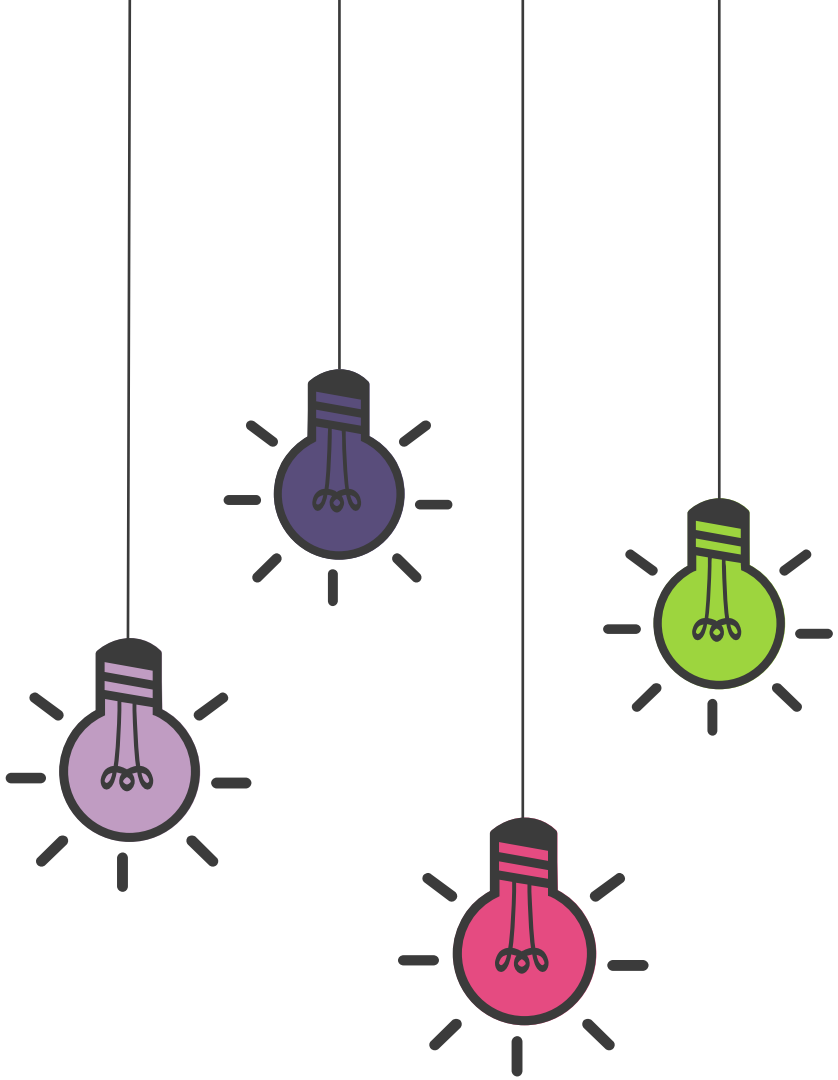
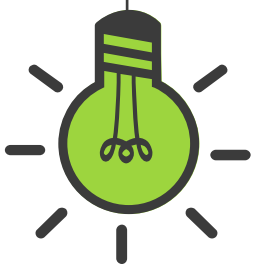


Python

파 이 썬

김 계 희
jenni7@naver.com

04 Visualization



1. Matplotlib

- 데이터를 Chart, Plot으로 시각화해주는 라이브러리이다.
- 2003년 0.1 출시, 17년이된 패키지
- 가장 많이 사용되는 시각화 패키지이다.
- 기본 구성 : 그림(figure), 축(axes)
- % matplotlib inline : jupyter notebook 내에서 output을 보여준다.

(1) Basic Attributes

- alpha : 투명도
- logy : Y축 Log scaling
- kind : line, bar, barh, kde
- subplots : 여러개의 plot 그리기
- legend : subplot의 범례 지정
- xlim, ylim : X축과 Y축의 경계(axis로 한번에 그릴 수 있음)
- grid : 그리드 표현(True, False)
- title : 그래프의 제목 지정
- linewidth : 라인 넓이
- color : 색

(1) Basic Attributes

- linestyle : 실선, 점선, 1점 쇄선 등
- marker : 마커 지정
- markerfacecolor : 마커 색
- markersize : 마커 크기
- 그 외 Attribute :

  https://matplotlib.org/3.1.1/tutorials/introductory/pyplot.html](https://matplotlib.org/3.1.1/tutorials/introductory/pyplot.html

