Good Morning Everyone

Welcome to Final Session of Python Programming

Todays Objectives

Data Visualization using Matplotlib

Creating Students Data

rollnumbers = year + collegecode + universitycode + branchcode + rollNO

100 Students

20APPY0501 - 100

In [4]:

```
roll = '20APPY05'
for i in range(1,101,1):
    if i < 10:
        print(roll + str(0) +str(i))
    else:
        print(roll + str(i))
```

```
...
```

In [7]:

```
roll = '20APPY05'
rollNumber = [roll + str(0) +str(num) if num<10 else roll + str(num) for num in range(
print(rollNumber)
```

```
['20APPY0501', '20APPY0502', '20APPY0503', '20APPY0504', '20APPY0505', '20AP
PY0506', '20APPY0507', '20APPY0508', '20APPY0509', '20APPY0510', '20APPY051
1', '20APPY0512', '20APPY0513', '20APPY0514', '20APPY0515', '20APPY0516', '2
0APPY0517', '20APPY0518', '20APPY0519', '20APPY0520', '20APPY0521', '20APPY0
522', '20APPY0523', '20APPY0524', '20APPY0525', '20APPY0526', '20APPY0527',
'20APPY0528', '20APPY0529', '20APPY0530', '20APPY0531', '20APPY0532', '20APP
Y0533', '20APPY0534', '20APPY0535', '20APPY0536', '20APPY0537', '20APPY053
8', '20APPY0539', '20APPY0540', '20APPY0541', '20APPY0542', '20APPY0543', '2
0APPY0544', '20APPY0545', '20APPY0546', '20APPY0547', '20APPY0548', '20APPY0
549', '20APPY0550', '20APPY0551', '20APPY0552', '20APPY0553', '20APPY0554',
'20APPY0555', '20APPY0556', '20APPY0557', '20APPY0558', '20APPY0559', '20APP
Y0560', '20APPY0561', '20APPY0562', '20APPY0563', '20APPY0564', '20APPY056
5', '20APPY0566', '20APPY0567', '20APPY0568', '20APPY0569', '20APPY0570', '2
0APPY0571', '20APPY0572', '20APPY0573', '20APPY0574', '20APPY0575', '20APPY0
576', '20APPY0577', '20APPY0578', '20APPY0579', '20APPY0580', '20APPY0581',
'20APPY0582', '20APPY0583', '20APPY0584', '20APPY0585', '20APPY0586', '20APP
Y0587', '20APPY0588', '20APPY0589', '20APPY0590', '20APPY0591', '20APPY059
2', '20APPY0593', '20APPY0594', '20APPY0595', '20APPY0596', '20APPY0597', '2
0APPY0598', '20APPY0599', '20APPY05100']
```

In [12]:

```
import numpy as np

np.random.randint(1,100)
```

Out[12]:

```
5
```

In [18]:

```
C = [np.random.randint(0,100) for i in range(1,101)]
Python = [np.random.randint(0,100) for i in range(1,101)]
Java = [np.random.randint(0,100) for i in range(1,101)]
Pandas = [np.random.randint(0,100) for i in range(1,101)]
subjects = ['C', 'Python', 'Java', 'Pandas']
print(C)
```

```
[98, 61, 42, 39, 8, 38, 49, 86, 52, 24, 19, 76, 92, 17, 10, 10, 13, 79, 16,
56, 11, 68, 96, 97, 13, 2, 39, 99, 31, 64, 65, 16, 11, 56, 79, 46, 90, 77, 2
1, 82, 73, 64, 15, 0, 74, 67, 23, 83, 98, 61, 98, 51, 51, 85, 73, 83, 72, 7
2, 2, 81, 9, 36, 15, 4, 3, 87, 3, 24, 77, 35, 45, 17, 5, 56, 35, 44, 73, 77,
2, 55, 6, 48, 81, 78, 70, 99, 17, 16, 62, 5, 89, 9, 93, 72, 81, 88, 81, 84,
32, 62]
```

Total_marks = C + Python + Java + Pandas

In [14]:

```
C = np.array(C)
Python = np.array(Python)
Java = np.array(Java)
Pandas = np.array(Pandas)
```

In [15]:

```
Total_marks = C + Python + Java + Pandas
print(Total_marks)
```

```
[194 243 134 172 153 277 156 256 204 106 151 272 247 144 243 177 250 232
 252 211 106 190 185 183 208 254 226 132 254 230 150 304 215  89 134 173
 182 204 135 270 245 165 174 114 222 217 243 172 244 103 297 239 213 175
 329 170 230 270 240 157 190 164 155 214 208 174 270 166 174 281 189 156
 199 236 128 246 154 260 148 161 275 249 176 177 186 202 143 173 154 117
 237 224 249  49 256 200 178 105 216 237]
```

In [20]:

```
percentage = (Total_marks / (len(subjects) * 100)) * 100
print(percentage)
```

```
[48.5  60.75 33.5  43.   38.25 69.25 39.   64.   51.   26.5  37.75 68.
 61.75 36.   60.75 44.25 62.5  58.   63.   52.75 26.5  47.5  46.25 45.75
 52.   63.5  56.5  33.   63.5  57.5  37.5  76.   53.75 22.25 33.5  43.25
 45.5  51.   33.75 67.5  61.25 41.25 43.5  28.5  55.5  54.25 60.75 43.
 61.   25.75 74.25 59.75 53.25 43.75 82.25 42.5  57.5  67.5  60.   39.25
 47.5  41.   38.75 53.5  52.   43.5  67.5  41.5  43.5  70.25 47.25 39.
 49.75 59.   32.   61.5  38.5  65.   37.   40.25 68.75 62.25 44.   44.25
 46.5  50.5  35.75 43.25 38.5  29.25 59.25 56.   62.25 12.25 64.   50.
 44.5  26.25 54.   59.25]
```

In [32]:

```
import pandas as pd

Student_report = pd.DataFrame([rollNumber, C, Python, Java, Pandas, Total_marks, perce
```

In [28]:

```
Student_report
```

...

