파이썬을 이용한 시각화 기본

01. 라이브러리 불러오기

In [1]:
```
import matplotlib.pyplot as plt
import matplotlib
```

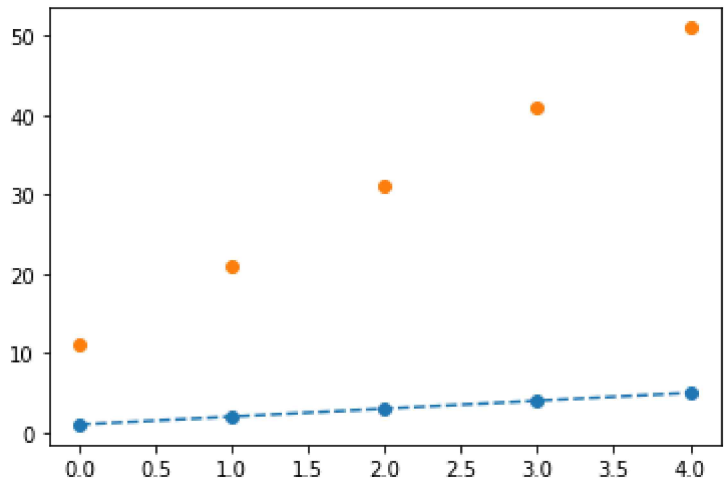
In [2]:
```
# 버전 확인 : version
print(matplotlib.__version__)
```

3.3.2

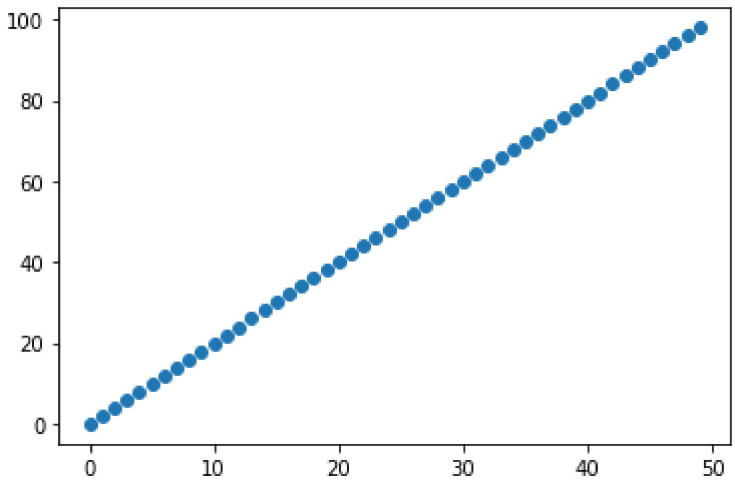
02. 그래프 그려보기

- 대부분의 Matplotlib 라이브러리에서 그래프를 그리는 것이 pyplot 서브 모듈 아래에 있다. 보통 약자로 plt를 이용합니다.
- 기본 plot() 함수로 그래프를 표시합니다.

In [3]:
```
import matplotlib.pyplot as plt


# marker='o'(점), '-'(선)
# marker 생략 가능
plt.plot([1,2,3,4,5], marker='o', linestyle="--")  # '-' 선 : 기본값
plt.plot([11,21,31,41,51], "o")  # '-' 선 : 기본값
plt.show()
```



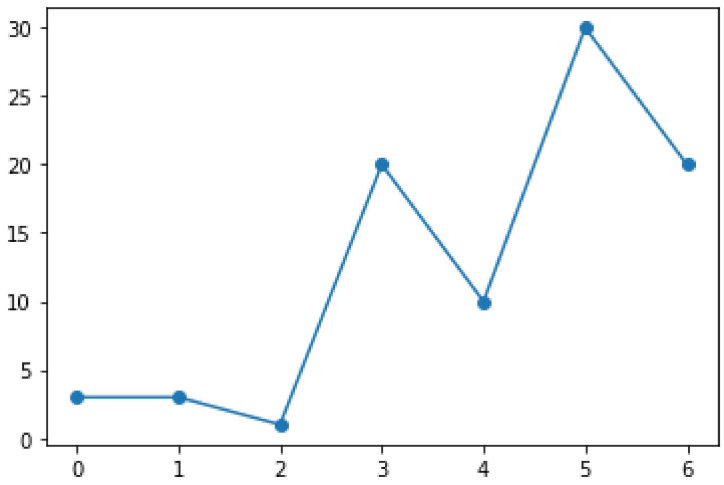
In [4]:
```
x = range(0, 50)
y = range(0, 100,2)
plt.plot(x, y, 'o')
plt.show()
```



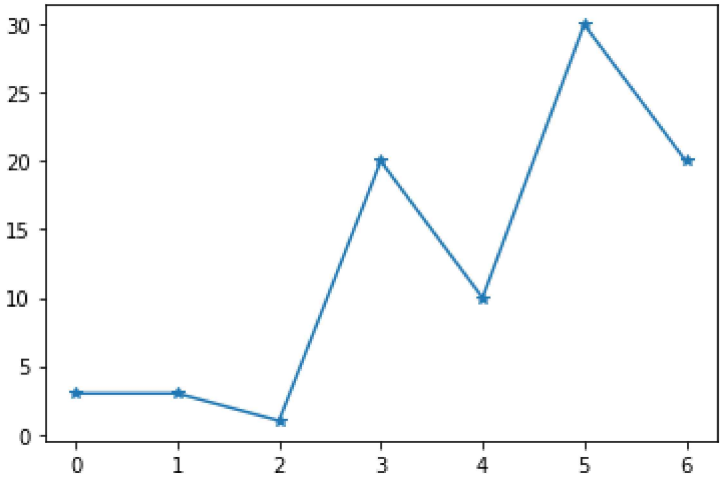
Matplotlib Markers

- marker 인자 키워드를 사용하여 특별한 점을 강조한다.

In [5]:
```
import numpy as np
ypoints = np.array([3, 3, 1, 20, 10, 30, 20])   # 7개의 y값 지정

plt.plot(ypoints, marker = 'o')
plt.show()
```



In [6]:
```
plt.plot(ypoints, marker = '*')
plt.show()
```

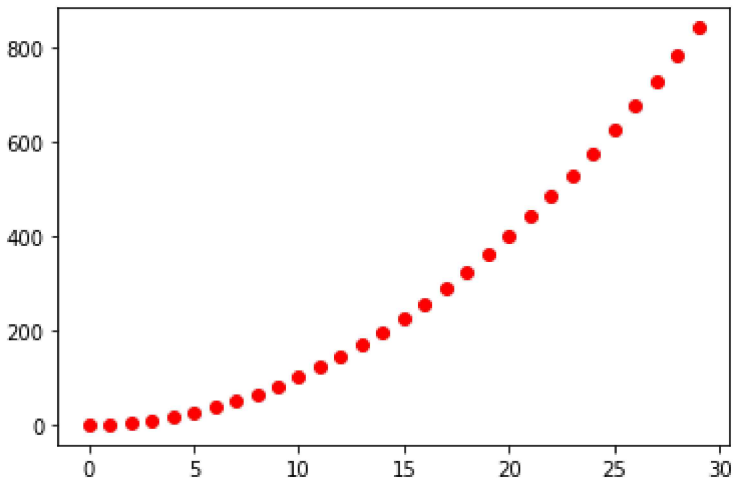


- Marker Reference (다양한 표현)
  - https://www.w3schools.com/python/matplotlib_markers.asp

y = x * x 의 그래프 그려보기

In [7]:
```
x = range(0, 30)
y = [ v * v for v in x ]
print(x)
print(y)

plt.plot(x, y, 'ro')  # r:빨간색, o:점
plt.show()
```

```
range(0, 30)
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361,
400, 441, 484, 529, 576, 625, 676, 729, 784, 841]
```



- 'ro'에서 'r'은 red를 의미하고, 'o'는 그래프의 마커 모양을 의미한다.

matplot의 주요 색상

- b : blue(파란색)
- g : green(녹색)
- r : red(빨간색)
- c : cyan(청록색)

- m : magenta(마젠타)
- y : yellow(노란색)
- k : black(검정색)
- w : white(흰색)

주요 마커

- o : circle(원)
- v : triangle_down(역삼각형)
- ^ : triangle_up(삼각형)
- s : square(네모)
- ■ : plus(플러스)
- . : point(점)

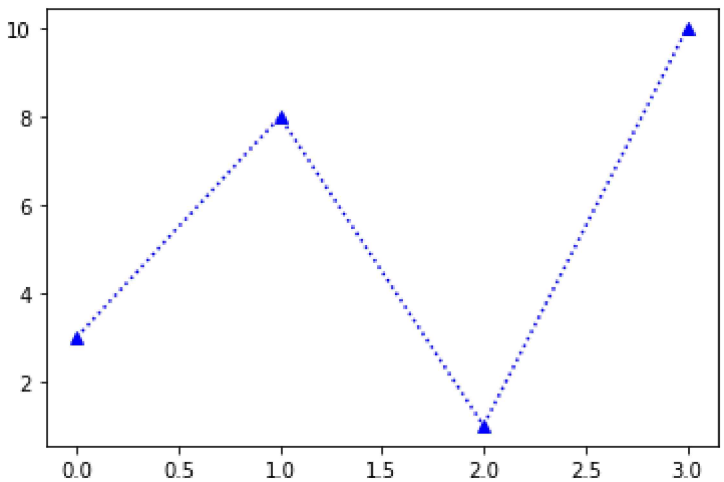
선

- '-' : Solid line
- ':' : Dotted line
- '--' : Dashed line
- '-.' : Dashed/dotted line

fmt의 파라미터를 다음과 같이 사용

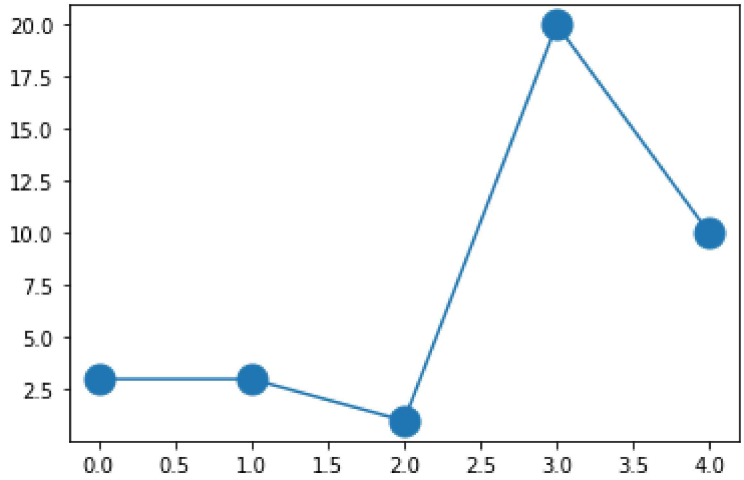
- marker | line | color

In [8]:
```
y = np.array([3, 8, 1, 10])

# ^:b => triangle_up(삼각형)| Dashed line | blue
plt.plot(y, '^:b')  #
plt.show()
```



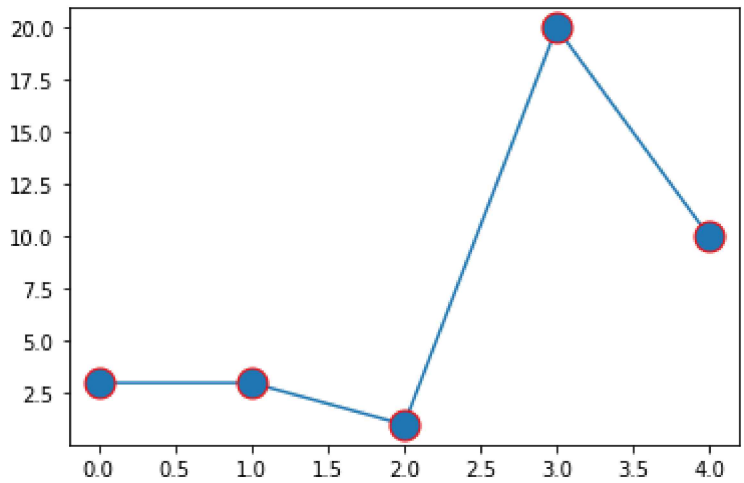
마커 사이즈**(ms)**

In [9]:
```
y = np.array([3, 3, 1, 20, 10])

plt.plot(y , 'o-', ms=15)   # o(marker), -(line)
plt.show()
```