(2) Line Plot

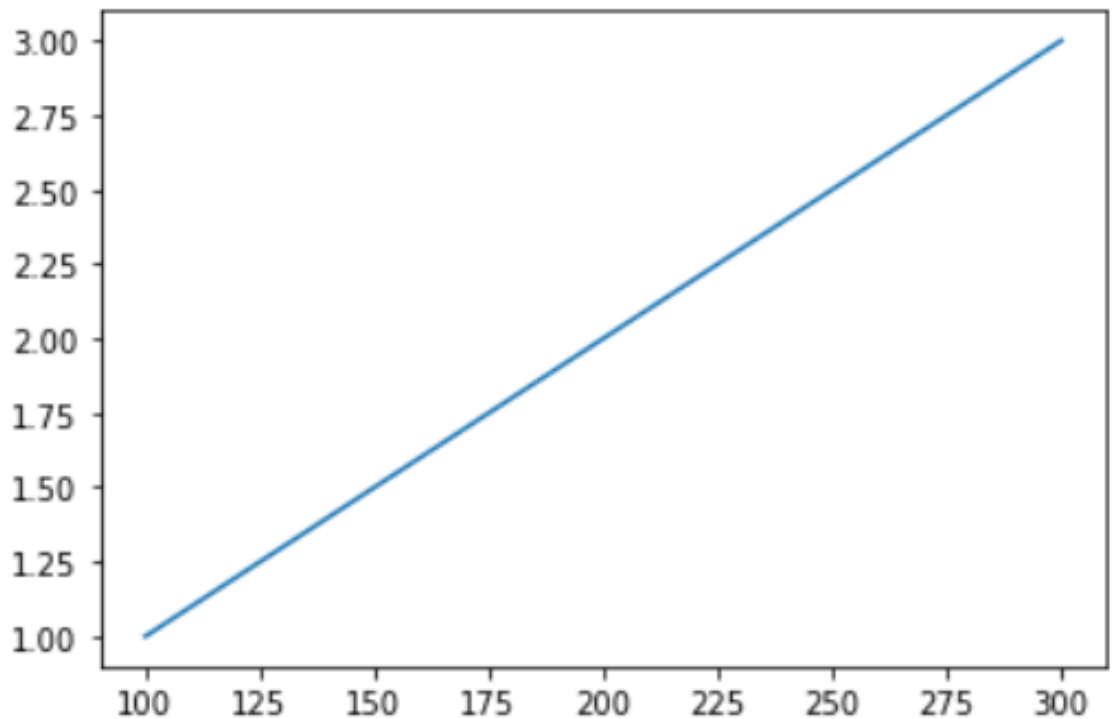
- **Line Plot**

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
%matplotlib inline
```

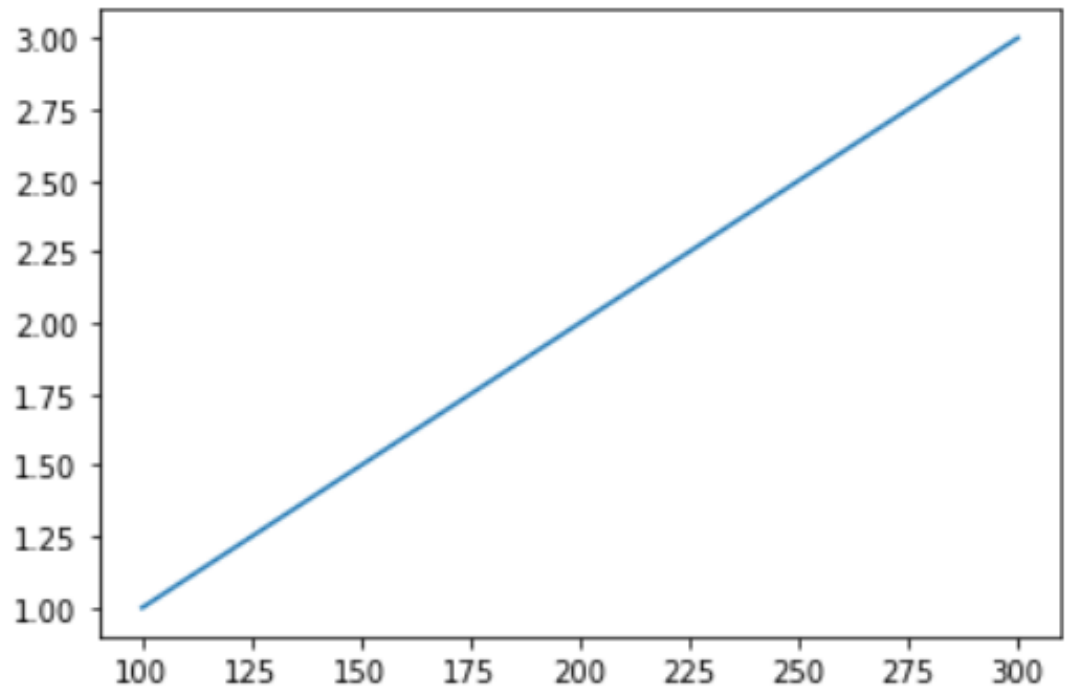
In [3]:
```
x=[100,200,300]
y=[1,2,3]
plt.plot(x,y)
```

Out[3]:
```
[<matplotlib.lines.Line2D at 0x1eaa13fd60>]
```



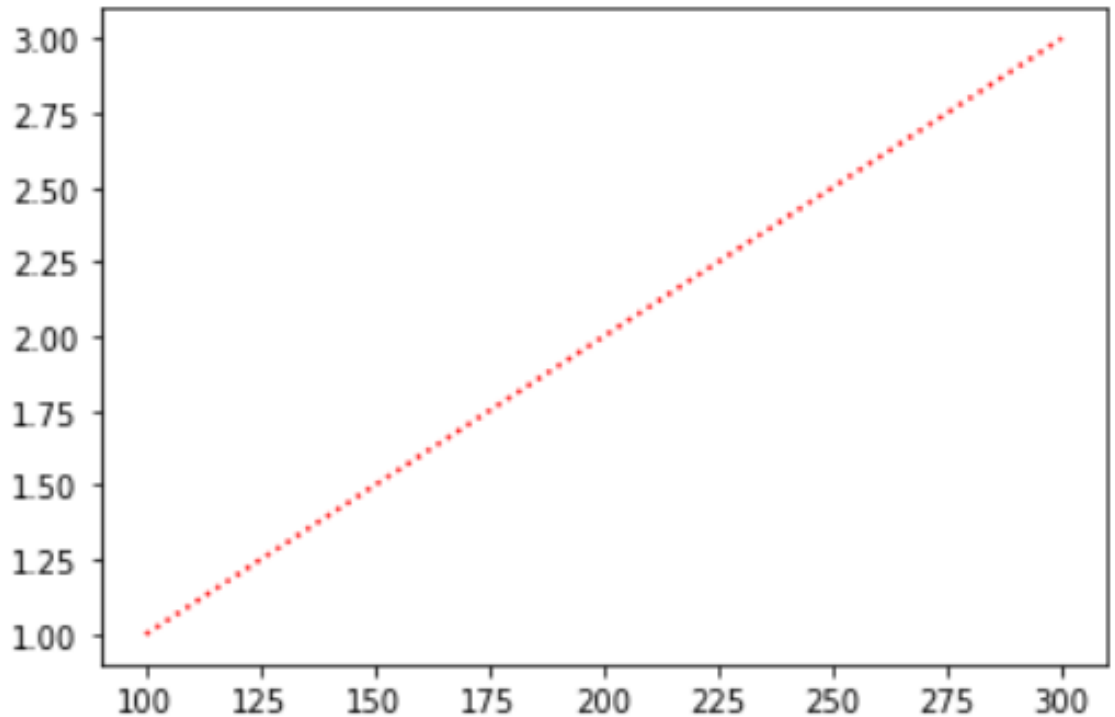
In [4]:
```
x=[100,200,300]
y=[1,2,3]
value=pd.Series([1,2,3],[100,200,300])
plt.plot(value)
```

Out[4]:
```
[<matplotlib.lines.Line2D at 0x9781095550>]
```



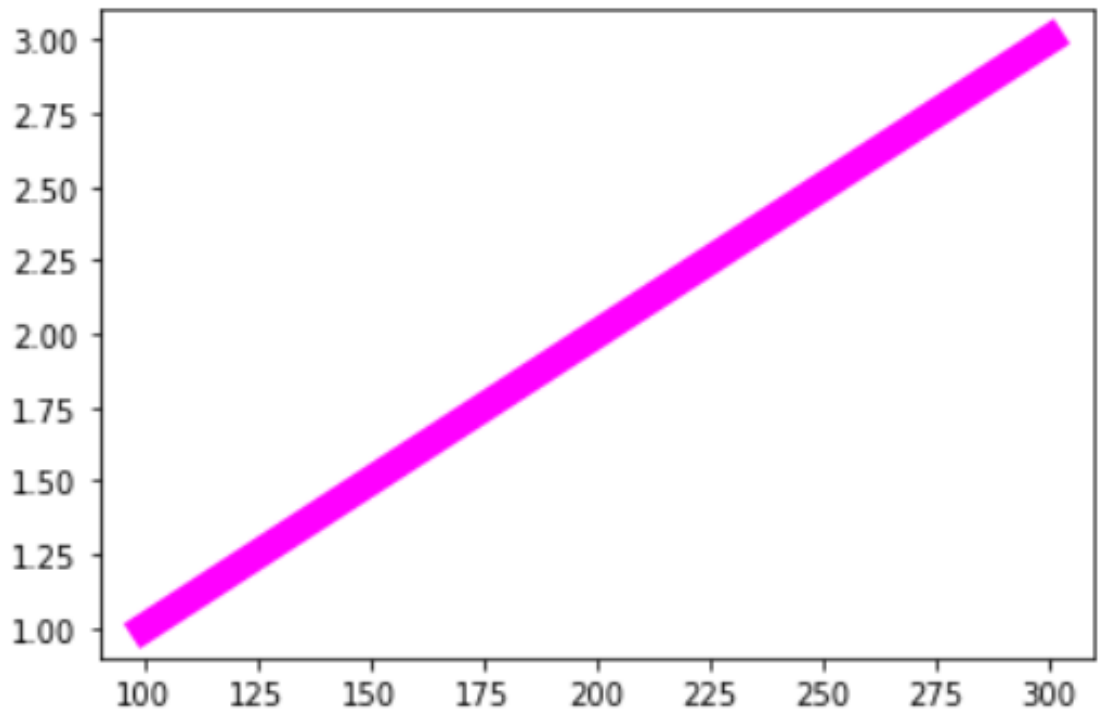
In [5]:
```
x=[100,200,300]
y=[1,2,3]
plt.plot(x,y,':r')  # g(green), r(red), b(blue), -->, --<, --or
```

Out[5]:
```
[<matplotlib.lines.Line2D at 0x9781100070>]
```



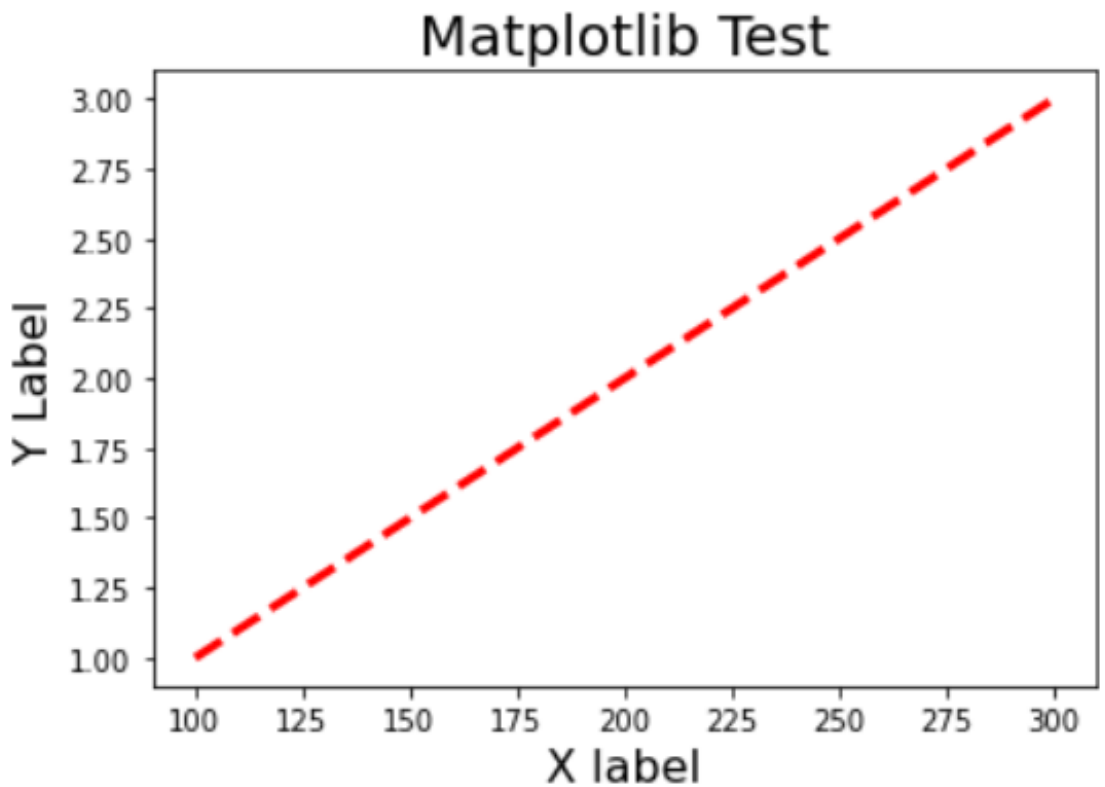
In [6]:
```
x=[100,200,300]
y=[1,2,3]
plt.plot(x,y,color='#ff00ff', linewidth='10')  # #ff00ff(핑크색), blue
```

Out[6]:
```
[<matplotlib.lines.Line2D at 0x1eaa2c99d0>]