In [29]:

```
Student_report.columns
```

Out[29]:

```
RangeIndex(start=0, stop=7, step=1)
```

In [33]:

```
Student_report.columns = ['RollNumber', 'C', 'Python', 'Java', 'Pandas', 'TotalMarks',
```

In [34]:

```
Student_report
```

...

In [35]:

```
Student_report.head()
```

Out[35]:

	RollNumber	C	Python	Java	Pandas	TotalMarks	Percentage
0	20APPY0501	98	33	43	38	194	48.5
1	20APPY0502	61	11	28	13	243	60.75
2	20APPY0503	42	25	45	78	134	33.5
3	20APPY0504	39	31	93	36	172	43
4	20APPY0505	8	37	7	42	153	38.25

In [36]:

```
Student_report.tail()
```

Out[36]:

	RollNumber	C	Python	Java	Pandas	TotalMarks	Percentage
95	20APPY0596	88	15	54	46	200	50
96	20APPY0597	81	74	18	63	178	44.5
97	20APPY0598	84	83	7	6	105	26.25
98	20APPY0599	32	64	91	65	216	54
99	20APPY05100	62	19	39	96	237	59.25

Line Plot

Scatter Plot

Hitogram

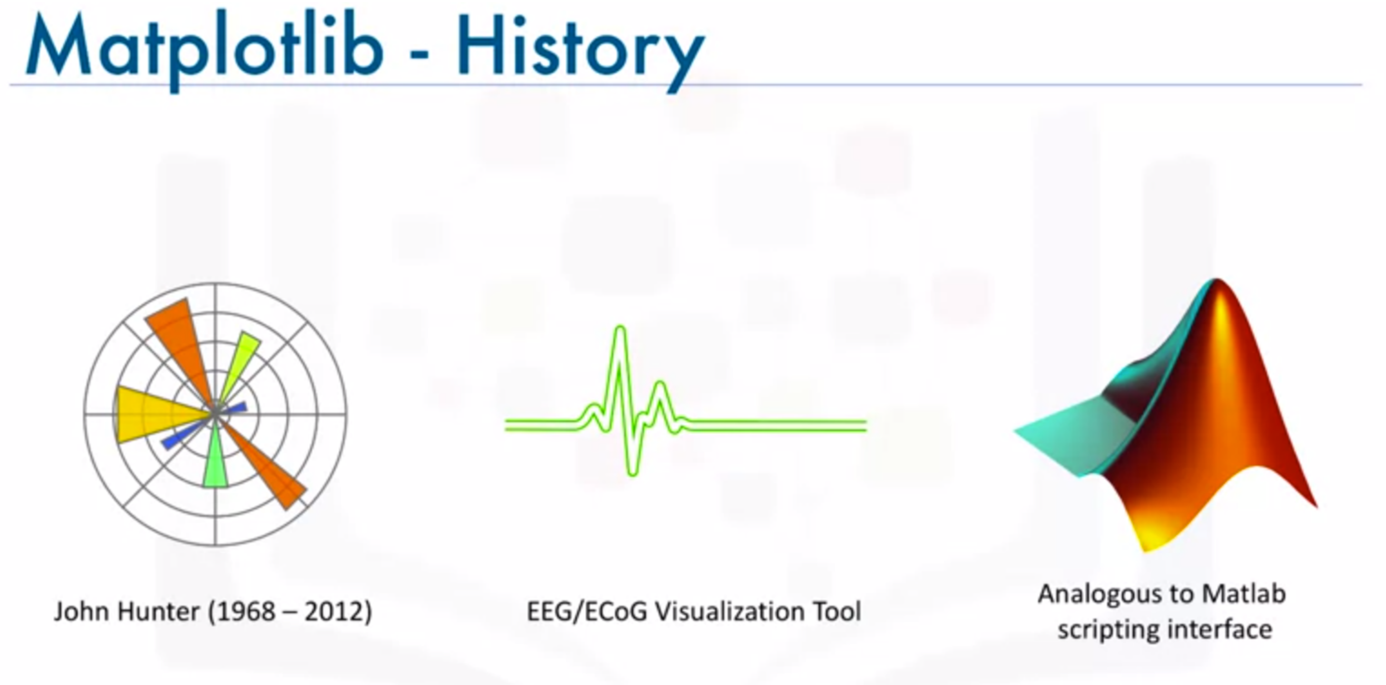
Bar Graph

Pie Chart

Box Plot

In [37]:

```
import matplotlib.pyplot as plt
```



Matplotlib - History

John Hunter (1968 – 2012)

