[4-2. Matplotlib 소개 및 사용 방법]

1. Matplotlib 소개

Matplotlib은 Python의 대표적인 2D 그래픽 라이브러리로, John D. Hunter가 2003년에 처음으로 만들었습니다. Hunter는 당시에 MATLAB(공학용 애플리케이션 소프트웨어)과 유사한 플로팅 기능을 Python에서 사용하고자 하는 목적에서 시작하여, Matplotlib이라는 이름의 과학 계산용 그래픽 라이브러리를 개발하게 되었습니다.

Matplotlib의 주요 목적은 Python에서 다양한 포맷과 환경에서 사용할 수 있는 2D 데이터 시각화 기능을 제공하는 것입니다. 이를 통해 사용자는 과학 계산과 데이터 분석 결과를 시각적으로 표현할 수 있습니다.

Matplotlib을 사용하기 위해서는 라이브러리가 설치되어 있어야 합니다. 아나콘다 네비게이터 "Environments" 탭에서 현재 사용 중인 가상환경을 선택한 후 라이브러리 이름을 조회하여 설치할 수 있습니다.

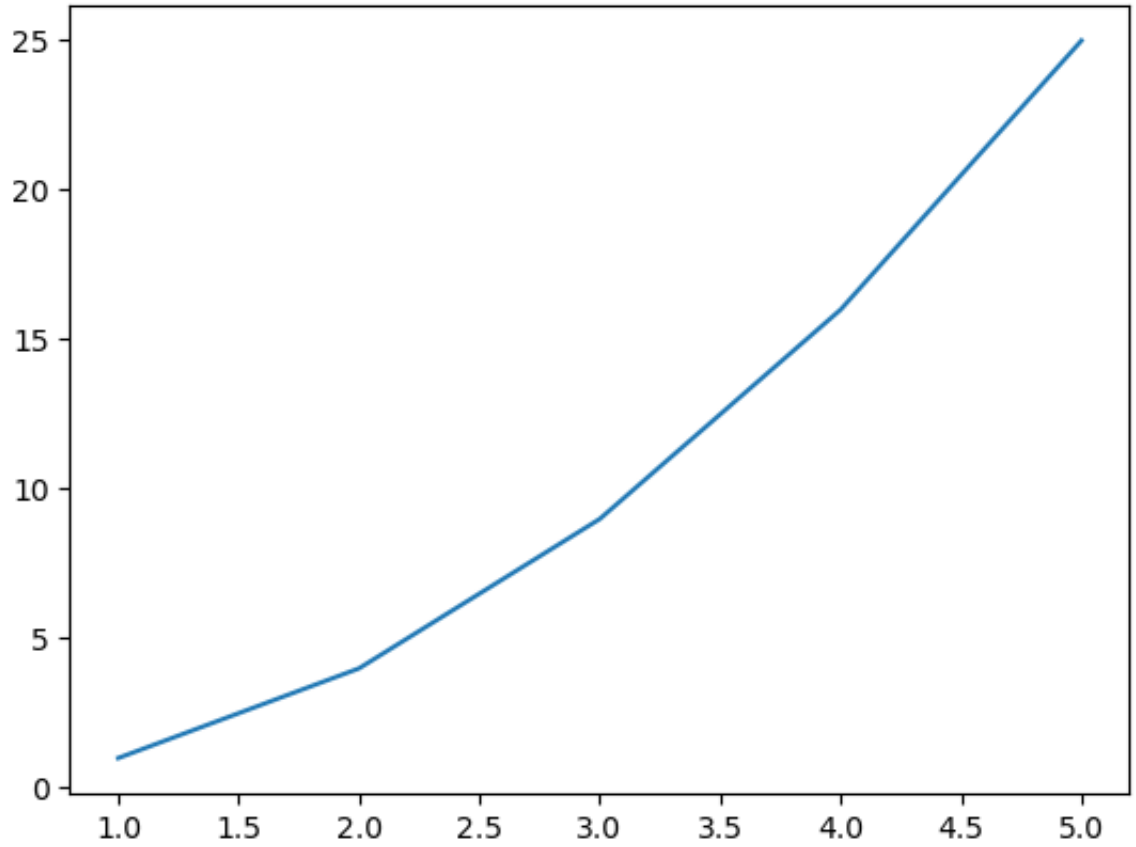
커맨드 라인(명령 프롬프트, 터미널 등) 환경에서는 "pip install matplotlib" 명령어를 사용해서 설치할 수 있습니다.

2. 기본적인 선그래프 생성

Matplotlib의 pyplot 모듈을 사용하여 간단한 선 그래프를 생성할 수 있습니다.