```

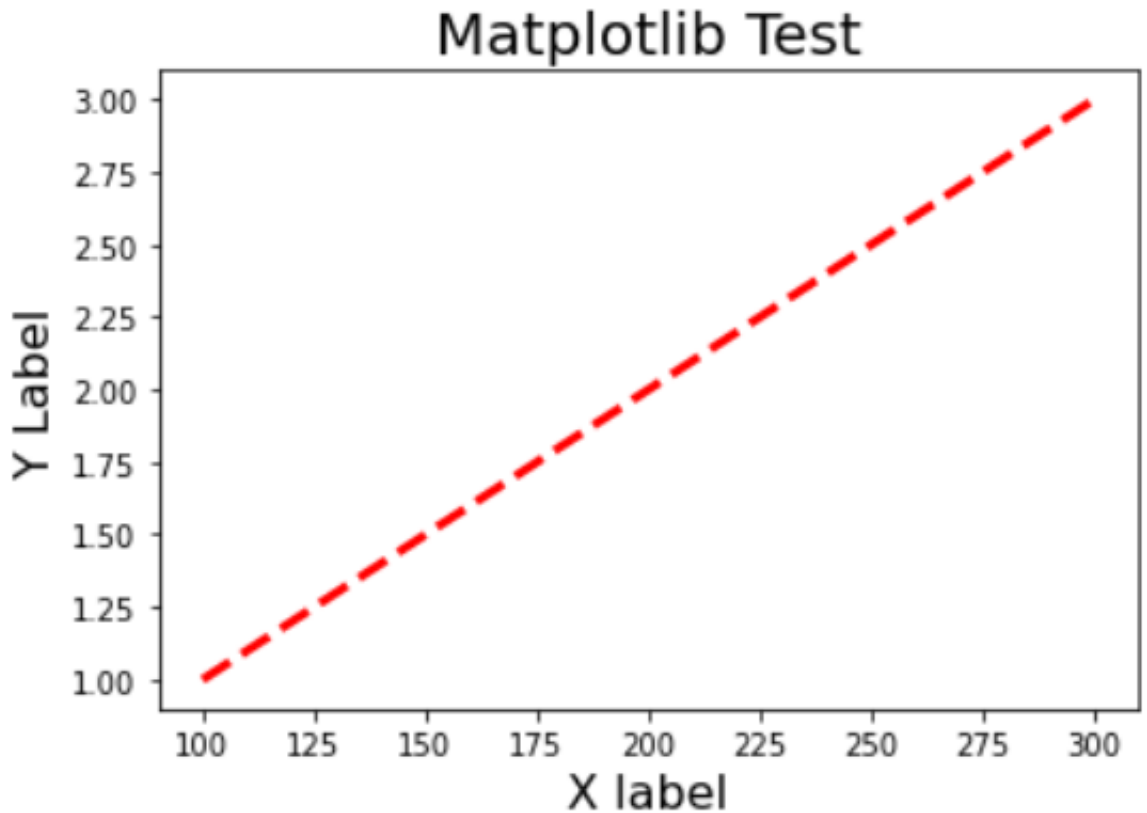


In [7]:
```
x=[100,200,300]
y=[1,2,3]
plt.plot(x,y,'--r', linewidth='3')
plt.xlabel('X label',fontsize=16)
plt.ylabel('Y Label',fontsize=16)
plt.title('Matplotlib Test',fontsize='20')
```

Out[7]:
```
Text(0.5, 1.0, 'Matplotlib Test')
```



In [8]:
```
x=[100,200,300]
y=[1,2,3]
plt.plot(x,y,'--r', linewidth='3')
plt.xlabel('X label',fontsize=16)
plt.ylabel('Y Label',fontsize=16)
plt.title('Matplotlib Test',fontsize='20')
plt.savefig('sample.png')
```



(3) label

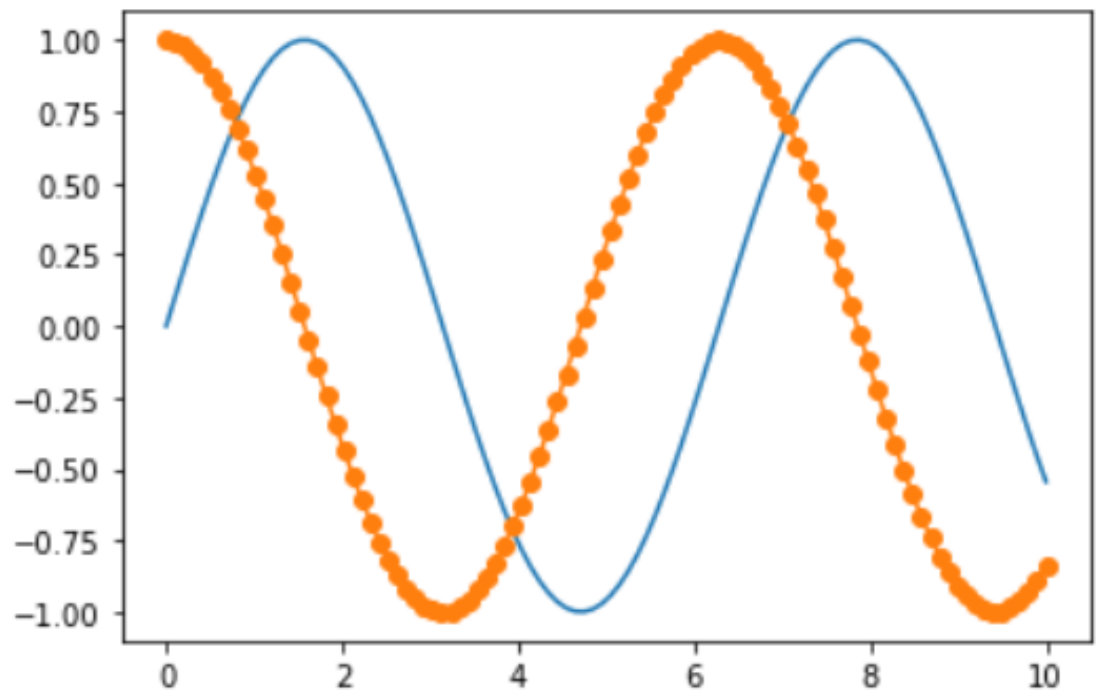
- 무슨 데이터인지 표시해주는 방법

(4) legend

- 좌표축에 범례를 추가해주는 방법
- loc : legend 위치
- 'best' 0
- 'upper right' 1
- 'upper left' 2
- 'lower left' 3
- 'lower right' 4
- 'right' 5
- 'center left' 6
- 'center right' 7
- 'lower center' 8
- 'upper center' 9
- 'center' 10

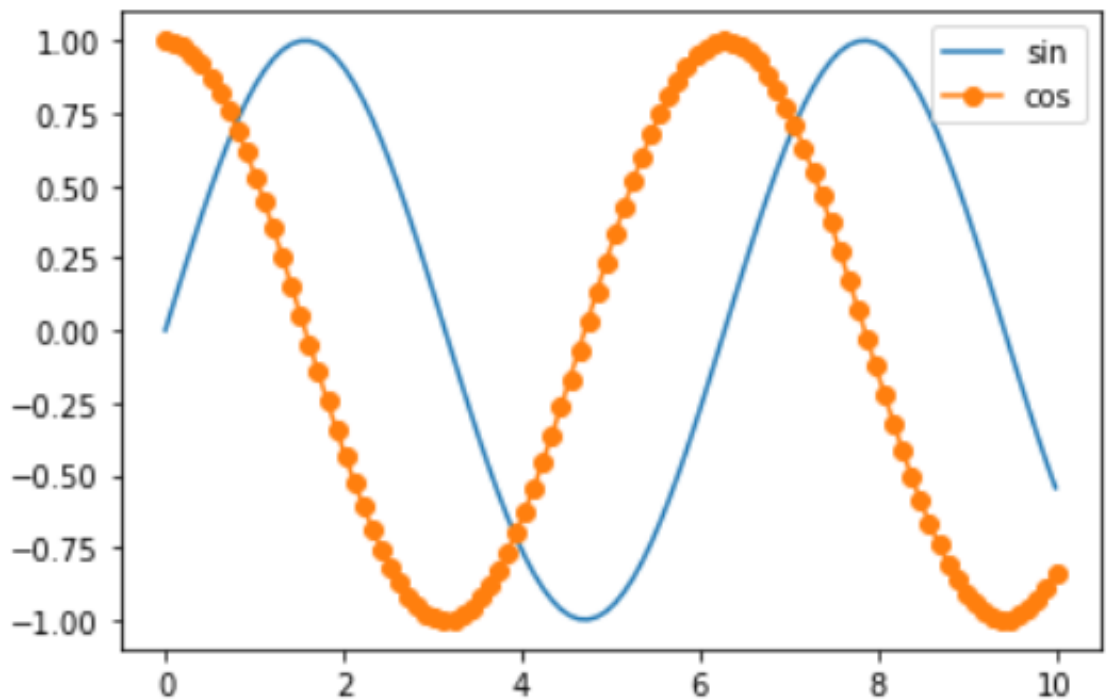
In [9]:
```
x=np.linspace(0,10,100)
y=np.sin(x)
y_=np.cos(x)