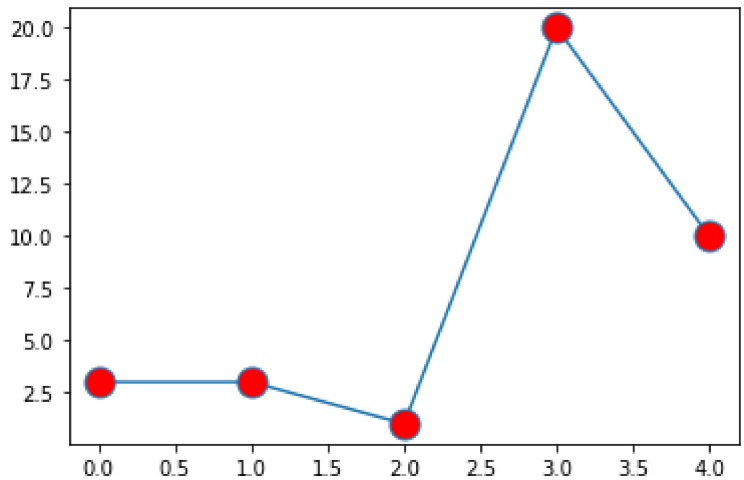
마커 Edge 색 - marker edge color(mec)

In [11]:
```
y = np.array([3, 3, 1, 20, 10])

plt.plot(y , marker = 'o', ms=15, mec = 'r')
plt.show()
```



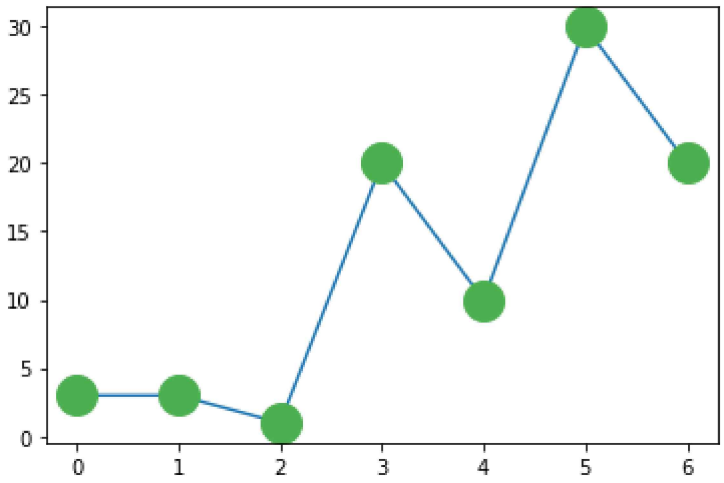
markerfacecolor

In [12]:
```
plt.plot(y , marker = 'o', ms=15, mfc = 'r')
plt.show()
```



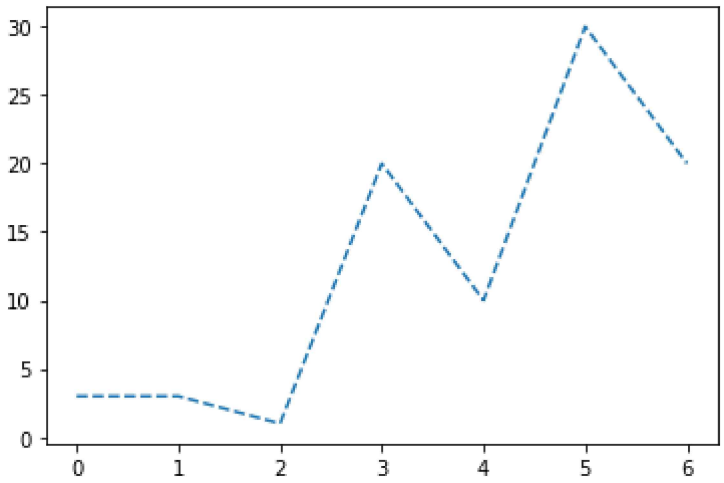
In [13]:
```
plt.plot(ypoints, marker = 'o', ms = 20, mec = '#4CAF50', mfc = '#4CAF50')
```

Out[13]:
```
[<matplotlib.lines.Line2D at 0x1a7d8f20100>]
```

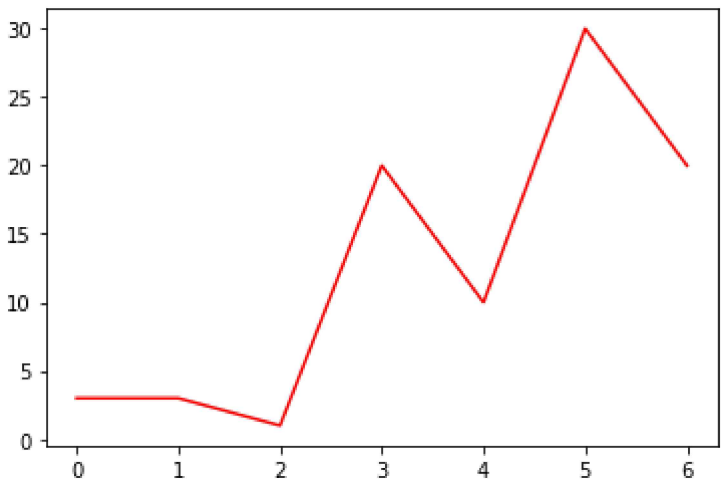


In [14]:
```
plt.plot(ypoints, linestyle = 'dashed')
```

Out[14]:
```
[<matplotlib.lines.Line2D at 0x1a7d8f79340>]
```

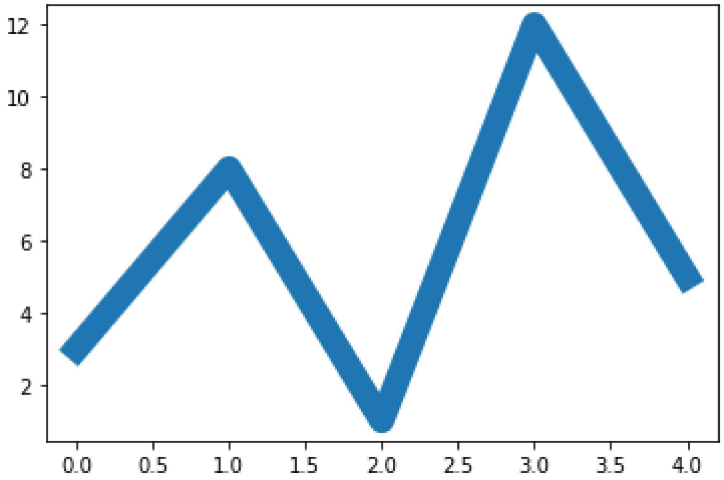


In [15]:
```
plt.plot(ypoints, color = 'r')
plt.show()
```



In [16]:
```
### 선 굵기
y = np.array([3, 8, 1, 12, 5])
plt.plot(y, linewidth = '12.5')
```

Out[16]: [<matplotlib.lines.Line2D at 0x1a7d8e5f640>]

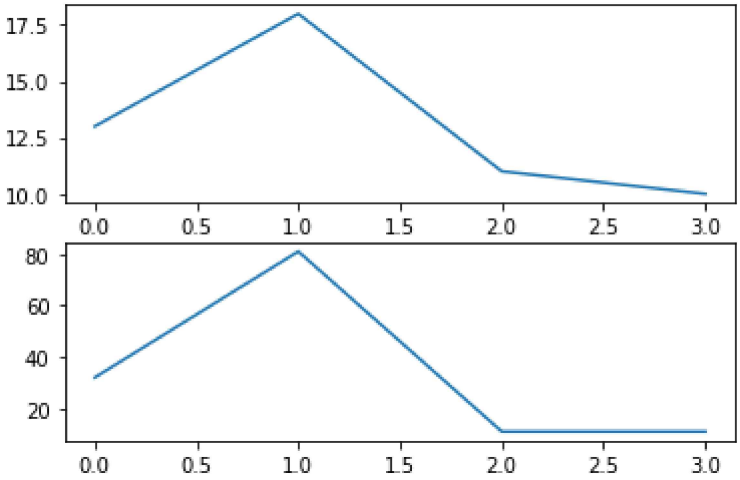


03. 여러 개의 그래프 그리기

- 한 화면에 여러개의 그래프를 그리기 위해서는

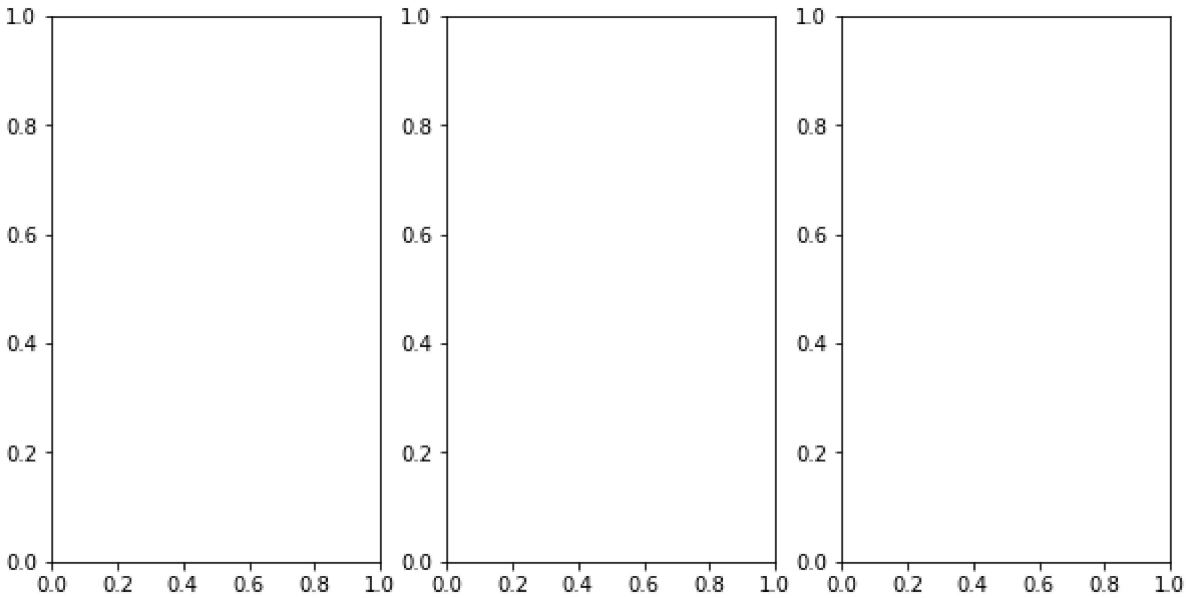
  (1) figure 함수를 통해 Figure 객체를 만든다.
  (2) add_subplot 메서드를 통해 그리려는 그래프 개수만큼 subplot를 만들면 된다.