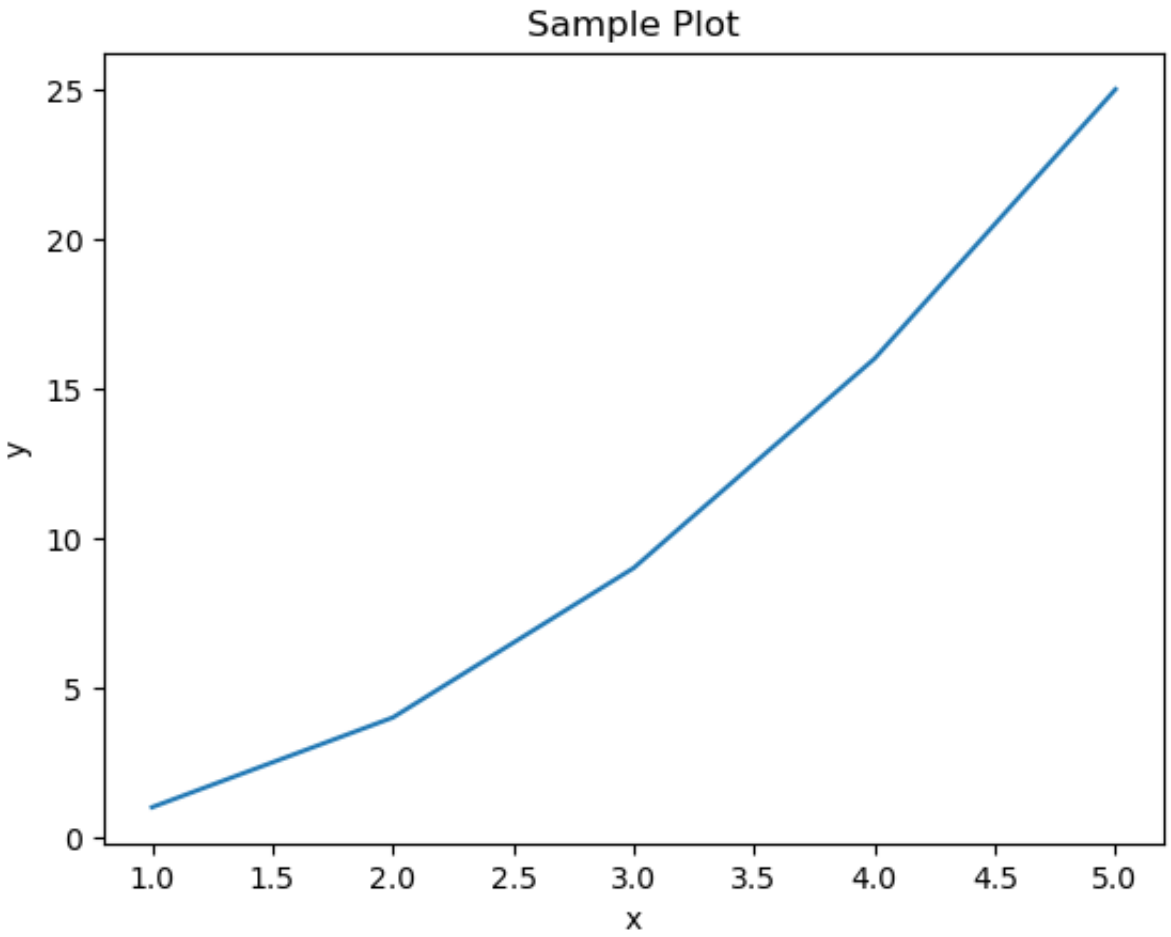
2-1. 기본 사용 방법

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

# 데이터 생성
x = [1, 2, 3, 4, 5]
y = [1, 4, 9, 16, 25]

# 선 그래프 생성
plt.plot(x, y)
plt.show()
```



2-2. 제목과 레이블 추가

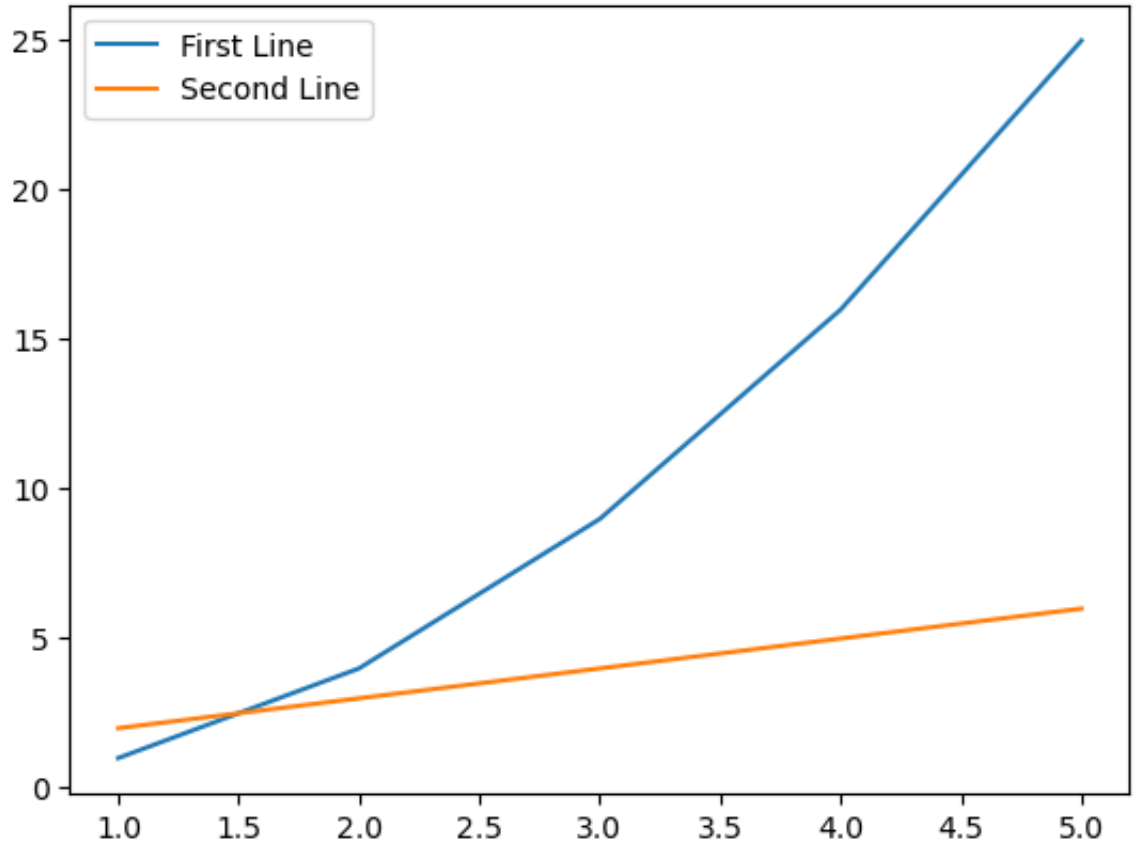
In [ ]:
```
plt.plot(x, y)
plt.title("Sample Plot")
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```



2-3. 여러개의 선그래프

하나의 그래프에 여러 개의 선을 그리고, 범례를 추가할 수 있습니다.

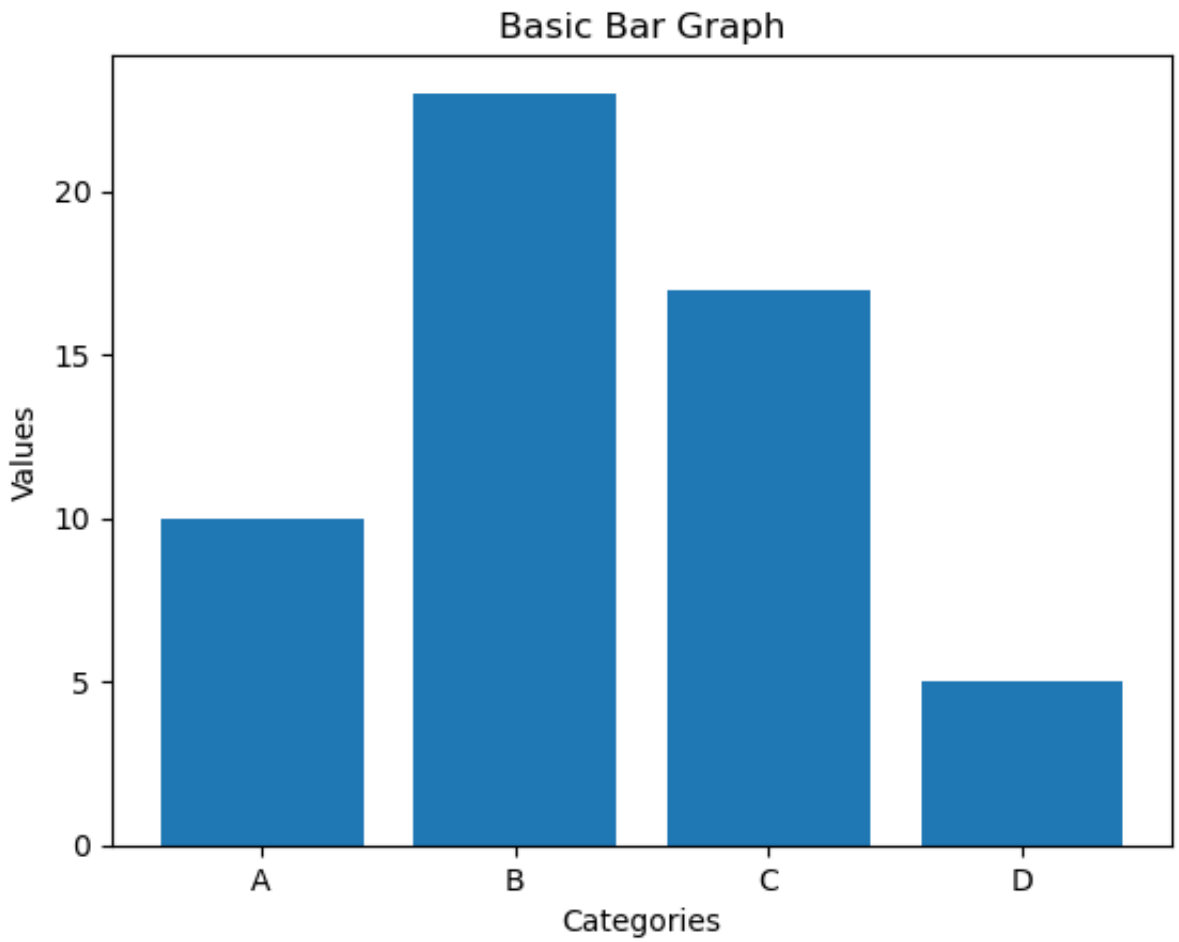
In [ ]:
```
x2 = [1, 2, 3, 4, 5]
y2 = [2, 3, 4, 5, 6]

plt.plot(x, y, label='First Line')
plt.plot(x2, y2, label='Second Line')
plt.legend()
plt.show()
```