plt.plot(x,y)
plt.plot(x,y_, '-o')
```

Out[9]:
```
[<matplotlib.lines.Line2D at 0x1eaa1c5f10>]
```



In [10]:
```
x=np.linspace(0,10,100)
y=np.sin(x)
y_=np.cos(x)

plt.plot(x,y, label='sin')
plt.plot(x,y_, '-o', label='cos')

plt.legend(loc=1)   # loc= 1,2,3,4
```

Out[10]:
```
<matplotlib.legend.Legend at 0x1eaa36f4c0>
```

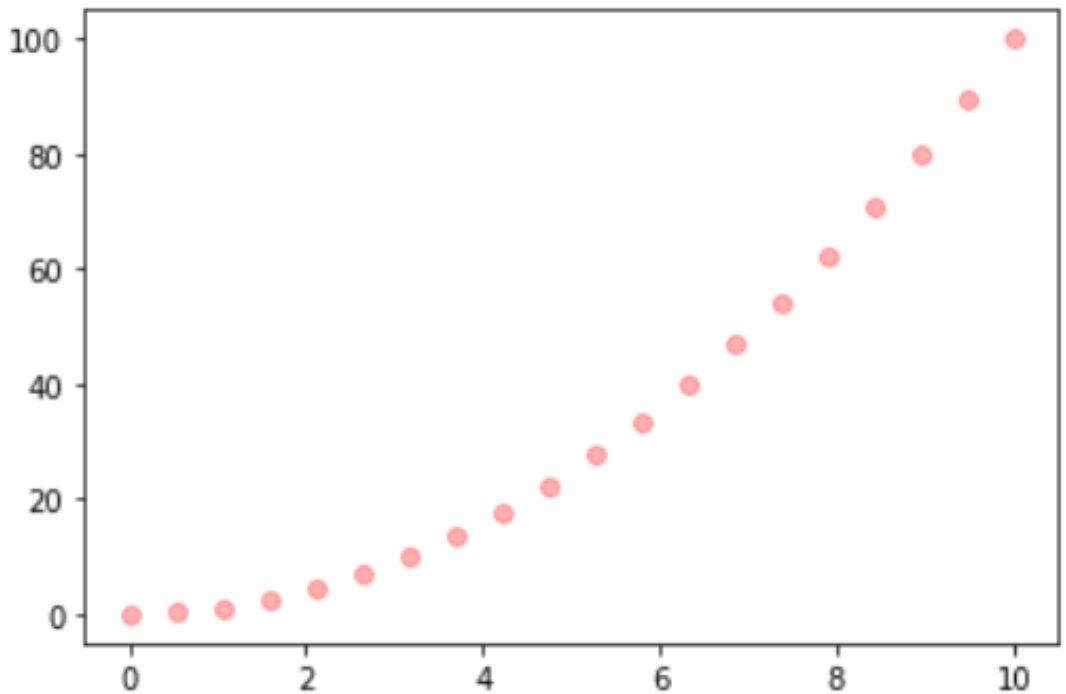


(5) Scatter Plot

- 변수들의 상관관계, 밀집 위치를 나타낸다.

- **scatter(산점도)**

In [11]:
```
x=np.linspace(0,10,20)
y=x**2

plt.scatter(x,y, c='r', alpha=0.3)  # color or c
plt.show()
```



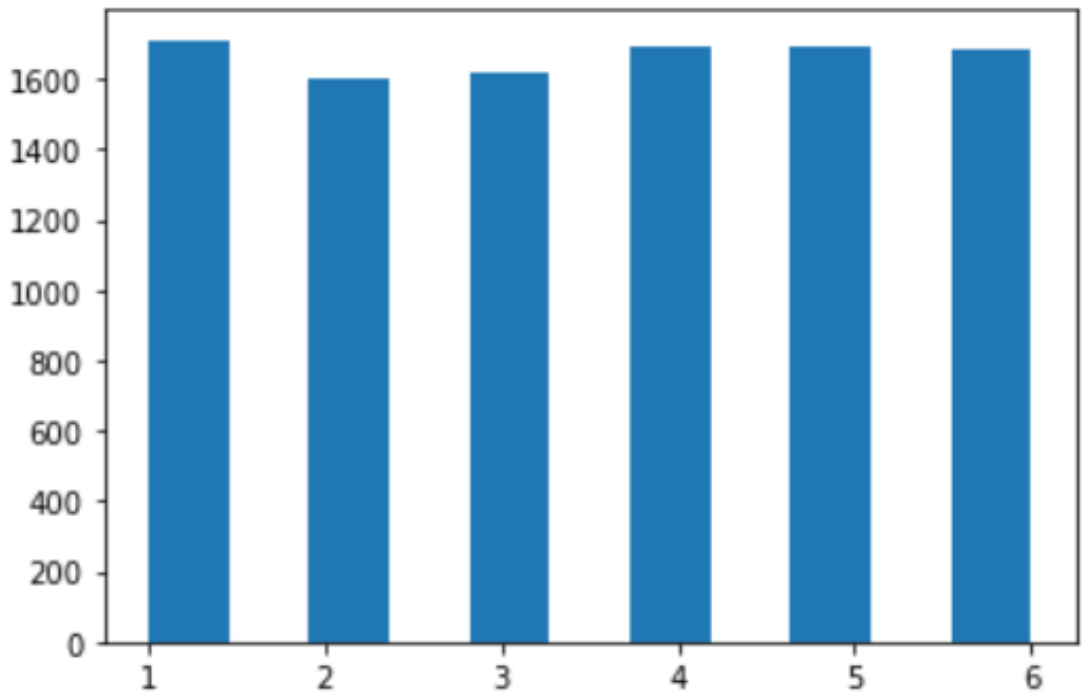
(6) histogram

- 도수분포표를 그래프로 나타낸 방법이다.
- 막대그래프는 계급 즉 가로를 생각하지 않고 세로의 높이로만 나타낸다. (출처 : 위키백과)
- 연속형 자료를 계급으로 나누어 계급별 도수를 막대로 나타낸다.

- **histogram**

In [12]:
```
x=[np.random.randint(1,7) for i in range(10000)]  # 10  100  10000
```

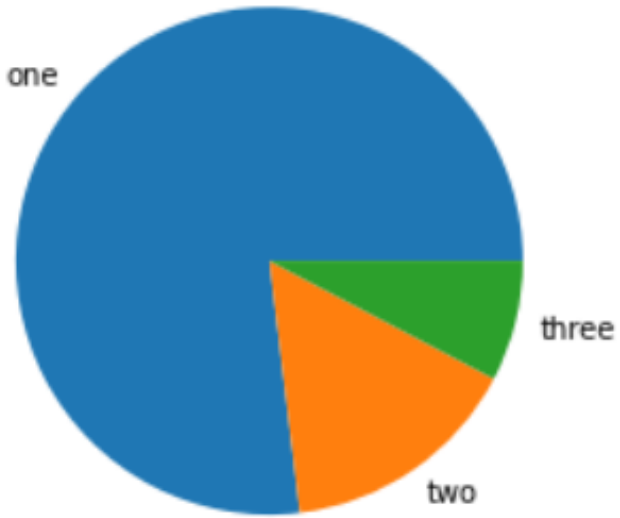
In [13]:
```
plt.hist(x, bins=11)
plt.show()
```



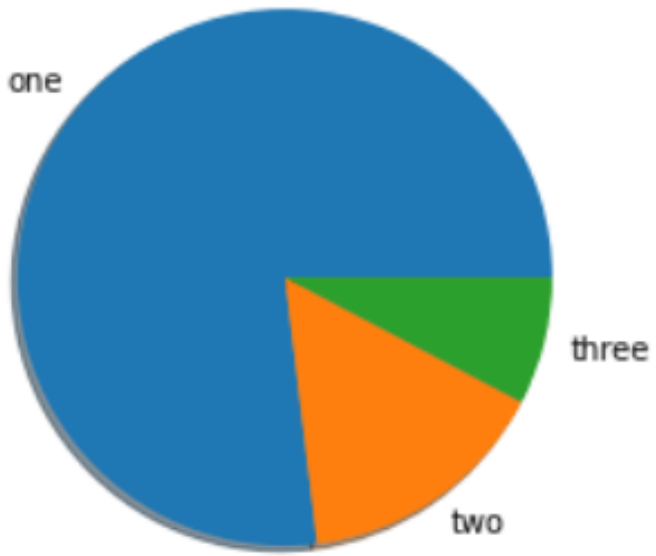
(7) Pie Chart

- **Pie Chart**

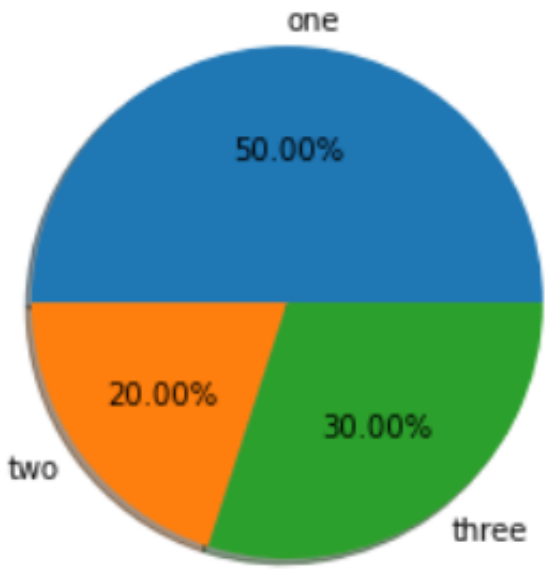
In [14]:
```
labels=['one','two','three']
size=[100,20,10]