EEG/ECoG Visualization Tool

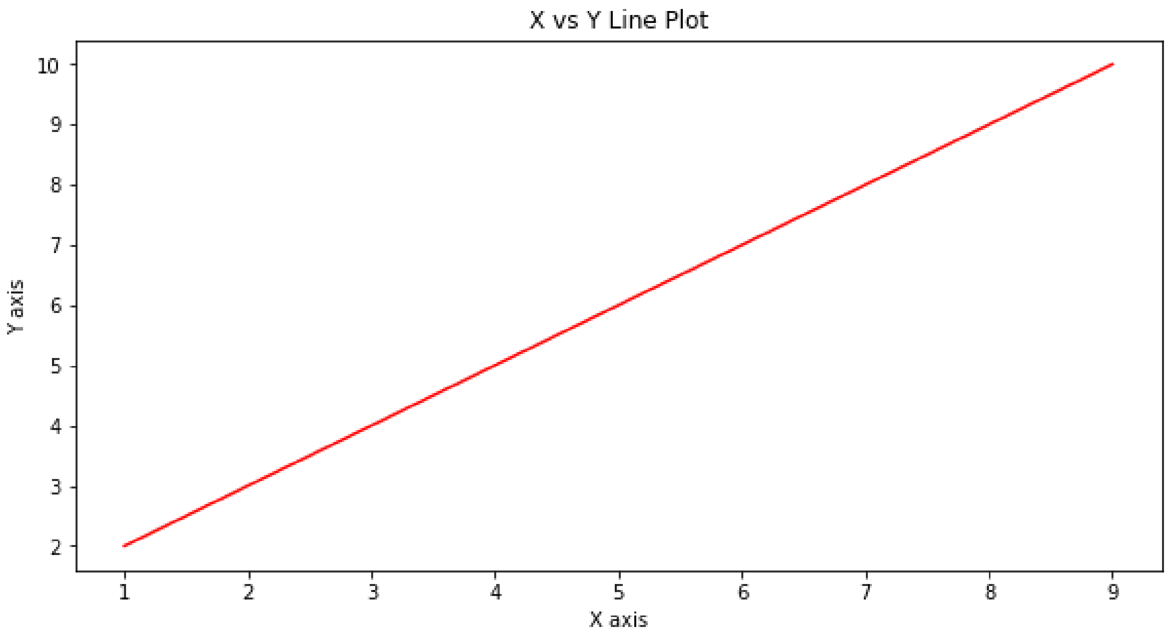
Analogous to Matlab scripting interface

Matplotlib package was implemented John Hunter

[Matplotlib Home Page (https://matplotlib.org/)](https://matplotlib.org/)

In [51]:

```
X = [1, 3, 5, 7, 9]
Y = [2,4,6,8,10]
plt.figure(figsize=(10, 5)) # width, height
plt.plot(X,Y, color = 'r')
plt.xlabel("X axis")
plt.ylabel("Y axis")
plt.title("X vs Y Line Plot")

plt.show()
```

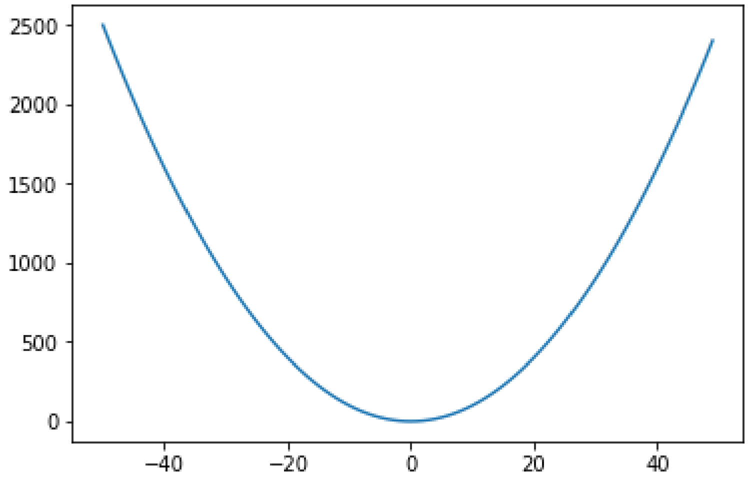


In [53]:

```
X = np.arange(-50, 50)
Y = X ** 2

plt.plot(X,Y)
```

Out[53]:

```
[<matplotlib.lines.Line2D at 0x23e75c67518>]
```



In [56]:

```
deg  = np.arange(0,361,15)
X = np.radians(deg)
Y = np.sin(X)

plt.plot(X,Y)
plt.xlabel("Deg in radians")
plt.ylabel("Sin values")
plt.title("Sin Graph from 0 deg to 360 deg")
plt.grid()
```

...

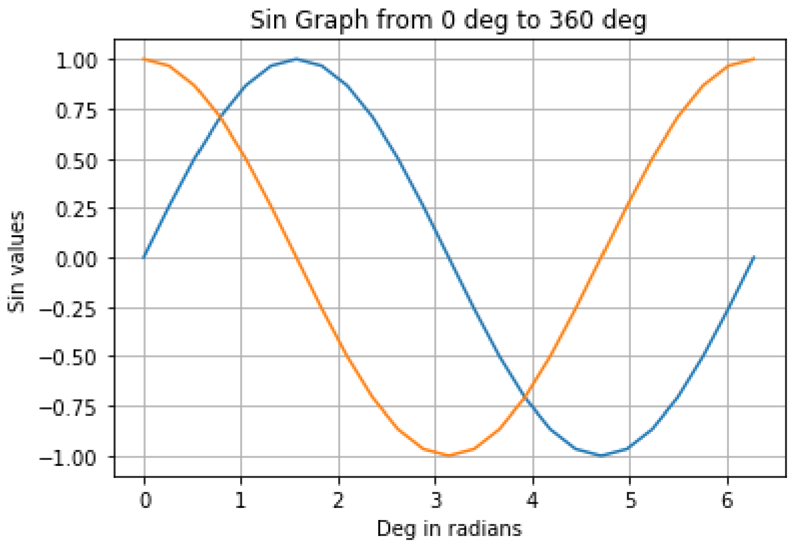
In [57]:

```
deg  = np.arange(0,361,15)
X = np.radians(deg)
Y = np.cos(X)

plt.plot(X,Y)
plt.xlabel("Deg in radians")
plt.ylabel("Cosine values")
plt.title("Cosine Graph from 0 deg to 360 deg")
plt.grid()
```

...

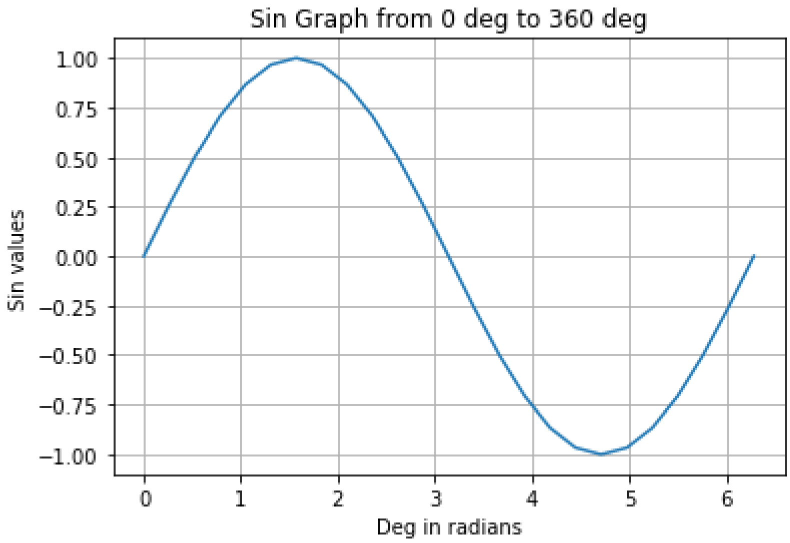
In [63]:

```
deg  = np.arange(0,361,15)
sinx = np.radians(deg)
siny = np.sin(X)
cosx = sinx
cosy = np.cos(X)
plt.plot(sinx,siny, label = 'Sin')
plt.plot(cosx,cosy, label = 'Cos')
plt.xlabel("Deg in radians")
plt.ylabel("Sin values")
plt.legend?
plt.title("Sin Graph from 0 deg to 360 deg")
plt.grid()
```