3. 막대그래프 그리기

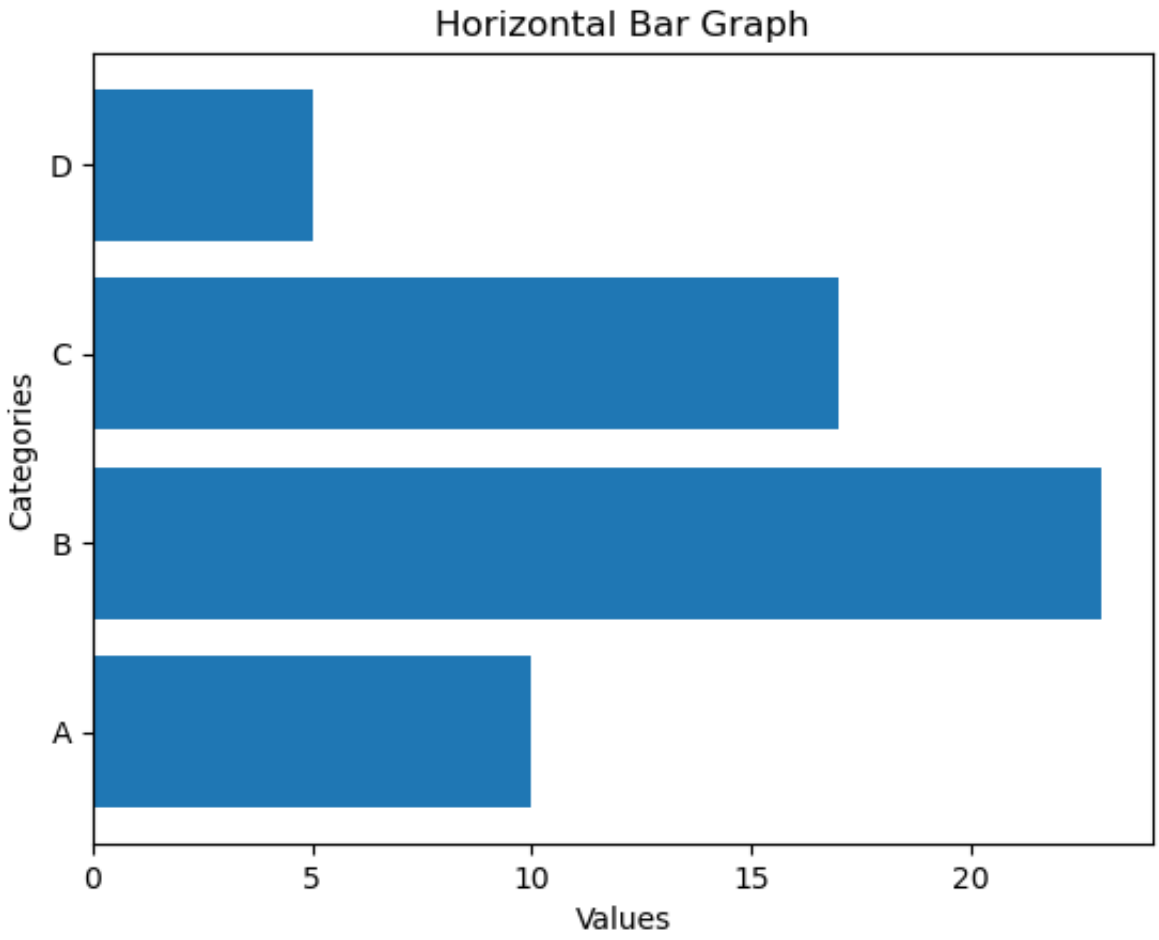
3-1. 기본 막대 그래프 그리기

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

# 데이터 설정
categories = ['A', 'B', 'C', 'D']
values = [10, 23, 17, 5]

# 막대 그래프 그리기
plt.bar(categories, values)

# 제목과 축 레이블 추가
plt.title('Basic Bar Graph')
plt.xlabel('Categories')
plt.ylabel('Values')

# 그래프 표시
plt.show()
```



3-2. 수평 막대 그래프 그리기

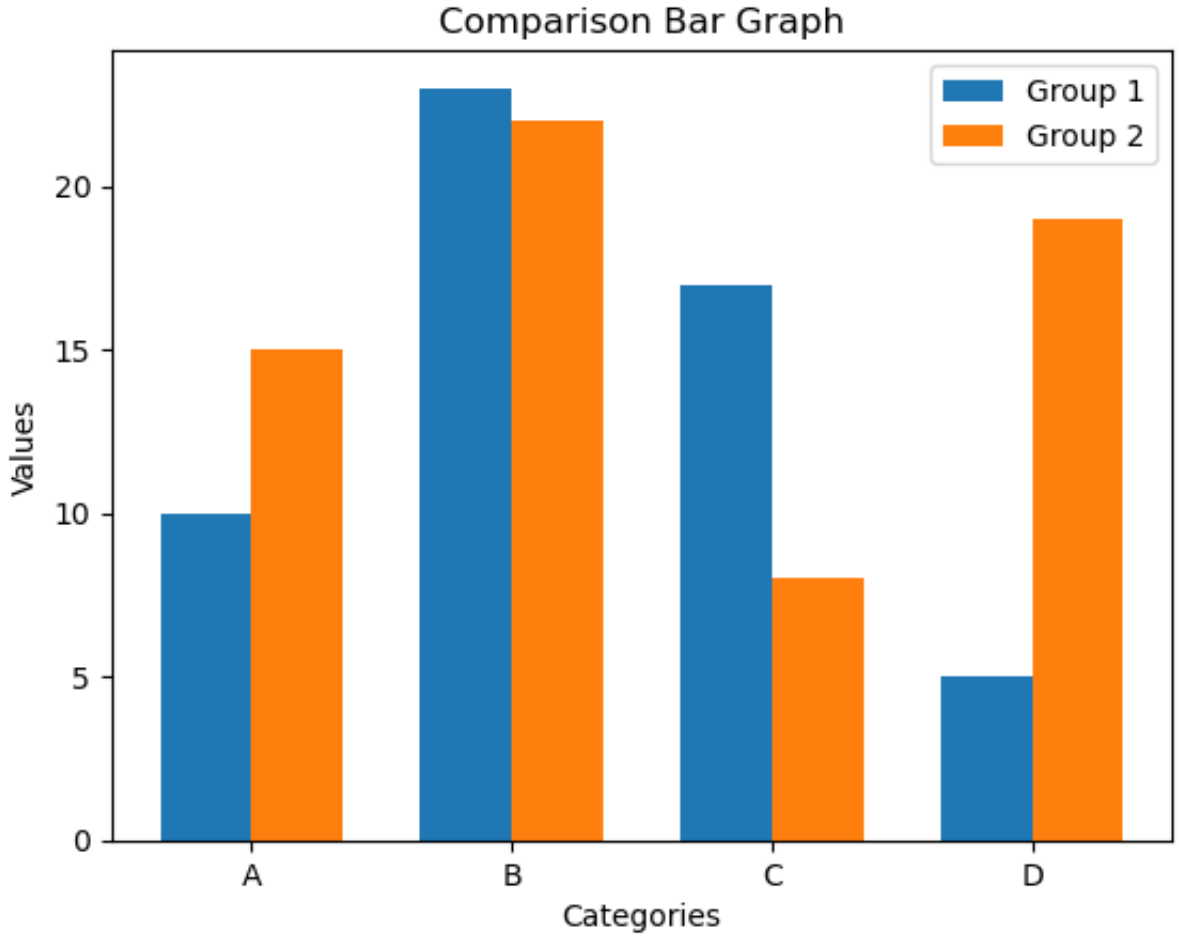
In [ ]:
```
plt.barh(categories, values)