In [17]:
```
fig = plt.figure()
ax1 = fig.add_subplot(2,1,1)
ax2 = fig.add_subplot(2,1,2)

x1 = np.array([0, 1, 2, 3])
y1 = np.array([13, 18, 11, 10])

x2 = np.array([0, 1, 2, 3])
y2 = np.array([32, 81, 11, 11])

ax1.plot(x1, y1)
ax2.plot(x2, y2)
plt.show()
```



In [18]:

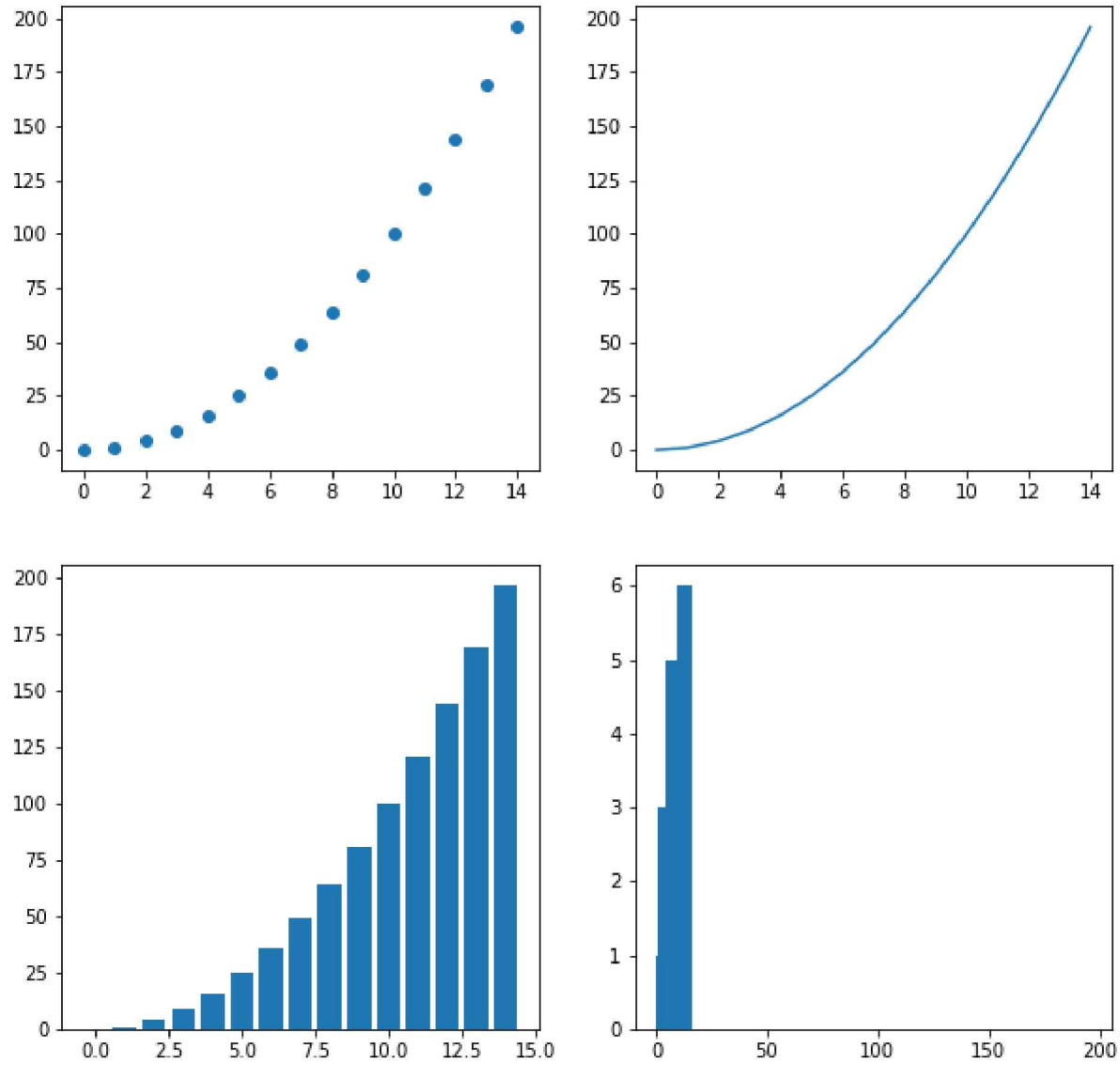
```
fig = plt.figure(figsize=(10,5))
ax1 = fig.add_subplot(1,3,1)  # 1행 3열 의 것중에 첫번째
ax2 = fig.add_subplot(1,3,2)  # 1행 3열 의 것중에 두번째
ax3 = fig.add_subplot(1,3,3)  # 1행 3열 의 것중에 두번째
plt.show()
```



plt.subplots(행, 열) 이용

In [19]:

```
### plot, bar 그래프 그리기
x = range(0,15)
y = [v*v for v in x]

fig, axs = plt.subplots(2,2, figsize=(10,10))
axs[0,0].plot(x, y, 'o')
axs[0,1].plot(x, y, '-')
axs[1,0].bar(x, y)
axs[1,1].hist(x, y)

plt.show()
```

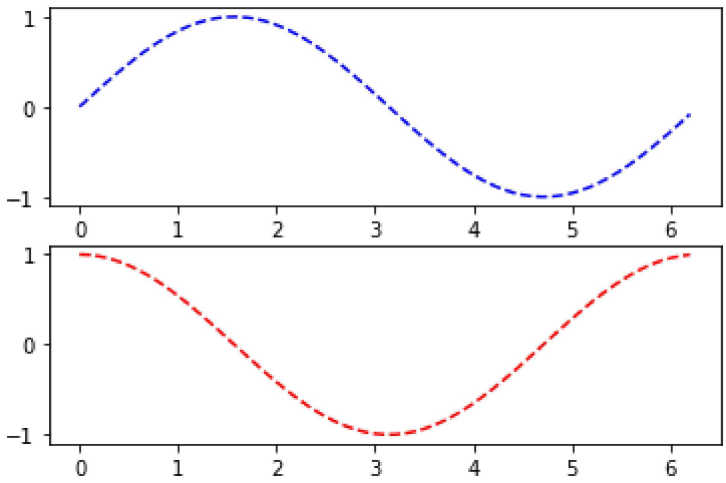


sin, cos 그래프 그려보기

In [20]:

```
import numpy as np
import matplotlib.pyplot as plt

x = np.arange(0.0, 2 * np.pi, 0.1)
sin_y = np.sin(x)
cos_y = np.cos(x)

plt.figure(figsize=(10,12))
fig, axs = plt.subplots(2)
axs[0].plot(x, sin_y, 'b--')
axs[1].plot(x, cos_y, 'r--')
plt.show()
```

<Figure size 720x864 with 0 Axes>



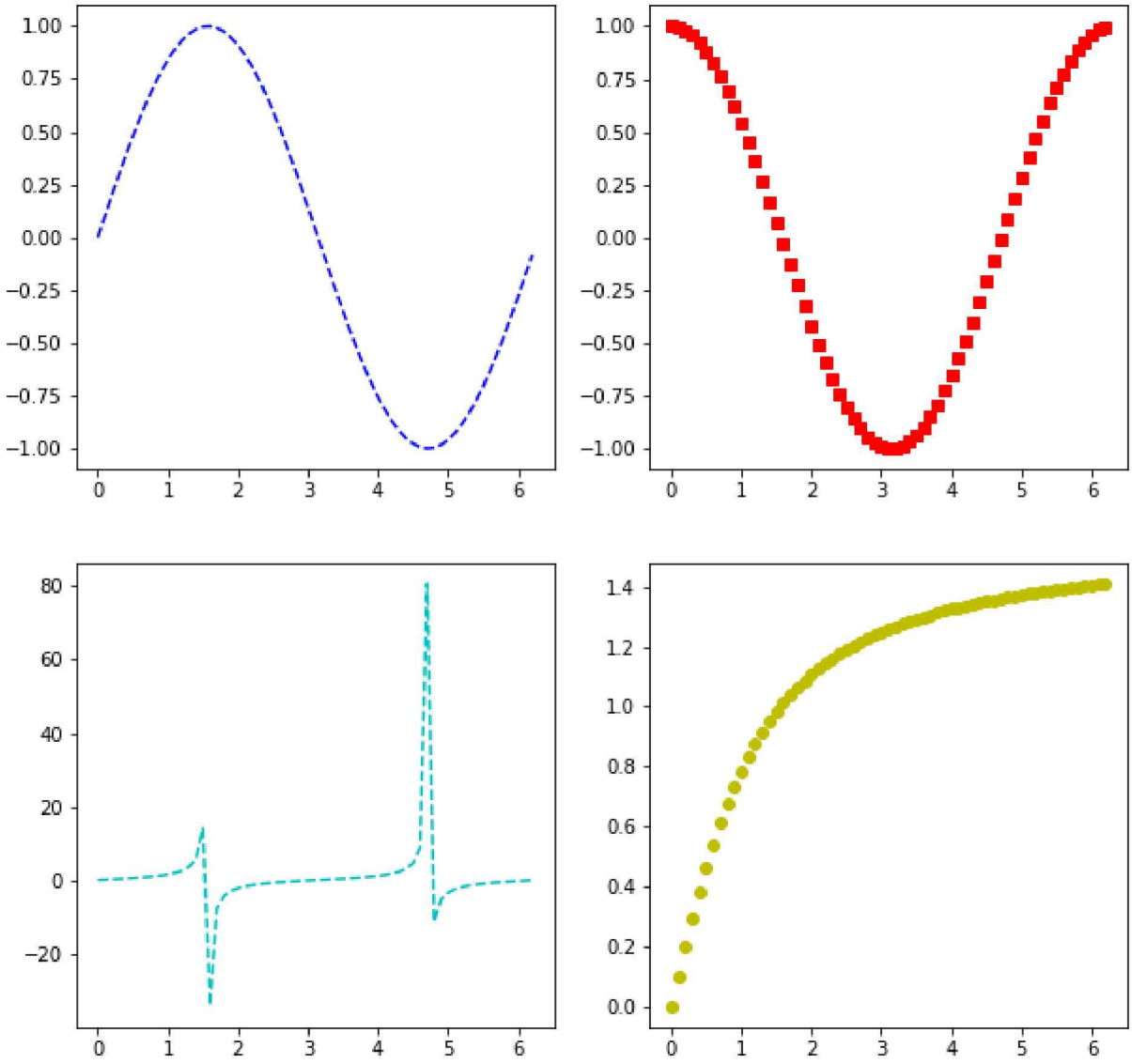
(실습2)

2행 2열의 그래프를 그려보자.
1행 1열: sin() 그래프
1행 2열: cos() 그래프 (표시 형식:빨간색 사각형)
2행 1열: tan() 그래프 (표시 형식:청록색 점)
2행 2열: arctan() 그래프 (표시 형식:노란색 원)

In [21]:

```
##### 2행 2열의 그래프
x = np.arange(0.0, 2 * np.pi, 0.1)
sin_y = np.sin(x)
cos_y = np.cos(x)
tan_y = np.tan(x)
arctan_y = np.arctan(x)

fig = plt.figure()
fig, axs = plt.subplots(2,2, figsize=(10,10)) #2행 2열

#1행 1열: sin() 그래프
axs[0,0].plot(x, sin_y, 'b--')

#1행 2열: cos() 그래프 (표시 형식:빨간색 사각형)
axs[0,1].plot(x, cos_y, 'rs')

#2행 1열: tan() 그래프 (표시 형식:청록색 점)
axs[1,0].plot(x, tan_y, 'c--')

#2행 2열: arctan() 그래프 (표시 형식:노란색 원)
axs[1,1].plot(x, arctan_y, 'yo')

plt.show()
```

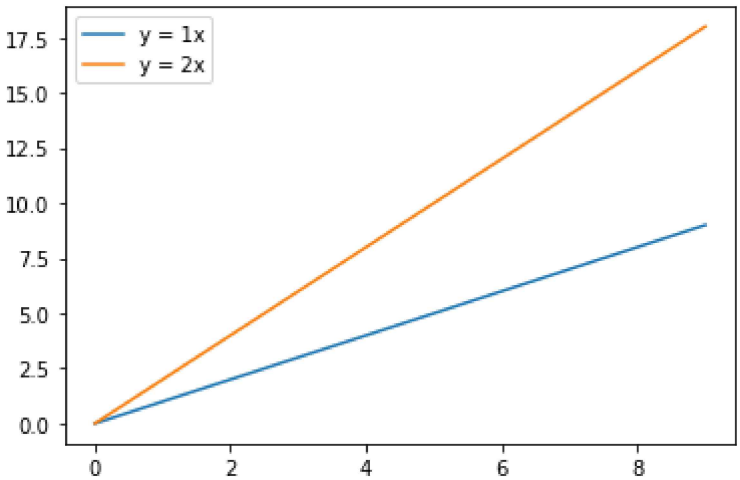
<Figure size 432x288 with 0 Axes>



04. 범례 표시 - 기본

In [22]:

```
x = np.arange(10)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.plot(x, 2*x, label='y = %ix' % 2)
plt.legend()
plt.show()
```