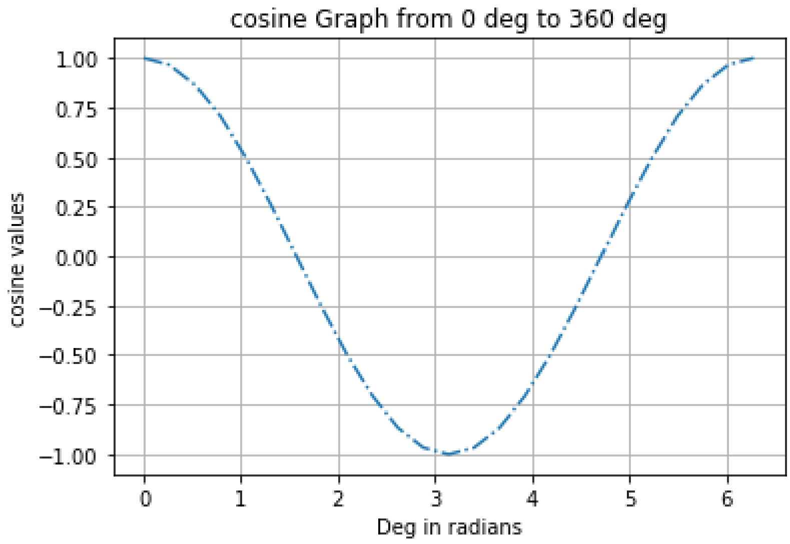


In [69]:

```
deg  = np.arange(0,361,15)
sinx = np.radians(deg)
siny = np.sin(X)
cosx = sinx
cosy = np.cos(X)
plt.plot(sinx,siny)
plt.xlabel("Deg in radians")
plt.ylabel("Sin values")
plt.title("Sin Graph from 0 deg to 360 deg")
plt.grid()
plt.show()
plt.plot(cosx,cosy, linestyle = '-.')
plt.xlabel("Deg in radians")
plt.ylabel("cosine values")
plt.title("cosine Graph from 0 deg to 360 deg")
plt.grid()
plt.show()
```



Sin Graph from 0 deg to 360 deg



cosine Graph from 0 deg to 360 deg

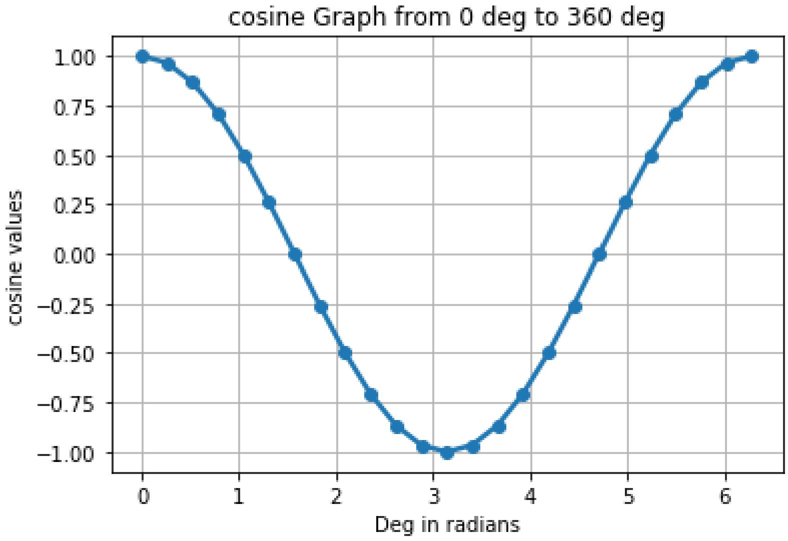
In [66]:

```
help(plt.plot)
```

```
Help on function plot in module matplotlib.pyplot:

plot(*args, scalex=True, scaley=True, data=None, **kwargs)
    Plot y versus x as lines and/or markers.

    Call signatures::

        plot([x], y, [fmt], data=None, **kwargs)
        plot([x], y, [fmt], [x2], y2, [fmt2], ..., **kwargs)

    The coordinates of the points or line nodes are given by *x*, *y*.

    The optional parameter *fmt* is a convenient way for defining basic
    formatting like color, marker and linestyle. It's a shortcut string
    notation described in the *Notes* section below.

    >>> plot(x, y)        # plot x and y using default line style and colo
r
    >>> plot(x, y, 'bo')  # plot x and y using blue circle markers
```

In [76]:

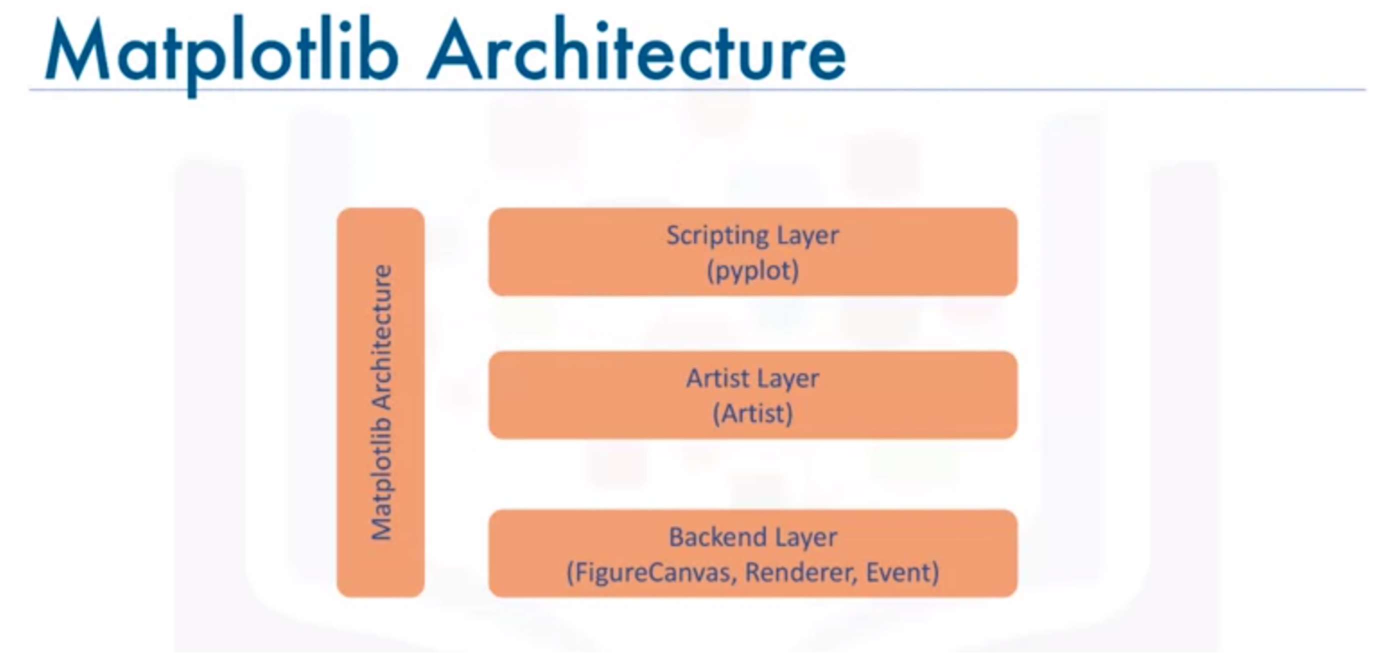
```
plt.plot(cosx,cosy, linewidth = 2.5, marker = 'o')
plt.xlabel("Deg in radians")
plt.ylabel("cosine values")
plt.title("cosine Graph from 0 deg to 360 deg")
plt.grid()
plt.show()
```



In [78]:

```python
import matplotlib
help(matplotlib)
```

```
Help on package matplotlib:

NAME
    matplotlib - This is an object-oriented plotting library.

DESCRIPTION
    A procedural interface is provided by the companion pyplot module,
    which may be imported directly, e.g.::

        import matplotlib.pyplot as plt

    or using ipython::

        ipython
```



Matplotlib Architecture

In [79]:

```python
X = Student_report['C']
Y = Student_report['Python']
plt.scatter(X,Y)
```

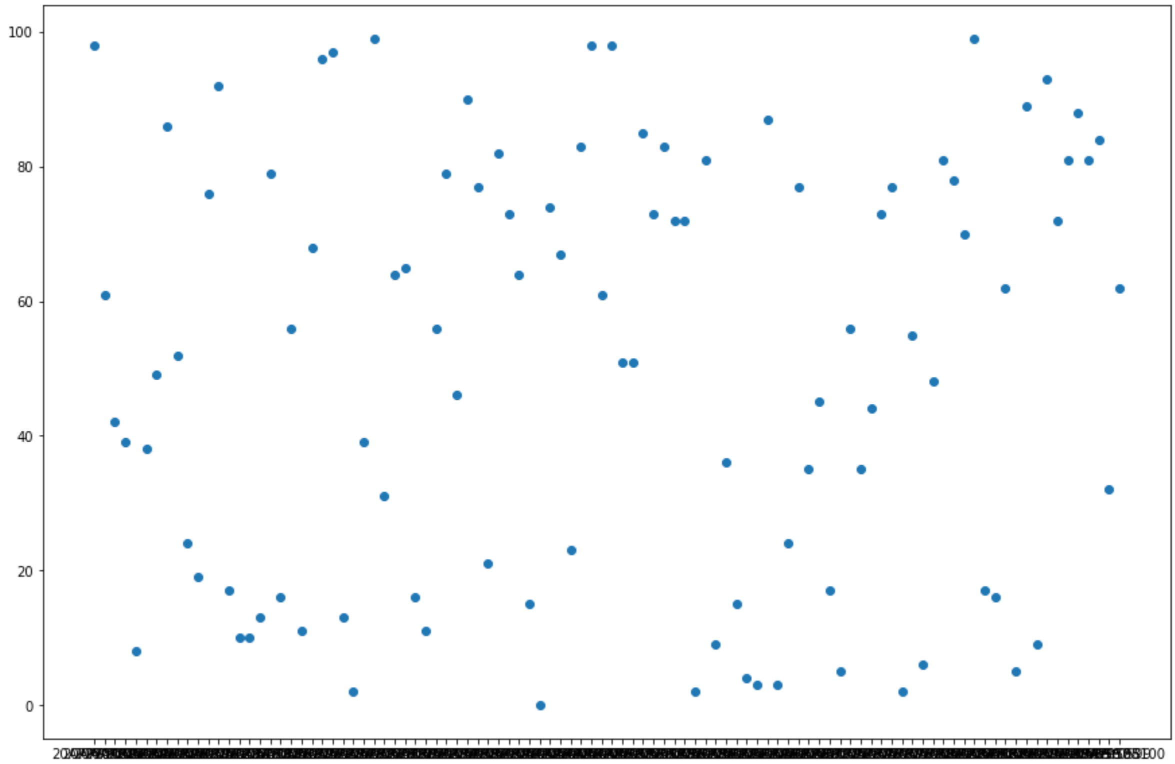
...