# 제목과 축 레이블 추가
plt.title('Horizontal Bar Graph')
plt.xlabel('Values')
plt.ylabel('Categories')

plt.show()
```



3-3. 여러 그룹의 데이터 비교하기

여러 그룹의 데이터를 비교하는 막대 그래프를 그리려면, 막대의 위치를 약간 조정하여 여러 막대가 서로 겹치지 않도록 해야 합니다.

In [ ]:
```
import numpy as np

# 추가 데이터 설정
values2 = [15, 22, 8, 19]
index = np.arange(len(categories))  # 카테고리별 인덱스 생성
bar_width = 0.35  # 막대 너비 설정

# 막대 그래프 그리기
plt.bar(index, values, bar_width, label='Group 1')
plt.bar(index + bar_width, values2, bar_width, label='Group 2')

# 축 레이블과 범례 추가
plt.xlabel('Categories')
plt.ylabel('Values')
plt.title('Comparison Bar Graph')
plt.xticks(index + bar_width / 2, categories)  # x축에 표시될 레이블 위치 조정
plt.legend()

plt.show()
```

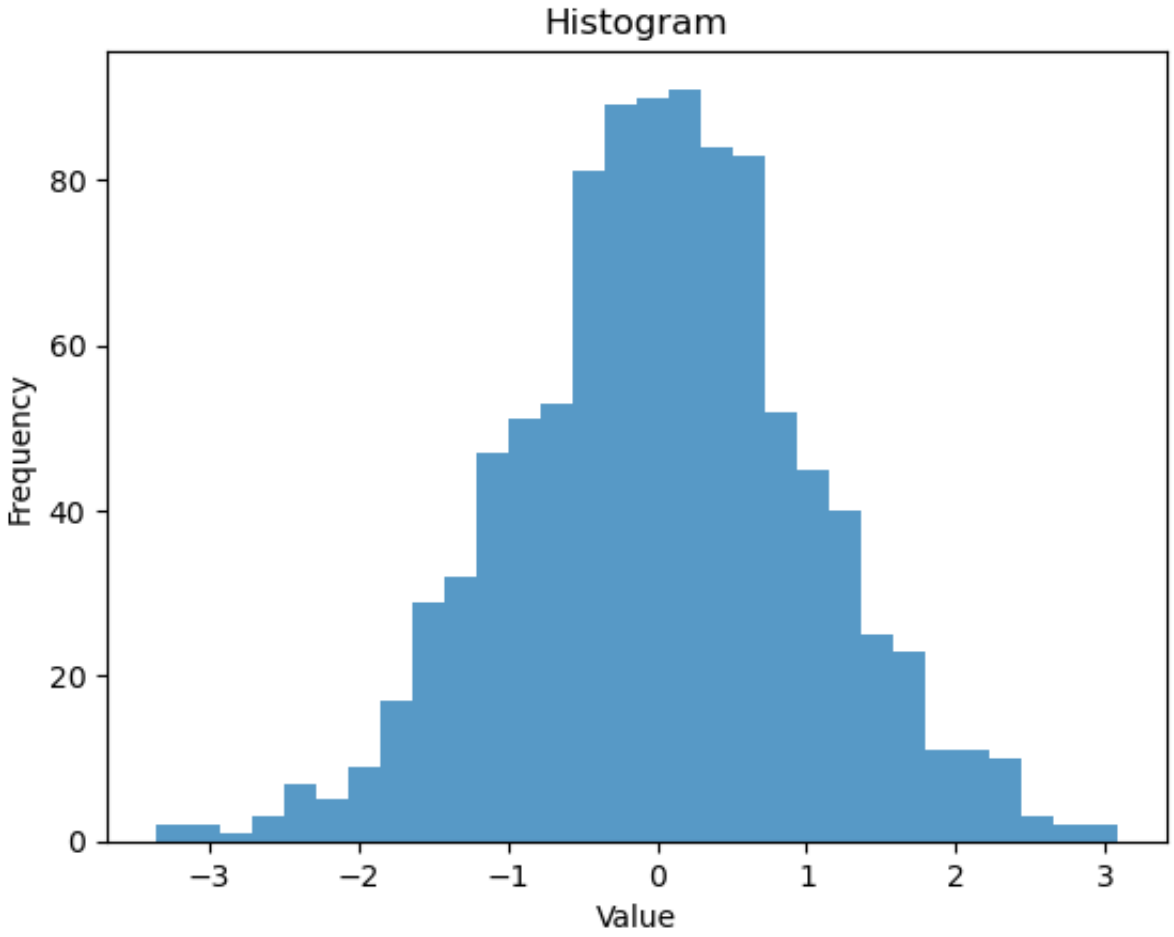


4. 히스토그램 그리기

4-1. 기본 히스토그램 그리기

아래 예시에서는 numpy를 사용하여 표준 정규 분포에서 무작위로 1000개의 데이터 포인트를 생성하고, 이 데이터를 사용하여 히스토그램을 그립니다. bins 매개변수는 히스토그램의 막대(bin) 개수를 조정하는 데 사용되며, alpha 매개변수는 막대의 투명도를 설정합니다.

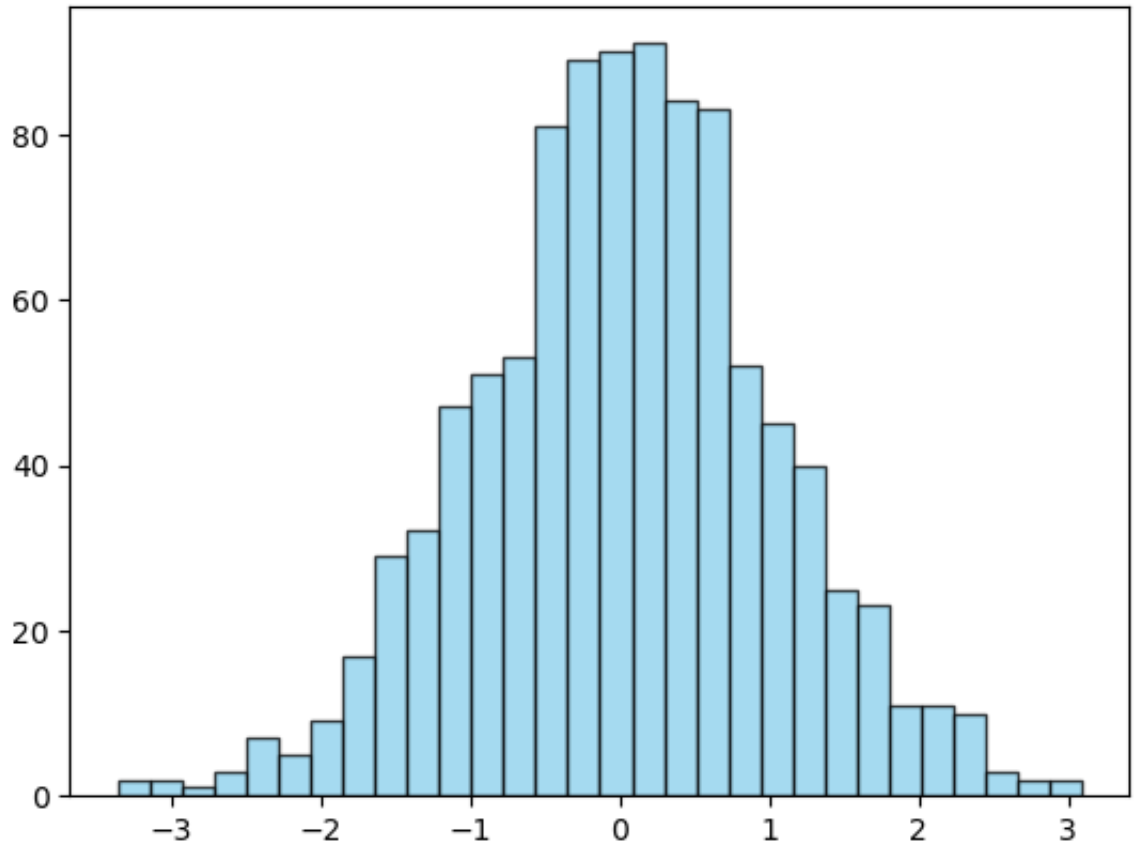
In [ ]:
```python
import matplotlib.pyplot as plt
import numpy as np