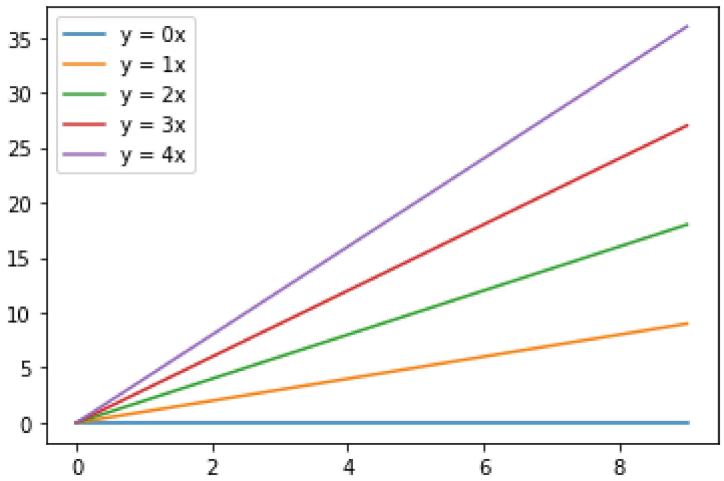


5개의 그래프 그려보기

In [23]:
```
x = np.arange(10)
list(x)
```

Out[23]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

In [24]:
```
for i in range(5):
    plt.plot(x, i*x, label='y = %ix' % i)
plt.legend()
plt.show()
```



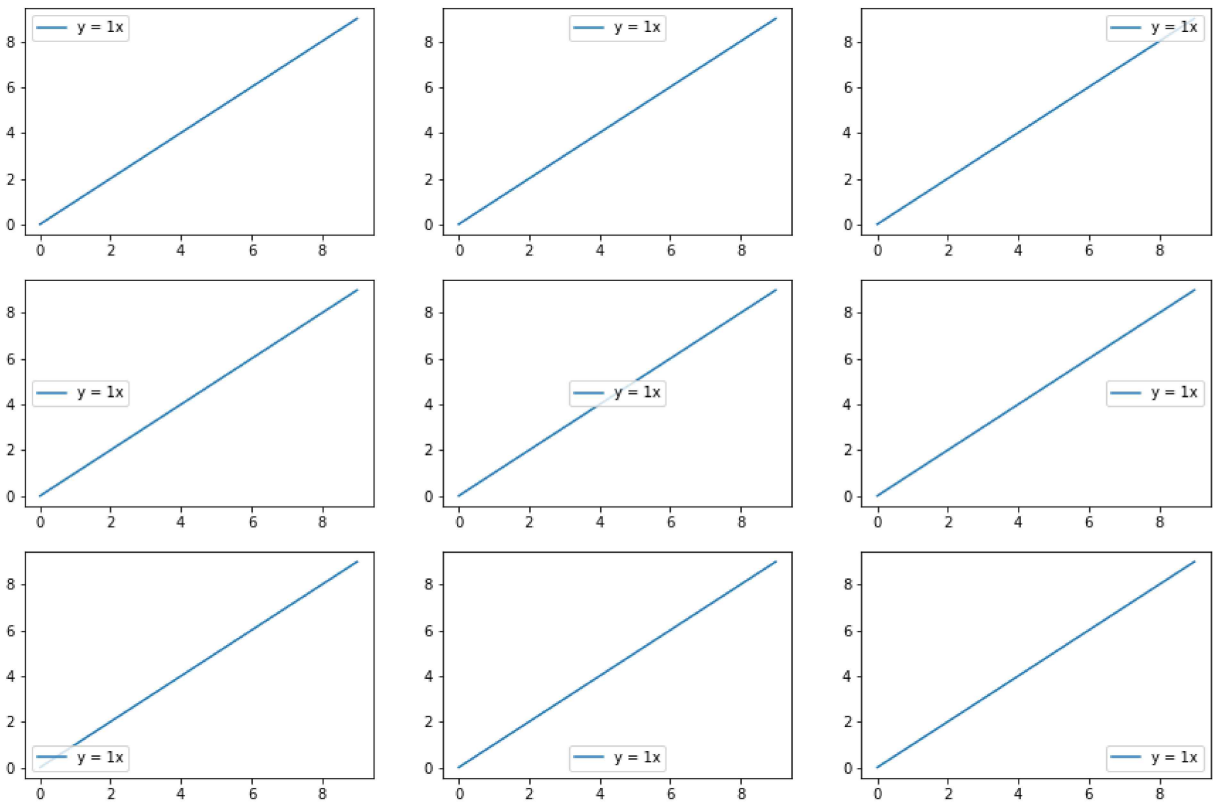
범례의 위치 변경과 **subplots()** 함수

- subplots() 함수는 3개의 인수를 사용한다.

In [25]:
```
plt.figure(figsize=(15,10))

plt.subplot(3,3,1)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='upper left')    # 2

plt.subplot(3,3,2)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='upper center')  # 9

plt.subplot(3,3,3)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='upper right')  # 1

plt.subplot(3,3,4)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='center left')  # 6
```

```
plt.subplot(3,3,5)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='center')  # 10

plt.subplot(3,3,6)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='center right')  # 5

plt.subplot(3,3,7)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='lower left')  # 3

plt.subplot(3,3,8)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='lower center')  # 8

plt.subplot(3,3,9)
plt.plot(x, 1*x, label='y = %ix' % 1)
plt.legend(loc='lower right')  # 4
```

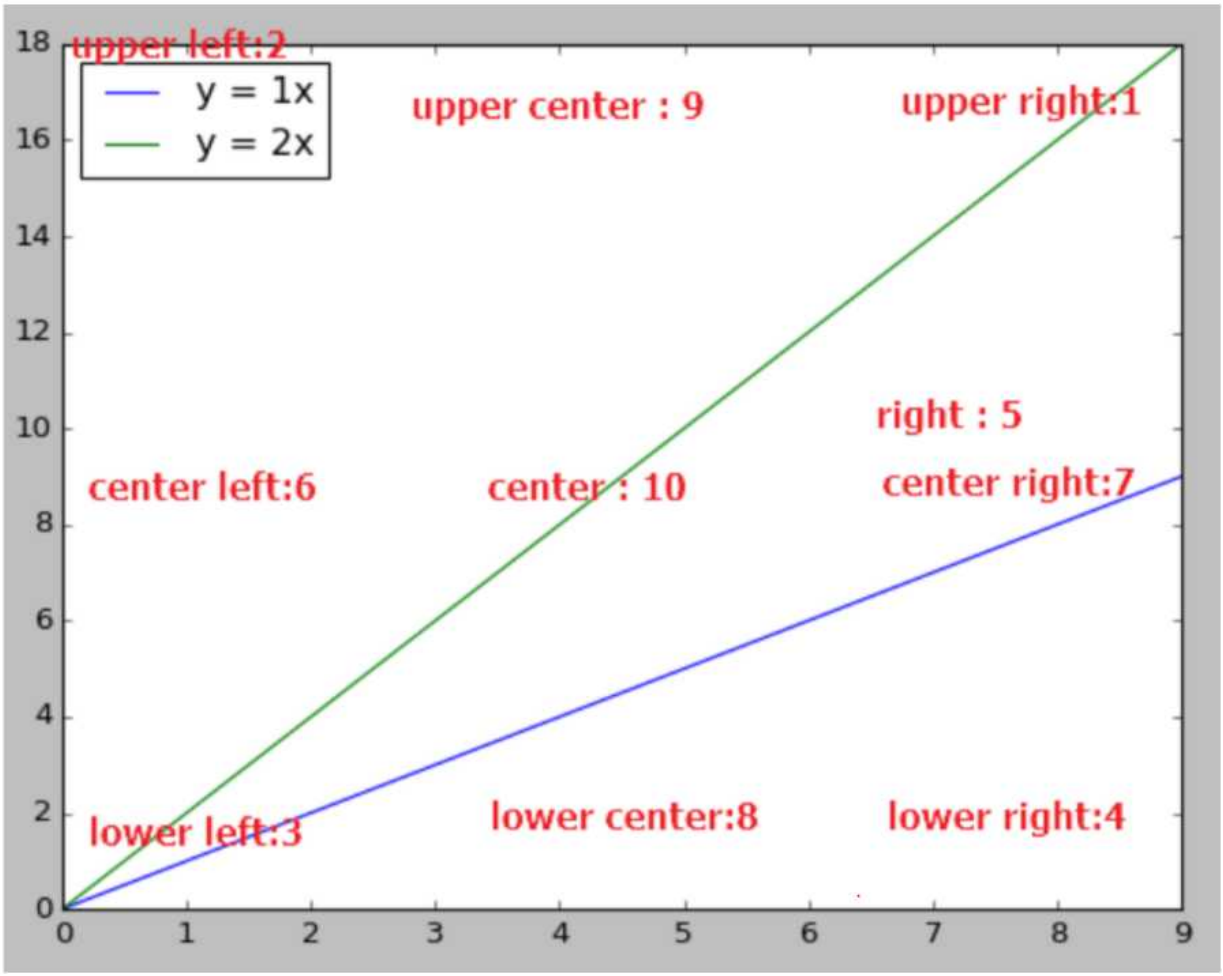
Out[25]: <matplotlib.legend.Legend at 0x1a7d8fa9d60>



In [26]:

```
import IPython.display as display
from PIL import Image
```

```
display.display(Image.open('./plt_legend_0622.png'))
```



In [27]:
```
import matplotlib.pyplot as plt
import matplotlib
import numpy as np

print(matplotlib.__version__)
```

3.3.2

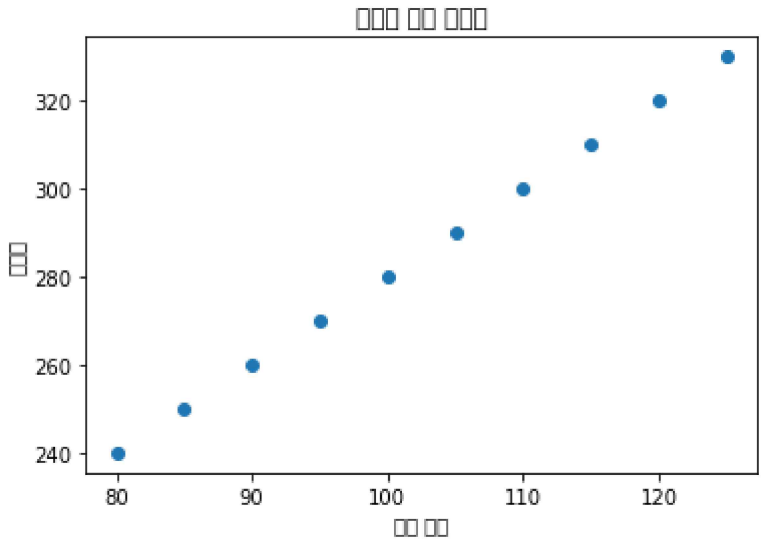
05. 타이틀과 레이블

In [28]:
```
import os, warnings
warnings.filterwarnings(action='ignore')
```

In [29]:
```
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.plot(x, y,'o')

plt.title("스포츠 시청 데이터")
```

```
plt.xlabel("평균 혈압")
plt.ylabel("칼로리")
```

Out[29]: Text(0, 0.5, '칼로리')



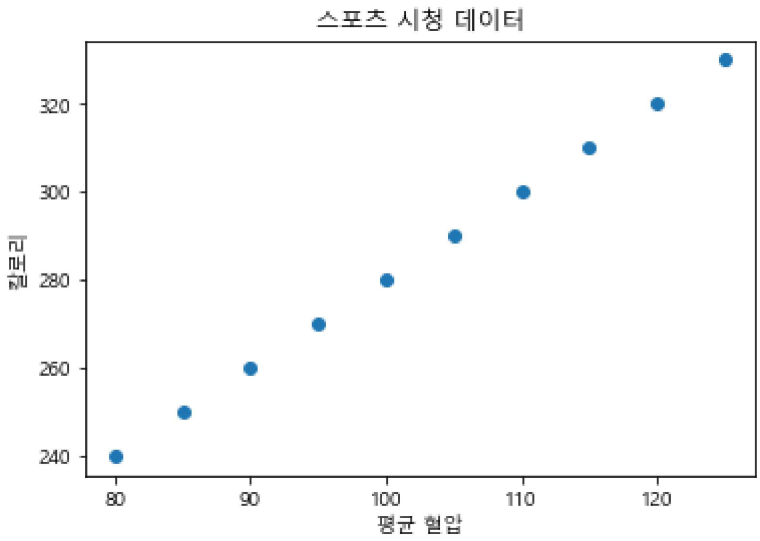
한글 표기

In [30]:
```
from matplotlib import font_manager, rc
import platform
import matplotlib.pyplot as plt
```