plt.pie(size, labels=labels)
plt.show()
```



In [15]:
```
labels=['one','two','three']
size=[100,20,10]

plt.pie(size, labels=labels, shadow=True)
plt.show()
```



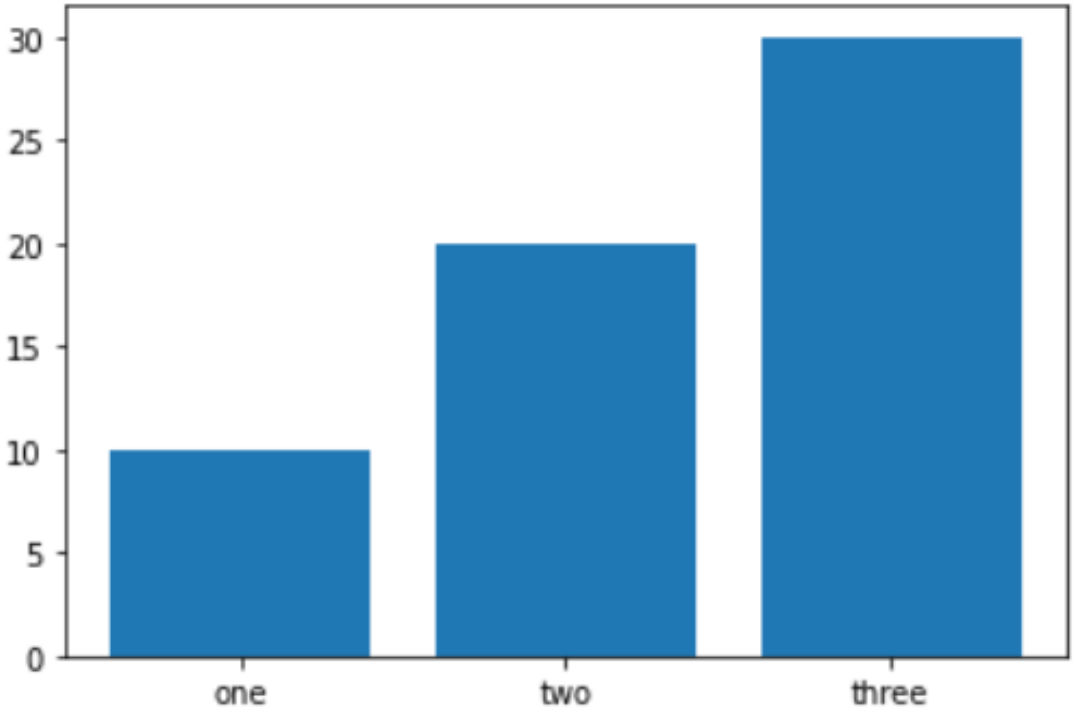
In [16]:
```
labels=['one','two','three']
size=[50,20,30]

plt.pie(size, labels=labels, shadow=True, autopct='%1.2f%%')
plt.show()
```



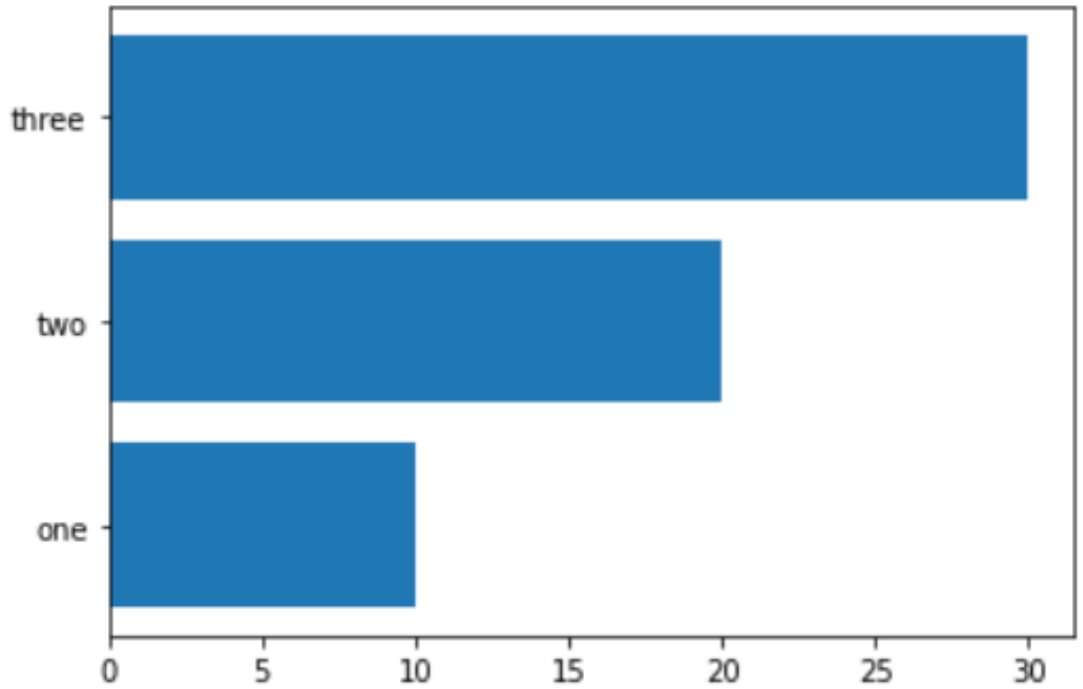
(8) Bar Chart

- 범주형 자료의 분포를 파악한다.
  - **Bar Chart**