# 임의의 데이터 생성
data = np.random.randn(1000)

# 히스토그램 그리기
plt.hist(data, bins=30, alpha=0.75)

# 제목과 축 레이블 추가
plt.title('Histogram')
plt.xlabel('Value')
plt.ylabel('Frequency')

# 그래프 표시
plt.show()
```



4-2. 히스토그램 스타일링

히스토그램의 막대 색상, 경계선 색상 등을 커스터마이징할 수 있습니다.

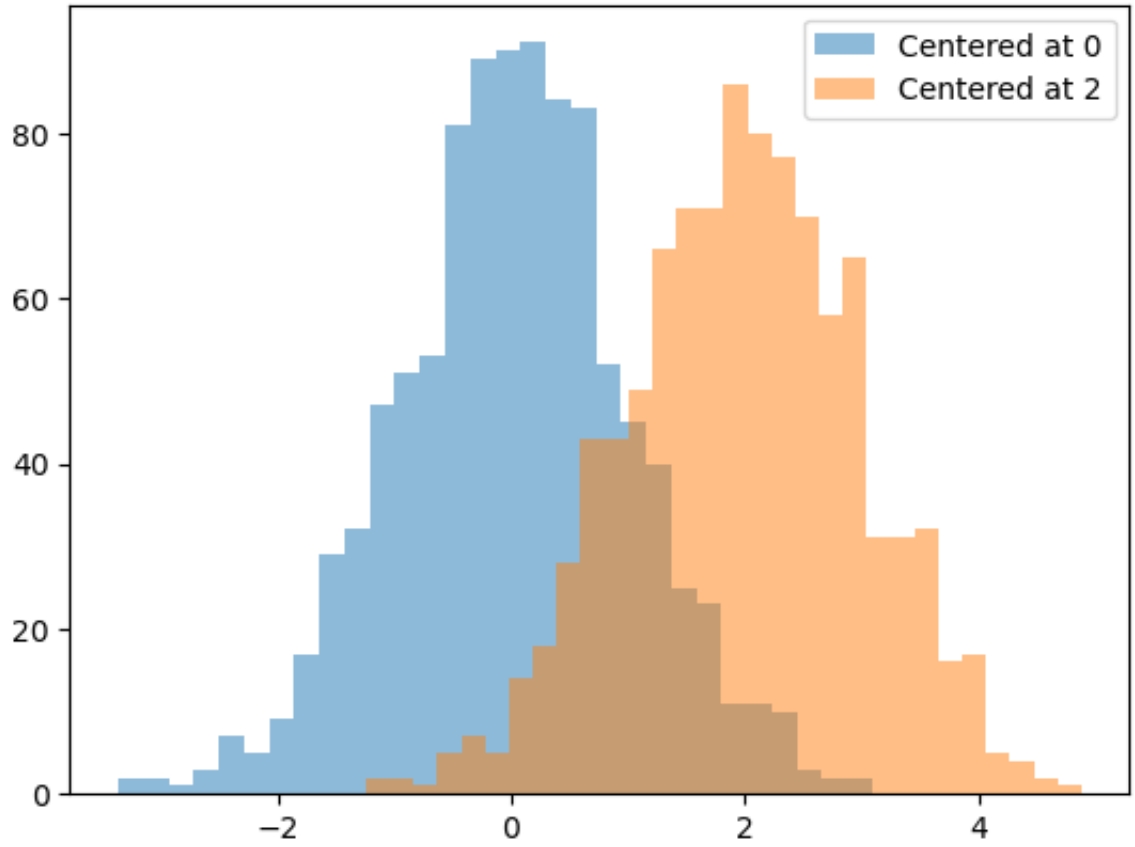
In [ ]:
```
plt.hist(data, bins=30, alpha=0.75, color='skyblue', edgecolor='black')

# 스타일링된 히스토그램 표시
plt.show()
```



4-3. 여러 데이터셋의 히스토그램 겹쳐 그리기

여러 데이터셋의 분포를 비교하기 위해 히스토그램을 겹쳐 그릴 수 있습니다.

In [ ]:
```
# 추가 데이터 생성
data2 = np.random.randn(1000) + 2  # 평균을 2만큼 이동

# 여러 데이터셋의 히스토그램 겹쳐 그리기
plt.hist(data, bins=30, alpha=0.5, label='Centered at 0')
plt.hist(data2, bins=30, alpha=0.5, label='Centered at 2')

# 범례 추가
plt.legend()

# 그래프 표시
plt.show()
```

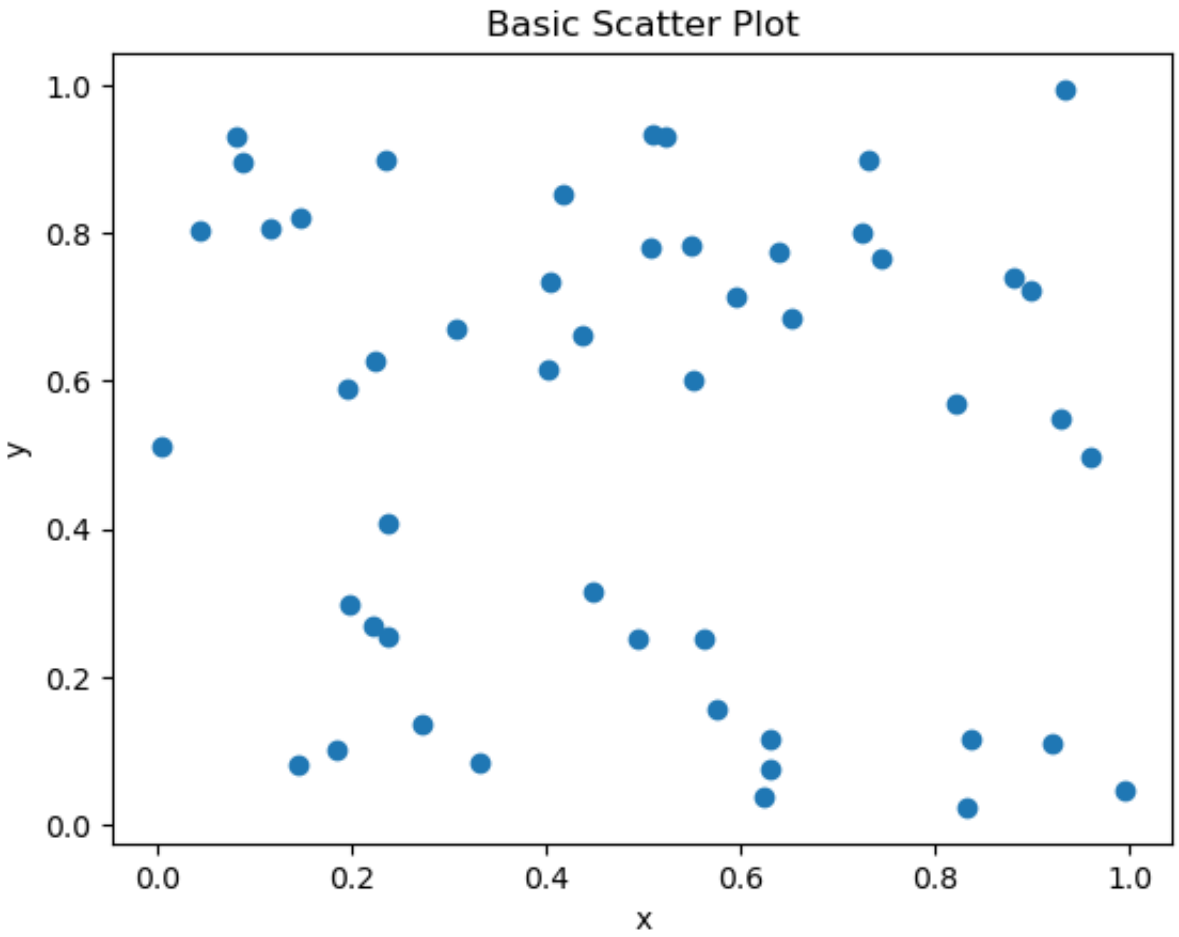


5. 산점도 그리기

5-1. 기본 산점도 그리기

plt.scatter() 메서드를 사용하여 각 데이터 포인트의 x와 y 위치를 기반으로 산점도를 그릴 수 있습니다.

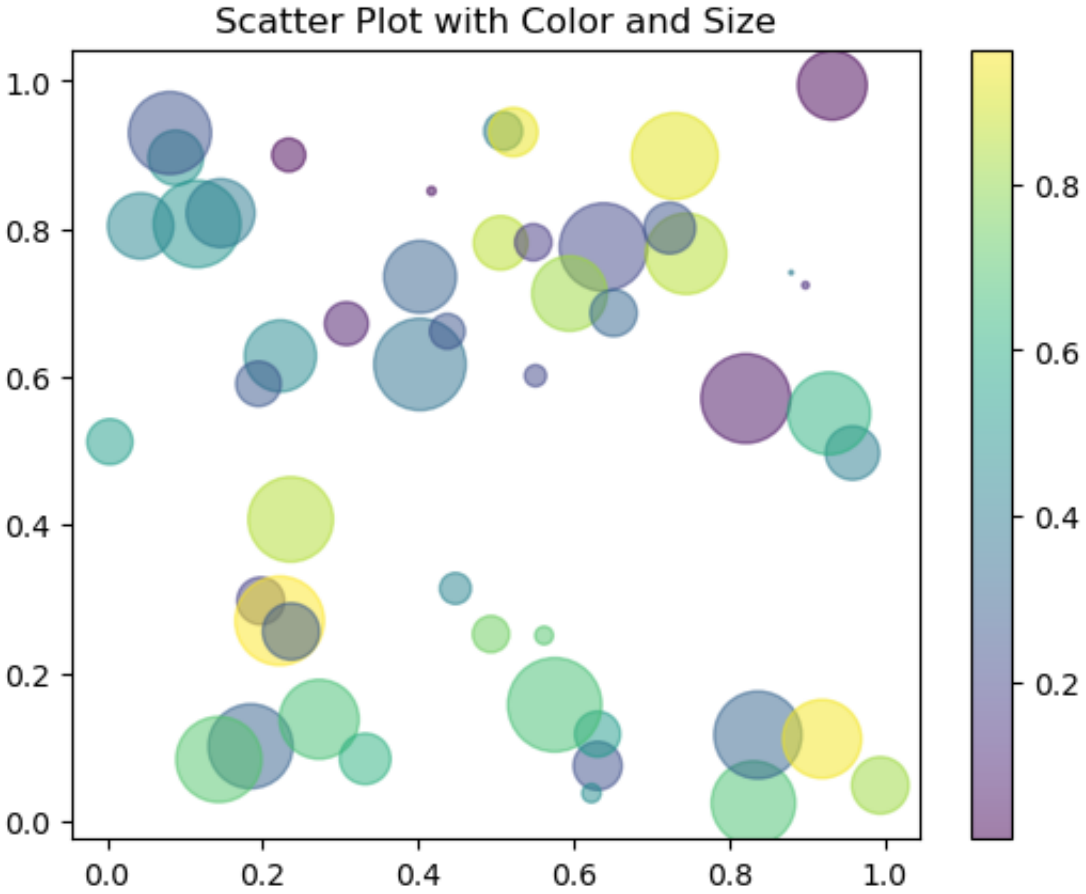
In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np

# 데이터 생성
x = np.random.rand(50)
y = np.random.rand(50)

# 산점도 그리기
plt.scatter(x, y)
plt.title('Basic Scatter Plot')
plt.xlabel('x')
plt.ylabel('y')

# 그래프 표시
plt.show()
```



5-2. 산점도에 색상과 크기 추가하기

산점도에서 각 점의 색상과 크기를 다르게 설정하여 추가 정보를 표현할 수 있습니다. 'c' 매개변수로 색상을, 's' 매개변수로 점의 크기를 설정할 수 있습니다.