In [88]:

```
plt.figure(figsize=(15,10))

plt.scatter(Student_report['RollNumber'], Student_report['C'])

```

Out[88]:

```
<matplotlib.collections.PathCollection at 0x23e76f2e160>
```



10 min Break 11:00 11:10

HackerEarth --> will give demo

2:00 to : 3:00PM

mail before 10min

20marks MCQs and 2 problem

Certificate will issued based on exam & attendance

In [91]:

```
import pandas as pd

itc_df =  pd.read_csv('https://raw.githubusercontent.com/AP-Skill-Development-Corporat
itc_df.head()
```

Out[91]:

	Unnamed: 0	Symbol	Series	Date	Prev Close	Open Price	High Price	Low Price	Last Price	Close Price	...	To Trad Quan
0	0	ITC	EQ	2017-05-15	274.95	275.90	278.90	275.50	278.50	277.95	...	54628
1	1	ITC	EQ	2017-05-16	277.95	278.50	284.30	278.00	283.00	283.45	...	112043
2	2	ITC	EQ	2017-05-17	283.45	284.10	284.40	279.25	281.50	281.65	...	82977
3	3	ITC	EQ	2017-05-18	281.65	278.00	281.05	277.05	277.65	277.90	...	79242
4	4	ITC	EQ	2017-05-19	277.90	282.25	295.65	281.95	286.40	286.20	...	357241

5 rows × 21 columns

In [93]:

```python
import pandas as pd

itc_df =  pd.read_csv('https://raw.githubusercontent.com/AP-Skill-Development-Corporat
itc_df.head()
```

Out[93]:

	Symbol	Series	Date	Prev Close	Open Price	High Price	Low Price	Last Price	Close Price	Average Price	Total Traded Quantity	
0	ITC	EQ	2017-05-15	274.95	275.90	278.90	275.50	278.50	277.95	277.78	5462855	1
1	ITC	EQ	2017-05-16	277.95	278.50	284.30	278.00	283.00	283.45	280.93	11204308	3
2	ITC	EQ	2017-05-17	283.45	284.10	284.40	279.25	281.50	281.65	281.56	8297700	2
3	ITC	EQ	2017-05-18	281.65	278.00	281.05	277.05	277.65	277.90	278.49	7924261	2
4	ITC	EQ	2017-05-19	277.90	282.25	295.65	281.95	286.40	286.20	290.08	35724128	1

In [95]:

```python
itc_df.columns
```

Out[95]:

```
Index(['Symbol', 'Series', 'Date', 'Prev Close', 'Open Price', 'High Price',
       'Low Price', 'Last Price', 'Close Price', 'Average Price',
       'Total Traded Quantity', 'Turnover', 'No. of Trades', 'Deliverable Qt
y',
       '% Dly Qt to Traded Qty', 'Month', 'Year', 'VWAP', 'Day_Perc_Change',
       'Trend'],
      dtype='object')
```

In [100]:

```
itc_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 494 entries, 0 to 495
Data columns (total 20 columns):
Symbol                    494 non-null object
Series                    494 non-null object
Date                      494 non-null object
Prev Close                494 non-null float64
Open Price                494 non-null float64
High Price                494 non-null float64
Low Price                 494 non-null float64
Last Price                494 non-null float64
Close Price               494 non-null float64
Average Price             494 non-null float64
Total Traded Quantity     494 non-null int64
Turnover                  494 non-null float64
No. of Trades             494 non-null int64
Deliverable Qty           494 non-null int64
% Dly Qt to Traded Qty    494 non-null float64
Month                     494 non-null int64
Year                      494 non-null int64
VWAP                      494 non-null float64
Day_Perc_Change           494 non-null float64
Trend                     494 non-null object
dtypes: float64(11), int64(5), object(4)
memory usage: 81.0+ KB
```

In [104]:

```
itc_df['Date'] = itc_df.Date.astype('datetime64')
```

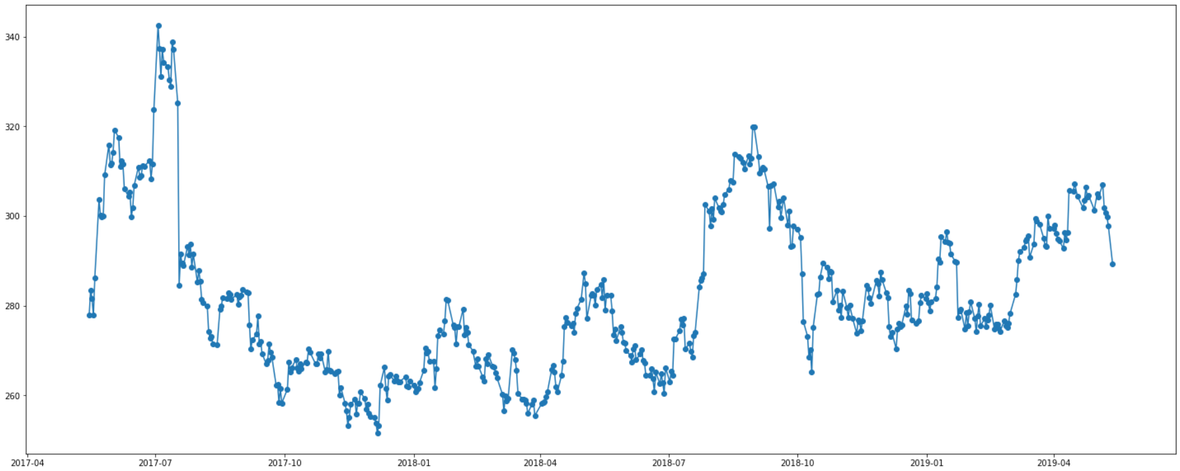
In [105]:

```python
x = itc_df['Close Price']
y = itc_df['Date']
plt.figure(figsize=(25,10))
plt.scatter(y,x)
plt.plot(y,x)
```

```
C:\Users\Jesus\Anaconda3\lib\site-packages\pandas\plotting\_converter.py:12
9: FutureWarning: Using an implicitly registered datetime converter for a ma
tplotlib plotting method. The converter was registered by pandas on import.
Future versions of pandas will require you to explicitly register matplotlib
converters.

To register the converters:
        >>> from pandas.plotting import register_matplotlib_converters
        >>> register_matplotlib_converters()
  warnings.warn(msg, FutureWarning)
```