```
In [17]: plt.bar(['one','two','three'],[10,20,30])
         plt.show()
```



In [18]:
```
plt.barh(['one','two','three'],[10,20,30])
plt.show()
```



(9) 선 색상과 스타일

- plt.plot(x, y, color='#ff0000') -> 16진수 색상 값

- plt.plot(x, y, linestyle='solid') -> 실선('-')
- plt.plot(x, y, linestyle='dashed') -> 파선('--')
- plt.plot(x, y, linestyle='dashdot') -> 1점 쇄선('-.')
- plt.plot(x, y, linestyle='dotted') -> 점선(':')

- plt.plot(x, y, '--r') -> 빨간색 파선

- '.' point marker
- 'o' circle marker(plt.plot(x, y_, '-o'))
- '^' triangle_up marker
- 's' square marker
- 'p' pentagon marker
- '*' star marker
- '+' plus marker
- 'x' x marker

- **공식문서 :**
- https://matplotlib.org/3.2.1/api/_as_gen/matplotlib.pyplot.plot.html](https://matplotlib.org/3.2.1/api/_as_gen/matplotlib.pyplot.plot.html
- https://matplotlib.org/3.2.1/api/pyplot_summary.html](https://matplotlib.org/3.2.1/api/pyplot_summary.html

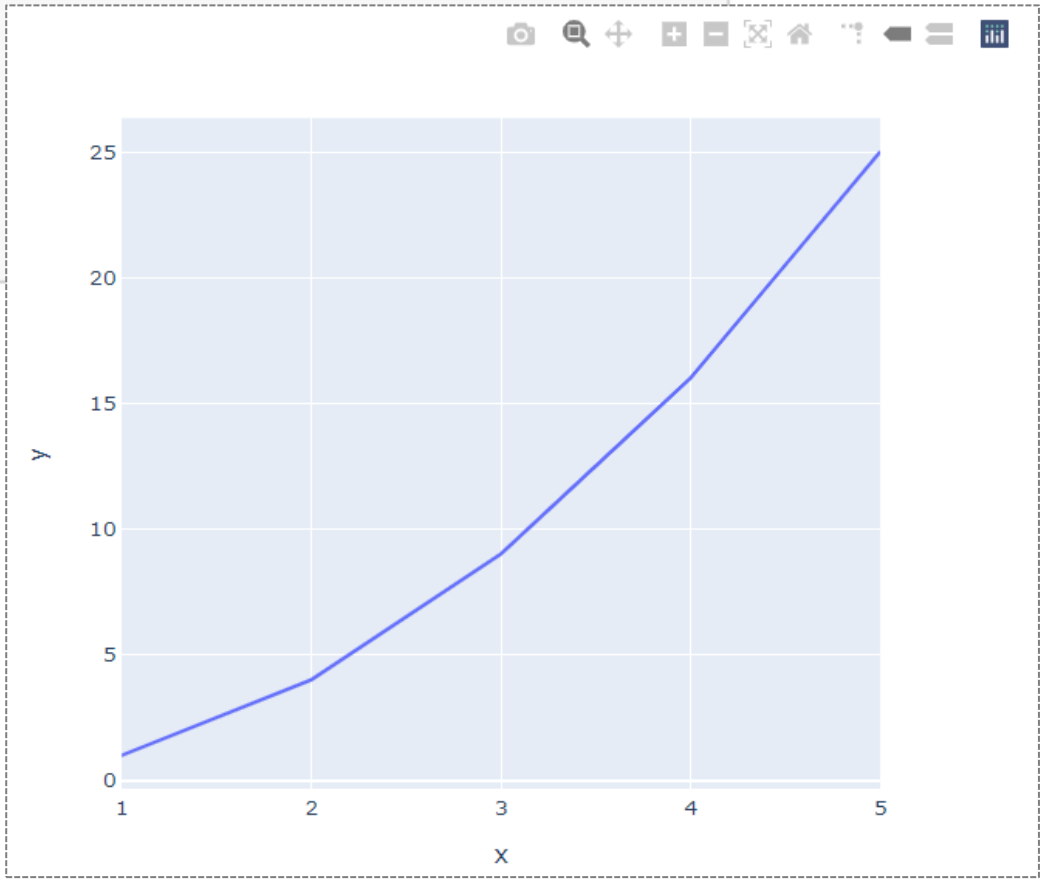
2. Plotly

- 인터랙티브한 그래프를 그리기에 적합한 패키지
- 웹 시각화인 자바스크립트의 라이브러리 D3를 이용해 그래프가 웹에서 빠르게 그려진다.
- 공식홈페이지(https://plotly.com/python/) 튜토리얼

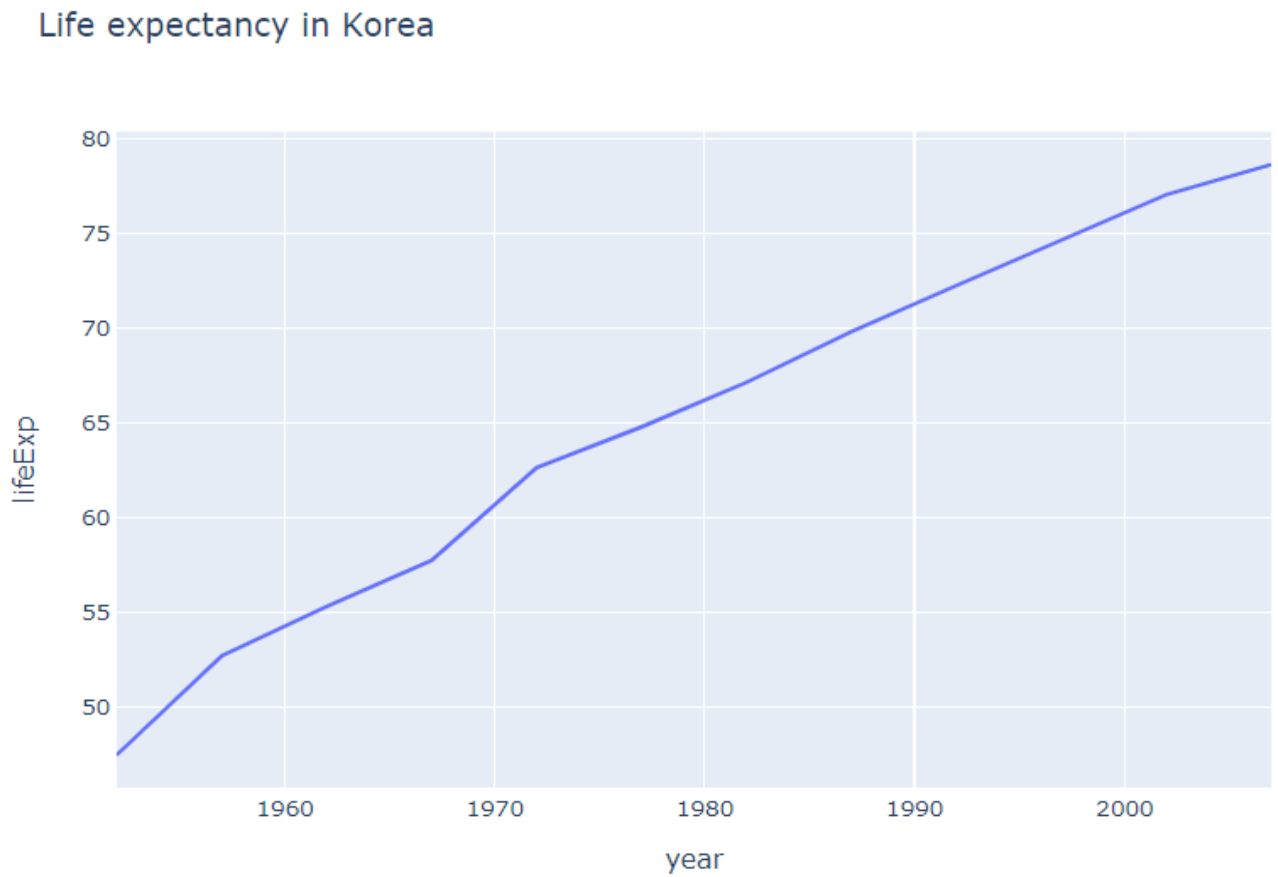
(1) Line Plot

- **Line Plot**

In [1]:
```
import plotly.express as px  # anaconda navigator에서 설치
import numpy as np

x_ = np.array([1,2,3,4,5])
y_ = x_**2
fig = px.line(x=x_,y=y_)
fig.show()
```



In [2]:
```
import plotly.graph_objects as go

# gapminder 데이터 셋 사용하기
korea_life = px.data.gapminder().query("country == 'Korea, Rep.'") # 한국인 기대 수명
fig = px.line(korea_life, x="year", y="lifeExp", title='Life expectancy in Korea')
fig.show()
```



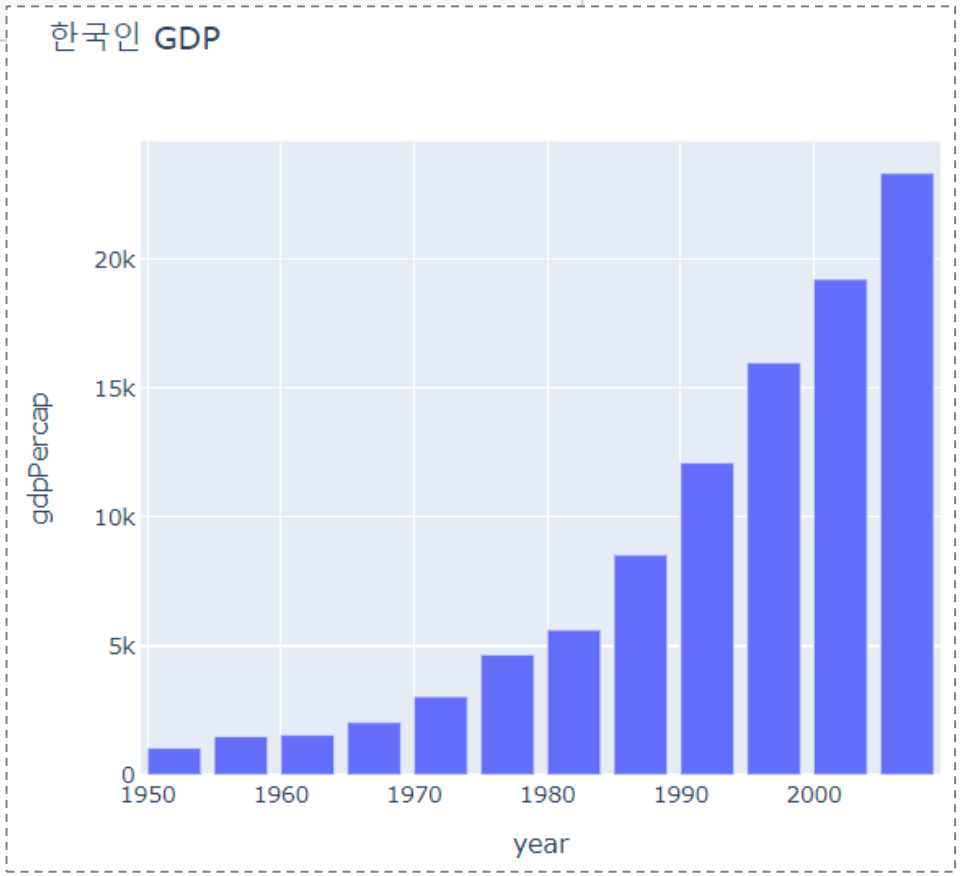
In [3]:
```
korea_life["year"]
korea_life["lifeExp"]
```

Out[3]:
```
840    47.453
841    52.681
842    55.292
843    57.716
844    62.612
845    64.766
846    67.123
847    69.810
848    72.244
849    74.647
850    77.045
851    78.623
Name: lifeExp, dtype: float64
```

(2) Bar Chart

- **Bar Chart**

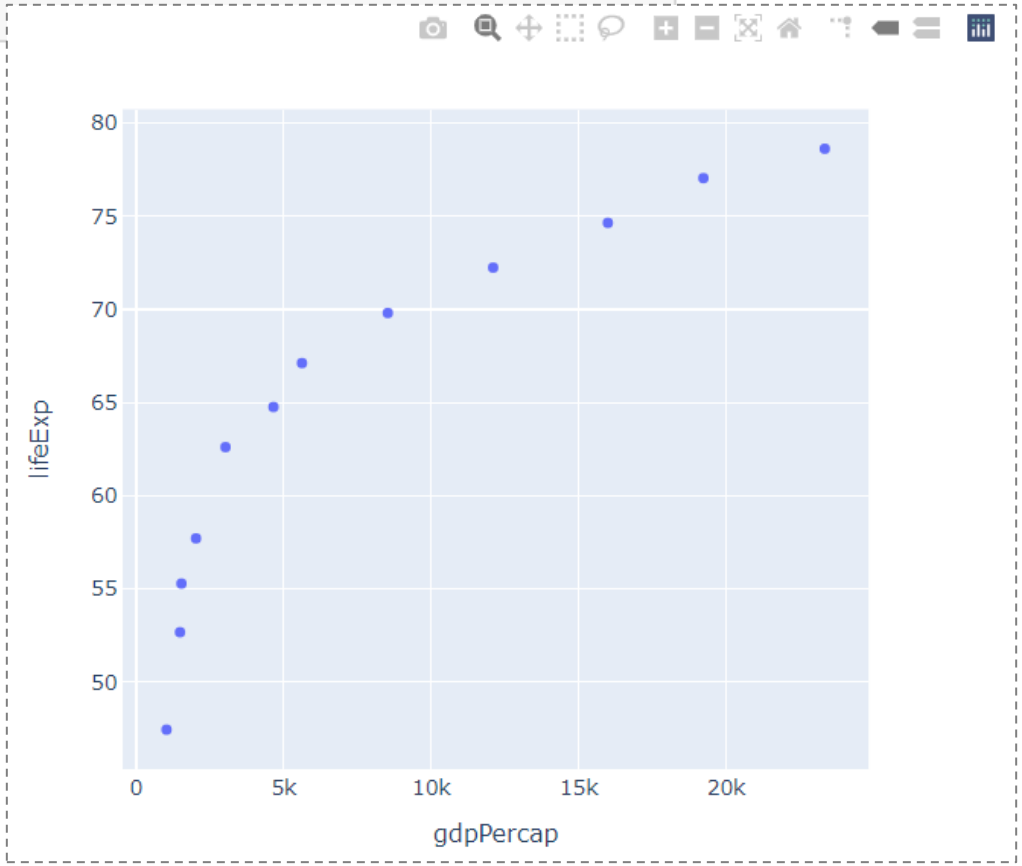
In [4]:
```
# 한국 GDP 데이터 셋
korea_gdp = px.data.gapminder().query("country == 'Korea, Rep.'")
fig = px.bar(korea_gdp, x='year', y='gdpPercap', title = "한국인 GDP")
fig.show()
```



(3) Scatter Plot

- **Scatter Plot**

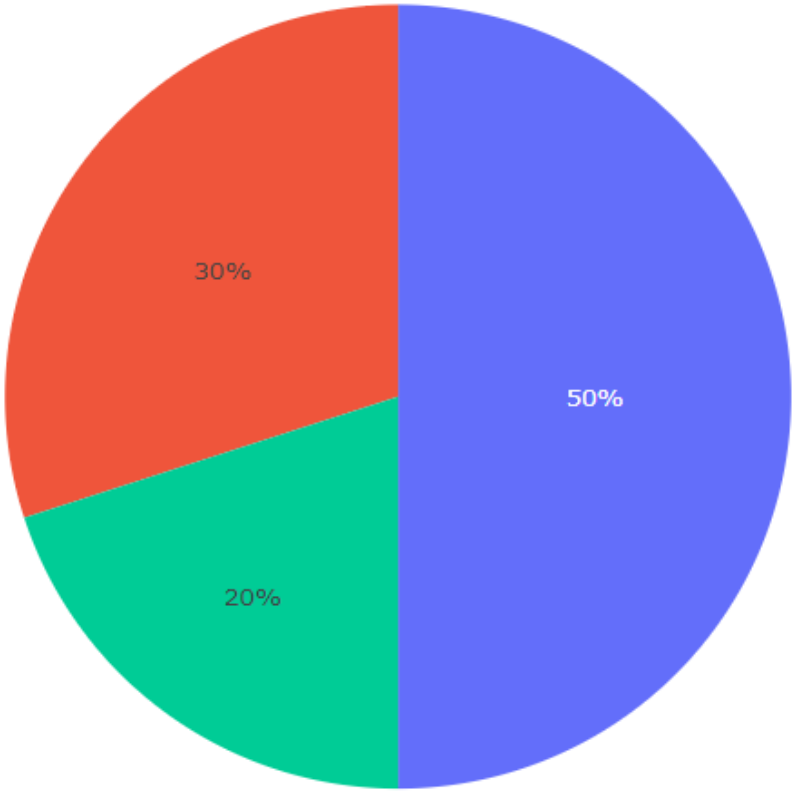
In [5]:
```
# 한국 기대 수명과 GDP의 상관관계
korea_data = px.data.gapminder().query("country == 'Korea, Rep.'")
fig = px.scatter(korea_data, x = 'gdpPercap', y = 'lifeExp')
fig.show()
```



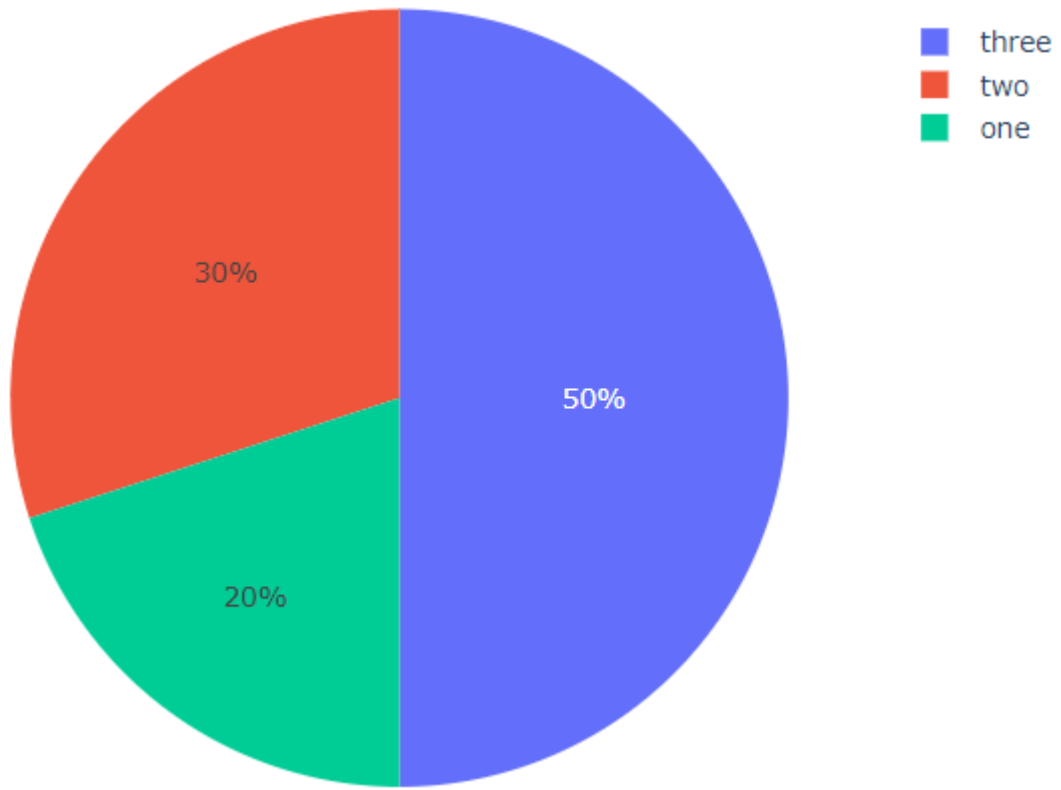
(4) Pie Chart

- **Pie Chart**

In [6]:
```
fig = px.pie(values = [20, 30, 50])
fig.show()
```



In [7]:
```
labels = ['one', 'two', 'three']
values = [20, 30, 50]
fig =go.Figure(data = [go.Pie(values = values, labels = labels)]) # 범례 추가하기
fig.show()
```



3. 이미지 분석

- 이미지는 작은 사각형 모양의 픽셀을 모아서 구성되어 있다.
- 이미지 크기는 (세로 픽셀수 X 가로 픽셀수)로 표현한다.
- 이미지를 저장할 때는 색을 표현하는 스칼라 값이나 2차원 vector로 표현한다.