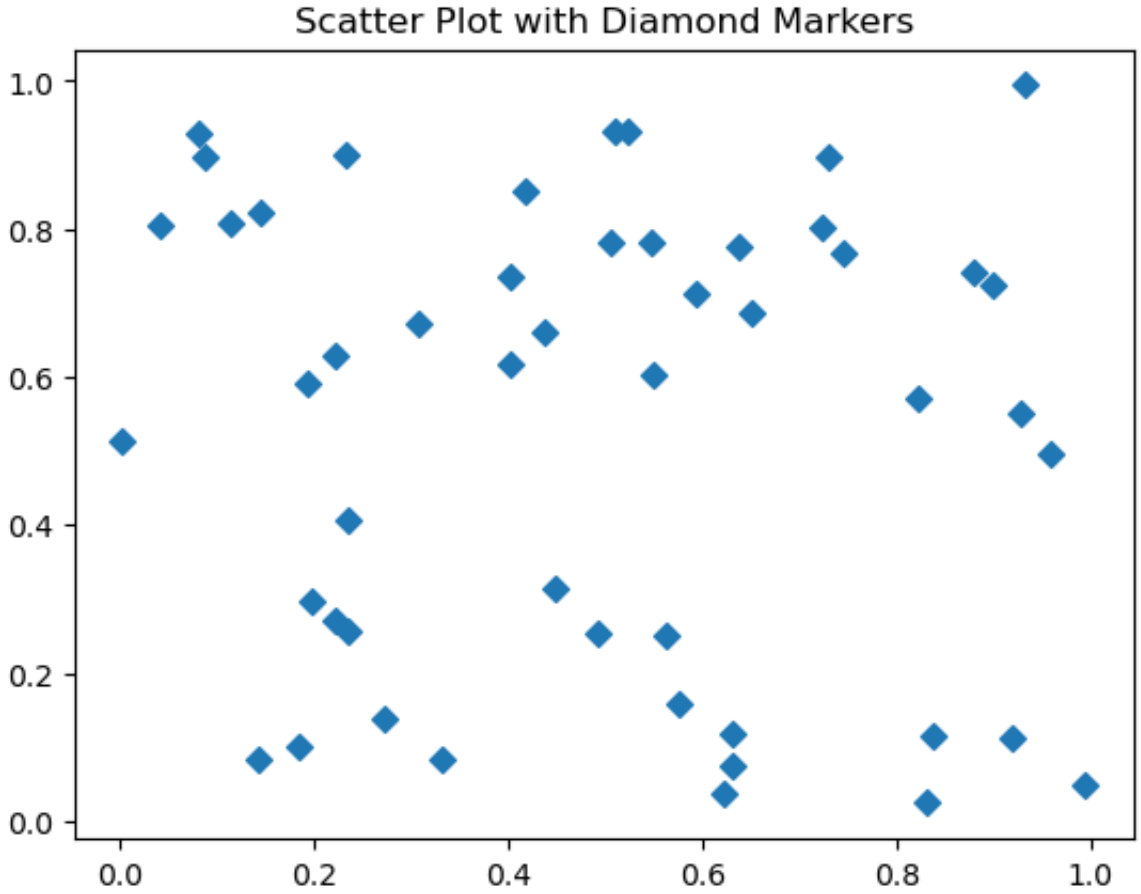
In [ ]:
```
# 추가 데이터 생성
colors = np.random.rand(50)  # 색상을 위한 데이터
sizes = 1000 * np.random.rand(50)  # 점 크기를 위한 데이터

# 산점도에 색상과 크기 추가하여 그리기
plt.scatter(x, y, c=colors, s=sizes, alpha=0.5, cmap='viridis')
plt.colorbar()  # 색상 스케일 표시
plt.title('Scatter Plot with Color and Size')

plt.show()
```



5-3. 산점도에서 마커 스타일 변경하기

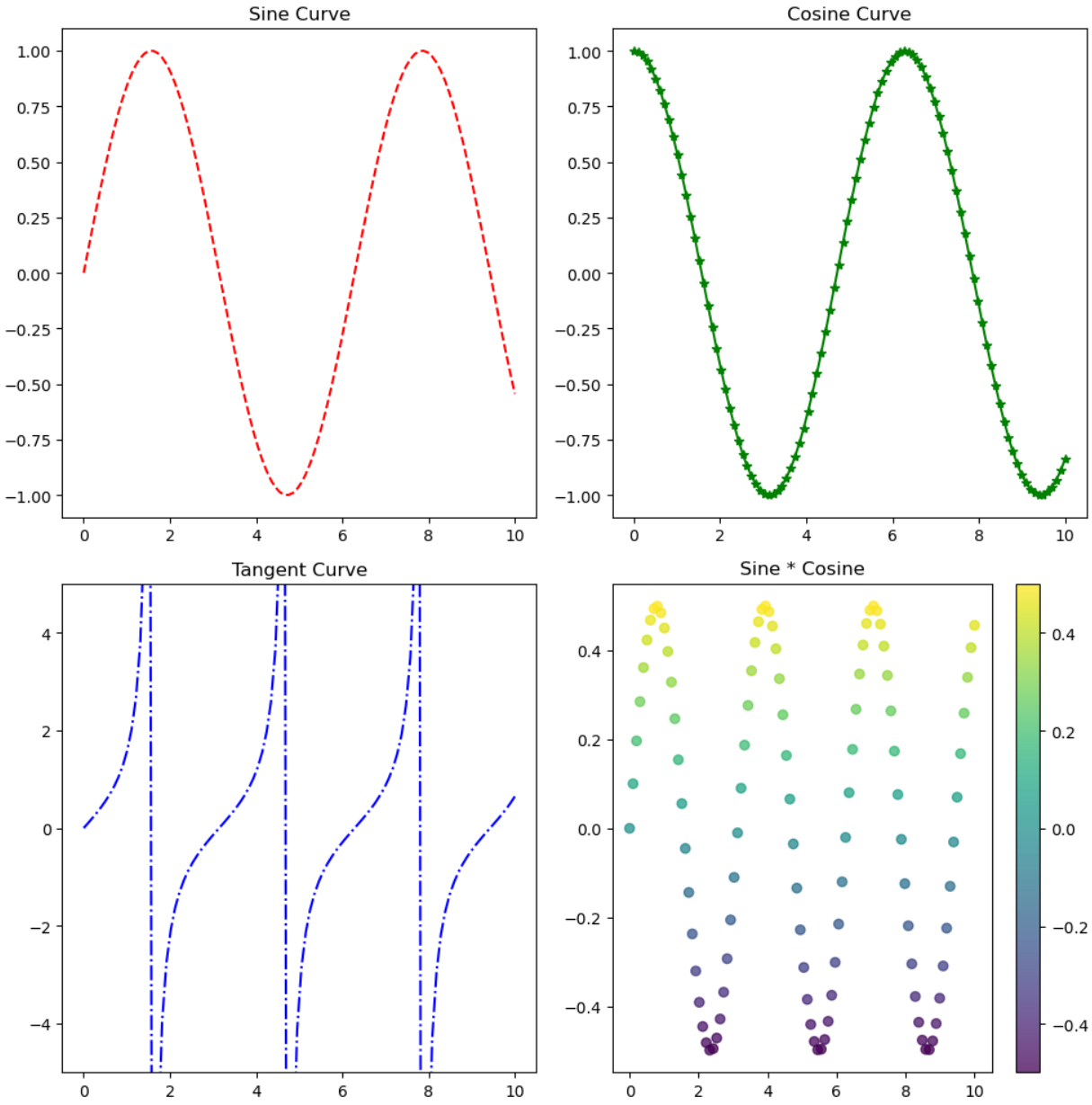
marker 매개변수를 사용하여 점의 모양을 변경할 수 있습니다.

In [ ]:
```
plt.scatter(x, y, marker='D')  # 'D'는 다이아몬드 모양의 마커를 의미합니다.
plt.title('Scatter Plot with Diamond Markers')

plt.show()
```

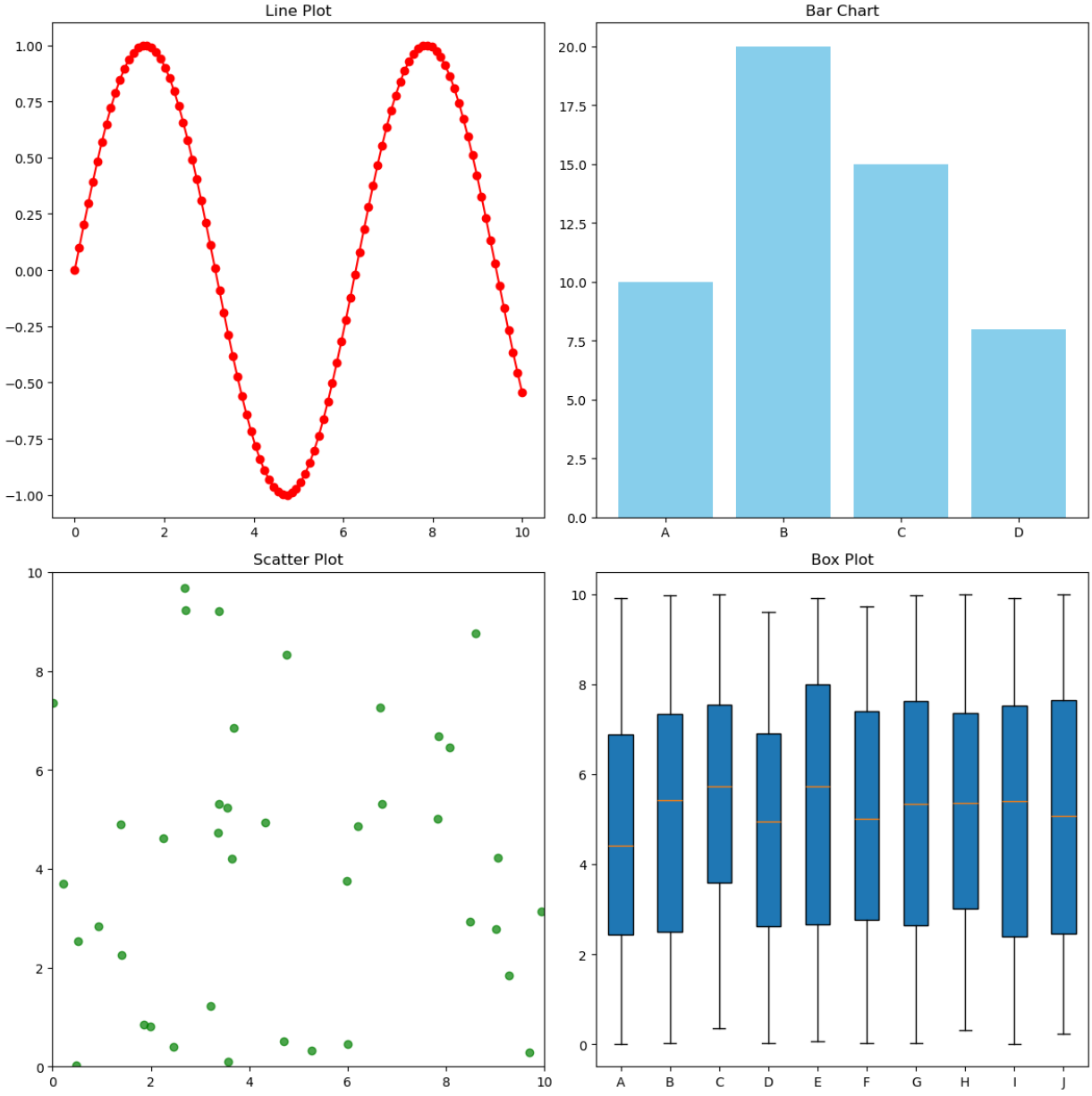


6. 복잡한 형태의 그래프 작성 예시

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

# 데이터 생성
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)
y3 = np.tan(x)
y4 = np.sin(x) * np.cos(x)

# 2x2 형태로 4개의 그래프 생성
fig, axs = plt.subplots(2, 2, figsize=(10, 10))

# 첫 번째 그래프: 사인 곡선
axs[0, 0].plot(x, y1, 'r--')  # 빨간색 점선
axs[0, 0].set_title('Sine Curve')

# 두 번째 그래프: 코사인 곡선
axs[0, 1].plot(x, y2, 'g-*')  # 초록색 별 모양 마커
axs[0, 1].set_title('Cosine Curve')

# 세 번째 그래프: 탄젠트 곡선
axs[1, 0].plot(x, y3, 'b-.')  # 파란색 대시-점선
axs[1, 0].set_title('Tangent Curve')
axs[1, 0].set_ylim(-5, 5)  # y축 범위 제한

# 네 번째 그래프: 사인과 코사인의 곱
sc = axs[1, 1].scatter(x, y4, c=y4, cmap='viridis', alpha=0.75)
    # 색상으로 값 표현
axs[1, 1].set_title('Sine * Cosine')
fig.colorbar(sc, ax=axs[1, 1], orientation='vertical')  # 컬러바 추가

# 전체 그래프에 대한 간격 조정
plt.tight_layout()

plt.show()
```


Sine Curve
Cosine Curve
Tangent Curve
Sine * Cosine

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

# 데이터 생성
x = np.linspace(0, 10, 100)
y = np.sin(x)
categories = ['A', 'B', 'C', 'D']
values = [10, 20, 15, 8]
data_for_scatter = np.random.rand(40, 2) * 10  # 산점도용 데이터
data_for_box = [np.random.rand(100) * 10 for _ in range(10)]
    # 박스 플롯용 데이터, 10개의 데이터 세트 생성

# 2x2 subplot 생성
fig, axs = plt.subplots(2, 2, figsize=(12, 12))

# 선 그래프
axs[0, 0].plot(x, y, marker='o', linestyle='-', color='r')
axs[0, 0].set_title('Line Plot')

# 막대 그래프
axs[0, 1].bar(categories, values, color='skyblue')
axs[0, 1].set_title('Bar Chart')

# 산점도
axs[1, 0].scatter(
    data_for_scatter[:, 0], data_for_scatter[:, 1],
    c='green', marker='o', alpha=0.7
)
axs[1, 0].set_title('Scatter Plot')
axs[1, 0].set_xlim(0, 10)
axs[1, 0].set_ylim(0, 10)

# 박스 플롯
axs[1, 1].boxplot(data_for_box, patch_artist=True, vert=True)
axs[1, 1].set_title('Box Plot')
axs[1, 1].set_xticks(
    range(1, 11),
    ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J']
)

# 전체 그래프 간격 조정
plt.tight_layout()

plt.show()
```


Line Plot
Bar Chart
Scatter Plot
Box Plot
A
B
C
D
E
F
G
H
I
J