Out[105]:

```
[<matplotlib.lines.Line2D at 0x23e7fcb2cc0>]
```

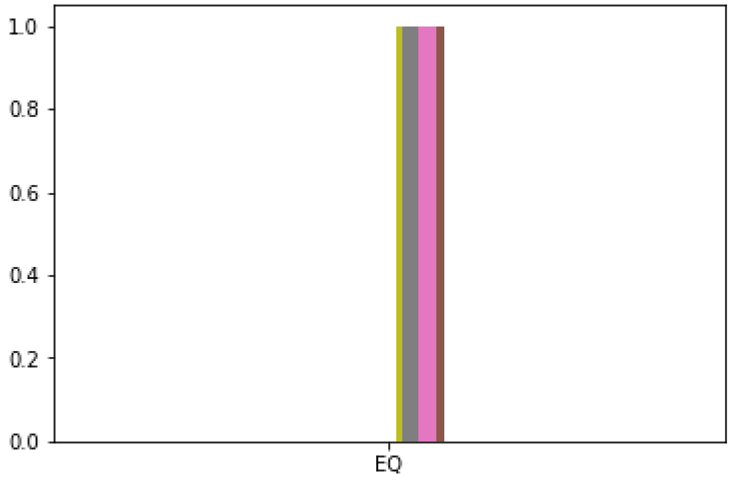


Histogram

it is an 1D plot it is to plot the frequency of the varaiabe in that array

In [108]:

```
plt.hist(itc_df['Series'])
plt.show()
```



In [109]:

```
help(plt.bar)
```

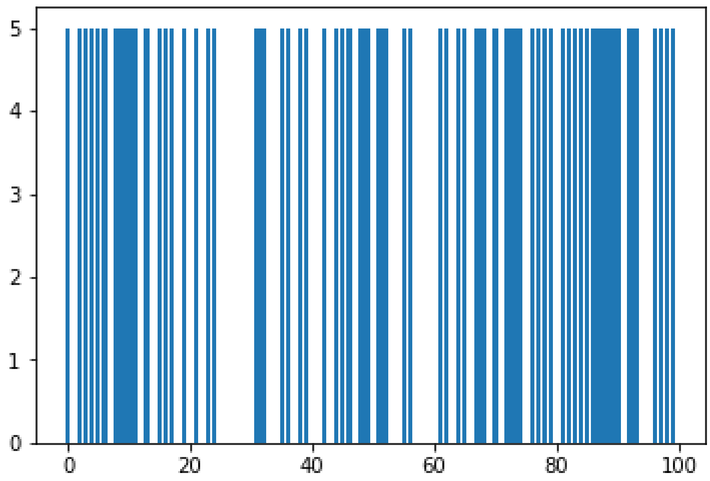
```
Help on function bar in module matplotlib.pyplot:

bar(x, height, width=0.8, bottom=None, *, align='center', data=None, **kwa
rgs)
    Make a bar plot.

    The bars are positioned at *x* with the given *align*\ment. Their
    dimensions are given by *width* and *height*. The vertical baseline
    is *bottom* (default 0).

    Each of *x*, *height*, *width*, and *bottom* may either be a scalar
    applying to all bars, or it may be a sequence of length N providing a
    separate value for each bar.

    Parameters
    ----------
    x : sequence of scalars
        The x coordinates of the bars. See also *align* for the
        alignment of the bars to the coordinates.
```

In [111]:

```python
plt.bar(Student_report['C'], height = 5)
```

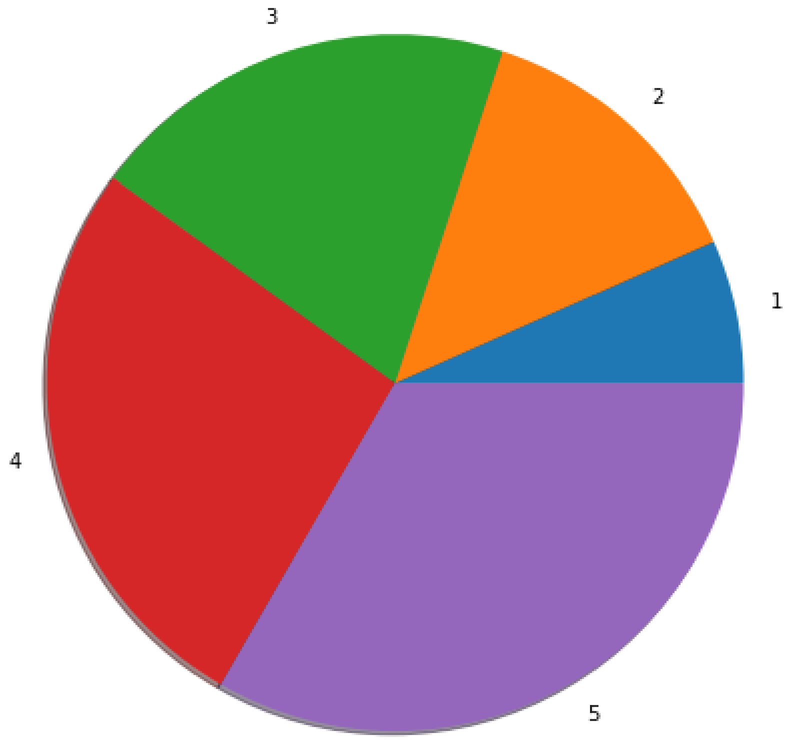
Out[111]:

```
<BarContainer object of 100 artists>
```



In [121]:

```python
x = [1,2,3,4,5]
plt.pie(x, labels=x, radius=2, shadow=True)
plt.show()
```



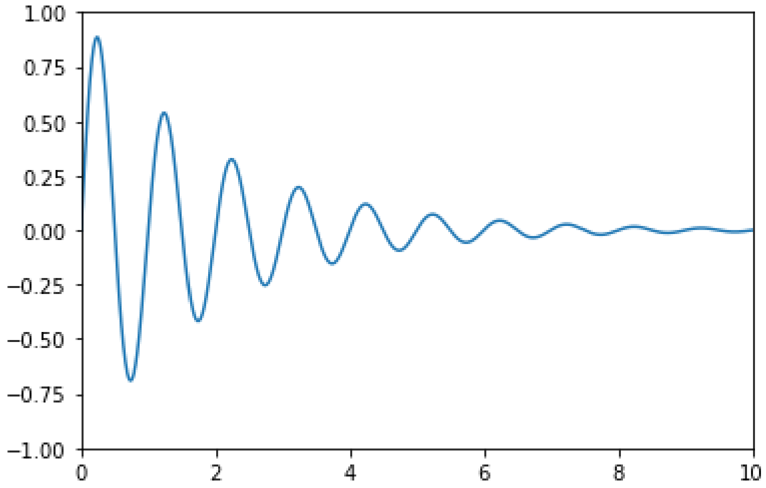
In [119]:

```python
help(plt.pie)
```

```
...
```

In [122]:

```python
x = np.arange(0, 10, 0.005)
y = np.exp(-x/2.) * np.sin(2*np.pi*x)

fig, ax = plt.subplots()
ax.plot(x, y)
ax.set_xlim(0, 10)
ax.set_ylim(-1, 1)

plt.show()
```

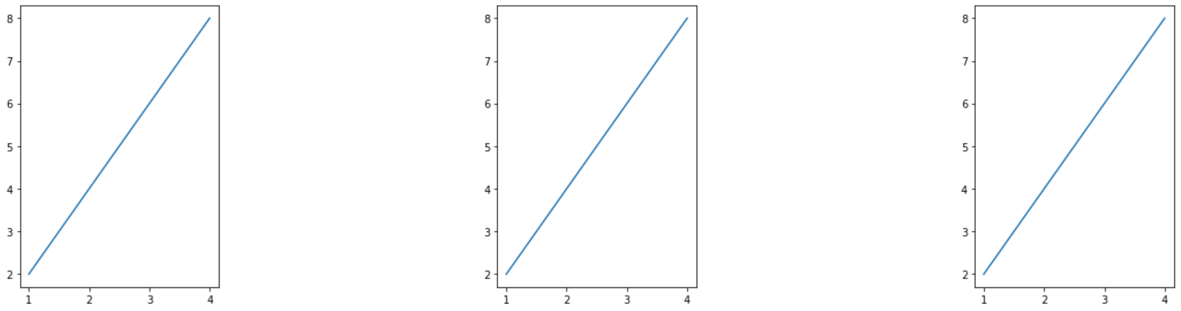


In [131]:

```python
plt.figure(figsize=(20,5))
plt.subplot(1,5,1)
plt.plot([1,2,3,4],[2,4,6,8])
plt.subplot(1,5,3)
plt.plot([1,2,3,4],[2,4,6,8])
plt.subplot(1,5,5)
plt.plot([1,2,3,4],[2,4,6,8])

```

Out[131]:

```
[<matplotlib.lines.Line2D at 0x23e0d82d3c8>]
```

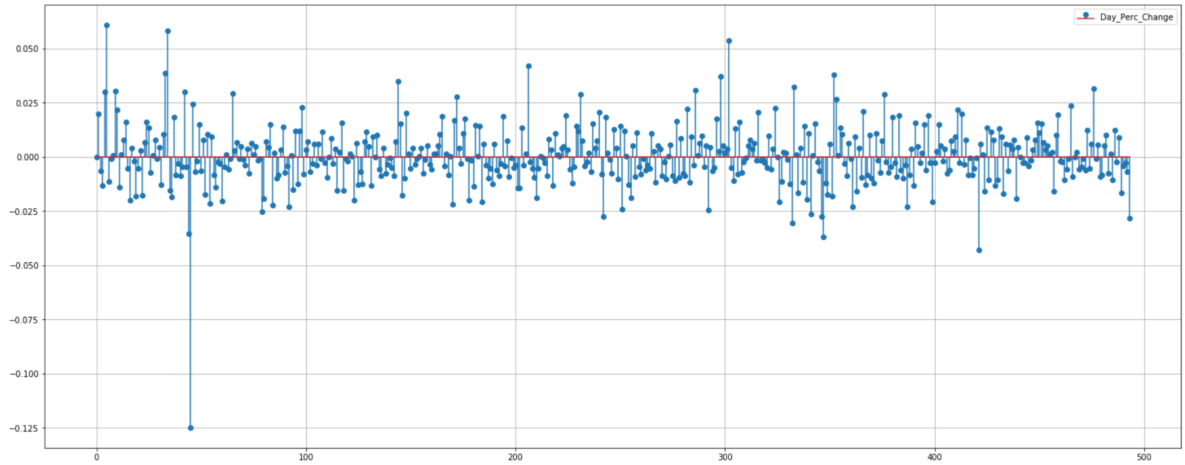


In [ ]:

```


```

In [136]:

```
plt.figure(figsize=(25,10))
plt.stem(itc_df['Day_Perc_Change'], label='Day_Perc_Change')

plt.legend()
plt.grid()
plt.show()
```



In [137]:

```
Student_report.to_csv("Student_report.csv")
```

In [138]:

```
ls
```

```
 Volume in drive C has no label.
 Volume Serial Number is 7866-1790

 Directory of C:\Users\Jesus\Desktop\Python FDP

30-May-20  12:06 PM    <DIR>          .
30-May-20  12:06 PM    <DIR>          ..
30-May-20  09:34 AM    <DIR>          .ipynb_checkpoints
30-May-20  12:06 PM         2,051,150 Day11.ipynb
23-May-20  08:28 AM             1,628 Day6.ipynb
30-May-20  11:11 AM    <DIR>          Python-FDP-TEAM-1-1
30-May-20  12:06 PM             3,821 Student_report.csv
25-May-20  09:02 PM           151,939 Summer_FDPS Attendance(upto 23-05-202
0).pdf
               4 File(s)      2,208,538 bytes
               4 Dir(s)  141,689,012,224 bytes free
```

Pandas (https://pandas.pydata.org/)

In [140]:

```python
import seaborn as sns
```

```
...
```

In [146]:

```python
s = 'abab'
print(s[::-1])
```

```
baba
```

In [ ]:

```python

```