In [31]:
```
path = "C:/Windows/Fonts/malgun.ttf"
if platform.system() == "Windows":
    font_name = font_manager.FontProperties(fname=path).get_name()
    rc('font', family=font_name)
elif platform.system()=="Darwin":
    rc('font', family='AppleGothic')
else:
    print("Unknown System")

matplotlib.rcParams['axes.unicode_minus'] = False
```

In [32]:
```
plt.plot(x, y,'o')

plt.title("스포츠 시청 데이터")
plt.xlabel("평균 혈압")
plt.ylabel("칼로리")
```

Out[32]: Text(0, 0.5, '칼로리')

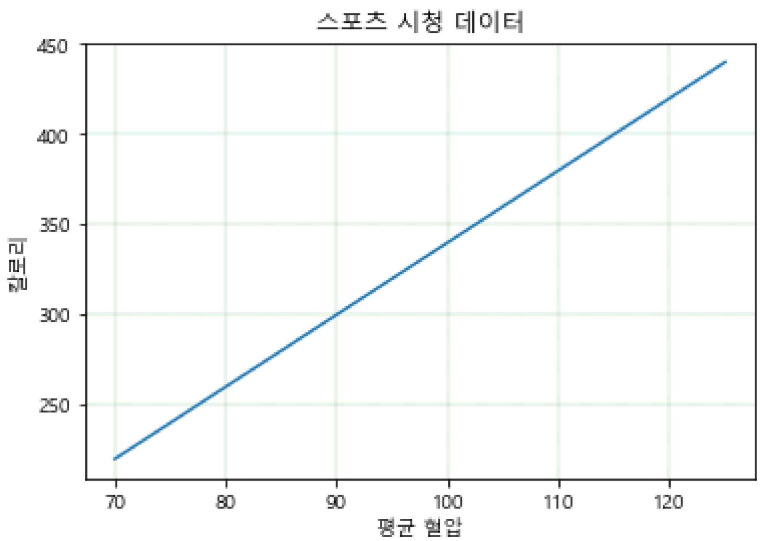


06. Grid

- 격자, 모눈이라는 뜻이다. 내용을 구성하는 데 사용되는 일련의 교차하는 직선 또는 곡선으로 구성된 구조.

In [33]:
```
x = np.array([70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125])
y = np.array([220, 240, 260, 280, 300, 320, 340, 360, 380, 400, 420,
440])

plt.title("스포츠 시청 데이터")
plt.xlabel("평균 혈압")
plt.ylabel("칼로리")

plt.grid(color = 'green', linestyle = '--', linewidth = 0.2)

plt.plot(x, y)
```

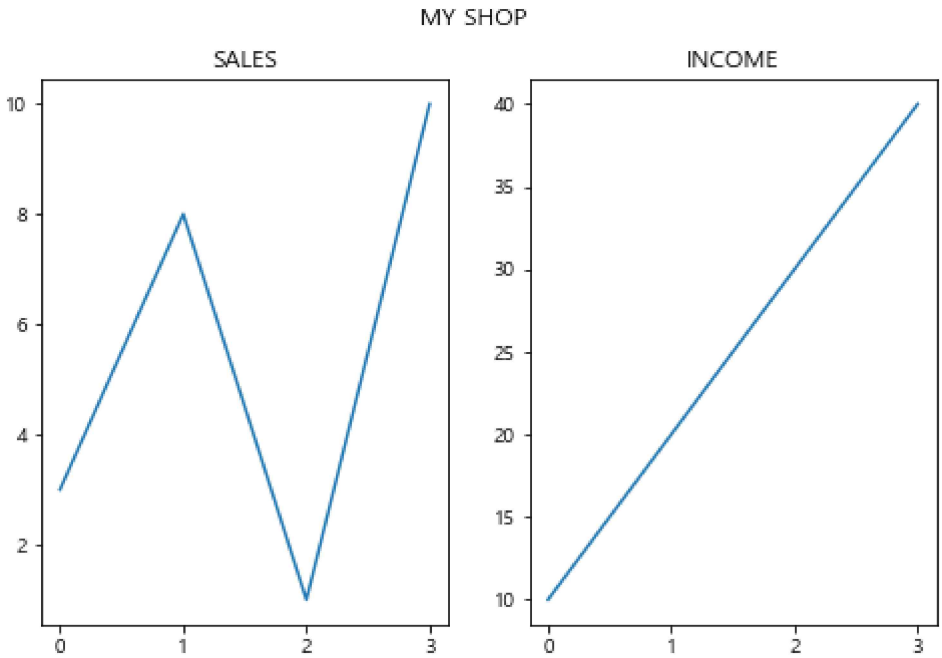
Out[33]: [<matplotlib.lines.Line2D at 0x1a7d92e8460>]



07. 여러개 그래프 전체 타이틀 표시 - SuperTitle

- 전체 이미지의 상위 타이틀을 subtitle()를 이용하여 제목을 추가할 수 있다.

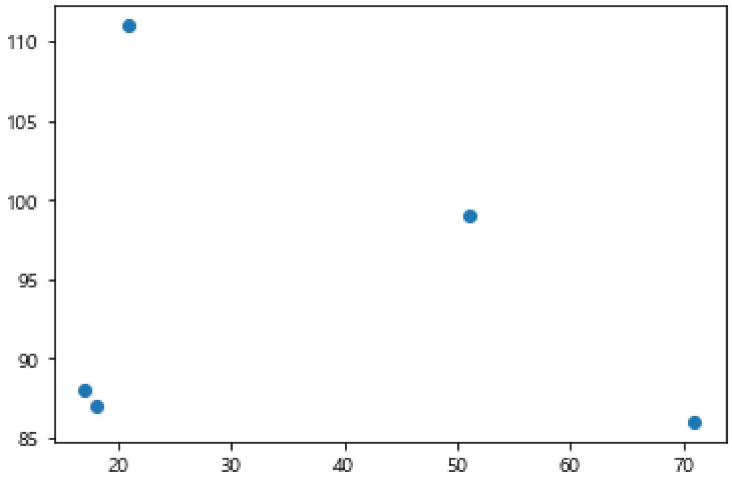
In [34]:
```
import matplotlib.pyplot as plt
import numpy as np

plt.figure(figsize=(8,5))

#plot 1:
x = np.array([0, 1, 2, 3])
y = np.array([3, 8, 1, 10])

# 1행 2열, 첫번째
plt.subplot(1, 2, 1)
plt.plot(x,y)
plt.title("SALES")

#plot 2:
x = np.array([0, 1, 2, 3])
y = np.array([10, 20, 30, 40])

# 1행 2열, 두번째
plt.subplot(1, 2, 2)
plt.plot(x,y)
plt.title("INCOME")

plt.suptitle("MY SHOP")
plt.show()
```



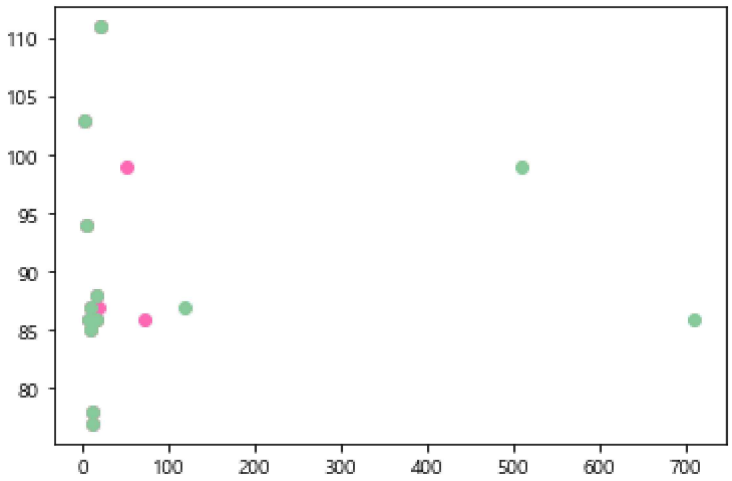
08. Scatter Plot(산점도)

- 직교 좌표계를 활용하여 좌표상의 점들을 표시.
- 두개 변수간의 관계를 나타낼 수 있다.

In [35]:
```
x = np.array([51,71,18,17,21])
y = np.array([99,86,87,88,111])

plt.scatter(x, y)
plt.show()
```

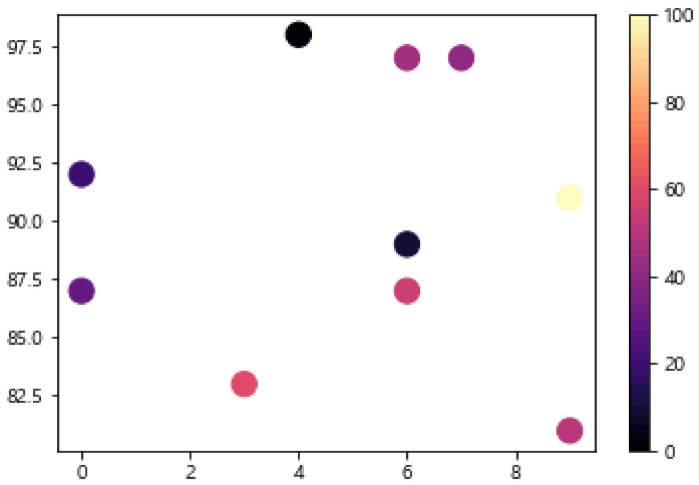


In [36]:
```
x = np.array([51,71,18,17,21,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

plt.scatter(x, y, color='hotpink')

x = np.array([511,711,118,17,21,17,2,9,4,11,12,9,6])
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

plt.scatter(x, y, color='#88c999')
plt.show()
```



In [37]:
```
x = np.random.randint(10, size=10)
y = np.random.randint(80,100, size=10)
print(x, y)
print( len(x), len(y) )

# 색에 숫자로 맵핑
colors = np.array([0, 10, 20, 30, 40, 45, 50, 55, 60, 100])

# 'viridis', 'plasma', 'inferno', 'magma', 'cividis'
plt.scatter(x, y, c=colors, cmap='magma', s=150)
plt.colorbar()   # colormap 을 포함할 수 있다.

plt.show()
```

```
[4 6 0 0 7 6 9 6 3 9] [98 89 92 87 97 97 81 87 83 91]
10 10
```



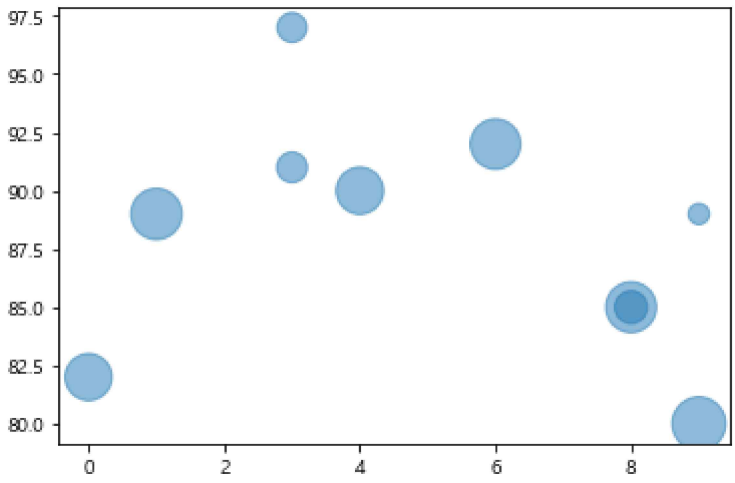
사용가능한 colorMaps

- https://www.w3schools.com/python/matplotlib_scatter.asp

09. Alpha(투명도)

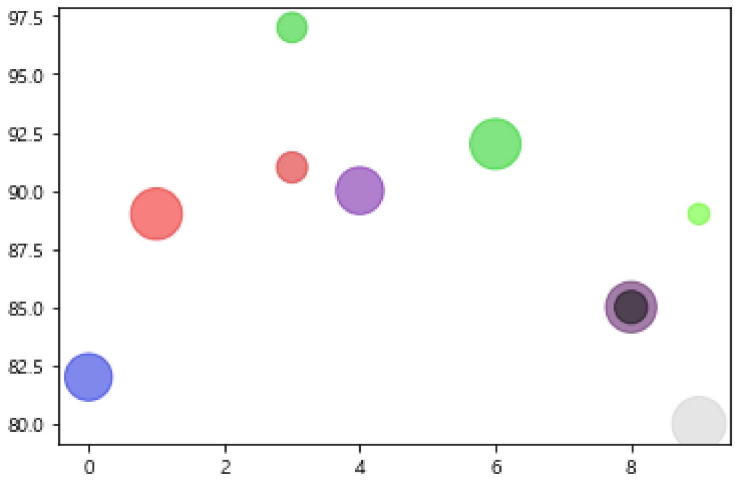
In [38]:
```
x = np.random.randint(10, size=10)
y = np.random.randint(80,100, size=10)

sizes = np.random.randint(20,800, size=10)
colors = np.random.randint(0,100, size=10)

plt.scatter(x, y, s=sizes, alpha=0.5)
```

