```



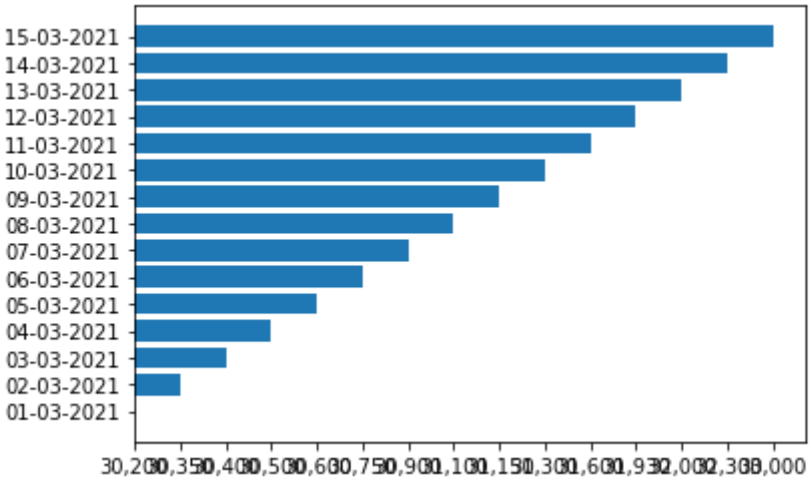
0.7

In [177]:

```
# we can create the above bar graph horizontally just by puting 'h'
plt.barh(x,y)
```

Out[177]:

```
<BarContainer object of 15 artists>
```



2nd May.

In [4]:

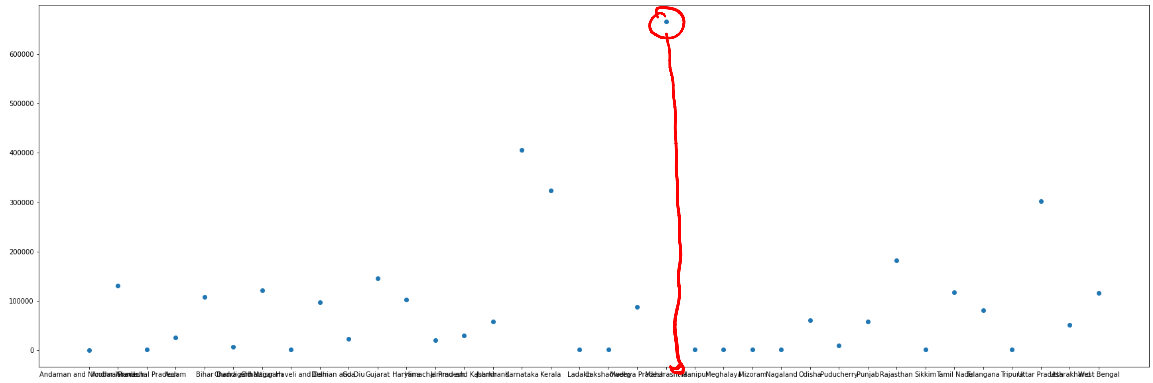
```
df= pd.read_csv('CovidIndia.csv')
print(df)
x=df['State/UT']
y= df['Active Cases']
#you can create scatter plot using following commands
plt.figure(figsize=(30, 10))
plt.scatter(x,y)
```

```
     #                              State/UT  Confirmed Cases  \
0    1           Andaman and Nicobar Islands             6046   
1    2                        Andhra Pradesh          1121102   
2    3                     Arunachal Pradesh            18636   
3    4                                 Assam           256576   
4    5                                 Bihar           484106   
5    6                            Chandigarh            43446   
6    7                          Chhattisgarh           744602   
7    8  Dadra and Nagar Haveli and Daman and Diu             7712   
8    9                                 Delhi          1174552   
9   10                                   Goa            93355   
10  11                               Gujarat           581624   
11  12                               Haryana           501566   
12  13                      Himachal Pradesh           102038   
13  14                     Jammu and Kashmir           179915   
14  15                             Jharkhand           239734   
15  16                             Karnataka          1564132   
16  17                                Kerala          1606819   
17  18                                Ladakh            14086   
18  19                           Lakshadweep             2923   
19  20                        Madhya Pradesh           575706   
20  21                           Maharashtra          4665754   
21  22                               Manipur            31905   
22  23                             Meghalaya            17108   
23  24                               Mizoram             6299   
24  25                              Nagaland            14134   
25  26                                Odisha           454607   
26  27                            Puducherry            60001   
27  28                                Punjab           377990   
28  29                             Rajasthan           615653   
29  30                                Sikkim             8211   
30  31                            Tamil Nadu          1186344   
31  32                             Telangana           450790   
32  33                               Tripura            35589   
33  34                         Uttar Pradesh          1282504   
34  35                           Uttarakhand           186014   
35  36                           West Bengal           845878   

    Active Cases  Cured/Discharged  Death  
0            205              5773     68  
1         130752            982297   8053  
2           1387             17190     59  
3          26374            228872   1330  
4         108203            373261   2642  
5           7222             35735    489  
6         121099            614693   8810  
7           1867              5841      4  
8          96747           1061246  16559  
9          23884             68249   1222  
10        145139            429130   7355  
11        102516            394709   4341  
12         19928             80585   1525  
13         30343            147242   2330  
14         58437            178468   2829  
15        405088           1143250  15794  
16        324169           1277294   5356  
17          1400             12542    144  
18          1438              1481      4  
19         88511            481477   5718  
20        665837           3930302  69615  
21          1652             29843    410  
```

```
22          1659              15275    174
23          1299               4985     15
24          1353              12674    107
25         61505             391048   2054
26         10263              48921    817
27         58229             310601   9160
28        182301             428953   4399
29          1647               6416    148
30        117405            1054746  14193
31         80695             367727   2368
32          1471              33720    398
33        301833             967797  12874
34         51127             132156   2731
35        116659             717772  11447
```

Out[4]:

```
<matplotlib.collections.PathCollection at 0x7f58b012ed90>
```

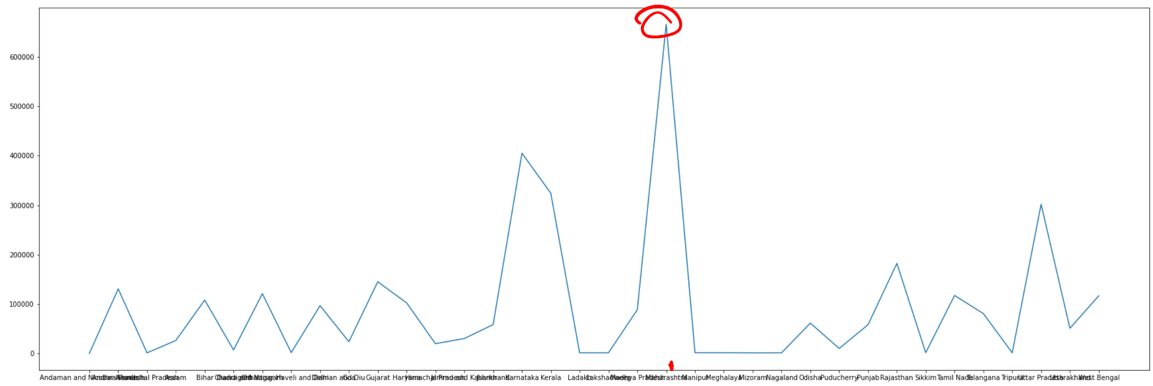


In [32]:

```
plt.figure(figsize=(30, 10))
plt.plot(x,y)
```

Out[32]:

```
[<matplotlib.lines.Line2D at 0x7f989e0786d0>]
```

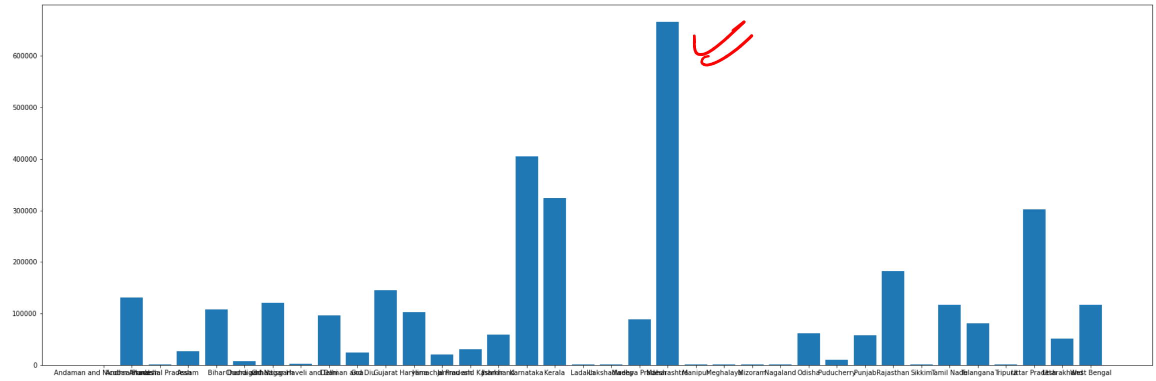


In [5]:

```
plt.figure(figsize=(30, 10))
plt.bar(x,y)
```

Out[5]:

```
<BarContainer object of 36 artists>
```



Matplotlib Intro

Numpy

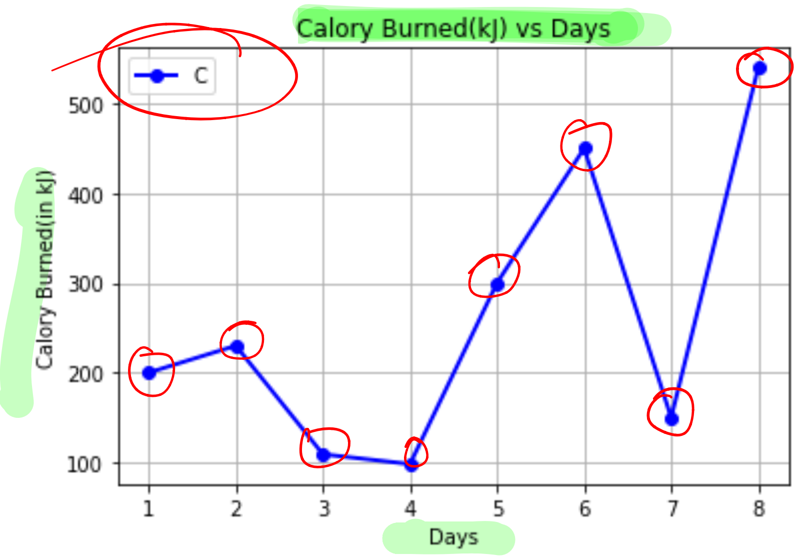
In [23]:

```
# Matplotlib is a Python 2D plotting library which produces publication-quality
 figures in a variety
# of hardcopy form and interactive environments across platforms.

#  The basic steps to creating plots with matplotlib are:
 # 1. Prepare data 2. Create plot 3. Plot 4. Customize plot 5. Save plot 6.Show
 plot

# Lets create the basic data and its plot
x= [1,2,3,4,5,6,7,8]
y= [200,230,109,98,300,450,150,540]
# command for ploting above created data is
plt.plot(x,y,color='b',marker='o',linewidth=2)

#you can add title , xlabel and ylabel with the help of
plt.title('Calory Burned(kJ) vs Days')
plt.xlabel('Days')
plt.ylabel('Calory Burned(in kJ)')
plt.grid()

plt.legend('C')
# there are lots of parameters we can use to customize our plots
# color,marker, linewidth, legend etc
```

Out[23]:

```
<matplotlib.legend.Legend at 0x7f8eee33a880>
```

help()

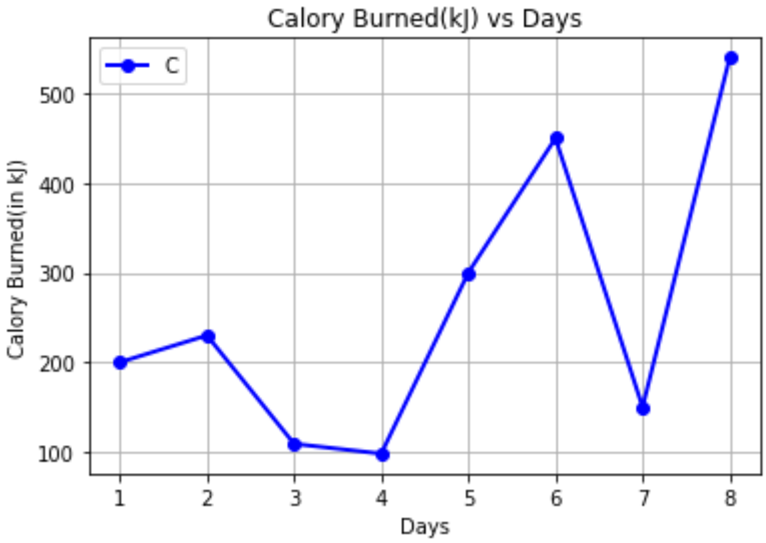


In [23]:

```
# Matplotlib is a Python 2D plotting library which produces publication-quality
 figures in a variety
# of hardcopy form and interactive environments across platforms.

#  The basic steps to creating plots with matplotlib are:
 # 1. Prepare data 2. Create plot 3. Plot 4. Customize plot 5. Save plot 6.Show
 plot

# Lets create the basic data and its plot
x= [1,2,3,4,5,6,7,8]
y= [200,230,109,98,300,450,150,540]
# command for ploting above created data is
plt.plot(x,y,color='b',marker='o',linewidth=2)

#you can add title , xlabel and ylabel with the help of
plt.title('Calory Burned(kJ) vs Days')
plt.xlabel('Days')
plt.ylabel('Calory Burned(in kJ)')
plt.grid()

plt.legend('C')
# there are lots of parameters we can use to customize our plots
# color,marker, linewidth, legend etc
```

Out[23]:

```
<matplotlib.legend.Legend at 0x7f8eee33a880>
```


Calory Burned(kJ) vs Days
C
Calory Burned(in kJ)
Days