- RGB는 색공간을 말한다. Red, Green, Blue로 세 개가 합쳐진 벡터이다.

- skimage(scikit-image) : 이미지 처리를 위한 파이썬 라이브러리이다. numpy array로 동작한다.
- PIL(Python Image Library)
  - Pillow : 이미지 처리와 그래픽 기능을 제공하는 파이썬 라이브러리이다.
    (더이상 PIL은 지원하지 않고 그 후속 프로젝트인 Pillow를 사용한다)
- cv2 : OpenCV는 Open Source Computer Vision Library의 약자로 오픈소스 컴퓨터 비전 및 머신러닝 라이브러리이다. 이 라이브러리를 불러올 때는 cv2를 사용한다.

In [1]:
```
import numpy as np
from skimage import io
# from PIL import Image
# from cv2
import matplotlib.pyplot as plt
```

In [2]:
```
pic = io.imread('pic.jpg') # 이미지 불러오기
```

In [3]:
```
type(pic)
```

Out[3]:
```
numpy.ndarray
```

In [4]:
```
pic.shape
```

Out[4]:
```
(183, 275, 3)
```

In [5]:
```
pic
```

Out[5]:
```
array([[[202, 224, 248],
        [203, 225, 249],
        [203, 225, 249],
        ...,
        [ 89, 154, 236],
        [ 89, 154, 236]
```

In [6]:
```
np.min(pic), np.max(pic)
```

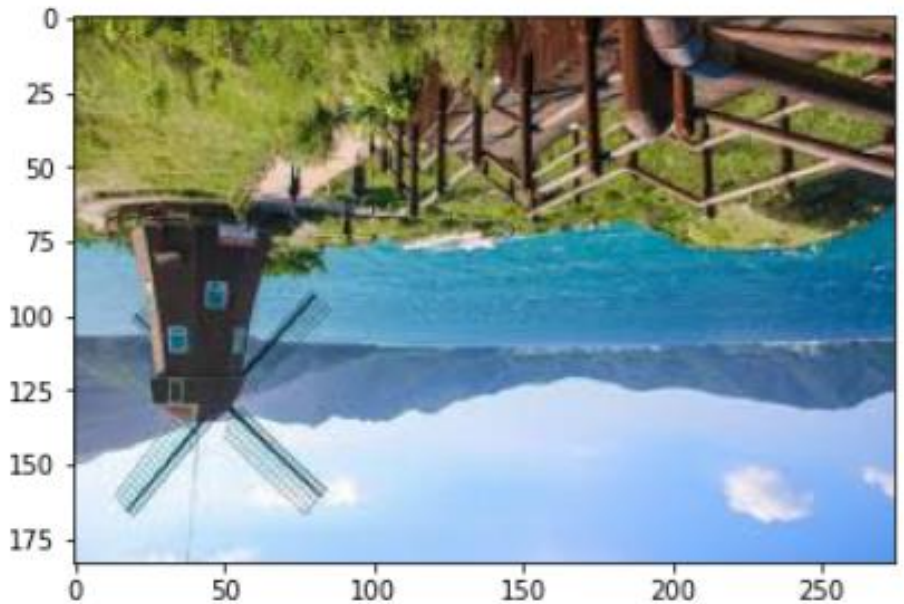
Out[6]: (0, 255)

In [7]:
```
plt.imshow(pic)
```

Out[7]: <matplotlib.image.AxesImage at 0xec76be1760>

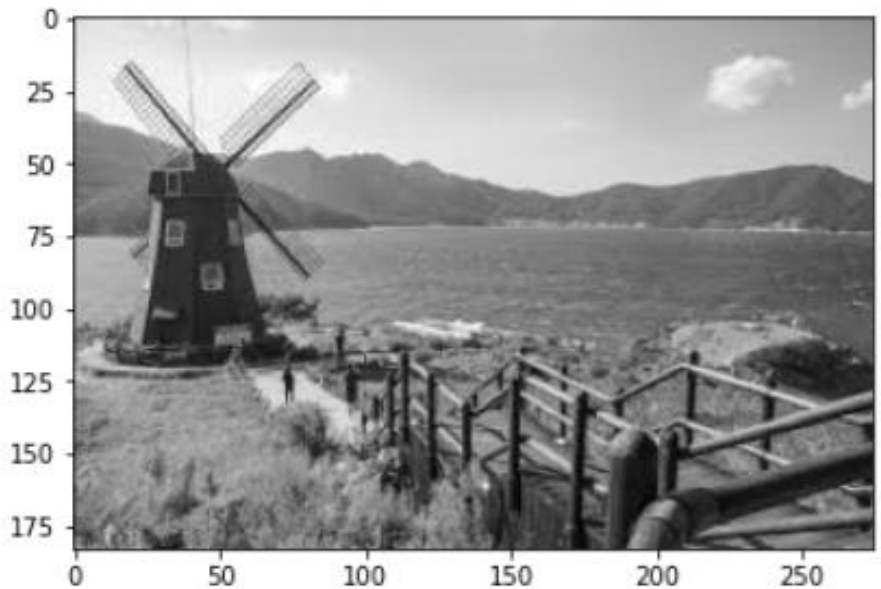
In [8]:
```
plt.imshow(pic[::-1]) # 상하로 뒤집기
```

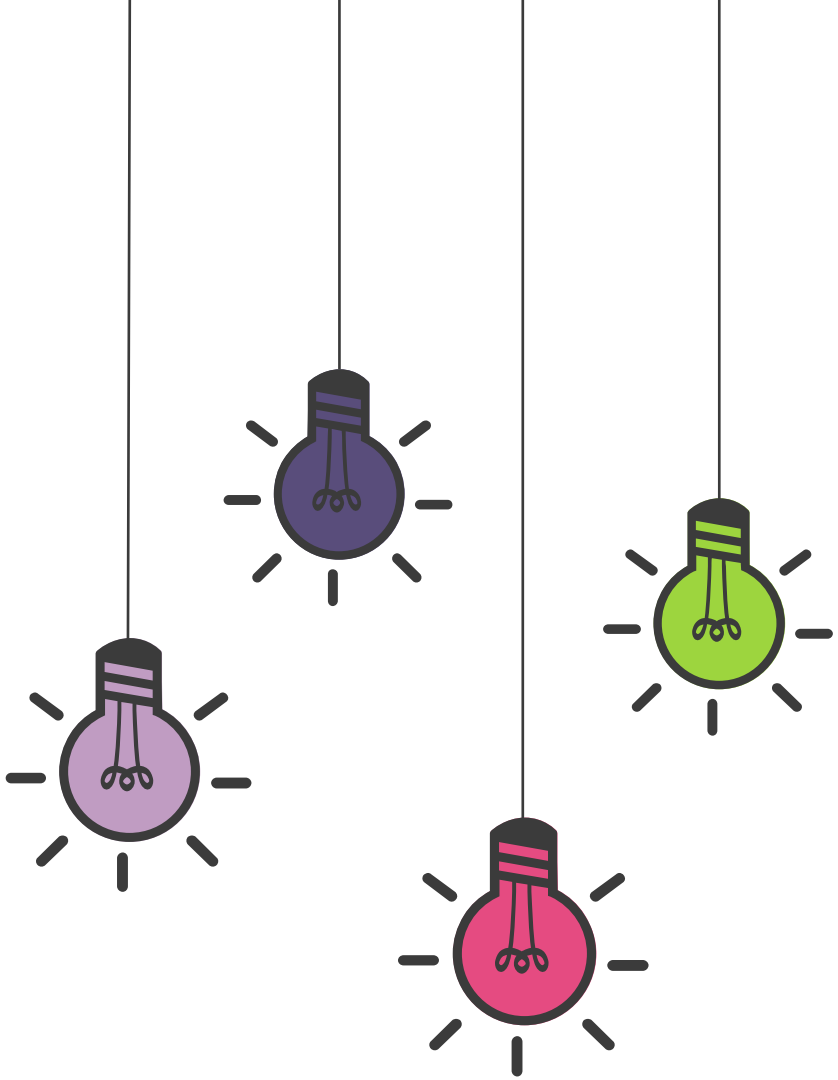
Out[8]: <matplotlib.image.AxesImage at 0xec76c8a910>



In [9]:
```
# 회색으로 이미지 출력하기
from skimage import color

plt.imshow(color.rgb2gray(pic), cmap=plt.cm.gray)
```

Out[9]: <matplotlib.image.AxesImage at 0xec772c0220>





감사합니다

THANK YOU FOR WATCHING

김 계 희
jenni7@naver.com