Out[38]:
```
<matplotlib.collections.PathCollection at 0x1a7d90807f0>
```



In [39]:
```
plt.scatter(x, y, c=colors, s=sizes, alpha=0.5, cmap='nipy_spectral')
```

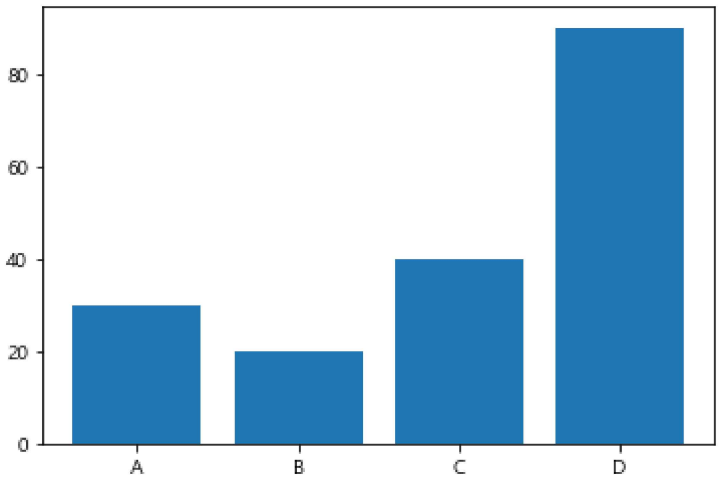
Out[39]: <matplotlib.collections.PathCollection at 0x1a7d90fe490>



10. Barplot(막대그래프)

In [40]:
```
x = np.array(['A', 'B', 'C', 'D'])
y = np.array([30,20,40,90])


plt.bar(x, y)
```

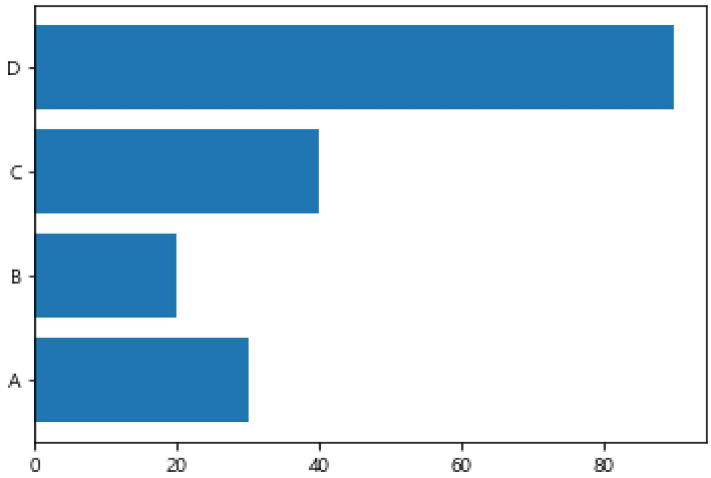
Out[40]: <BarContainer object of 4 artists>



11. Horizontal Bars(수평 막대그래프)

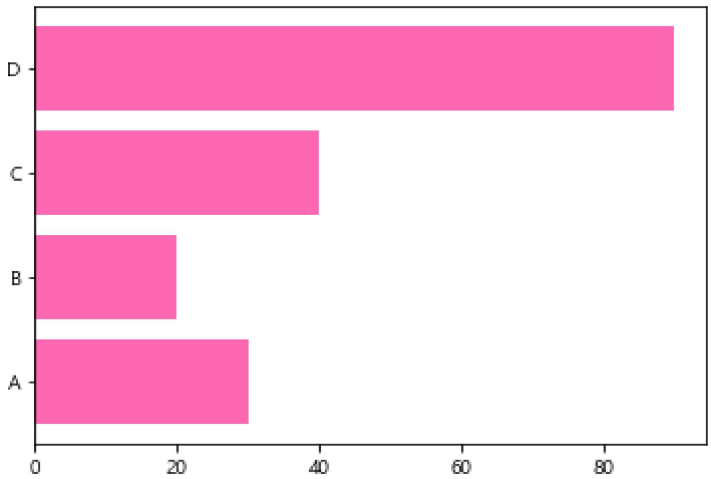
In [41]:
```
x = np.array(['A', 'B', 'C', 'D'])
y = np.array([30,20,40,90])


plt.barh(x, y)
```

Out[41]: <BarContainer object of 4 artists>



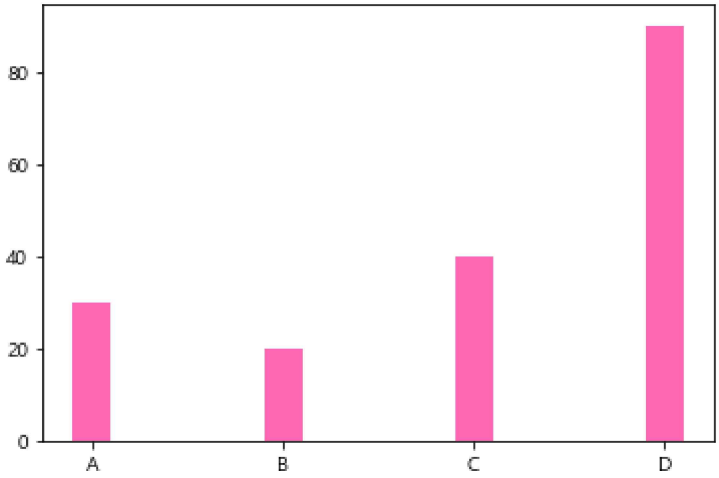
In [42]:
```
plt.barh(x, y, color = 'hotpink')
```

Out[42]: <BarContainer object of 4 artists>



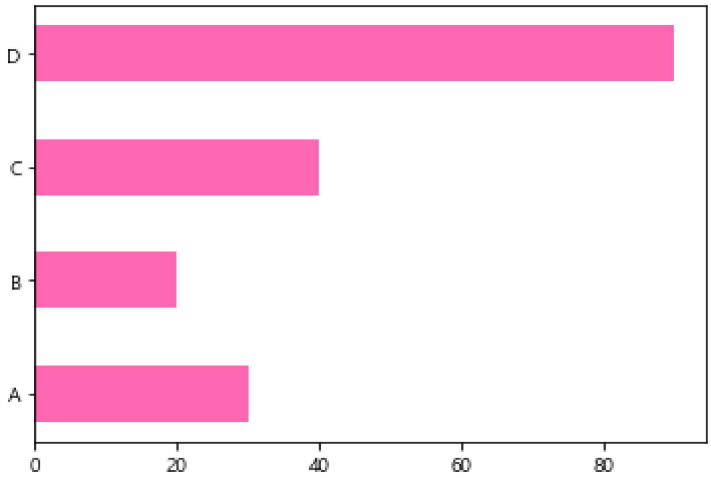
In [43]:
```
plt.bar(x, y, width=0.2, color = 'hotpink')
```

Out[43]: <BarContainer object of 4 artists>



In [44]:
```
plt.barh(x, y, height=0.5, color = 'hotpink')
```

Out[44]: <BarContainer object of 4 artists>



12. 히스토그램

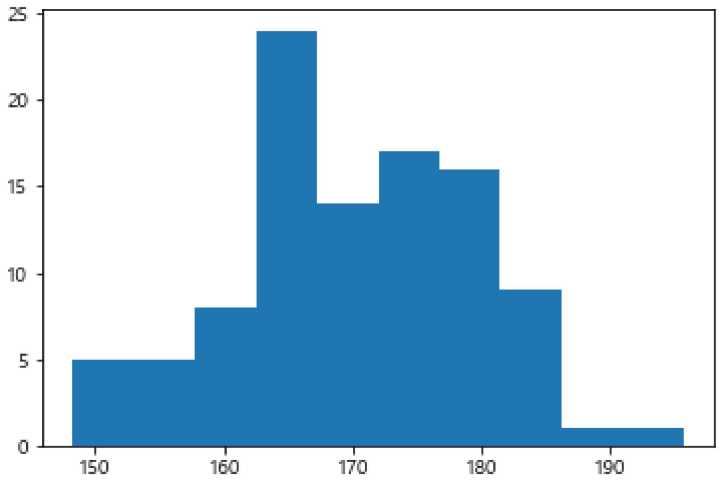
- 연속형 값을 표시할 때 사용.
- 가로축이 계급, 세로축이 도수(구간의 값의 개수)를 의미

In [45]:
```
# 정규 분포를 따르는 값. 100개 생성
x = np.random.normal(loc=170, scale=10, size=100)  # 평균, 표준편차, 개수


print(x)
plt.hist(x)
```

```
[154.35166278 163.10488052 181.97649238 164.77200627 195.66230795
 177.57329029 182.93317455 163.95787856 162.57791981 174.70376978
 179.34668705 155.21458933 173.45114013 173.17803513 151.90578735
 170.52233645 171.11448802 173.93770494 163.70107788 173.38393395
 167.18431257 167.91586448 167.82176355 174.01203691 187.53016407
 185.76618833 163.55175032 165.02584881 165.23044681 168.24442028
 162.71739687 157.39436612 173.78395127 184.72914578 163.39838882
 170.10445673 148.28180632 160.5670301  166.08458594 179.80998905
 172.04830747 161.63458083 151.88356815 176.27230266 162.19562211
 172.58340566 180.19942756 163.6548879  185.23852424 163.84353628
 165.27606072 180.46210065 161.24036929 156.4378359  151.21698133
 162.22880591 174.28381205 178.8318843  182.12653971 164.23796582
```

```
 166.32274449 181.67897716 180.14183957 170.78315347 172.30886905
 170.71289473 160.13980069 178.78974849 152.43055962 165.28910279
 179.96148122 166.22612775 173.15642392 181.43134782 184.19336752
 167.35920286 160.41310819 162.42037935 179.16995362 163.56417452
 179.33565487 164.51201503 176.83464962 183.90398511 168.02265
 181.10244236 163.06181028 175.14276831 168.88447935 169.29169077
 163.50696523 180.57628654 178.12377388 174.88898107 154.51838191
 173.82099966 162.69589949 171.53446999 172.11404621 169.33652427]
```

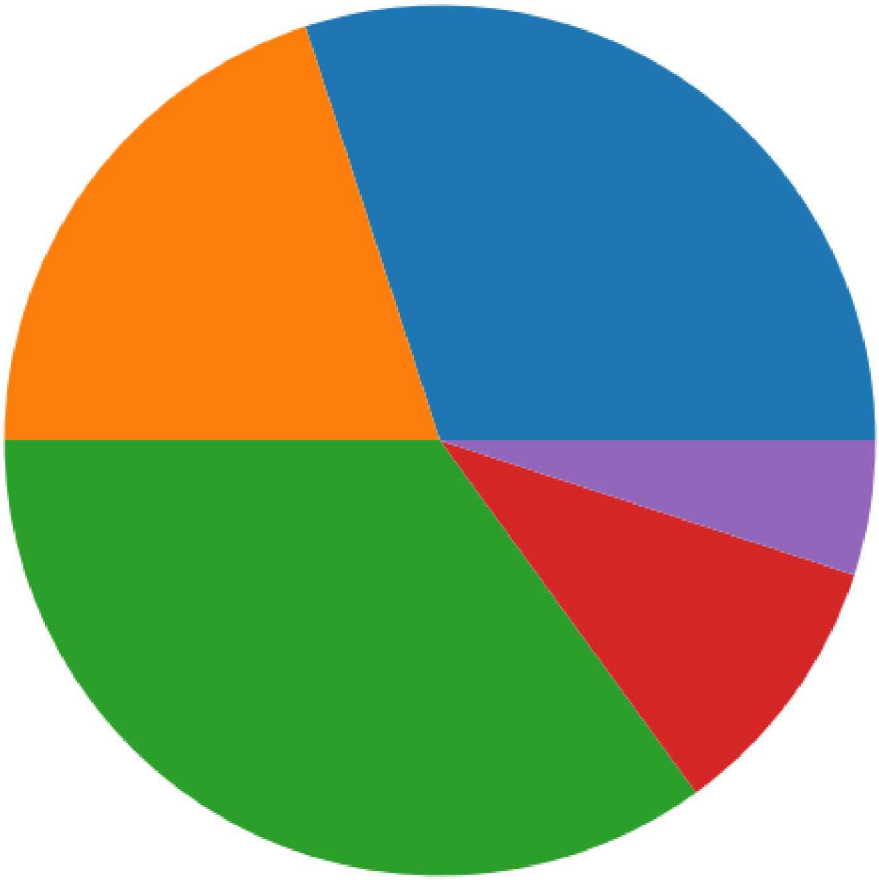
Out[45]:
```
(array([ 5.,  5.,  8., 24., 14., 17., 16.,  9.,  1.,  1.]),
 array([148.28180632, 153.01985649, 157.75790665, 162.49595681,
        167.23400697, 171.97205714, 176.7101073 , 181.44815746,
        186.18620762, 190.92425779, 195.66230795]),
 <BarContainer object of 10 artists>)
```



13. pie chart(원그래프)

In [46]:
```
y = np.array([30,20,35,10, 5])
plt.figure(figsize=(10,10))
plt.pie(y)
```

Out[46]:
```
([<matplotlib.patches.Wedge at 0x1a7d98316a0>,
  <matplotlib.patches.Wedge at 0x1a7d9831b80>,
  <matplotlib.patches.Wedge at 0x1a7d9831fa0>,
  <matplotlib.patches.Wedge at 0x1a7d983e460>,
  <matplotlib.patches.Wedge at 0x1a7d983e8e0>],
 [Text(0.6465637441936395, 0.8899187180267095, ''),
  Text(-0.8899187482945419, 0.6465637025335369, ''),
  Text(-0.4993894763020947, -0.9801072140121813, ''),
  Text(0.8899187331606258, -0.6465637233635886, ''),
  Text(1.0864571863351944, -0.1720778377961938, '')])
```

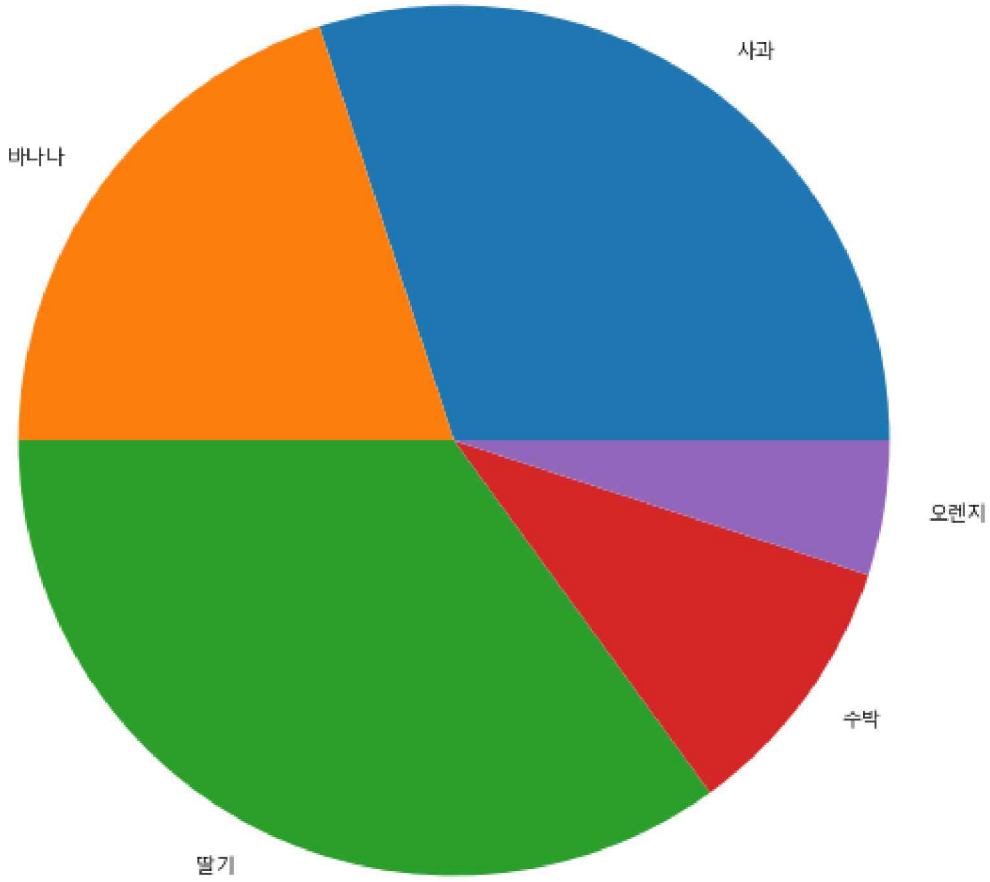


In [47]:

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


y = np.array([30,20,35,10, 5])
addlbl = ['사과', '바나나', '딸기', '수박', '오렌지']
plt.pie(y, labels=addlbl)
```

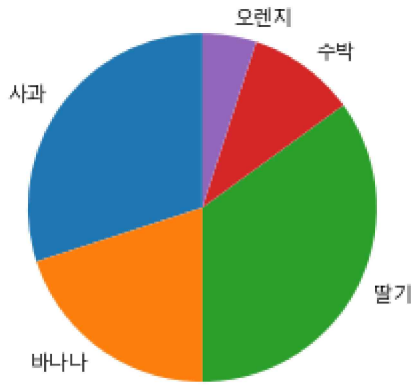
Out[47]:

```
([<matplotlib.patches.Wedge at 0x1a7d9881370>,
  <matplotlib.patches.Wedge at 0x1a7d9881850>,
  <matplotlib.patches.Wedge at 0x1a7d9881cd0>,
  <matplotlib.patches.Wedge at 0x1a7d988e190>,
  <matplotlib.patches.Wedge at 0x1a7d988e610>],
 [Text(0.6465637441936395, 0.8899187180267095, '사과'),
  Text(-0.8899187482945419, 0.6465637025335369, '바나나'),
  Text(-0.4993894763020947, -0.9801072140121813, '딸기'),
  Text(0.8899187331606258, -0.6465637233635886, '수박'),
  Text(1.0864571863351944, -0.1720778377961938, '오렌지')])
```



In [48]:
```
y = np.array([30,20,35,10, 5])
addlbl = ['사과', '바나나', '딸기', '수박', '오렌지']
plt.pie(y, labels = addlbl, startangle = 90)
```

Out[48]:
```
([<matplotlib.patches.Wedge at 0x1a7d9a04e20>,
  <matplotlib.patches.Wedge at 0x1a7d9a112e0>,
  <matplotlib.patches.Wedge at 0x1a7d9a11760>,
  <matplotlib.patches.Wedge at 0x1a7d9a11be0>,
  <matplotlib.patches.Wedge at 0x1a7d9a20040>],
 [Text(-0.8899187180267095, 0.6465637441936395, '사과'),
  Text(-0.6465637025335373, -0.8899187482945414, '바나나'),
  Text(0.9801072140121813, -0.4993894763020948, '딸기'),
  Text(0.6465637233635887, 0.8899187331606258, '수박'),
  Text(0.17207783779619384, 1.0864571863351942, '오렌지')])
```

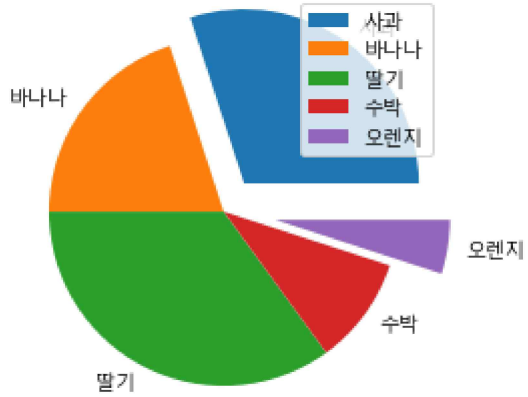


In [49]:
```
y = np.array([30,20,35,10, 5])
addlbl = ['사과', '바나나', '딸기', '수박', '오렌지']
myexplode = [0.2, 0, 0, 0, 0.3]


plt.pie(y, labels = addlbl, explode = myexplode)
```

Out[49]:
```
([<matplotlib.patches.Wedge at 0x1a7d954d460>,
  <matplotlib.patches.Wedge at 0x1a7d9543940>,
  <matplotlib.patches.Wedge at 0x1a7d9543e80>,
  <matplotlib.patches.Wedge at 0x1a7d9237d90>,
  <matplotlib.patches.Wedge at 0x1a7d93752e0>],
 [Text(0.764120788592483, 1.051722121304293, '사과'),
  Text(-0.8899187482945419, 0.6465637025335369, '바나나'),
  Text(-0.4993894763020947, -0.9801072140121813, '딸기'),
  Text(0.8899187331606258, -0.6465637233635886, '수박'),
  Text(1.3827636916993382, -0.21900815719515573, '오렌지')])
```

In [50]:
```
y = np.array([30,20,35,10, 5])
addlbl = ['사과', '바나나', '딸기', '수박', '오렌지']
myexplode = [0.2, 0, 0, 0, 0.3]


plt.pie(y, labels = addlbl, explode = myexplode)
plt.legend()
```

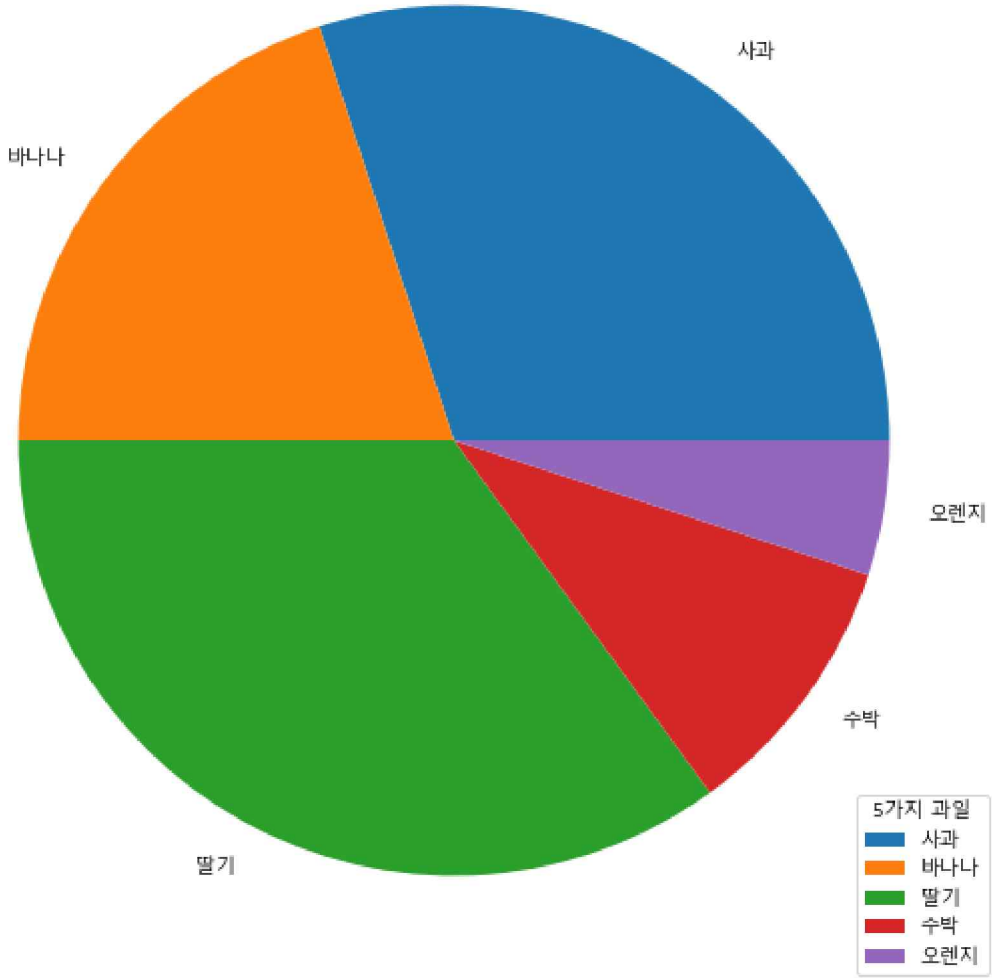
Out[50]:
```
<matplotlib.legend.Legend at 0x1a7d9324d00>
```



In [51]:

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

y = np.array([30,20,35,10, 5])
addlbl = ['사과', '바나나', '딸기', '수박', '오렌지']
myexplode = [0.2, 0, 0, 0, 0.3]

plt.pie(y, labels = addlbl)
plt.legend(title = "5가지 과일")
```

Out[51]: <matplotlib.legend.Legend at 0x1a7d9a30b50>



14. 3D Plots

- 과거에서는 다음과 같이 썼지만, 이제는 통합되었다.

  ```
  from mpl_toolkits.mplot3d import Axes3D
  ax = Axes3D(fig)
  ```

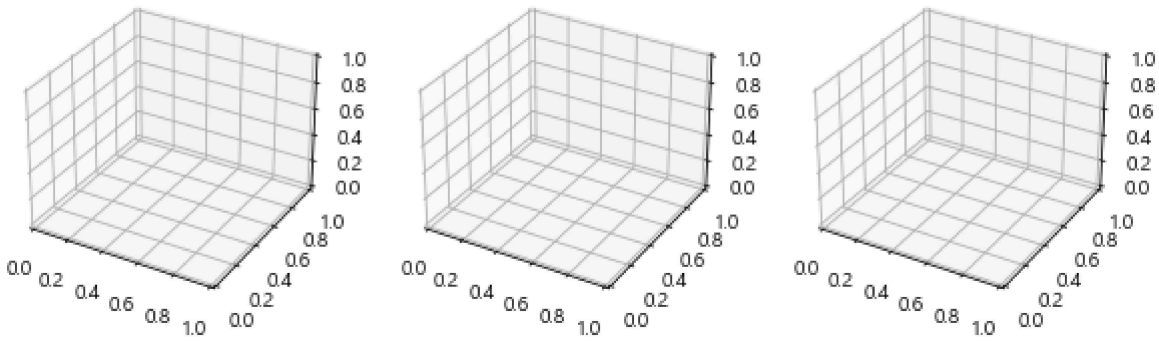
- 현재는 아래와 같이 써도 가능

  `fig.add_subplot(projection='3d')`만으로 Axes3D를 사용 가능.

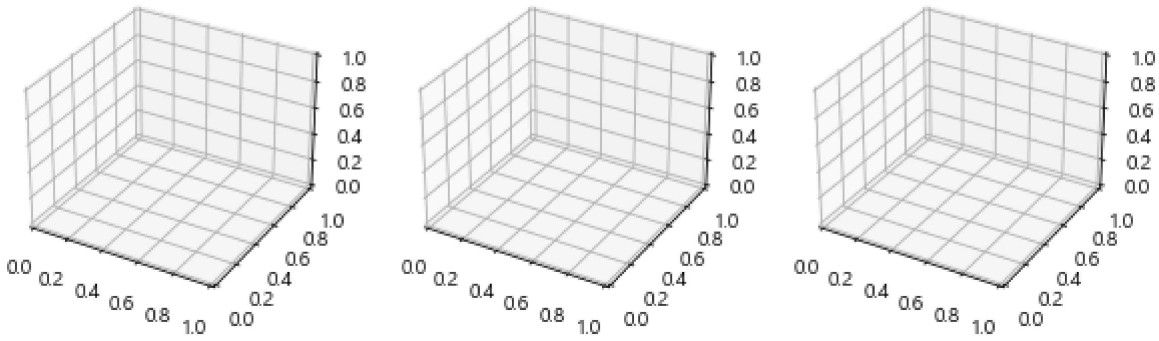
1행 3열의 3D axes를 만들기

In [52]:

```
import matplotlib.pyplot as plt
import numpy as np

fig = plt.figure(figsize=(10, 3))
ax0 = fig.add_subplot(131, projection="3d")
ax1 = fig.add_subplot(132, projection="3d")
ax2 = fig.add_subplot(133, projection="3d")
```

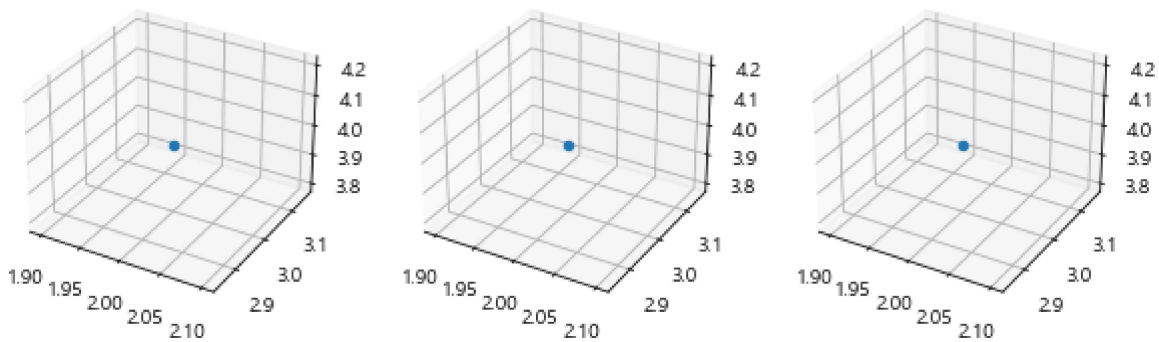


In [53]:
```
# 다음과 같이 좀 더 간단히 가능
fig, axs = plt.subplots(ncols=3, figsize=(10, 3),
                        subplot_kw={"projection":"3d"})
```



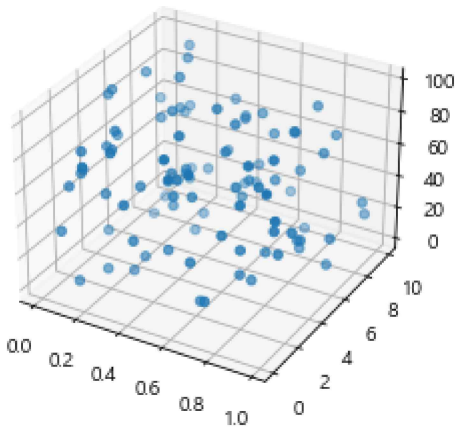
In [54]:
```
# 다음과 같이 좀 더 간단히 가능
fig, axs = plt.subplots(ncols=3, figsize=(10, 3),
                        subplot_kw={"projection":"3d"})

axs[0].scatter(2,3,4)
axs[1].scatter(2,3,4)
axs[2].scatter(2,3,4)
```

Out[54]: <mpl_toolkits.mplot3d.art3d.Path3DCollection at 0x1a7db1ccf40>



In [55]:
```
np.random.seed(42)

xs = np.random.random(100)
```

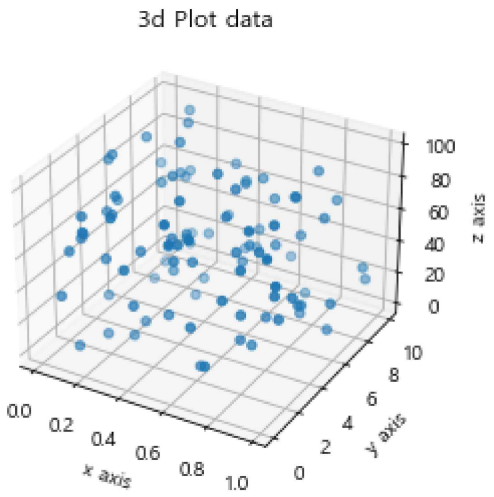
```
ys = np.random.random(100) * 10
zs = np.random.random(100) * 100

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(xs,ys,zs)
plt.show()
```



title, label 넣기

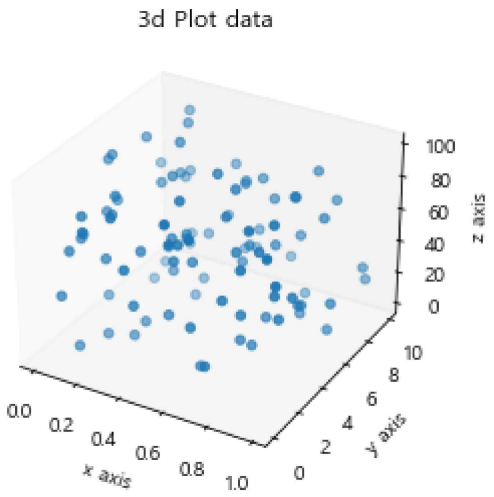
In [56]:

```
np.random.seed(42)

xs = np.random.random(100)
ys = np.random.random(100) * 10
zs = np.random.random(100) * 100

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(xs,ys,zs)

# 제목
ax.set_title("3d Plot data")

# 레이블
ax.set_xlabel("x axis")
ax.set_ylabel("y axis")
ax.set_zlabel("z axis")

plt.show()
```



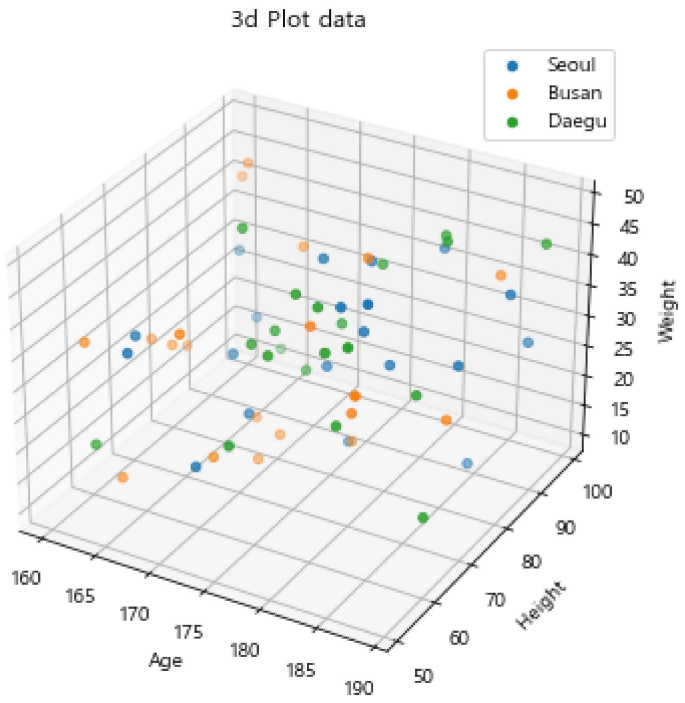
In [57]:

```
np.random.seed(42)

xs = np.random.random(100)
ys = np.random.random(100) * 10
zs = np.random.random(100) * 100

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(xs,ys,zs)

# 제목
ax.set_title("3d Plot data")
ax.grid(False)

# 레이블
ax.set_xlabel("x axis")
ax.set_ylabel("y axis")
ax.set_zlabel("z axis")

plt.show()
```



In [58]:

```
### 범주

labels = ["Seoul", "Busan", "Daegu"]

fig = plt.figure(figsize=(12,6))
ax = fig.add_subplot(111, projection='3d')

for one in labels:
    ages = np.random.randint(low=10, high=50, size=20)
    heights = np.random.randint(low=160, high=190, size=20)
    weights = np.random.randint(low=50, high=100, size=20)

    ax.scatter(xs=heights, ys=weights, zs=ages, label=one)

# 제목
ax.set_title("3d Plot data")

# 레이블
ax.set_xlabel("Age")
ax.set_ylabel("Height")
ax.set_zlabel("Weight")

ax.legend(loc="best")

plt.show()
```



Reference :

- https://www.w3schools.com/colors/colors_names.asp
- matplotlib colormap : https://matplotlib.org/3.5.0/tutorials/colors/colormaps.html
- 3D Plot : https://likegeeks.com/3d-plotting-in-python/

In [ ]: