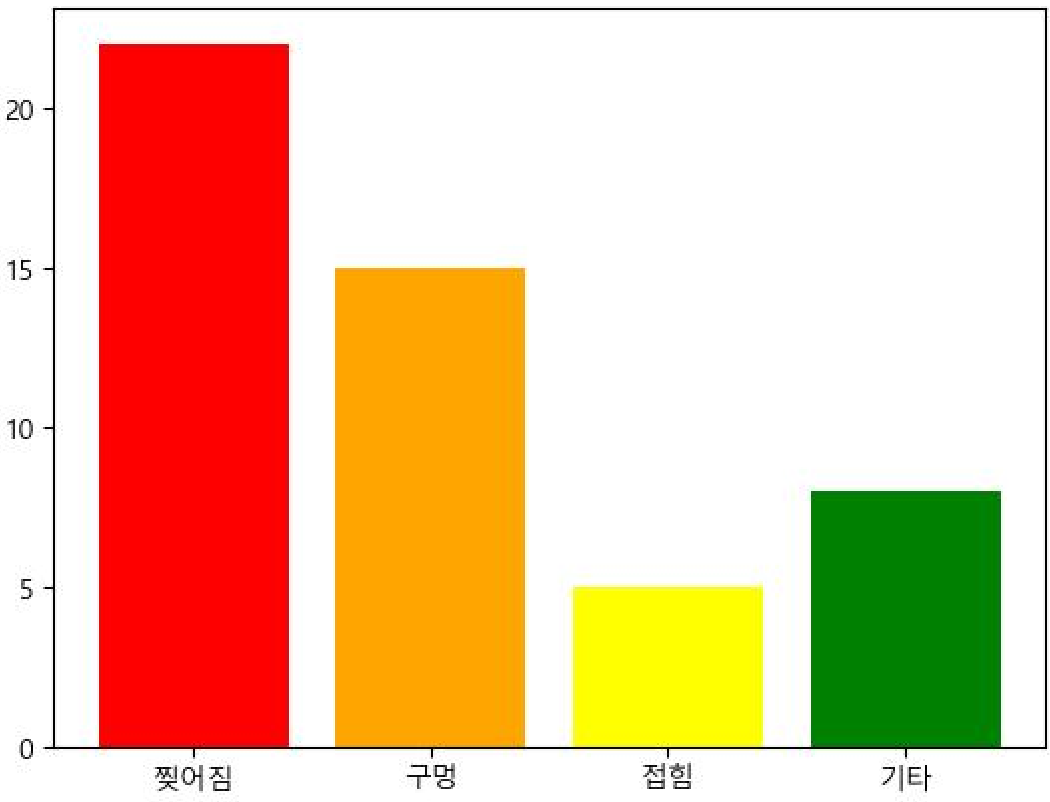
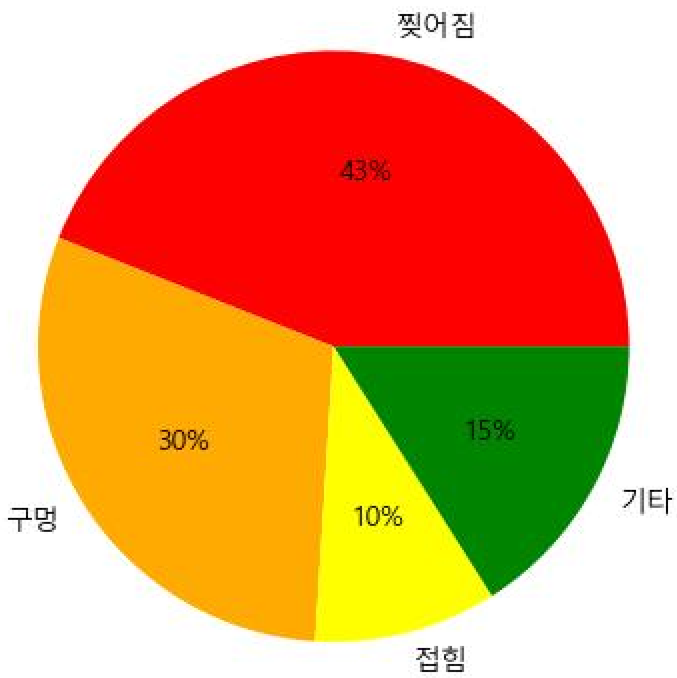
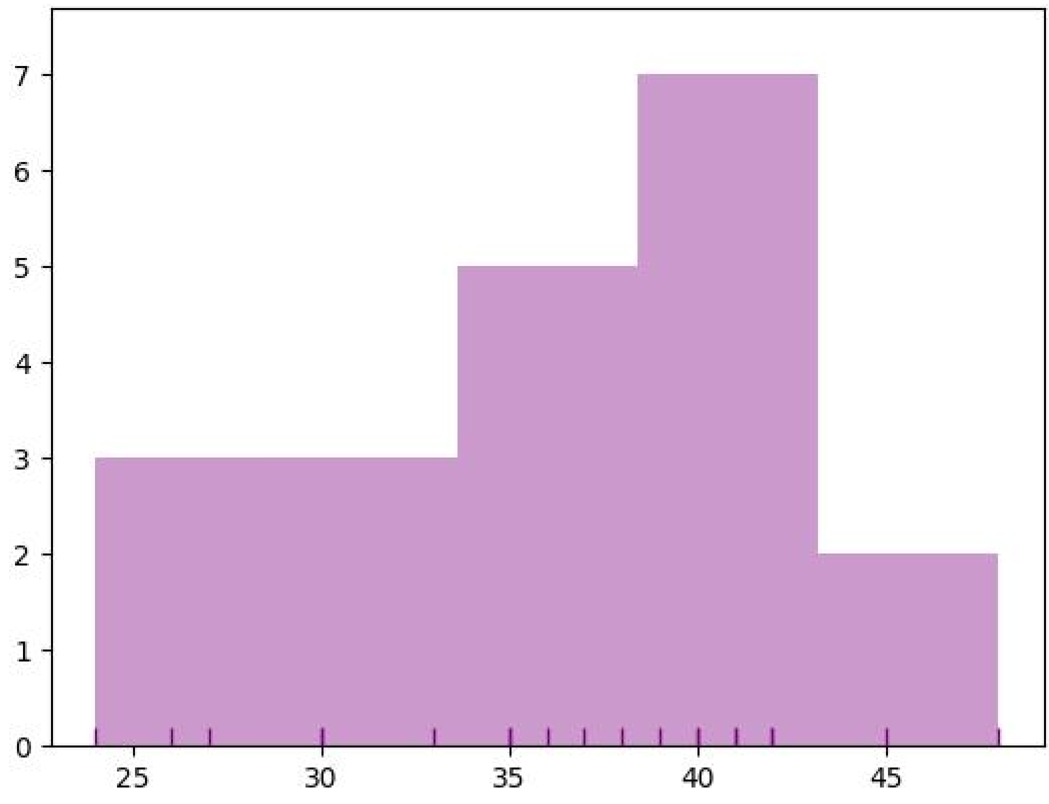
```
# 예제 1.3 p15

import matplotlib.pyplot as plt
import numpy as np
# 패키지 선언

plt.rc('font', family='Malgun Gothic')
# 한글 폰트 설정

x = np.arange(4) #values 개수만큼 필요

desc = ['찢어짐', '구멍', '접힘', '기타']
values = [22, 15, 5, 8]
# 설명과 변수 입력

plt.bar(x, values)
# 기본 막대그래프 생성

plt.bar(x, values, color=['red', 'orange', 'yellow', 'green'])
# 색상이 있는 막대그래프 생성
plt.xticks(x, desc)
plt.show()
```



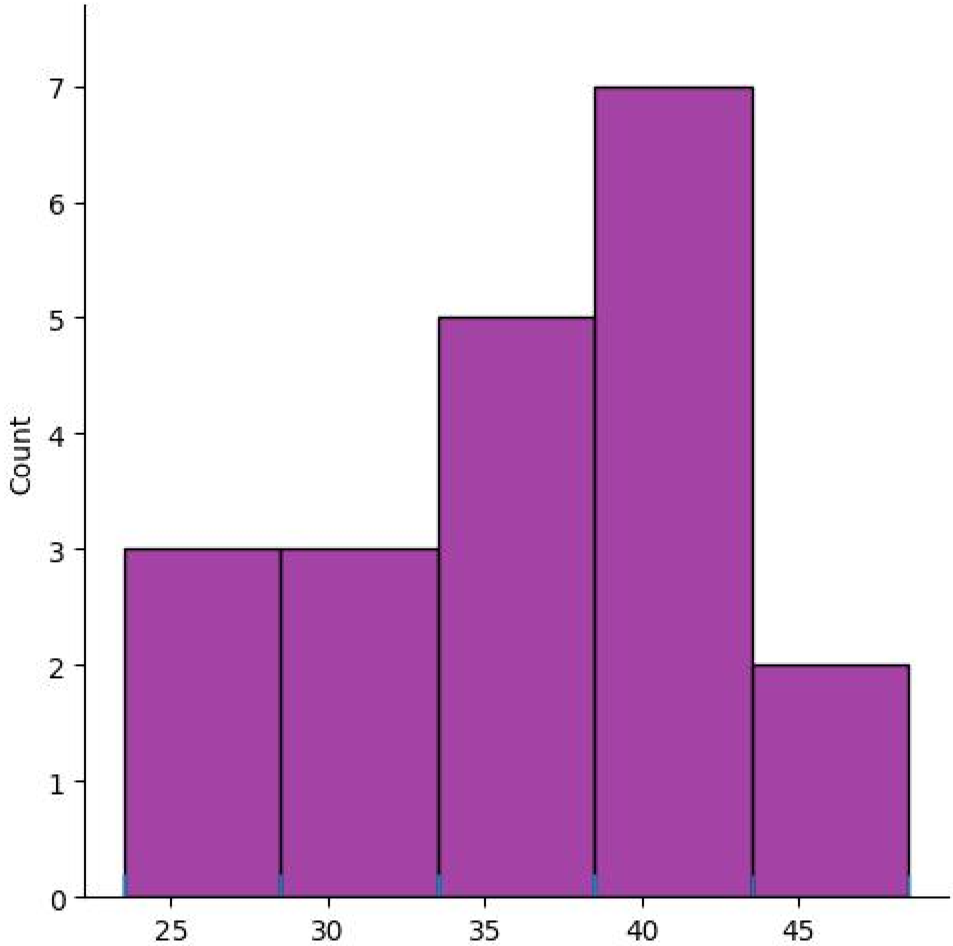
```
# 예제 1-3 p15

import matplotlib.pyplot as plt
import numpy as np
# 패키지 선언

plt.rc('font', family='Malgun Gothic')
# 한글 폰트 설정

desc = ['찢어짐', '구멍', '접힘', '기타']
values = [22, 15, 5, 8]
color=['red', 'orange', 'yellow', 'green']
# 설명과 변수 입력

# plt.pie(values, labels=desc, autopct='%d%%')
# 원 그래프 생성, 정수로만 표시

# plt.pie(values, labels=desc, autopct='%.1f%%')
# 원 그래프 생성, 소수점 한 자리까지 표시
plt.pie(values, labels=desc, autopct='%d%%', colors=color)
# 원 그래프 생성, 정수로만 표시, 색상 지정

plt.show()
```



```
# 예제 1.6 p15
import seaborn as sns
import matplotlib.pyplot as plt
# 패키지 선언

degree = [33, 36, 37, 38, 39, 27, 30, 35, 26, 48, 40, 41, 41, 42, 45, 35, 24, 30, 40, 42]

sns.distplot(degree, bins=5, color='purple', kde=False, rug=True)
plt.show()
```




```
# 최신문법
import seaborn as sns
import matplotlib.pyplot as plt

degree = [33, 36, 37, 38, 39, 27, 30, 35, 26, 48, 40, 41, 41, 42, 45, 35, 24, 30, 40, 42]

sns.displot(degree, bins=5, color='purple', kde=False)
sns.rugplot(degree)
plt.show()
```



Key: aggr|stem|leaf
25 | **18** | *9* = **18** .*9*x10 = 189.0

25	18	9
24	17	
24	16	5
23	15	4
22	14	5
21	13	002
18	12	0027
14	11	02
12	10	3
11	9	049
8	8	3469
4	7	37
2	6	5
1	5	6

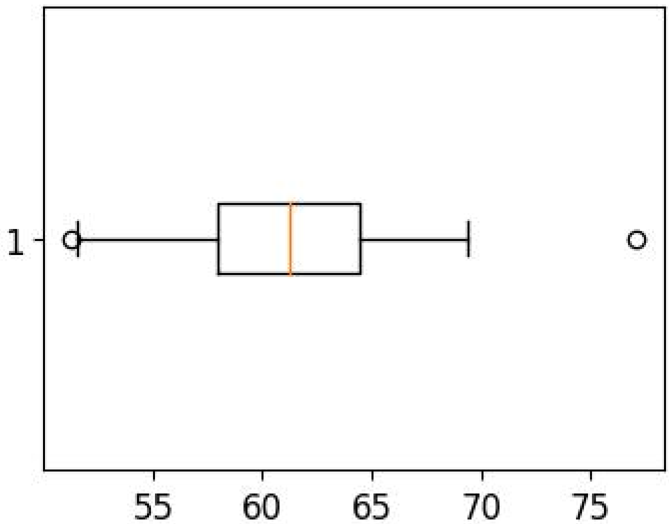
```
#예제 1.8 줄기 잎 그림 , p24
import stemgraphic

value = [56, 89, 165, 73, 83, 145, 90, 189, 127, 77, 110, 112, 132, 120, 94, 130, 84, 65, 99, 154, 86, 120, 122, 103, 130]
# 데이터 입력

stemgraphic.stem_graphic(value, scale=10)
# 데이터 출력

# 4.1 예제 (1.10) , p28
from statistics import *
unnecessary_surgery = [2, 2, 8, 20, 33]

# q1, 중앙값과 평균을 구하여라
print(f"q1 중앙값 : {median(unnecessary_surgery)}")
print(f"q1 평균 : {mean(unnecessary_surgery)}\n")

# q2, 만약 5번째 의사가 수술을 추가 제안한다면,
unnecessary_surgery[4] = 58
print(f"q2 중앙값 : {median(unnecessary_surgery)}")
print(f"q2 평균 : {mean(unnecessary_surgery)}")
```

q1 중앙값 : 8
q1 평균 : 13

q2 중앙값 : 8
q2 평균 : 18

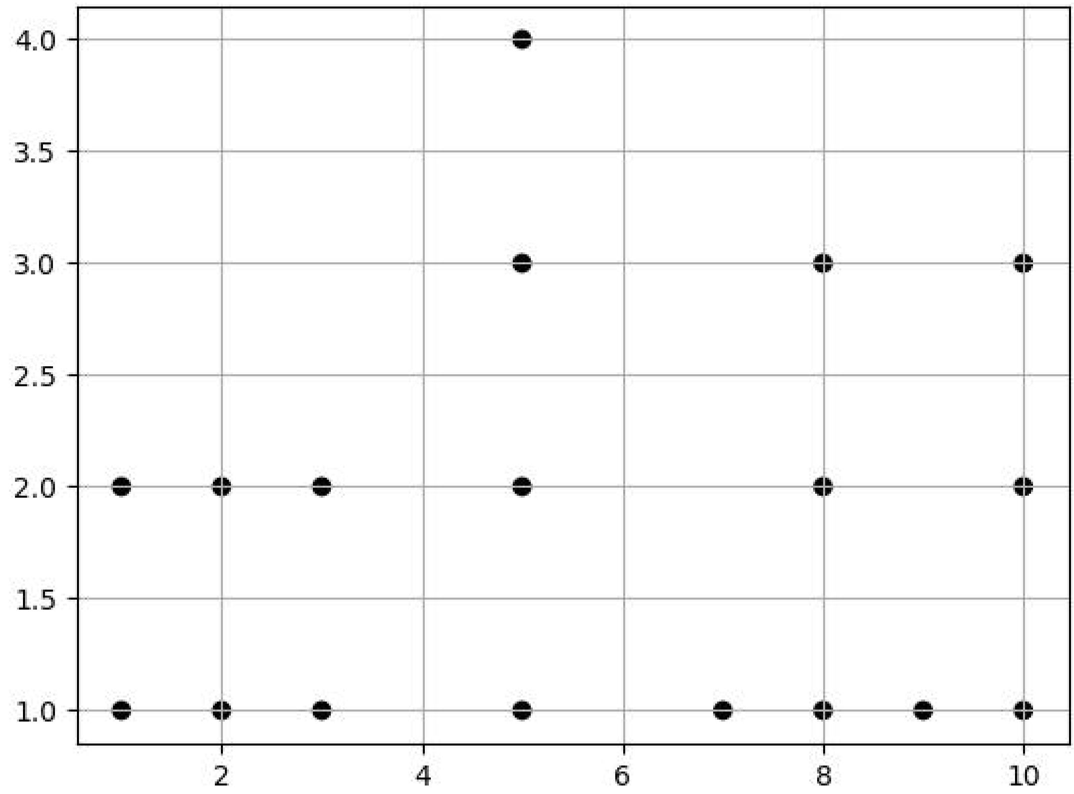
```
# 예제 (1.13) , p32
import matplotlib.pyplot as plt
import numpy as np

# 기본 스타일 설정
plt.style.use('default')
plt.rcParams['figure.figsize'] = (4, 3)
plt.rcParams['font.size'] = 12

# 데이터 준비
np.random.seed(0)
values = [55.9, 63.8, 57.2, 59.8, 65.7, 62.7, 60.8, 51.3, 61.8, 56.0, 66.9, 56.8, 66.2,
```

```
64.6, 59.5, 63.1, 60.6, 62.0, 59.4, 67.2, 63.6, 60.5, 66.8, 61.8, 64.8, 55.8, 55.7, 77.1,
62.1, 61.0, 58.9, 60.0, 66.9, 61.7, 60.3, 51.5, 67.0, 60.2, 56.2, 59.4, 67.9, 64.9, 55.7,
61.4, 62.6, 56.4, 56.4, 69.4, 57.6, 63.8]

# 그래프 그리기
fig, ax = plt.subplots()

box = ax.boxplot([values], notch=False, whis=1, vert=False)
# vert : True(수직), False(수평), whis : 이상치 경계값, notch : 데이터의 모양을 홈 모양
으로 표시

plt.show()
```




```
# 예제 (1.14) , p33
from statistics import *

work_time = [45, 43, 41, 39, 39, 35, 37, 40, 39, 36, 37]

print(f"평균 : {round(mean(work_time),1)}")
print(f"분산 : {variance(work_time)}")
print(f"표준편차 : {stdev(work_time)}")
# 평균은 round 함수 이용 소수점 첫째자리까지 표시/반올림, 분산, 표준편차는 전부 표시

평균 : 39.2
분산 : 8.963636363636363
표준편차 : 2.9939332597164494

# 연습문제 1, p39
import pandas as pd
```

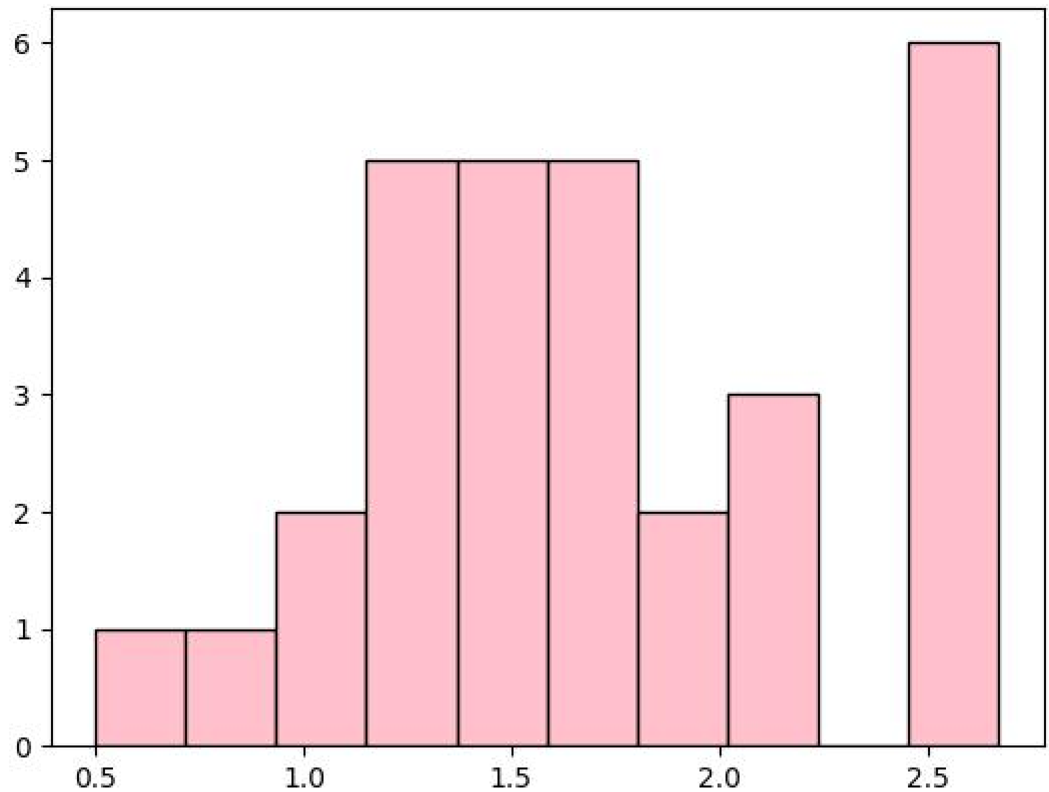
```
# 데이터 입력
blood = ['O', 'O', 'A', 'B', 'A', 'O', 'A', 'A', 'A', 'O', 'B', 'O', 'B', 'O', 'O', 'A', 'O', 'O', 'A', 'A', 'A', 'A', 'AB', 'A', 'B', 'A', 'A', 'O', 'O', 'A', 'O', 'O', 'A', 'A', 'A', 'O', 'A', 'O', 'O', 'AB']

# 데이터프레임으로 개수 구하기
pd.Series(blood).value_counts()

A     18
O     16
B      4
AB     2
Name: count, dtype: int64

# 연습문제 3, p39
import numpy as np
import matplotlib.pyplot as plt

values = [15, 3, 18, 10, 5, 12, 8, 5, 8, 10, 7, 2, 1, 5, 3, 5, 15, 10, 15, 9, 8, 18, 1, 2, 11]

bins = np.arange(11) + 0.5
# 히스토그램 계급구간 만들기

hist, edges = np.histogram(values, bins=bins)
# 계급구간 도수 구하기

y = np.arange(1,hist.max()+1)
# y = 도수 범위
x = np.arange(10) + 1
# 데이터 속성값 범위
X, Y = np.meshgrid(x, y)
# x-y 평면 범위 ( 격자 형태 )

plt.scatter(X, Y, c=Y<=hist, cmap="Greys", )
# 산점도 플롯 그리기

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




```
# 연습문제 5, p40
import pandas as pd

speed = [1.28, 1.36, 1.24, 2.47, 1.94, 2.52, 2.67, 1.37, 1.56, 2.66, 2.17, 1.57, 2.10, 2.54, 1.63, 2.11, 2.57, 1.72, 0.76, 1.02, 1.78, 0.50, 1.49, 1.57, 1.04, 1.92, 1.55, 1.78, 1.70, 1.20]
label = ['0.45~0.89','0.90~1.34','1.35~1.79','1.80~2.24','2.25~2.70']

speed_cut = pd.cut(speed, 5, labels=label)
pd.value_counts(speed_cut).sort_index()

0.45~0.89     2
0.90~1.34     6
1.35~1.79    11
1.80~2.24     5
2.25~2.70     6
Name: count, dtype: int64

# 연습문제 5, p40
import matplotlib.pyplot as plt
```

speed = [1.28, 1.36, 1.24, 2.47, 1.94, 2.52, 2.67, 1.29, 1.56, 2.66, 2.17, 1.57, 2.10, 2.54, 1.63, 2.11, 2.57, 1.72, 0.76, 1.02, 1.78, 0.50, 1.49, 1.57, 1.04, 1.92, 1.55, 1.78, 1.70, 1.20]

plt.hist(speed, color='pink', edgecolor='black')
plt.show()



연습문제 4, p40

import stemgraphic

value = [20, 18, 25, 26, 17, 14, 20, 40, 18, 15, 22, 15, 17, 25, 22, 12, 52, 27, 24, 41, 34, 20, 17, 20, 19, 20, 16, 20, 15, 34, 22, 29, 29, 34, 27, 13, 6, 24, 47, 32, 12, 17, 36, 35, 41, 36, 32, 46, 30, 51]
# 데이터 입력

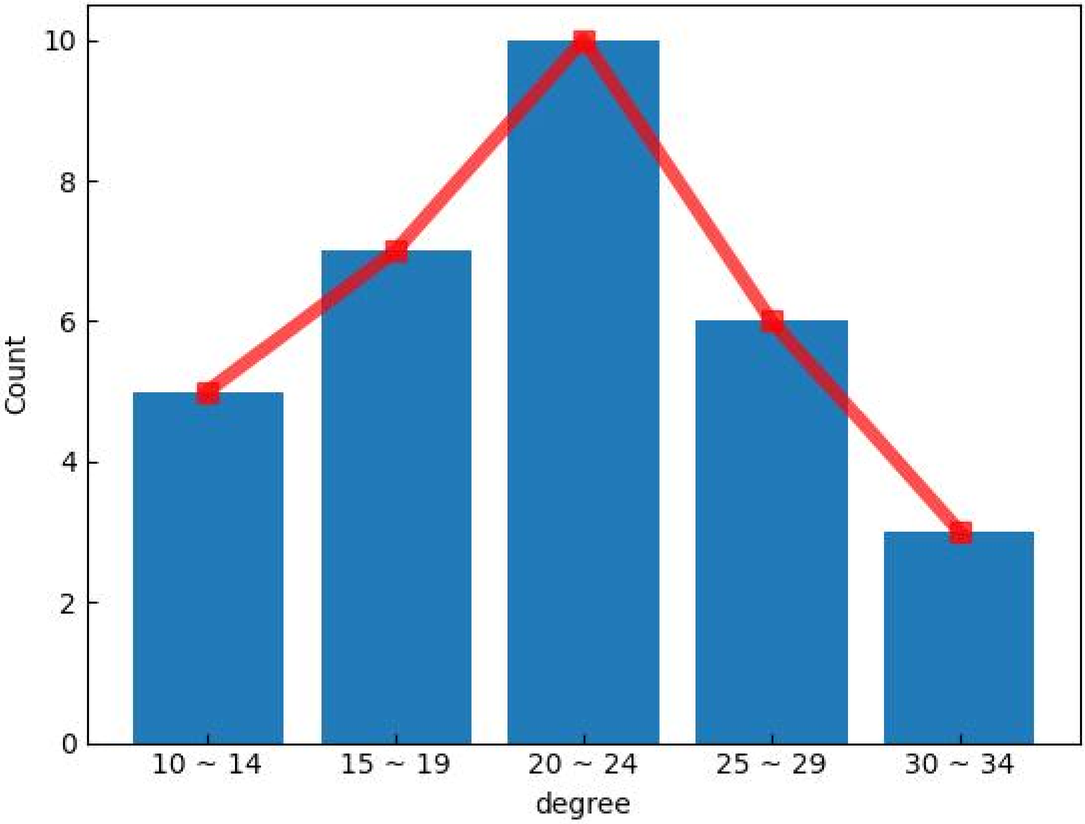
stemgraphic.stem_graphic(value, scale=10)
# 데이터 출력


```
# 연습문제 9, p41
from statistics import *

temp = [15.2, 11.0, 16.8, 23.2, 14.3, 21.9, 22.4, 20.5, 15.0, 17.0, 12.8, 21.0, 27.7, 28.0, 18.8, 16.4, 14.9, 20.0, 23.5, 23.9, 24.0, 13.2, 13.6, 24.1, 25.9, 30.8, 26.3, 32.1, 29.2, 31.5, 28.5]

print(f"최대 : {max(temp)}")
print(f"최소 : {min(temp)}")
print((max(temp) - min(temp)) // 5)
```

최대 : 32.1
최소 : 11.0
4.0

```
# 연습문제 9, p41
data = [15.2, 15.3, 16.8, 23.2, 14.3, 21.9, 22.4, 20.5, 15.0, 17.0, 12.8, 21.0, 27.7, 28.0, 18.8, 16.4, 14.9, 20.0, 23.5, 23.9, 24.0, 13.2, 13.6, 24.1, 25.9, 30.8, 26.3, 32.1, 29.2, 31.5, 28.5]
bins = [10,15,20,25,30,35]
labels = ['10 ~ 14', '15 ~ 19', '20 ~ 24', '25 ~ 29', '30 ~ 34']

freq_table = {}
for label in labels:
    freq_table[label] = 0

for value in data:
    for i in range(len(bins)-1):
        if bins[i] <= value < bins[i+1]:
            freq_table[labels[i]] +=1
```

```
            break

print(freq_table)
{'10 ~ 14': 5, '15 ~ 19': 7, '20 ~ 24': 10, '25 ~ 29': 6, '30 ~ 34': 3}


# 연습문제 9, p41
import matplotlib.pyplot as plt
import numpy as np

x = np.arange(5)
label = ['10 ~ 14', '15 ~ 19', '20 ~ 24', '25 ~ 29', '30 ~ 34']
values = [freq_table['10 ~ 14'], freq_table['15 ~ 19'], freq_table['20 ~ 24'], freq_table['25 ~ 29'], freq_table['30 ~ 34']]

fig, ax1 = plt.subplots()
ax1.plot(label, values, '-s', color='red', markersize=7, linewidth=5, alpha=0.7, label='Count')
ax1.set_xlabel('degree')
ax1.set_ylabel('Count')
ax1.tick_params(axis='both', direction='in')

plt.bar(x, values)
plt.xticks(x, label)

plt.show()
```




```
# 연습문제 17, p43
import stemgraphic

time = [5.9, 5.3, 1.6, 7.4, 9.8, 1.7, 8.9, 1.2, 2.1, 4.0, 6.5, 7.2, 7.3, 8.4, 8.9, 6.7, 9.2,
2.8, 4.5, 6.3, 7.6, 9.7, 9.4, 8.8, 3.5, 1.1, 4.3, 3.3, 3.1, 1.3, 8.4, 1.6, 8.2, 6.5, 4.1, 3.1,
1.1, 5.0, 9.4, 6.4, 7.7, 2.7]

stemgraphic.stem_graphic(time, scale=1)
```

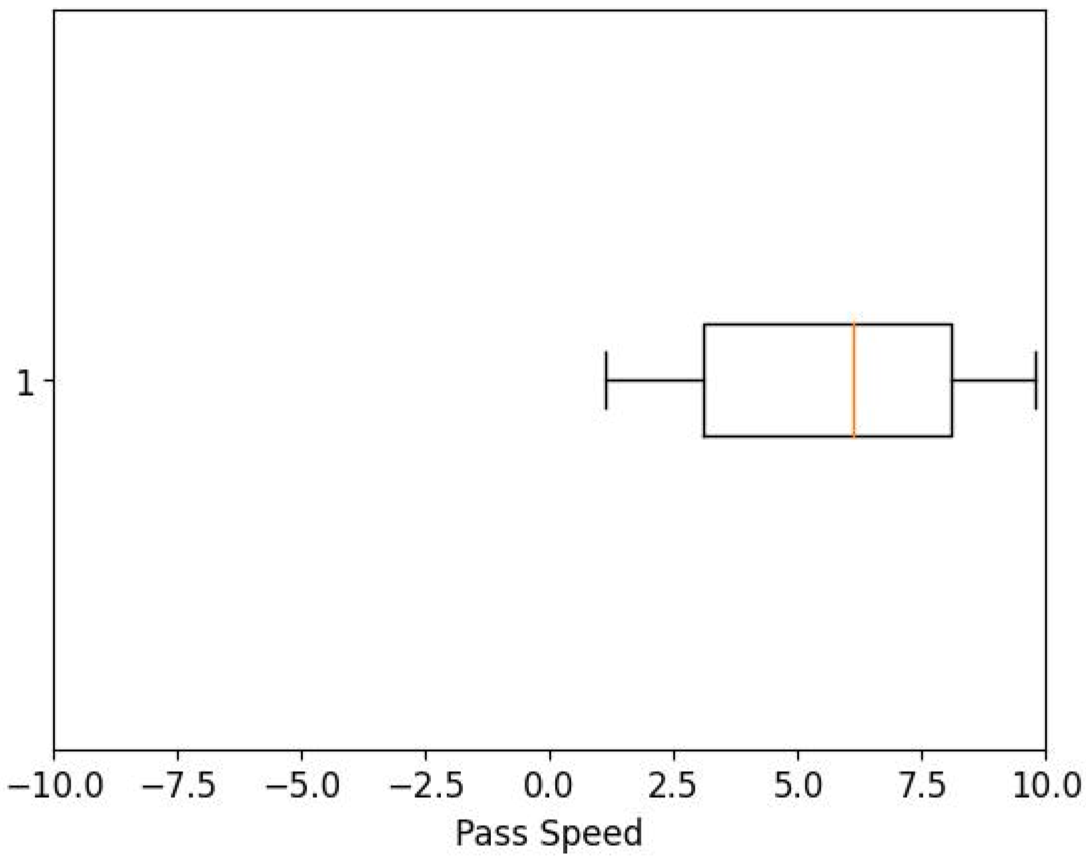

```
# 연습문제 17, p43
from statistics import *

time = [5.9, 5.3, 1.6, 7.4, 9.8, 1.7, 8.9, 1.2, 2.1, 4.0, 6.5, 7.2, 7.3, 8.4, 8.9, 6.7, 9.2,
2.8, 4.5, 6.3, 7.6, 9.7, 9.4, 8.8, 3.5, 1.1, 4.3, 3.3, 3.1, 1.3, 8.4, 1.6, 8.2, 6.5, 4.1, 3.1,
1.1, 5.0, 9.4, 6.4, 7.7, 2.7]

print("평균: ", mean(time))
print("중위수(중앙값): ", median(time))
print("최빈수(최빈값): ", mode(time))
```

```
평균:  5.523809523809524
중위수(중앙값):  6.1
최빈수(최빈값):  1.6
```

```
# 연습문제 17, p43
from statistics import *

time = [5.9, 5.3, 1.6, 7.4, 9.8, 1.7, 8.9, 1.2, 2.1, 4.0, 6.5, 7.2, 7.3, 8.4, 8.9, 6.7, 9.2,
2.8, 4.5, 6.3, 7.6, 9.7, 9.4, 8.8, 3.5, 1.1, 4.3, 3.3, 3.1, 1.3, 8.4, 1.6, 8.2, 6.5, 4.1, 3.1,
1.1, 5.0, 9.4, 6.4, 7.7, 2.7]

print("범위: ", max(time) - min(time))
print("표준편차: ", stdev(time))
```

```
# 연습문제 17, p43
import matplotlib.pyplot as plt
import numpy as np
```

```
time = [5.9, 5.3, 1.6, 7.4, 9.8, 1.7, 8.9, 1.2, 2.1, 4.0, 6.5, 7.2, 7.3, 8.4, 8.9, 6.7, 9.2,
2.8, 4.5, 6.3, 7.6, 9.7, 9.4, 8.8, 3.5, 1.1, 4.3, 3.3, 3.1, 1.3, 8.4, 1.6, 8.2, 6.5, 4.1, 3.1,
1.1, 5.0, 9.4, 6.4, 7.7, 2.7]

plt.style.use('default')
# 플롯 스타일 설정

plt.rcParams['font.size'] = 12
# 폰트 사이즈 지정

fig, ax = plt.subplots()
# 서브플롯 할당

ax.boxplot([time], notch=False, whis=2, vert=False)
# 박스플롯 생성

ax.set_xlim(-10.0, 10.0)
ax.set_xlabel('Pass Time')

plt.show()
```

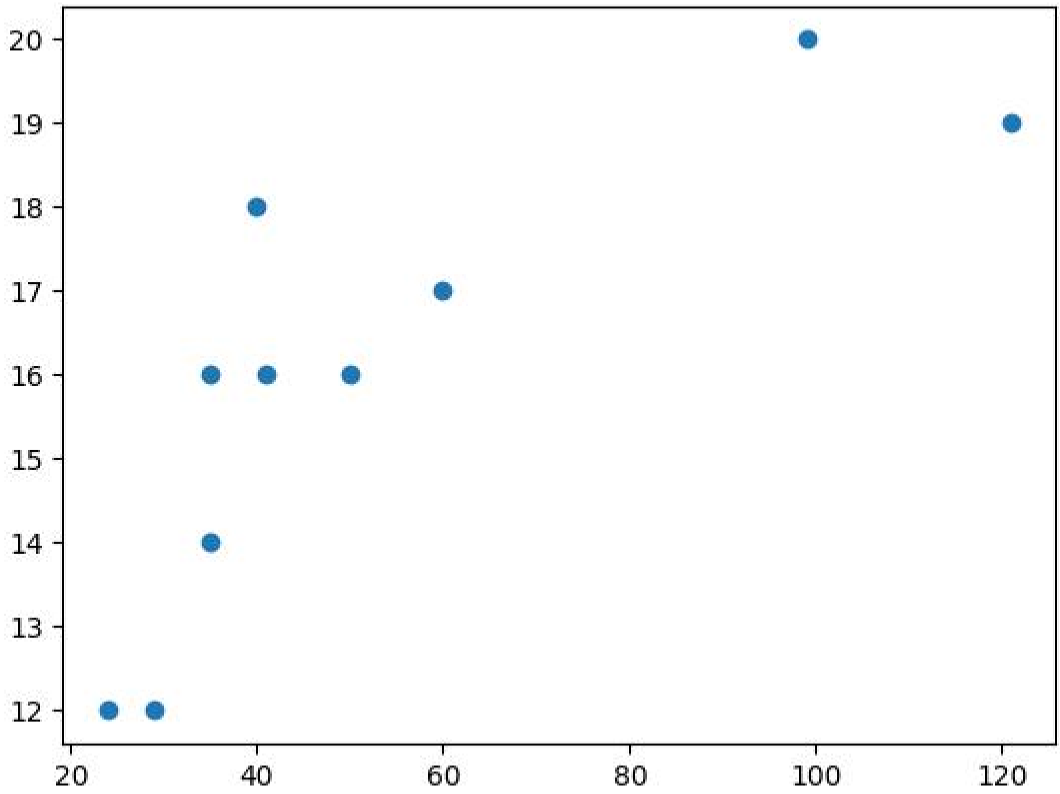



```
# 연습문제 14, p42
from statistics import *

sample = [66.9, 66.2, 71.0, 68.6, 65.4, 68.4, 71.9]

print(f"표본중앙값 : {median(sample)}")
print(f"표본평균 : {mean(sample):.3f}")
print(f"표본표준편차 : {stdev(sample):.3f}")

표본중앙값 : 68.4
표본평균 : 68.343
표본표준편차 : 2.419

# 연습문제 15, p43
import stemgraphic

values = [2.0, 3.0, 0.3, 3.3, 1.3, 0.4, 0.2, 6.0, 5.5, 6.5, 0.2, 2.3, 1.5, 4.0, 5.9, 1.8, 4.7, 0.7, 4.5, 0.3, 1.5, 0.5, 2.5, 5.0, 1.0, 6.0, 5.6, 6.0, 1.2, 0.2]

stemgraphic.stem_graphic(values, scale=1)
```

scale1 = 소수점 왼쪽 1 기반 분석

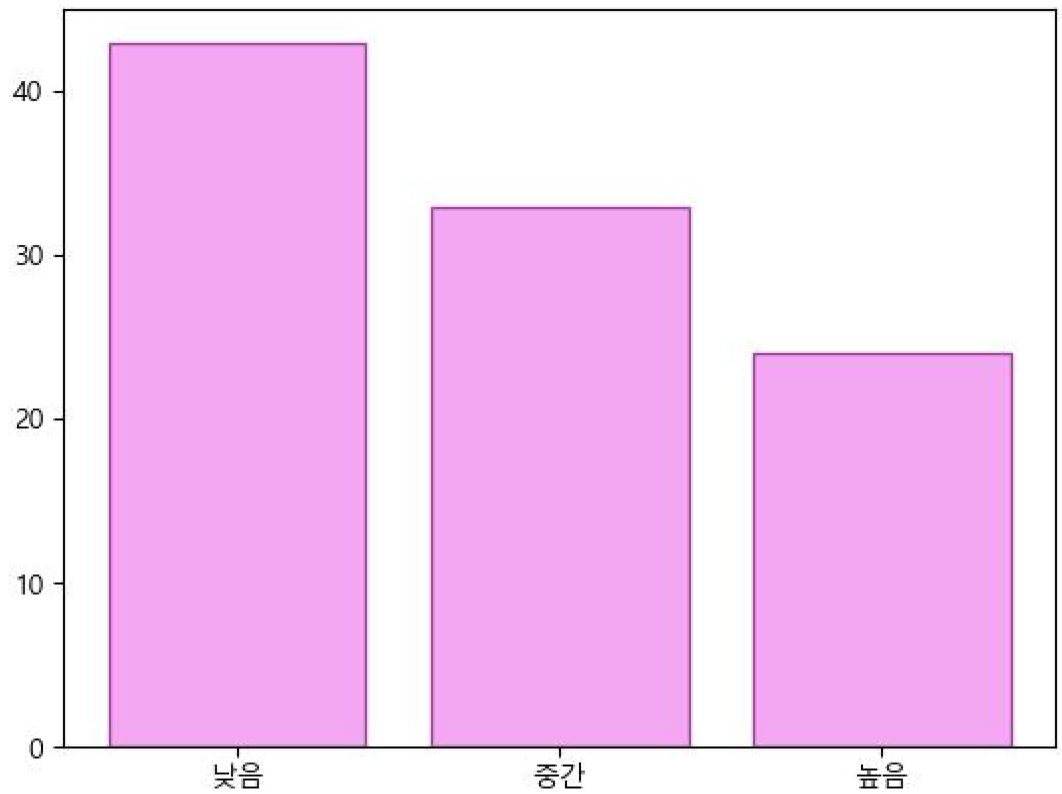

```
# 연습문제 15, p43
import pandas as pd

values = [2.0, 3.0, 0.3, 3.3, 1.3, 0.4, 0.2, 6.0, 5.5, 6.5, 0.2, 2.3, 1.5, 4.0, 5.9, 1.8,
4.7, 0.7, 4.5, 0.3, 1.5, 0.5, 2.5, 5.0, 1.0, 6.0, 5.6, 6.0, 1.2, 0.2]

df_values = pd.DataFrame(values)
draw = df_values.value_counts(normalize=True)
print("상대도수분포")
print(draw)
상대도수분포
0.2    0.100000
6.0    0.100000
1.5    0.066667
0.3    0.066667
3.0    0.033333
5.9    0.033333
5.6    0.033333
5.5    0.033333
5.0    0.033333
4.7    0.033333
4.5    0.033333
4.0    0.033333
3.3    0.033333
2.5    0.033333
2.3    0.033333
2.0    0.033333
1.8    0.033333
1.3    0.033333
1.2    0.033333
```

```
1.0      0.033333
0.7      0.033333
0.5      0.033333
0.4      0.033333
6.5      0.033333
Name: proportion, dtype: float64
# 연습문제 15, p43
from statistics import *

values = [2.0, 3.0, 0.3, 3.3, 1.3, 0.4, 0.2, 6.0, 5.5, 6.5, 0.2, 2.3, 1.5, 4.0, 5.9, 1.8, 4.7, 0.7, 4.5, 0.3, 1.5, 0.5, 2.5, 5.0, 1.0, 6.0, 5.6, 6.0, 1.2, 0.2]

print(f"표본평균: {mean(values):.3f}")
print("표본범위: ", max(values) - min(values))
print(f"표본표준편차: {stdev(values):.3f}")

표본평균: 2.797
표본범위:  6.3
표본표준편차: 2.227

# 추가문제 - 설문조사 #18, p45
# 다음은 직장인 10명의 수입과 교육받은 기간을 조사한 자료이다.
# 산점도와 상관계수를 구하고, 수입과 교육기간 사이에 어떤 상관관계가 있는지 설명하시오.
import matplotlib.pyplot as plt

# 수입과 교육기간 데이터
income = [121, 99, 41, 35, 40, 29, 35, 24, 50, 60]
learning_time = [19, 20, 16, 16, 18, 12, 14, 12, 16, 17]

plt.scatter(income, learning_time)
plt.show()
```



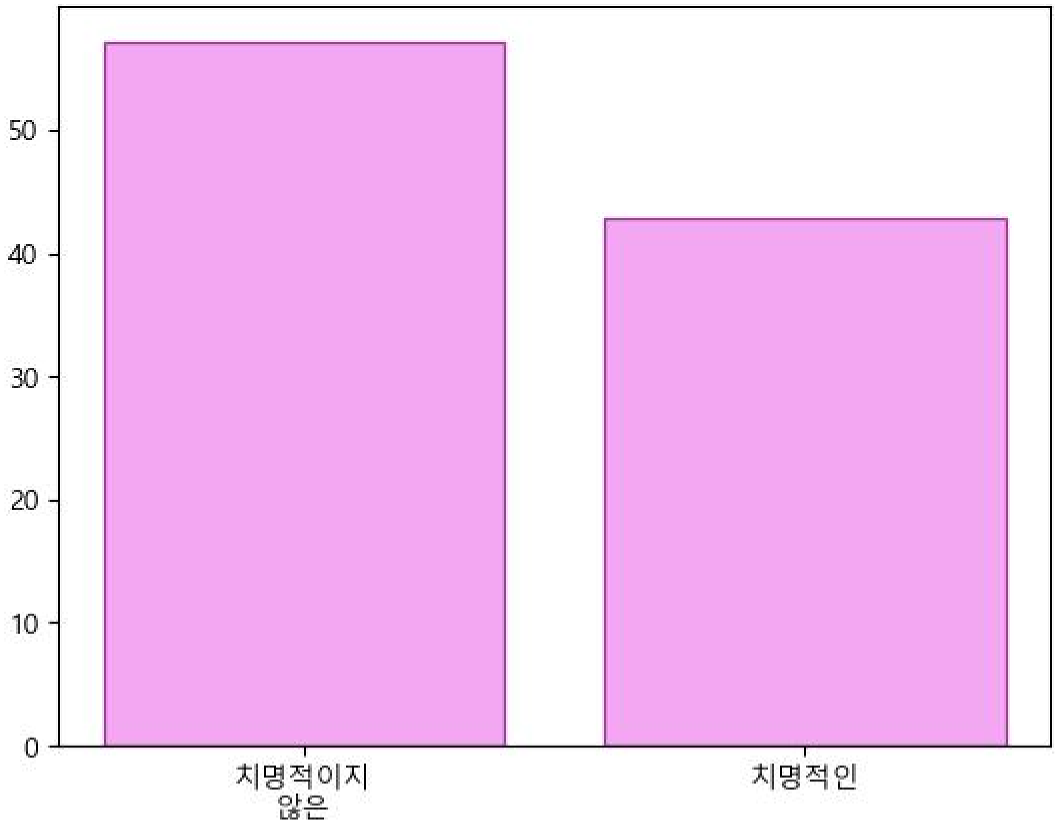

```
# 연습문제 2.8, p55 두변수 자료의 요약
# d type = low, middle, high
from matplotlib import pyplot as plt

non_serious_heart_attack = [29, 17, 18]
serious_heart_attack = [19, 20, 9]
values, re_values, tx_values = [], [], []
label = ['낮음', '중간', '높음']

# 콜레스테롤 수치에 따른 막대그래프, 주변 분포 구하기
# serious_heart_attack의 변수 수를 계산하여 반복하여 더해 저장
for i in range(len(non_serious_heart_attack)):
    xs = non_serious_heart_attack[i] + serious_heart_attack[i]
    values.append(xs)

# 토탈 값 구하기
total = sum(non_serious_heart_attack) + sum(serious_heart_attack)
for i in range(3):
    data = round((values[i] / total), 3)
    data = round((data * 100), 1)
```

```
    re_values.append(data)

plt.rc('font', family='Malgun Gothic')
plt.bar(label, re_values, color='violet', alpha=0.7, edgecolor='purple')
plt.show()
```



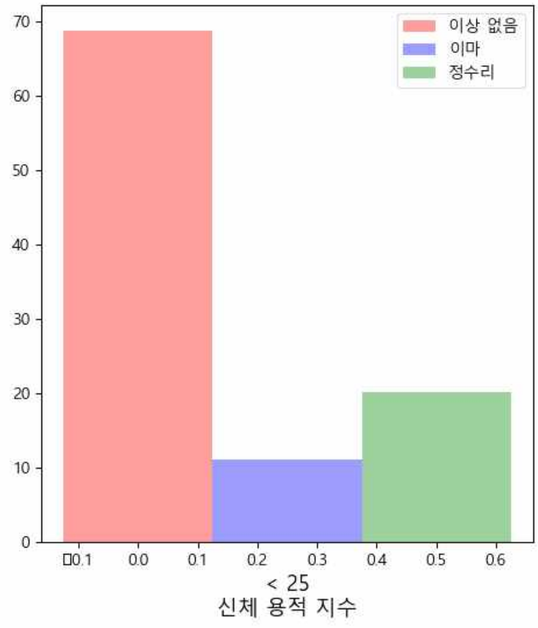

```
# 연습문제 2.8, p55
from matplotlib import pyplot as plt

non_serious_heart_attack = [29, 17, 18]
serious_heart_attack = [19, 20, 9]

thread = sum(non_serious_heart_attack), sum(serious_heart_attack)
values = []
desc = ['치명적이지\n않은', '치명적인']

for i in range(len(thread)):
    total = sum(thread)
    data = round((thread[i] / total), 3)
    data = round((data * 100), 1)
    values.append(data)
```

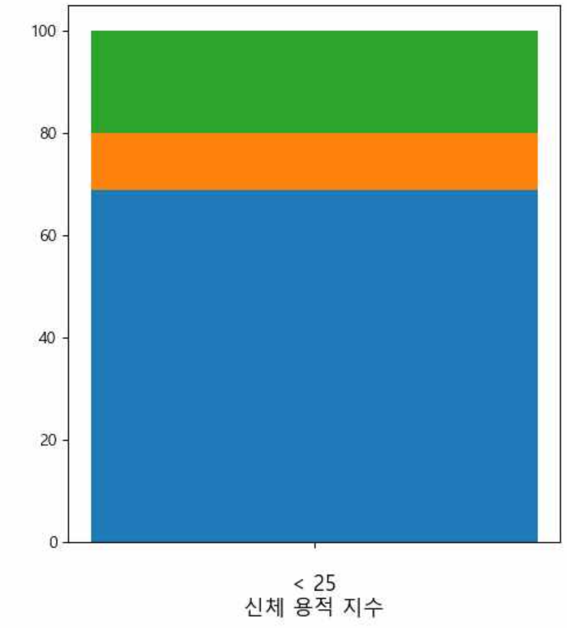
```
plt.rc('font', family='Malgun Gothic')
plt.bar(desc, values, color='violet', alpha=0.7, edgecolor='purple')
plt.show()
```




```
# 연습문제 2.9, p55
import matplotlib.pyplot as plt
import numpy as np

# 폰트값 설정
plt.rc('font', family='Malgun Gothic')

# 데이터 입력
label = ['이상 없음', '이마', '정수리']
vunder_25 = [137, 22, 40]
v25_to_28 = [218, 34, 57]
vhigh_28  = [153, 30, 68]
values = []
colors = ['red', 'green', 'blue']
valve = ['vunder_25', 'v25_to_28', 'vhigh_28']

# under_25
```

```
total = sum(vunder_25)
for i in range(len(vunder_25)):
    data = round((vunder_25[i] / total * 100) ,1)
    values.append(data)

fig, ax = plt.subplots(figsize=(12,6))
bar_width = 0.25
index = np.arange(1)

plt.subplot(121)
b1 = plt.bar(index, values[0], bar_width, alpha=0.4, color='red', label=label[0])

b2 = plt.bar(index + bar_width, values[1], bar_width, alpha=0.4, color='blue', label=label[1])

b3 = plt.bar(index + 2 * bar_width, values[2], bar_width, alpha=0.4, color='green', label=label[2])

plt.xlabel('< 25\n신체 용적 지수', size = 13)
plt.legend()

plt.subplot(122)

plt.xlabel('< 25\n신체 용적 지수', size = 13)

quarters = [' ']
# 탑 형식의 데이터 지정
plt.bar(quarters, values[0])
plt.bar(quarters, values[1], bottom=values[0])
plt.bar(quarters, values[2], bottom=values[0]+values[1])

plt.show()
```

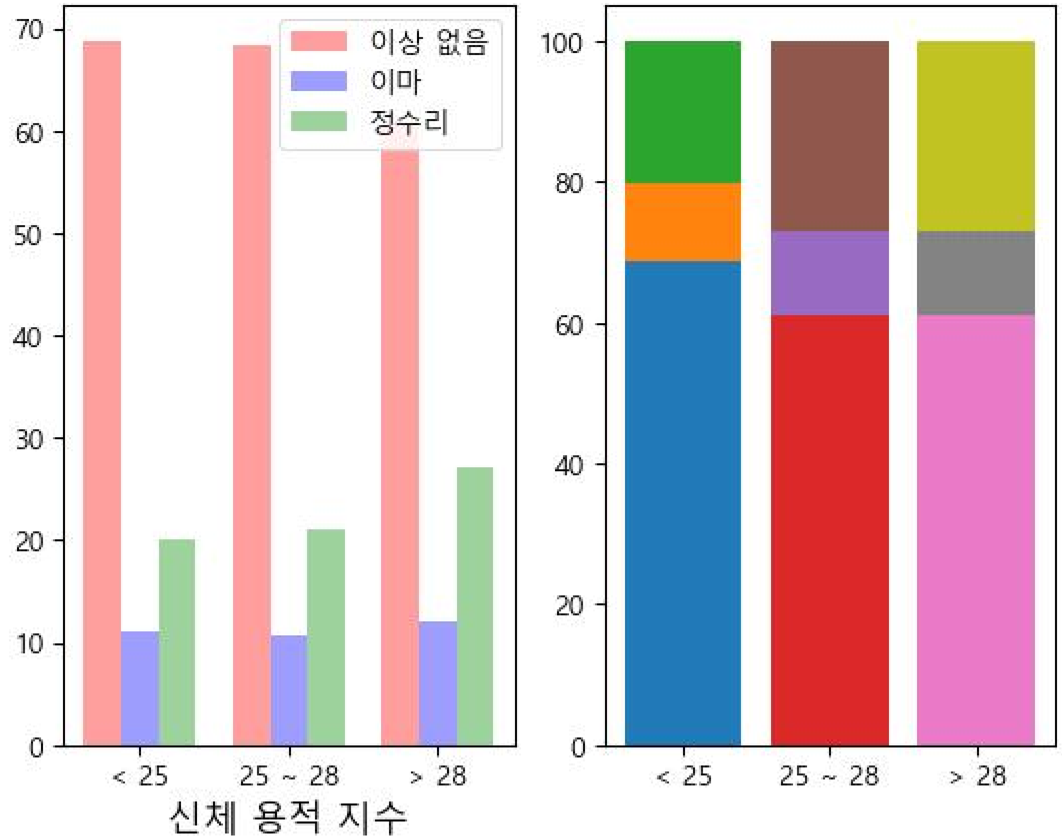





```
# 연습문제 2.9, p55
import matplotlib.pyplot as plt
import numpy as np

bar_width = 0.25

plt.rc('font', family='Malgun Gothic')

label = ['이상 없음', '이마', '정수리']
vunder_25 = [137, 22, 40]
v25_to_28 = [218, 34, 67]
vhigh_28  = [153, 30, 68]
values, values2, values3 = [], [], []
colors = ['red', 'green', 'blue']
valve = ['vunder_25', 'v25_to_28', 'vhigh_28']
desc = ['< 25', '25 ~ 28', '> 28']
x_save, x_save1, x_save2 = [], [], []

# under_25
total = sum(vunder_25)
for i in range(len(vunder_25)):
    data = round((vunder_25[i] / total * 100) ,1)
    x_save.append(data)
    if i == 0:
        values.append(data)
```

```
        elif i == 1:
            values2.append(data)
        else:
            values3.append(data)

total = sum(v25_to_28)
for i in range(len(v25_to_28)):
    data = round((v25_to_28[i] / total * 100) ,1)
    x_save1.append(data)
    if i == 0:
        values.append(data)
    elif i == 1:
        values2.append(data)
    else:
        values3.append(data)

total = sum(vhigh_28)
for i in range(len(vhigh_28)):
    data = round((vhigh_28[i] / total * 100) ,1)
    x_save2.append(data)
    if i == 0:
        values.append(data)
    elif i == 1:
        values2.append(data)
    else:
        values3.append(data)

plt.subplot(121)
index = np.arange(3)
b1 = plt.bar(index, values, bar_width, alpha=0.4, color='red', label=label[0])
b2 = plt.bar(index + bar_width, values2, bar_width, alpha=0.4, color='blue', label=label[1])
b3 = plt.bar(index + 2 * bar_width, values3, bar_width, alpha=0.4, color='green', label=label[2])

plt.xticks(np.arange(bar_width, 3 + bar_width, 1), desc)
plt.xlabel('신체 용적 지수', size = 13)
plt.legend()

plt.subplot(122)
```

```
print(x_save, x_save1, x_save2)

print(values)
# plot2
plt.bar("< 25", x_save[0])
plt.bar("< 25", x_save[1], bottom=x_save[0])
plt.bar("< 25", x_save[2], bottom=x_save[0]+x_save[1])

# plot 2-1
plt.bar("25 ~ 28", x_save2[0])
plt.bar("25 ~ 28", x_save2[1], bottom=x_save2[0])
plt.bar("25 ~ 28", x_save2[2], bottom=x_save2[0]+x_save2[1])

# plot 2-2
plt.bar("> 28", x_save2[0])
plt.bar("> 28", x_save2[1], bottom=x_save2[0])
plt.bar("> 28", x_save2[2], bottom=x_save2[0]+x_save2[1])

plt.show()
```



```
# 예제(5.4), p156 05 여러가지 확률분포
from scipy.integrate import quad
from scipy.stats import *

meeting_hours = 3

def f(x):
  if 0 <= x <= 4:
    return 1/4
  else:
    return 0

print(f"확률밀도함수 : {uniform.pdf(meeting_hours, loc=0, scale=4)}")
result, _ = quad(f, meeting_hours, 4)
print(f"회의가 최소한 3시간 이상일 확률 : {result}")
```

```
확률밀도함수 : 0.25
회의가 최소한 3시간 이상일 확률 : 0.25
```

```
# 예제(5.6), p159 + 함수를 사용하는 방식
import math
from math import comb

n = 20
# 납품시 검사하는 장비의 수
p = 0.02
# 불량률
k = 10
# 납품 횟수

# 함수를 사용하는 방식
def binomial_distribution(n: int, p: float, x: int) -> float:
    return math.comb(n, x) * p ** x * (1 - p) ** (n - x)

def at_least_one_failure(n: int, p: float) -> float:
    return 1 - binomial_distribution(n, p, 0)

print(f"적어도 한 대의 불량률이 있을 확률 : {round(((at_least_one_failure(n, p)) % 100), 4)}")
```

```
p_v = 1 - (1 - 0.02) ** 20
fcs = comb(k, 2) * (p_v ** 2) * ((1 - p_v) ** (k - 2))
print(f"적어도 한 대의 불량품이 포함될 확률이 2번 있는 경우 : {round((fcs), 4)}")
```

적어도 한 대의 불량률이 있을 확률 : 0.3324
적어도 한 대의 불량품이 포함될 확률이 2번 있는 경우 : 0.1962

```
# 예제(5.11), p161
from math import comb

n = 5     # 전체 제품 수
N = 40    # 전체 제품 중 불량품 수
k = 3     # 뽑은 제품 중 불량품 수

p = comb(3, 1) * comb(37, 4) / comb(40, 5)
print(f"정확히 한 개의 불량품이 있을 확률 : {round((p), 4)}")
```

정확히 한 개의 불량품이 있을 확률 : 0.3011

```
# 예제(5.20), p168

p = 0.05    # 5%의 확률로 상대방과 통화할 수 있음
# 1 try

value = (p) * (1-p) ** 3
print(f"네 번째 시도에서 상대방과 통화할 수 있는 확률 : {round(value, 4)}")
```

네 번째 시도에서 상대방과 통화할 수 있는 확률 : 0.0429

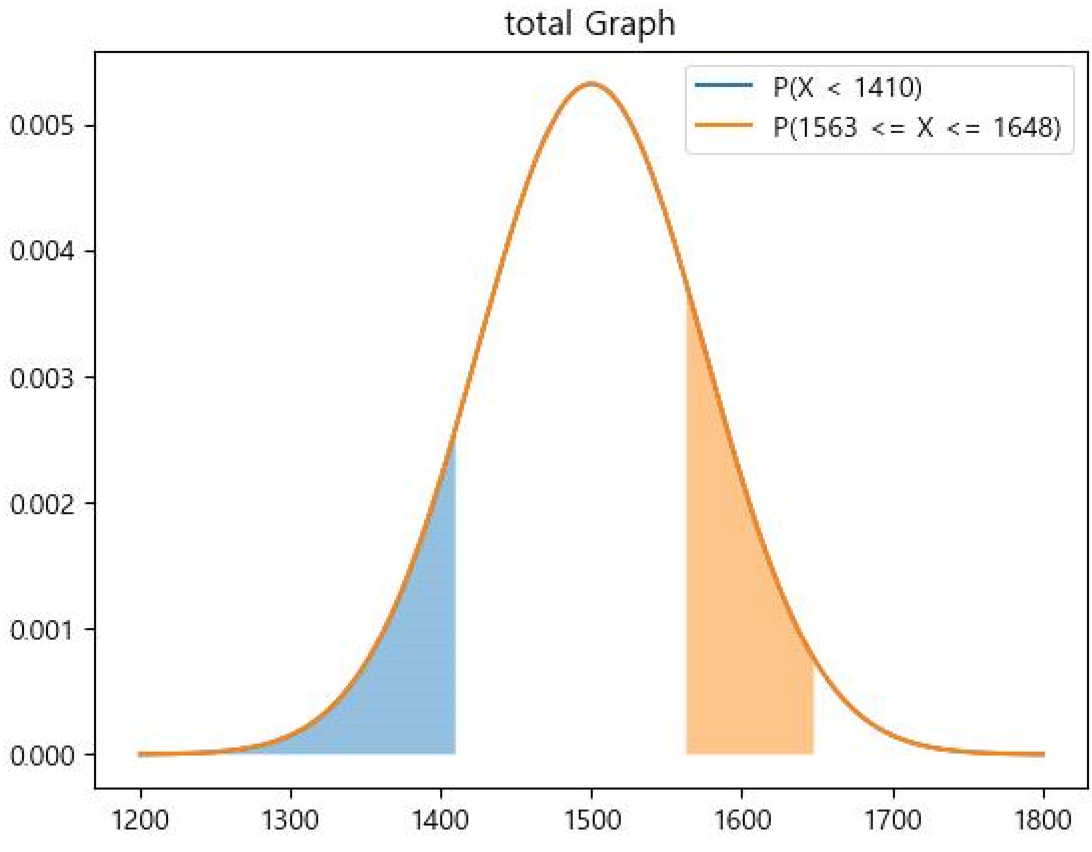
```
# 예제(5.21), p173
from scipy.stats import norm

mu     = 50   # 평균
sigma = 10   # 표준편차

# P(60 < X < 65)
prob = norm.cdf(65, mu, sigma) - norm.cdf(60, mu, sigma)

print(f"P(60 < X < 65) = {prob}")
```

P(60 < X < 65) = 0.09184805266259899

```
# 예제(5.23), p174
from scipy.stats import norm
import matplotlib.pyplot as plt
import numpy as np

plt.rc('font', family='Malgun Gothic')

mu = 1500
sigma = 75

# P(X < 1410)
prob = norm.cdf(1410, mu, sigma)

# P(1563 <= X <= 1648)
prob2 = norm.cdf(1648, mu, sigma) - norm.cdf(1563, mu, sigma)

# 백열전구의 수명이 1410시간 이하일 확률
print(f"P(X < 1410) = {prob:.4f}")
print(f"P(1536 < X < 1648) = {prob2:.4f}")

x = np.linspace(mu - 4 * sigma, mu + 4 * sigma, 1000)
y = norm.pdf(x, mu, sigma)

# P(X < 1410)
x_fill = np.linspace(mu - 4 * sigma, 1410, 100)
y_fill = norm.pdf(x_fill, mu, sigma)

plt.plot(x, y, label="P(X < 1410)")
plt.fill_between(x_fill, y_fill, alpha=0.5)
plt.title("P(X < 1410) / P(Z <= -1.2)")

x = np.linspace(mu - 4 * sigma, mu + 4 * sigma, 1000)
y = norm.pdf(x, mu, sigma)

# P(1563 <= X <= 1648)
x_fill = np.linspace(1563, 1648, 100)
y_fill = norm.pdf(x_fill, mu, sigma)

plt.plot(x, y, label="P(1563 <= X <= 1648)")
plt.fill_between(x_fill, y_fill, alpha=0.5)
plt.title("total Graph")
```

plt.legend()
# legend = 범례

plt.show()



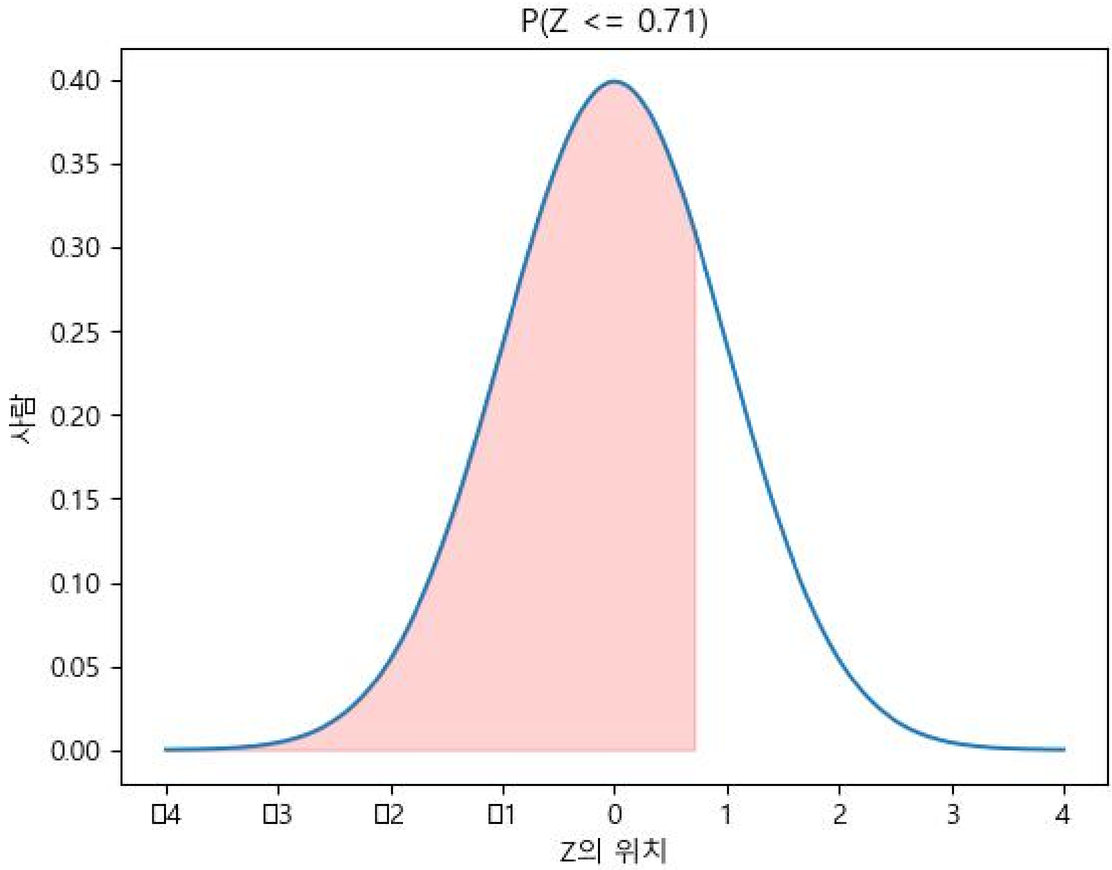
# 예제(5.25), p176
from scipy.stats import norm

mu = 6
sigma = 2.366
x = 5

# P(4.5 <= X <= 5.5)
prob = norm.cdf(x + 0.5, mu, sigma) - norm.cdf(x - 0.5, mu, sigma)
print(f"5대가 결점이 있을 확률 : {round((prob), 4)}")
5대가 결점이 있을 확률 : 0.1533

# 예제(5.27), p177
from scipy.stats import norm

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

plt.rc('font', family='Malgun Gothic')

x = 155
mu = (250 * 0.6)
sigma = 7.746

prob = norm.cdf(x + 0.5, mu, sigma)
print(f"예산삭감을 지지할 확률 : {round((prob), 4)}")

x = np.linspace(-4, 4, 1000)
y = norm.pdf(x)

fig, ax = plt.subplots()
ax.plot(x, y)
ax.fill_between(x, y, where=(x <= 0.71), color='red', alpha=0.2)
ax.set_xlabel('Z의 위치')
ax.set_ylabel('사람')
ax.set_title('P(Z <= 0.71)')

plt.show()
```




```
# 예제(5.29), p180
from scipy.stats import expon

mu = 3
x = 9
x1 = 6

prob = expon.cdf(x, scale=mu)
print(f"반응시간이 9초보다 작을 확률 : {round((prob), 4)}")

prob = expon.cdf(x1, scale=mu) - expon.cdf(x, scale=mu)
print(f"반응시간이 6초와 9초 사이일 확률 : {round((abs(prob)), 4)}")
# abs = 절대값 구하는 함수
```

```
반응시간이 9초보다 작을 확률 : 0.9502
반응시간이 6초와 9초 사이일 확률 : 0.0855
```

```
# 예제(5.13), p163
```

```
# 은행에서는 오전 11시 ~ 12시 사이에 평균 60명의 손님이 방문할 때, 조건을 만족하는 확률을 구하시오

# [ 조건 ]
# 1. 오전 11시에서 12시 사이에 어느 1분 동안 두 손님이 방문할 확률은?
# 2. 어느 1분 동안에 3명 이하의 손님이 도착할 확률을 계산하시오

from math import exp, factorial

mean = 60 / 60

prob_2 = (mean ** 2) * exp(-mean) / factorial(2)
print(f"1분 동안 두 손님이 방문할 확률 : {prob_2:.3f}")
prob_3_or_fewer = sum([(mean ** i) * exp(-mean) / factorial(i) for i in range(4)])
print(f"1분 동안 3명 이하의 손님이 도착할 확률 : {prob_3_or_fewer:.3f}")

1분 동안 두 손님이 방문할 확률 : 0.184
1분 동안 3명 이하의 손님이 도착할 확률 : 0.981

# 5장 연습문제
# 연습문제 01 , p181
from scipy.stats import uniform

a = 0
b = 10
x = 7
x1 = 2

prob = 1 - uniform.cdf(x, a, b-a)
print(f"승객이 7분 이상 기다릴 확률 : {round((prob), 1)}")
prob = uniform.cdf(x1, a, b-a) - uniform.cdf(x, a, b-a)
print(f"승객이 2분에서 7분 이상 기다릴 확률 : {round((abs(prob)), 1)}")

승객이 7분 이상 기다릴 확률 : 0.3
승객이 2분에서 7분 이상 기다릴 확률 : 0.5

# 연습문제 02 , p181
# 이항 분포 :  n번의 독립적인 시행에서 성공 확률이 p인 사건이 k번 이상 발생할 확률
from scipy.stats import binom

n = 20
```

```
p = 0.3
k = 10
sig = 4

prob = 1 - binom.cdf(k-1, n, p)
print(f"작업자 실수에 의해 발생할 확률 : {prob:.3f}%")
prob = binom.cdf(sig, n, p)
print(f"20번의 사고 중 4번 이하가 작업자 실수에 의해 발생할 확률 : {prob:.3f}%")
prob = binom.pmf(sig, n, p)
print(f"20번의 사고 중 5번만이 작업자 실수에 의해 발생할 확률 : {prob:.3f}%")
```

```
작업자 실수에 의해 발생할 확률 : 0.048%
20번의 사고 중 4번 이하가 작업자 실수에 의해 발생할 확률 : 0.238%
20번의 사고 중 5번만이 작업자 실수에 의해 발생할 확률 : 0.130%
```

```
# 연습문제 07 , p182
from scipy.stats import hypergeom

M = 25 # 전체 동물 수
n = 10 # 전체 동물 중 꼬리표가 있는 동물 수
N = 15 # 뽑은 동물 수
k = 5  # 뽑은 동물 중 꼬리표가 있는 동물 수

result = hypergeom.pmf(k, M, n, N)
print(f"생포한 동물 5마리에 꼬리표가 있을 확률 : {result:.4f}")
```

```
생포한 동물 5마리에 꼬리표가 있을 확률 : 0.2315
```

```
# 연습문제 09 , p182
from scipy.stats import hypergeom

M = 20 # 전체 기업수
n = 3  # 규정위반 기업수
N = 5  # 조사할 기업수
k = 0  # 규정위반 기업이 없을 확률
u = 2  # 규정위반 기업이 2개일 확률

result = hypergeom.pmf(k, M, n, N)
print(f"5개의 기업이 모두 규정위반기업이 아닐 확률 : {result:.3f}")
result = hypergeom.pmf(u, M, n, N)
print(f"5개의 기업 중 2개의 위반기업이 있을 확률 : {result:.3f}")
```

5개의 기업이 모두 규정위반기업이 아닐 확률 : 0.399
5개의 기업 중 2개의 위반기업이 있을 확률 : 0.132

```
# 연습문제 11 , p183
from scipy.stats import poisson

mu = 5 # 연간 평균 결함있는 차의 수
k = 3  # 기껏해야 3대일 확률
x = 1  # 결함차가 1대일 확률

result = poisson.cdf(k, mu)
print(f"연간 결함있는 차가 3대일 확률 : {result:.3f}")
result = 1 - poisson.cdf(x, mu)
print(f"결함을 나타내는 차가 연간 1대를 초과할 확률 : {result:.3f}")
```

연간 결함있는 차가 3대일 확률 : 0.265
결함을 나타내는 차가 연간 1대를 초과할 확률 : 0.960

```
# 연습문제 13 , p183
from scipy.stats import poisson

mu = 3 # 평균 3건의 교통사고 가정
k = 5  # 5건의 교통사고가 발생할 확률
u = 2  # 2건의 교통사고가 발생할 확률

result = poisson.pmf(k, mu)
print(f"정확히 4건의 교통사고가 일어날 확률 : {result:.3f}")
result = poisson.cdf(u, mu)
print(f"3건 미만의 교통사고가 일어날 확률 : {result:.3f}")
```

정확히 4건의 교통사고가 일어날 확률 : 0.101
3건 미만의 교통사고가 일어날 확률 : 0.423

```
# 연습문제 15 , p183
from scipy.stats import norm

z = [1.43, -0.89, -2.16, -0.65, -1.39, 1.96, -0.48, 1.74]

result = norm.cdf(z[0])
print(f"1 : Z=1.43의 왼쪽 면적 확률 : {result:.3f}")
result = 1 - norm.cdf(z[1])
```

```
print(f"2 : Z=-0.89의 오른쪽 면적 확률 : {result:.3f}")
result = norm.cdf(z[3]) - norm.cdf(z[2])
print(f"3 : Z=-2.16과 Z=-0.65 사이의 면적 확률 : {result:.3f}")
result = norm.cdf(z[4])
print(f"4 : Z=-1.39의 왼쪽 면적 확률 : {result:.3f}")
result = 1 - norm.cdf(z[5])
print(f"5 : Z=1.96의 오른쪽 면적 확률 : {result:.3f}")
result = norm.cdf(z[7]) - norm.cdf(z[6])
print(f"6 : Z=-0.48과 Z=1.74 사이의 면적 확률 : {result:.3f}")

1 : Z=1.43의 왼쪽 면적 확률 : 0.924
2 : Z=-0.89의 오른쪽 면적 확률 : 0.813
3 : Z=-2.16과 Z=-0.65 사이의 면적 확률 : 0.242
4 : Z=-1.39의 왼쪽 면적 확률 : 0.082
5 : Z=1.96의 오른쪽 면적 확률 : 0.025
6 : Z=-0.48과 Z=1.74 사이의 면적 확률 : 0.643

# 연습문제 17 , p184
from scipy.stats import norm

mu = 800    # 수명 평균값
sigma = 40 # 표준편차
x1 = 778    # 수명 최소값
x2 = 834    # 수명 최대값

result = norm.cdf(x2, mu, sigma) - norm.cdf(x1, mu, sigma)
print(f"전구의 수명이 778시간과 834시간 사이에 있을 확률 : {result:.3f}")

# 연습문제 21 , p184
from scipy.stats import binom

n = 100   # 로트에 포함된 부품의 수
p = 0.05 # 불량 부품의 비율
k = 2      # 불량 부품의 수
u = 10    # 불량 부품의 수(2)

prob = 1 - binom.cdf(k, n, p)
print(f"불량 부품이 2개를 초과할 확률 : {prob:.3f}")
prob = 1 - binom.cdf(u, n, p)
print(f"불량 부품이 10개를 초과할 확률 : {prob:.3f}")
```

불량 부품이 2개를 초과할 확률 : 0.882
불량 부품이 10개를 초과할 확률 : 0.011

```
# 연습문제 23 , p185
from scipy.stats import expon

mu = 3  # 평균 3초
x1 = 5  # 5초 이상
x2 = 10 # 10초 이상

prob1 = 1 - expon.cdf(x1, scale=mu)
prob2 = 1 - expon.cdf(x2, scale=mu)

print(f"반응시간이 {x1}초를 초과할 확률 : {prob1:.3f}")
print(f"반응시간이 {x2}초를 초과할 확률 : {prob2:.3f}")
```

반응시간이 5초를 초과할 확률 : 0.189
반응시간이 10초를 초과할 확률 : 0.036

```
# 06 표본평균
# 예제(6.2), p190
from scipy.stats import binom

n = 400              # 차가 지나간 수
p = 0.48             # 안전벨트를 하고 있을 확률
k1 = int(0.45 * n)  # 안전벨트를 하고 있을 비율이 45%일 확률
k2 = int(0.55 * n)  # 안전벨트를 하고 있을 비율이 55%일 확률

prob = binom.cdf(k2, n, p) - binom.cdf(k1 - 1, n, p)
print(f'안전벨트를 하고 있을 비율이 45%에서 55%일 확률 : {prob:.4f}')
```

안전벨트를 하고 있을 비율이 45%에서 55%일 확률 : 0.8925

```
# 예제(6.3). p192
from scipy.stats import norm

prob = 1 - norm.cdf(-0.20)
print(f'남자의 퍼센트와 여자의 퍼센트가 10%를 넘을 확률 : {prob:.4f}')
```

남자의 퍼센트와 여자의 퍼센트가 10%를 넘을 확률 : 0.5793

```
# 예제(6.4), p193
from scipy.stats import norm
import math

n1 = 200       # 혼인한 커플
p1 = 0.43      # 혼인한 커플의 집 소유 비율
n2 = 180       # 독신
p2 = 0.19      # 독신의 집 소유 비율
mu = p1 - p2  # 두 모집단의 차이

sd = math.sqrt(p1*(1-p1)/n1 + p2*(1-p2)/n2)  # 표준오차
print(f"표준오차 : {sd}")

sc = norm.ppf(0.1, mu, sd)  # 표준정규분포의 누적분포함수
print(f"퍼센트 차이가 몇 퍼센트보다 클 확률 : {round((sc), 3)}")
```

```
표준오차 : 0.04561249828720194
퍼센트 차이가 몇 퍼센트보다 클 확률 : 0.182
```

```
# 예제(6.7), p196
from scipy.stats import norm

mu = 650 # 평균 미결제 잔액
sd = 420 # 표준편차
n = 100  # 커플의 수
sem = 42 # 표준오차
cx = 700 # 700달러 이상 미결제 잔액을 가진 커플의 수

z = (cx - mu) / sem    # z스코어 구하기
p = 1 - norm.cdf(z)     # p벨류
print(f'700달러를 넘을 확률 : {p:.3f}')
```

```
700달러를 넘을 확률 : 0.117
```

```
# 예제(6.8), p197
from scipy.stats import norm

mu_a = 4.5  # 비료 A를 쓰는 분포표의 평균 수확
sd_a = 0.7  # 비료 A를 쓰는 분포표의 표준편차

mu_b = 4.3  # 비료 B를 쓰는 분포표의 평균 수확
```

```
sd_b = 0.4  # 비료 B를 쓰는 분포표의 표준편차

n_a = 45     # 비료 A를 쓰는 분포표의 표본 수
n_b = 50     # 비료 B를 쓰는 분포표의 표본 수

sem_a = sd_a / (n_a ** 0.5)
sem_b = sd_b / (n_b ** 0.5)
mu_diff = mu_a - mu_b
sd_diff = (sem_a ** 2 + sem_b ** 2) ** 0.5
z = (0 - mu_diff) / sd_diff
p = norm.cdf(z)

print(f'비료 A를 쓰는 분포표의 평균 수확이 B를 평균 분포표보다 낮을 확률 : {p:.4f}')
비료 A를 쓰는 분포표의 평균 수확이 B를 평균 분포표보다 낮을 확률 : 0.0460

# 예제(6.9), p198
import math
mu_public = 8.5  # 주립학교 결석한 날 평균 수
sd_public = 4.1  # 주립학교 결석한 날 표준편차

mu_private = 5.3 # 사립학교 결석한 날 평균 수
sd_private = 2.9 # 사립학교 결석한 날 표준편차

n_public = 200   # 주립학교 학생 수
n_private = 150  # 사립학교 학생 수

probability = 0.95 # 확률
sd = math.sqrt(sd_public**2/n_public + sd_private**2/n_private)
ex = (mu_public - mu_private) - 1.645 * sd
print(f'확률이 0.95일 때 결석한 날 평균수의 차이 : {round((ex), 1)}')
print(f"넘을 확률 : {probability}")

확률이 0.95일 때 결석한 날 평균수의 차이 : 2.6
넘을 확률 : 0.95

# 예제(6.10), p202
from scipy.stats import t

p = t.cdf(-1.415, df=10)
tc = t.ppf(1 - 0.05, df=25)
print(f"P(T<=-1.415) = 1-P(T<=1.415) : {round((p), 3)}")
```

```
print(f"자유도가 n = 25, α = 0.05인 t : {round((tc), 3)}")
P(T<=-1.415) = 1-P(T<=1.415) : 0.094
자유도가 n = 25, α = 0.05인 t : 1.708

# 예제(6.14), p205
from scipy.stats import chi2

sample = [1.9, 2.4, 3.0, 3.5, 4.2]
n = len(sample)
mu = 3
sigma = 1

x2 = sum(((x - mu) / sigma) ** 2 for x in sample)
p = 1 - chi2.cdf(x2, df=n-1)

print(f"x^2의 값 : {round((x2), 3)}")
print(f"P(X^2 > X(0)^2) : {round((p), 3)}")

x^2의 값 : 3.26
P(X^2 > X(0)^2) : 0.515

# 예제(6.15), p205
from scipy.stats import chi2

df = 9 # 자유도

x = chi2.ppf(1 - 0.05, df=df)
print(f"x의 값 : {round((x), 2)}")

x의 값 : 16.92

# 예제(6.17), p208
from scipy.stats import f

p = 1 - f.cdf(3.18, dfn=9, dfd=9)
print(f"두 집단의 표본분산비가 3.18 이상일 확률 : {round((p), 3)}")
두 집단의 표본분산비가 3.18 이상일 확률 : 0.05

# 예제(6.17), p208
from scipy.stats import f
```

```
alpha = 0.05
dfn, dfd = 2, 4
f_value = f.ppf(1 - alpha, dfn, dfd)
f_value2 = f.ppf(1 - 0.1, dfn, dfd)

print(f'F0.05,(2,4) : {round((f_value), 2)}')
print(f'F0.01,(2,4) : {round((f_value2), 2)}')
F0.05,(2,4) : 6.94
F0.01,(2,4) : 4.32

# 연습문제 01 p210
from scipy.stats import norm

n = 200     # 유권자 200명
p = 0.455   # A 후보자 지지율 45.5%
k = 110     # A 후보자 지지할 최소 유권자

mu = n * p
sigma = (n * p * (1 - p)) ** 0.5
z = (k - mu) / sigma

prob = 1 - norm.cdf(z)
print(f'적어도 {k}명이 A 후보자를 지지할 확률: {prob:.4f}')
적어도 110명이 A 후보자를 지지할 확률: 0.0035

# 연습문제 03 p210
from scipy.stats import norm

n1 = 470        # 미혼 남자의 수
n2 = 619        # 미혼 여자의 수
p1 = 245 / n1  # 성인 전용 극장의 찬성 미혼 남자의 비율
p2 = 223 / n2  # 성인 전용 극장의 찬성 미혼 여자의 비율

mu = p1 - p2
sigma = ((p1 * (1 - p1) / n1) + (p2 * (1 - p2) / n2)) ** 0.5
z = (0.20 - mu) / sigma

prob = norm.cdf(z)
print(f'지지율의 차가 20% 이하가 될 확률 : {prob:.4f}')

지지율의 차가 20% 이하가 될 확률 : 0.9027
```

```
# 연습문제 05 p210
from scipy.stats import norm

n1 = 100    # 월요일 생산 차량 수
n2 = 200    # 다른 요일 생산 차량 수
p1 = 0.08   # 월요일 생산 차량 결함률
p2 = 0.06   # 다른 요일 생산 차량 결함률

mu = p1 - p2
sigma = ((p1 * (1 - p1) / n1) + (p2 * (1 - p2) / n2)) ** 0.5
z = (0.03 - mu) / sigma

prob = 1 - norm.cdf(z)
print(f'월요일 생산 차가 다른 요일의 생산 차보다 결함이 3% 많을 확률 : {prob:.3f}')
```

```
월요일 생산 차가 다른 요일의 생산 차보다 결함이 3% 많을 확률 : 0.377
```

```
# 연습문제 11 p211
from scipy.stats import norm
import math

z1 = (75 - 75) / (10 / math.sqrt(25))
z2 = (79 - 75) / (10 / math.sqrt(25))
print(z1, z2)

prob = norm.cdf(z2) - norm.cdf(z1)
print(f'P({z1:.4f} <= Z <= {z2:.4f}) : {prob:.4f}')
```

```
0.0 2.0
P(0.0000 <= Z <= 2.0000): 0.4772
```

```
# 연습문제 15 p212
from scipy.stats import norm

sample_mean = 187.5    # 체납된 금액의 평균
std_dev = 54.5         # 체납된 금액의 표준편차
n = 50                 # 랜덤 50개 추출

sample_std_dev = std_dev / (n ** 0.5)
prob = 1 - norm.cdf(200, sample_mean, sample_std_dev)
```

```
print(f"체납된 평균 금액이 200달러 이상일 확률 : {prob:.4f}")
체납된 평균 금액이 200달러 이상일 확률 : 0.0524

# 연습문제 17 p212
from math import sqrt
from scipy.stats import norm

mean1 = 50
var1 = 9
mean2 = 40
var2 = 4

n1 = 5
n2 = 4
mean_diff = mean1 - mean2
std_dev_diff = sqrt(var1 / n1 + var2 / n2)
prob = norm.cdf(8.2, mean_diff, std_dev_diff)

print(f"P(X-Y < 8.2)의 확률: {prob:.4f}")
P(X-Y < 8.2)의 확률: 0.1410

# 연습문제 22 p213
from scipy.stats import t

df = [17, 6, 18, 17]
t_value = [-1.740, 3.143, 1.330, -2.567]

p_value = t.cdf(t_value[0], df[0])

print(f'P(T < {t_value[0]}) = {p_value:.2f}')
p_value = t.cdf(t_value[1], df[1]) - t.cdf(-t_value[1], df[1])

print(f'P(|T| < {t_value[1]}) = {p_value:.2f}')
p_value = t.cdf(-t_value[2], df[2]) - t.cdf(t_value[2], df[2])

print(f'P({-t_value[2]} < T < {t_value[2]}) = {round((abs(p_value)), 3)}')
p_value = 1 - t.cdf(t_value[3], df[3])

print(f'P(T > {t_value[3]}) = {p_value:.2f}')
```

```
P(T < -1.74) = 0.05
P(|T| < 3.143) = 0.98
P(-1.33 < T < 1.33) = 0.8
P(T > -2.567) = 0.99
```

```
# 연습문제 23 p213
from scipy.stats import t

df = [6, 15, 8, 11]
p = [0.95, 0.975, 0.01, 0.99]

k_value = t.ppf(p[0], df[0])

print(f'P(T<k) : {k_value:.3f}')
k_value = t.ppf(p[1], df[1])

print(f'P(-k < T < k) : {k_value:.3f}')
k_value = t.ppf(1 - p[2], df[2])

print(f'P(T > k) : {k_value:.3f}')
k_value = t.ppf(1 - p[3], df[3])

print(f'P(T > k) : {k_value:.3f}')
```

```
P(T<k) : 1.943
P(-k < T < k) : 2.131
P(T > k) : 2.896
P(T > k) : -2.718
```

```
# 연습문제 25 p214
from scipy.stats import chi2

df = [8, 17, 11, 23]
alpha = [0.05, 0.01, 0.025, 0.95]

x = chi2.ppf(1 - alpha[0], df[0])

print(f'x0.05^2 = {round((x), 2)}')
x = chi2.ppf(1 - alpha[1], df[1])

print(f'x0.01^2 = {round((x), 2)}')
```

```
x = chi2.ppf(1 - alpha[2], df[2])

print(f'x0.025^2 = {round((x), 2)}')
x = chi2.ppf(1 - alpha[3], df[3])

print(f'x0.95^2 = {round((x), 2)}')

x0.05^2 = 15.51
x0.01^2 = 33.41
x0.025^2 = 21.92
x0.95^2 = 13.09

# 연습문제 26 p214
from scipy.stats import chi2

df = [19, 5, 10, 18]
x = [30.14, 5, 3.24, 3.49]
c = [15.99, 17.53]

p_value = 1 - chi2.cdf(x[0], df[0])
print(f'P(X^2 > {x[0]}) = {round((p_value), 2)}')

p_value = 1 - chi2.cdf(x[1], df[1])
print(f'P(X^2 > {x[1]}) = {round((p_value), 3)}')

p = chi2.cdf(c[0], df[2]) - chi2.cdf(x[2], df[2])
print(f'P({x[2]} < X^2 < {c[0]}) : {round((p), 3)}')

p = chi2.cdf(c[1], df[3]) - chi2.cdf(x[3], df[3])
print(f'P({x[3]} < X^2 < {c[1]}) : {round((p), 3)}')

P(X^2 > 30.14) = 0.05
P(X^2 > 5) = 0.416
P(3.24 < X^2 < 15.99) : 0.875
P(3.49 < X^2 < 17.53) : 0.513

# 연습문제 27 p214
from scipy import stats
from scipy.stats import chi2

df = [4, 19, 25]
```

```
p = [0.99, 0.025, 0.045]
c = [37.652]

x = chi2.ppf(1 - p[0], df[0])
print(f"P(X^2 > Xa^2) : {round((x), 3)}")

x = chi2.ppf(1 - p[1], df[1])
print(f"P(X^2 > Xa^2) : {round((x), 3)}")

Xa2 = chi2.ppf(1 - p[2], df[2])
P = chi2.cdf(Xa2, df[2]) - chi2.cdf(c[0], df[2])

print(f'P({c} < X^2 < {Xa2:.3f}) = {P:.3f}')
X_0_005_squared = stats.chi2.ppf(1 - 0.005, df)

print("X0.005^2 =", round((X_0_005_squared[2]),3))

P(X^2 > Xa^2) : 0.297
P(X^2 > Xa^2) : 32.852
P([37.652] < X^2 < 38.123) = 0.005
X^2 for p=0.005: 14.860
X0.005^2 = 46.928

# 연습문제 29 p215
from scipy.stats import chi2

n = [31]
sigma2 = [5]
alpha = [0.10, 0.95]

dof = n[0] - 1
upper_bound = chi2.ppf(1 - alpha[0], dof)

C = upper_bound * sigma2[0] / dof
print(f"P(S^2 <= C)=0.90 , C : {round((C), 2)}")

upper_bound2 = chi2.ppf(1 - alpha[1], dof)

C = upper_bound2 * sigma2[0] / dof
print(f"P(S^2 <= C)=0.95 , C : {round((C), 2)}")
```

```
P(S^2 <= C)=0.90 , C : 6.71
P(S^2 <= C)=0.95 , C : 3.08

# 연습문제 32 p215
from scipy.stats import f

alpha = 0.25
dfn, dfd = 3, 5
f_value = f.ppf(1 - alpha, dfn, dfd)
f_value2 = f.ppf(1 - 0.05, dfn, dfd)
f_value3 = f.ppf(1 - 0.975, dfn, dfd)
f_value4 = f.ppf(1 - 0.95, dfn, dfd)

print(f'F0.25,(3,5) : {round((f_value), 2)}')
print(f'F0.95,(3,5) : {round((f_value2), 2)}')
print(f'F0.975,(3,5) : {round((f_value3), 3)}')
print(f'F0.95,(3,5) : {round((f_value4), 4)}')

F0.25,(3,5) : 1.88
F0.95,(3,5) : 5.41
F0.975,(3,5) : 0.067
F0.95,(3,5) : 0.1109

# 연습문제 34 p216
from scipy.stats import f

n1 = 8
n2 = 12
f_ratio = 4.5

dof1 = n1 - 1
dof2 = n2 - 1

p_value = 1 - f.cdf(f_ratio, dof1, dof2)
print(f"표본분산비가 4.5 이상일 확률 : {round((p_value), 2)}")
표본분산비가 4.5 이상일 확률 : 0.01

# 통계적 추정
# 예제(7.2), p220
from math import sqrt
from scipy.stats import norm
```

```
n = 225
x = 18
p_hat = x / n
z = norm.ppf(0.975)
se = sqrt(p_hat * (1 - p_hat) / n)
lower = p_hat - z * se
upper = p_hat + z * se

print(f"95% 신뢰구간 : {round((lower), 3)}, {round((upper), 3)}")
95% 신뢰구간 : 0.045, 0.115

# 예제(7.5), p223
import math
from scipy.stats import norm

n1 = 125
p1 = 0.84
n2 = 150
p2 = 0.72
alpha = 0.1

phat1 = p1
phat2 = p2
phat = (n1 * phat1 + n2 * phat2) / (n1 + n2)
z = norm.ppf(1 - alpha / 2)
se = math.sqrt(phat * (1 - phat) * (1 / n1 + 1 / n2))

ci_low = (phat1 - phat2) - z * se
ci_high = (phat1 - phat2) + z * se

diff = phat1 - phat2
margin_of_error = z * se

print(f'차이: {diff:.2f} ± {margin_of_error:.3f}')
차이: 0.12 ± 0.083

# 예제(7.7), p226
from math import sqrt

x = 5.32         # 무작위 표본의 평균 박동수
```

```
sigma = 2.49      # 모표준편차
n = 50            # 표본의 크기(인수)
v = 1.96          # 95% 신뢰구간의 z-값

lower = x - (v * (sigma / sqrt(n)))
upper = x + (v * (sigma / sqrt(n)))

print(f"95% 신뢰구간 추정 : {round((lower), 3)} < μ < {round((upper), 3)}")
95% 신뢰구간 추정 : 4.63 < μ < 6.01

# 예제(7.11), p229
from math import sqrt

n      = 10       # 표본의 크기
x1     = 27.2     # 표본평균
s      = 1.8      # 표본표준편차
t      = 2.262    # t분포표에서 검정유의수준 0.05

lower = x1 - (t * s / sqrt(n))
upper = x1 + (t * (s / sqrt(n)))

print(f"95% 신뢰구간 추정 : {round((lower), 1)} < μ < {round((upper), 1)}")

95% 신뢰구간 추정 : 25.9 < μ < 28.5

# 연습문제 5 / 예제(7.12), p230
from scipy.stats import *
import numpy as np
import math

data = [0.43, 0.52, 0.46, 0.49, 0.60, 0.56]
xbar = np.mean(data)
sd    = round((np.std(data, ddof=1)), 4)
n     = len(data)
t     = 2.015

lower = xbar - (t * (sd / math.sqrt(n)))
upper = xbar + (t * (sd / math.sqrt(n)))

print(f"90% 신뢰구간 추정 : {round((lower), 3)} < μ < {round((upper), 3)}")
```

90% 신뢰구간 추정 : 0.458 < µ < 0.562

```
# 예제(7.14), p232
from math import sqrt

simga       = 0.05 # 유의수준 5%
z1          = 1.96 # 95% 구간의 z값

month       = 30   # 연구 기간 30개월
teamA       = 3.5  # teamA의 표준편차
teamA_mean = 12.3 # teamA의 사고 평균
teamB       = 3.4  # teamB의 표준편차
teamB_mean = 7.6  # teamB의 사고 평균

lower = (teamA_mean - teamB_mean) - (z1 * sqrt(((teamA**2) / month) + ((teamB**2)/month)))
upper = (teamA_mean - teamB_mean) + (z1 * sqrt(((teamA**2) / month) + ((teamB**2)/month)))

print(f"95% 신뢰구간 추정 : {round((lower), 2)} < µ1 - µ2 < {round((upper), 2)}")
```

95% 신뢰구간 추정 : 2.95 < µ1 - µ2 < 6.45

```
# 예제(7.16), p235
from math import sqrt
import numpy as np

data1 = [40, 49, 38, 48, 40]
data2 = [51, 41, 53, 39, 40, 47]

x1 = int(np.mean(data1))
x2 = int(np.mean(data2))
s1 = 5.517  # s1
n1 = len(data1)       # 구획 A
n2 = len(data2)       # 구획 B
t = 2.821

lower = (x1-x2) - (t * (s1 * sqrt(1/n1 + 1/n2)))
upper = (x1-x2) + (t * (s1 * sqrt(1/n1 + 1/n2)))
```

```
print(f"98% 신뢰구간 : {lower:.3f} < μ1 - μ2 < {upper:.3f}")
# round는 반올림, 그러나 이 함수에서는 반올림이 일어나지 않아 :.xf로 표현( 결과는 같음 / 간단함의 여부 )

98% 신뢰구간 : -11.424 < μ1 - μ2 < 7.424

# 예제(7.17), p236
from math import sqrt
from scipy.stats import t

x = [19, 11, 14, 17, 23, 11, 15, 19, 11, 8]
y = [22, 18, 17, 19, 22, 12, 14, 11, 19,7]

d = [a - b for a, b in zip(x, y)]
n = len(d)
d_bar = sum(d) / n
s_d = sqrt(sum((x - d_bar)**2 for x in d) / (n - 1))
t_value = t.ppf(0.975, n - 1)
lower = d_bar - t_value * s_d / sqrt(n)
upper = d_bar + t_value * s_d / sqrt(n)

print(f"95% 신뢰구간 : {round((lower), 2)} < μ(D) < {round((upper), 2)}")
95% 신뢰구간 : -4.55 < μ(D) < 1.95

# 예제(7.19), p240
from scipy.stats import chi2

weight = [46.4, 46.1, 45.8, 47.0, 46.1, 45.9, 45.8, 46.9, 45.2, 46.0]
n = len(weight)
x_bar = sum(weight) / n
s2 = sum((x - x_bar)**2 for x in weight) / (n - 1)
lower = (n - 1) * s2 / chi2.ppf(0.975, n - 1)
upper = (n - 1) * s2 / chi2.ppf(0.025, n - 1)

print(f"95% 신뢰구간 : ({round((lower), 3)} < σ^2 < {round((upper), 3)})")

95% 신뢰구간 : (0.135 < σ^2 < 0.954)

# 7장 연습문제
# 연습문제 1 p241 , node (3)
from math import sqrt
```

```
from scipy.stats import norm

n = 40           # 샘플 사이즈
x = 34           # 성공한 샘플의 수
p_hat = x / n    # 성공률
z = norm.ppf(0.975)
se = sqrt(p_hat * (1 - p_hat) / n)
lower = p_hat - z * se
upper = p_hat + z * se

print(f"성공률 p의 95% 신뢰구간 : ({round((lower), 3)} < p < {round((upper), 3)})")
성공률 p의 95% 신뢰구간 : (0.739 < p < 0.961)

# 연습문제 7 p242
from math import sqrt
from scipy.stats import norm

a = [928, 72]  # 품목 A
b = [772, 28]  # 품목 B

n1 = sum(a)
n2 = sum(b)

p1 = a[1] / n1
p2 = b[1] / n2

p_hat = p1 - p2
z = norm.ppf(0.975)

se = sqrt((p1 * (1 - p1) / n1) + (p2 * (1 - p2) / n2))
lower = -(p_hat - z * se)
upper = p_hat + z * se

print(f"신뢰 상한과 하한 : ({lower:.4f}, {upper:.4f})\n : ({lower:.4f} < p1 - p2 < {upper:.4f})")
신뢰 상한과 하한 : (-0.0165, 0.0575)
 : (-0.0165 < p1 - p2 < 0.0575)

# 연습문제 9 p242
from scipy.stats import *
```

```
n = 100
sample_mean = 1.022
sigma = 0.021

sem = sigma / (n ** 0.5)
ci = norm.interval(0.95, loc=sample_mean, scale=sem)

print(f"95% 신뢰구간 : ({round((ci[0]), 3)} < μ < {round((ci[1]), 3)}) [단위 : mm]")
95% 신뢰구간 : (1.018 < μ < 1.026) [단위 : mm]

# 연습문제 12 p243
import numpy as np
from scipy import stats

x = [4.6, 3.6, 4.0, 6.1, 8.8, 5.3, 1.2, 5.6, 3.3, 1.6]

x_mean = np.mean(x)
s = np.std(x, ddof=1)
sem = s / np.sqrt(len(x))
t_value = stats.t.ppf((1 + 0.90) / 2, len(x) - 1)
margin_of_error = t_value * sem

ci_lower = x_mean - margin_of_error
ci_upper = x_mean + margin_of_error

print(f"90%에 대한 모평균 뮤(μ)의 신뢰구간 : ({ci_lower:.2f} < μ < {ci_upper:.2f})")
90%에 대한 모평균 뮤(μ)의 신뢰구간 : (3.12 < μ < 5.70)

# 연습문제 18 p244
import numpy as np
from scipy import stats

man = [52, 60, 55, 46, 33, 75, 58, 45, 57, 88]
girl = [62, 58, 65, 56, 53, 45, 56, 65, 77, 47]

man_mean = np.mean(man)
girl_mean = np.mean(girl)

man_std = np.std(man, ddof=1)
girl_std = np.std(girl, ddof=1)
```

```
pooled_std = np.sqrt(((len(man) - 1) * man_std **2 + (len(girl) -1) * girl_std **2) / (len(man) + len(girl) -2))

sem = pooled_std * np.sqrt(1 / len(man) +1 / len(girl))
t_value = stats.t.ppf((1 +0.90) /2 , len(man) + len(girl) -2)

margin_of_error = t_value * sem

ci_lower = (man_mean - girl_mean) - margin_of_error
ci_upper = (man_mean - girl_mean) + margin_of_error

print(f"합동표준편차 : {pooled_std:.2f}")
print(f"평균생존연령의 차이에 대한 90% 신뢰구간 : ({ci_lower:.2f} < μ < {ci_upper:.2f})")
합동표준편차 : 12.83
평균생존연령의 차이에 대한 90% 신뢰구간 : (-11.45 < μ < 8.45)

# 연습문제 20 p245
import numpy as np
from scipy import stats

data = [[1, 34400, 36700], [2, 45500, 46800], [3, 36700, 37700], [4, 32000, 31100], [5, 48400, 47800], [6, 32800, 36400], [7, 38100, 38900], [8, 30100, 31500]]

differences = [row[1] - row[2] for row in data]
d_mean = np.mean(differences)
s = np.std(differences, ddof=1)
sem = s / np.sqrt(len(differences))
t_value = stats.t.ppf((1 + 0.99) / 2, len(differences) - 1)
margin_of_error = t_value * sem

ci_lower = d_mean - margin_of_error
ci_upper = d_mean + margin_of_error

print(f"μ(d) 의 99% 신뢰구간 : ({round((ci_lower), 1)}km < μ(d) < {round((ci_upper), 1)}km)")

μ(d) 의 99% 신뢰구간 : (-2912.1km < μ(d) < 687.1km)

# 연습문제 23 p246
import numpy as np
from scipy import stats
```

```
SBP = [35, 48, 65, 33, 61, 54, 49, 37, 58, 65]
conventional = [33, 40, 55, 41, 62, 54, 40, 35, 59, 56]

SBP_mean = np.mean(SBP)
conventional_mean = np.mean(conventional)

SBP_std = np.std(SBP, ddof=1)
conventional_std = np.std(conventional, ddof=1)

pooled_std = np.sqrt(((len(SBP) - 1) * SBP_std **2 + (len(conventional) -1) * conventional_std **2) / (len(SBP) + len(conventional) -2))

sem = pooled_std * np.sqrt(1 / len(SBP) +1 / len(conventional))

t_value = stats.t.ppf((1 +0.98) /2 , len(SBP) + len(conventional) -2)

margin_of_error = t_value * sem

ci_lower = (SBP_mean - conventional_mean) - margin_of_error
ci_upper = (SBP_mean - conventional_mean) + margin_of_error

print(f"450g당 비타민 양의 평균차이에 대한 98% 신뢰구간 : ({round((ci_lower), 1)} < μ < {round((ci_upper), 1)})")

450g당 비타민 양의 평균차이에 대한 98% 신뢰구간 : (-10.1 < μ < 16.1)

# 연습문제 25 p246
import numpy as np
from scipy import stats

life = [1.9, 2.4, 3.0, 3.5, 4.2]

x_mean = np.mean(life)
s = np.std(life, ddof=1)

chi2_lower = stats.chi2.ppf((1 - 0.95) / 2, len(life) - 1)
chi2_upper = stats.chi2.ppf((1 + 0.95) / 2, len(life) - 1)

ci_lower = (len(life) - 1) * s ** 2 / chi2_upper
ci_upper = (len(life) - 1) * s ** 2 / chi2_lower
```

```
print(f"α^2에 대한 95% 신뢰구간 : ({round((ci_lower), 2)} < α^2 < {round((ci_upper), 2)})")

α^2에 대한 95% 신뢰구간 : (0.29 < α^2 < 6.73)

# 연습문제 28 p247
import numpy as np
from scipy import stats

weight = [46.4, 46.1, 45.8, 47.0, 46.1, 45.9, 45.8, 46.9, 45.2, 46.0]

x_mean = np.mean(weight)
s = np.std(weight, ddof=1)

chi2_lower = stats.chi2.ppf((1 - 0.95) / 2, len(weight) - 1)
chi2_upper = stats.chi2.ppf((1 + 0.95) / 2, len(weight) - 1)

ci_lower = (len(weight) - 1) * s ** 2 / chi2_upper
ci_upper = (len(weight) - 1) * s ** 2 / chi2_lower

print(f"무게 분산에 대한 95% 신뢰구간 : ({round((ci_lower), 3)} < α^2 < {round((ci_upper), 3)})")
무게 분산에 대한 95% 신뢰구간 : (0.135 < α^2 < 0.954)

# 08 검정
# 예제 8.1
from statsmodels.stats.proportion import proportions_ztest

# 귀무 가설: 연합 일원들 중 쟁의를 지지하는 비율은 75% 이하이다.
# 대립 가설: 연합 일원들 중 쟁의를 지지하는 비율은 75% 초과이다.

n = 125 # 샘플 크기
x = 87 # 샘플 중 쟁의를 지지하는 인원 수
p = 0.75 # 귀무 가설 하에서의 비율

# z-검정
z_stat, p_value = proportions_ztest(x, n, p, alternative='smaller')

alpha = 0.10 # 유의 수준
if p_value < alpha:
```

```
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

p-value는 0.0947로, 유의 수준 0.1보다 작다.
따라서 귀무 가설을 기각한다.

```
# 예제(8.4), p256
from statsmodels.stats.proportion import proportions_ztest

# 귀무 가설: 두 조립 절차 간의 결점 비율 차이는 없다.
# 대립 가설: 두 조립 절차 간의 결점 비율 차이가 있다.

n1 = 350 # 첫 번째 조립 절차의 샘플 크기
x1 = 28 # 첫 번째 조립 절차에서 결점이 있는 차량 수
n2 = 500 # 두 번째 조립 절차의 샘플 크기
x2 = 32 # 두 번째 조립 절차에서 결점이 있는 차량 수

# z-검정
z_stat, p_value = proportions_ztest([x1, x2], [n1, n2])

alpha = 0.10 # 유의 수준

if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

p-value는 0.3701로, 유의 수준 0.1보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

```
# 예제(8.6), p258
from scipy import stats
import numpy as np

# 귀무 가설: 판매원의 평균 판매고는 1000달러 이하이다.
# 대립 가설: 판매원의 평균 판매고는 1000달러 초과이다.
```

```
sales = np.array([1280, 1250, 990, 1100, 880, 1300, 1100, 950, 1050]) # 판매고 데이터
mu = 1000    # 귀무 가설 하에서의 평균
sigma = 100 # 모표준편차

# z-검정
z_stat = (np.mean(sales) - mu) / (sigma / np.sqrt(len(sales)))
p_value = stats.norm.sf(z_stat)

alpha = 0.01 # 유의 수준 / 1%유의 수준
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각하지 않는다")
```

p-value는 0.0013로, 유의 수준 0.01보다 작다.
따라서 귀무 가설을 기각하고 대립 가설을 채택한다

```
# 예제(8.7), p260
from scipy import stats
import numpy as np

# 귀무 가설: 에어컨의 새 브랜드가 전기를 하루에 6.5 킬로와트 이하로 사용한다.
# 대립 가설: 에어컨의 새 브랜드가 전기를 하루에 6.5 킬로와트 초과로 사용한다.

n      = 15  # 샘플 크기
x_bar = 7.0 # 표본 평균
s      = 1.4 # 표본 표준편차
mu     = 6.5 # 귀무 가설 하에서의 평균

# t-검정
t_stat = (x_bar - mu) / (s / np.sqrt(n))
p_value = stats.t.sf(t_stat, df=n-1)

alpha = 0.05 # 유의 수준
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
```

```
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다")
```

p-value는 0.0941로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없다

```
# 예제(8.9), p262
from scipy import stats
import numpy as np

# 귀무 가설: 유럽의 성인 몸무게와 미국 성인의 몸무게는 같다.
# 대립 가설: 유럽의 성인 몸무게가 미국 성인의 몸무게보다 작다.

n1     = 15  # 유럽 성인 샘플 크기
x1     = 154 # 유럽 성인 표본 평균
sigma1 = 10  # 유럽 성인 모표준편차

n2     = 18  # 미국 성인 샘플 크기
x2     = 162 # 미국 성인 표본 평균
sigma2 = 13  # 미국 성인 모표준편차

# z-검정
z_stat = (x1 - x2) / np.sqrt(sigma1**2 / n1 + sigma2**2 / n2)
p_value = stats.norm.cdf(z_stat)

alpha = 0.05 # 유의 수준
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다.")
```

p-value는 0.0229로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각하고 대립 가설을 채택한다.

```
# 예제(8.12), p266
from scipy import stats

# 귀무 가설: 남성과 여성 공무원들은 경찰서에 승진을 위해 똑같은 시간을 기다린다.
# 대립 가설: 남성과 여성 공무원들은 경찰서에 승진을 위해 다른 시간을 기다린다.
```

```
male_waiting_times = [8, 7, 10, 5, 7] # 남성 공무원들의 기다린 시간
female_waiting_times = [9, 5, 12, 8]  # 여성 공무원들의 기다린 시간

# t-검정
t_stat, p_value = stats.ttest_ind(male_waiting_times, female_waiting_times, equal_var=False)

alpha = 0.05 # 유의 수준
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다")
```

p-value는 0.5369로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없다

```
# 예제(8.14), p269
from scipy import stats

# 귀무 가설: 새로운 정신안정제의 효과와 가짜약에 의한 효과는 같다.
# 대립 가설: 새로운 정신안정제의 효과가 가짜약에 의한 효과보다 좋다.

d = [-3, -7, -3, -2, 1, -1, 1, 8, -8, 1] # 차이

# t-검정
t_stat, p_value = stats.ttest_1samp(d, popmean=0)

alpha = 0.05 # 유의 수준
if p_value / 2 < alpha:
    print(f"p-value는 {p_value / 2:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value / 2:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다.")
```

p-value는 0.1948로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없다.

```
# 예제(8.16), p271
from scipy import stats
import numpy as np

# 귀무 가설: 햄스터의 몸무게의 분산은 2.25 이하이다.
# 대립 가설: 햄스터의 몸무게의 분산은 2.25 초과이다.

weights = np.array([12, 8, 7, 12, 14, 13]) # 햄스터들의 몸무게
n = len(weights)                            # 샘플 크기
s2 = np.var(weights, ddof=1)                # 표본 분산
sigma2 = 2.25                               # 귀무 가설 하에서의 분산

# 카이제곱 검정
chi2_stat = (n - 1) * s2 / sigma2
p_value = stats.chi2.sf(chi2_stat, df=n-1)

alpha = 0.05 # 유의 수준
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다.")
```

p-value는 0.0032로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각하고 대립 가설을 채택한다.

```
# 8장 연습문제
# 연습문제 3 p273
import numpy as np
from statsmodels.stats.proportion import proportions_ztest

# 귀무 가설: 시장의 지지율은 50% 이하이다.
# 대립 가설: 시장의 지지율은 50% 초과이다.

n = 300 # 샘플 크기
x = 158 # 지지하는 유권자 수
p = 0.5 # 귀무 가설 하에서의 비율

# z-검정
z_stat, p_value = proportions_ztest(x, n, p, alternative='larger')
```

```
alpha = 0.05 # 유의 수준
print("1. 재선의 가능성에 대한 검정")
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다.")
# 검정 결과에 따른 결론 : 시장은 재선 가능성이 있다
```

1. 재선의 가능성에 대한 검정
p-value는 0.1775로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없다.

```
# 연습문제 3
from scipy import stats
from statsmodels.stats.proportion import proportions_ztest

# 귀무 가설: 시장의 지지율은 4년전과 같다.
# 대립 가설: 시장의 지지율은 4년전보다 낮다.

n = 300  # 샘플 크기
x = 158  # 지지하는 유권자 수
p = 0.56 # 귀무 가설 하에서의 비율

# z-검정
z_stat, p_value = proportions_ztest(x, n, p, alternative='smaller')

alpha = 0.05 # 유의 수준
print("2. 4년 전에 비하여 지지율이 내려갔다고 할 수 있는가? ")
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다.")
# 검정 결과에 따른 결론 : 지지율이 4년전에 비해 내려갔다고 할 수 없다.
```

2. 4년 전에 비하여 지지율이 내려갔다고 할 수 있는가?
p-value는 0.1238로, 유의 수준 0.05보다 크거나 같다.

따라서 귀무 가설을 기각할 수 없다.

```
# 연습문제 3 신뢰구간 95% 구간 추정
import math

n = 300    # 샘플 크기
x = 158    # 지지하는 유권자 수
p = x / n # 지지율
se = math.sqrt(p * (1 - p) / n)
me = 1.96 * se

lower = p - me
upper = p + me

print(f"재선에 대한 95% 신뢰구간 : ({lower:.2f} < X < {upper:.2f}).")
재선에 대한 95% 신뢰구간 : (0.47 < X < 0.58).


# 연습문제 1 p273, node (3) - 통합 코드 / 재선의 가능성을 구하는 코드
# 연습문제 1 p273, node (3) - A
import numpy as np
from statsmodels.stats.proportion import proportions_ztest

# 귀무 가설: 시장의 지지율은 50% 이하이다.
# 대립 가설: 시장의 지지율은 50% 초과이다.

n = 300   # 샘플 크기
x = 158   # 지지하는 유권자 수
p = 0.5   # 귀무 가설 하에서의 비율
P = 0.56 # 비율2

# z-검정
z_stat, p_value = proportions_ztest(x, n, p, alternative='larger')

alpha = 0.05 # 유의 수준
print("1. 재선의 가능성에 대한 검정")
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무
```

가설을 기각할 수 없다.\n")
검정 결과에 따른 결론 : 시장은 재선 가능성이 있다
z_stat, p_value = proportions_ztest(x, n, P, alternative='smaller')

alpha = 0.05 # 유의 수준
print("2. 4년 전에 비하여 지지율이 내려갔다고 할 수 있는가?")
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다. \n따라서 귀무 가설을 기각할 수 없다.")

검정 결과에 따른 결론 : 지지율이 4년전에 비해 내려갔다고 할 수 없다.
1. 재선의 가능성에 대한 검정
p-value는 0.1775로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없다.

2. 4년 전에 비하여 지지율이 내려갔다고 할 수 있는가?
p-value는 0.1238로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없다.

# 연습문제 4 p273
from statsmodels.stats.proportion import proportions_ztest

귀무 가설: 새로운 약의 효과는 기존의 약의 효과와 같다.
대립 가설: 새로운 약의 효과가 기존의 약의 효과보다 높다.

n = 100 # 샘플 크기
x = 70 # 효과를 보인 사람 수
p = 0.6 # 귀무 가설 하에서의 비율

z-검정
z_stat, p_value = proportions_ztest(x, n, p, alternative='larger')

alpha = 0.05 # 유의 수준
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같습니다. \n따라서

귀무 가설을 기각할 수 없다.")

p-value는 0.0145로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각하고 대립 가설을 채택한다.

```
# 연습문제 9 p274
from scipy import stats

# 도시 거주자의 지지율
city = [1] * 63 + [0] * 37
# 교외 거주자의 지지율
suburb = [1] * 59 + [0] * 66

# 두 집단 간의 비율 차이 검정
oddsratio, p_value = stats.fisher_exact([[63, 37], [59, 66]])

print(f'P-값: {p_value:.4f}')
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다. \n따라서 귀무 가설을 기각하고 대립 가설을 채택한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같습니다. \n따라서 귀무 가설을 기각할 수 없다.")
```

P-값: 0.0221
p-value는 0.0221로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각하고 대립 가설을 채택한다.

```
# 연습문제 10 p274
from scipy import stats

# 도시지역의 성인여성 발병률
city = [1] * 20 + [0] * 180
# 농촌지역의 성인여성 발병률
rural = [1] * 10 + [0] * 140

# 두 집단 간의 비율 차이 검정
oddsratio, p_value = stats.fisher_exact([[20, 180], [10, 140]])
alpha = 0.05
if p_value < alpha:
```

```
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
p-value는 0.3360로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

# 연습문제 7 p274
import numpy as np
from scipy.stats import norm

# 귀무가설: 어떤 도시의 가정 중 1/5가 기름난방을 한다
# 대립가설: 어떤 도시의 가정 중 1/5가 기름난방을 하지 않는다

n = 1000 # 무작위로 추출한 가정의 수
x = 136 # 기름난방을 하는 가정의 수
p = 0.2 # 귀무가설에 따른 기름난방을 하는 가정의 비율

# 정규 근사를 사용한 검정
z = (x - n * p) / np.sqrt(n * p * (1 - p))
p_value = 2 * norm.sf(np.abs(z))
print(f'p-value: {p_value}')

alpha = 0.02
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

p-value: 4.2003939760219985e-07
p-value는 0.0000로, 유의 수준 0.02보다 작다.
따라서 귀무 가설을 기각한다.

# 연습문제 13 p275
from scipy.stats import chi2_contingency

# 귀무가설: 시와 인접지역의 유권자 지지율은 같다
# 대립가설: 시와 인접지역의 유권자 지지율은 다르다
```

```
observed = [[120, 80], [240, 260]] # 시와 인접지역의 찬성/반대 표

# 카이제곱 검정
chi2, p_value, dof, expected = chi2_contingency(observed)
print(f'p-value: {p_value:.4f}')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

p-value: 0.0053
p-value는 0.0053로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

# 연습문제 14 p275
from scipy.stats import norm

# 귀무가설: 한국사람의 평균수명은 70년 이하이다
# 대립가설: 한국사람의 평균수명은 70년 초과이다

n     = 100  # 표본 크기
x_bar = 71.8 # 표본 평균
mu    = 70   # 귀무가설에 따른 모평균
sigma = 8.9  # 모표준편차

# z-검정
z = (x_bar - mu) / (sigma / n**0.5)
p_value = norm.sf(z)
print(f'p-value: {p_value:.4f}')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무
```

가설을 기각할 수 없음")

p-value: 0.0216
p-value는 0.0216로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 17 p276
from scipy.stats import t
import numpy as np

# 귀무가설: 신제품의 평균인장강도는 8kg이다
# 대립가설: 신제품의 평균인장강도는 8kg이 아니다

n     = 50  # 표본 크기
x_bar = 7.8 # 표본 평균
mu    = 8   # 귀무가설에 따른 모평균
s     = 0.5 # 표본 표준편차

# t-검정
t_stat = (x_bar - mu) / (s / n**0.5)
p_value = t.sf(np.abs(t_stat), n-1) * 2
print(f'p-value: {p_value:.4f}')

alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

p-value: 0.0068
p-value는 0.0068로, 유의 수준 0.01보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 18 p276
from scipy.stats import t
# type = t검정

# 귀무가설: 에디슨 진공청소기의 연평균 전력사용량은 46kwh 이상이다
```

```
# 대립가설: 에디슨 진공청소기의 연평균 전력사용량은 46kwh 미만이다

n      = 12   # 표본 크기
x_bar = 42   # 표본 평균
mu     = 46   # 귀무가설에 따른 모평균
s      = 11.9 # 표본 표준편차

# t-검정
t_stat = (x_bar - mu) / (s / n**0.5)
p_value = t.cdf(t_stat, n-1)

# -일 경우 abs로 절대값으로 취하면 된다. ( 예제 : 8e8 참조, t.cdf(np.abs(tstat)) )

print(f'p-value: {p_value:.4f}')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
p-value: 0.1344
p-value는 0.1344로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

```
# 연습문제 19 p276
from scipy.stats import t

# 귀무가설: μ가 10이다        | μ = 10
# 대립가설: μ가 10보다 적다 | μ > 10

n      = 16 # 표본 크기
x_bar = 11 # 표본 평균
mu     = 10 # 귀무가설에 따른 모평균
s      = 3  # 표본 표준편차

# 기각역 계산
alpha = 0.2 # 유의수준
t_crit = t.ppf(alpha, n-1)
```

```
# 유의수준 기반 기각역 계산
rejection_region = (t_crit)
print(f'기각역: {round((rejection_region), 4)}')

# t-검정
t_stat = (x_bar - mu) / (s / n**0.5)
print(f'검정통계량: {round((t_stat), 2)}\n')

if t_stat < t_crit:
    print(f"t_stat는 {round((t_stat), 2)}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"t_stat는 {round((t_stat), 2)}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
기각역: -0.8662
검정통계량: 1.33

t_stat는 1.33로, 유의 수준 0.2보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

```
# 연습문제 20 p276
from scipy import stats

# 귀무가설: 약물복용 청소년의 평균 IQ는 이 지역의 평균 IQ인 110과 같다
# 대립가설: 약물복용 청소년의 평균 IQ는 이 지역의 평균 IQ인 110보다 낮다

iq = [125, 105, 117, 109, 118, 104, 98, 111, 107, 108, 135, 94, 90, 100, 99] # 약물복용 청소년의 IQ
mu = 110 # 귀무가설에 따른 모평균

# t-검정
t_stat, p_value = stats.ttest_1samp(iq, mu)
print(f'p-value: {p_value/2:.4f}')
print(f"t-value {t_stat}")
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무
```

가설을 기각할 수 없음")

p-value: 0.2637
t-value -0.648074069840786
p-value는 0.5274로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

```
# 연습문제 21 p277
import math
from scipy import stats

# 귀무가설: 전구의 평균 수명은 1000시간이다
# 대립가설: 전구의 평균 수명은 1000시간이 아니다

n     = 20   # 표본 크기
x_bar = 1216 # 표본 평균
mu    = 1000 # 귀무가설에 따른 모평균
s     = 495  # 표본 표준편차

# 이 문제를 풀기 위해서는 반드시 정규분포여야 한다.

# t-검정
t_statistic = (x_bar - mu) / (s / math.sqrt(n))

# 자유도 계산
df = n - 1

# p-value 계산 (양측 검정)
p_value = stats.t.sf(abs(t_statistic), df) * 2

# 결과 출력
print(f't-statistic: {t_statistic}')
print(f'p-value: {p_value}\n')

# 유의 수준 설정
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무
```

```
가설을 기각할 수 없음")

t-statistic: 1.9514775076361803
p-value: 0.06590109707725218
p-value는 0.0659로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

# 연습문제 22 p277
import numpy as np
from scipy import stats

# 딕셔너리로 데이터 입력 / np.array 혹은 리스트도 가능
sampleA = {'n1': 12, 'x1': 35, 's1': 4.2}
sampleB = {'n2': 16, 'x2': 43, 's2': 3.7}

# 통합분산 계산
pooled_var = ((sampleA['n1'] - 1) * sampleA['s1']**2 + (sampleB['n2'] - 1) * sampleB['s2']**2) / (sampleA['n1'] + sampleB['n2'] - 2)
print(f'통합분산: {pooled_var:.4f}')

# t-검정
t_stat, p_value = stats.ttest_ind_from_stats(sampleA['x1'], sampleA['s1'], sampleA['n1'], sampleB['x2'], sampleB['s2'], sampleB['n2'], equal_var=True)
print(f'p-value: {p_value}\n')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다. \n")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음 \n")

# 두 모평균의 차이에 대한 95% 신뢰구간 계산
diff_mean = sampleA['x1'] - sampleB['x2']
se = np.sqrt(pooled_var * (1/sampleA['n1'] + 1/sampleB['n2']))
margin_of_error = stats.t.ppf(0.975, df=sampleA['n1']+sampleB['n2']-2) * se
ci = [diff_mean - margin_of_error, diff_mean + margin_of_error]
print(f"t-value {t_stat}")
print(f'두 모평균의 차이에 대한 95% 신뢰구간 : {diff_mean:.4f} ± {margin_of_error:.4f}')
```

통합분산: 15.3612
p-value: 1.3550195956491572e-05

p-value는 0.0000로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

t-value -5.345024082400142
두 모평균의 차이에 대한 95% 신뢰구간 : -8.0000 ± 3.0766

```
# 연습문제 23 p277
from scipy import stats
import numpy as np

# 귀무가설: 4년제 대학생과 2년제 대학생의 주당 과제물 수행 시간의 평균은 같다
# 대립가설: 4년제 대학생의 주당 과제물 수행 시간의 평균이 2년제 대학생보다 높다

n1 = 47              # 4년제 대학생 표본 크기
x1 = 18.6            # 4년제 대학생 표본 평균
s1 = np.sqrt(22.4) # 4년제 대학생 표본 표준편차

n2 = 36              # 2년제 대학생 표본 크기
x2 = 14.7            # 2년제 대학생 표본 평균
s2 = np.sqrt(20.9) # 2년제 대학생 표본 표준편차

# t-검정
t_stat, p_value = stats.ttest_ind_from_stats(x1, s1, n1, x2, s2, n2, equal_var=False)
print(f'p-value: {p_value/2:.4f}')

alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

p-value: 0.0001
p-value는 0.0003로, 유의 수준 0.01보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 26 p278
from scipy import stats

# 귀무가설: 두 암기법의 평균은 같다
# 대립가설: 두 암기법의 평균은 다르다

learnA = [5, 2, 4, 7, 4, 4, 8, 3, 7, 6]  # 암기법 A
learnB = [6, 5, 4, 9, 4, 6, 8, 5, 6, 7]  # 암기법 B

# t-검정
t_stat, p_value = stats.ttest_ind(learnA, learnB)
print(f'p-value: {p_value:.4f}')

alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
p-value: 0.2289
p-value는 0.2289로, 유의 수준 0.01보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

```
# 연습문제 27 p278

# 두 강재의 마모량을 비교하기 위한 실험을 할 때, 강재A와 강재B에 대하여 각각 12회 및 10회씩 실험한 결과 강재A의 마모량은 평균이 85, 표본표준편차가 4이고 강재B에서는 81과 5로 나타났다. 강재A의 마모량이 강재B의 마모량보다 2이상 심한지를 유의수준 0.05로 검정하라.

# [ 조건 ] : 단 분산이 같은 정규분포를 근사적으로 따른다.

from scipy import stats

# 귀무가설: 강재A의 마모량이 강재B의 마모량보다 2 이상 심하지 않다
# 대립가설: 강재A의 마모량이 강재B의 마모량보다 2 이상 심하다

n1 = 12 # 강재A 표본 크기
x1 = 85 # 강재A 표본 평균
```

```
s1 = 4  # 강재A 표본 표준편차

n2 = 10 # 강재B 표본 크기
x2 = 81 # 강재B 표본 평균
s2 = 5  # 강재B 표본 표준편차

# t-검정
t_stat, p_value = stats.ttest_ind_from_stats(x1, s1, n1, x2 + 2, s2, n2)
print(f'p-value: {p_value:.4f}')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
p-value: 0.3093
p-value는 0.3093로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

```
# 연습문제 28 p279

# 어느 회사에서 직업 훈련이 근로자의 능력 향상에 효과가 있는지 알아보려고 한다. 이를 위해 16명의 근로자를 추출해 직업 훈련 전과 후의 작업능률 점수를 통해 알아본 결과 아래의 데이터와 같을 때, 이 조사결과에서 훈련 전과 훈련 후의 능률이 같은지 검정하시오.

# [ 조건 ] : 유의수준 1% 사용

from scipy import stats

# 귀무가설: 훈련 전과 훈련 후의 능률은 같다
# 대립가설: 훈련 전과 훈련 후의 능률은 다르다

after = [80, 90, 92, 75, 86, 90, 81, 70, 89, 88, 82, 79, 91, 90, 78, 89]
before = [75, 83, 96, 77, 81, 90, 82, 67, 94, 85, 78, 82, 98, 80, 87, 81]

# t-검정
t_stat, p_value = stats.ttest_rel(after,before)
print(f'p-value: {p_value:.4f}')
```

```
alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
p-value: 0.5411
p-value는 0.5411로, 유의 수준 0.01보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

```
# 연습문제 29 p279

# 대학교 신입생을 대상으로 쌍을 이루는 IQ가 비슷한 두 사람을 뽑아 10쌍을 골랐을 때, 각 쌍의 임의의 한 사람에게는 교수방법A로, 다른 사람은 교수방법B로 통계학을 가르친 후 시험을 본 결과가 다음과 같을 때, 두 교수방법의 점수차이에 대한 진짜 평균의 95% 신뢰구간을 구하시오.

# [ 조건 ] : 두 모집단의 점수 차에 대한 분포는 정규분포이다.

import numpy as np
from scipy import stats

# 귀무가설 : 두 교수방법의 점수 차이는 없다.
# 대립가설 : 두 교수방법의 점수 차이는 있다.

A = [76, 60, 85, 58, 91, 75, 82, 64, 70, 88]
B = [81, 52, 87, 70, 86, 77, 90, 63, 58, 83]

diff = np.array(A) - np.array(B)
mean_diff = np.mean(diff)
std_diff = np.std(diff, ddof=1)

t_value = stats.t.ppf(0.975, len(diff)-1)
margin_of_error = t_value * std_diff / np.sqrt(len(diff))

lower_bound = mean_diff - margin_of_error
upper_bound = mean_diff + margin_of_error
```

```
print(f"95% 신뢰구간 : ({lower_bound:.2f} < X < {upper_bound:.2f})")

t_stat = mean_diff / (std_diff / np.sqrt(len(diff)))
p_value = stats.t.sf(np.abs(t_stat), len(diff)-1) * 2

print(f'p-value: {p_value}')

alpha = 0.05 # 유의 수준
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
95% 신뢰구간 : (-5.13 < X < 5.53)
p-value: 0.9342097170313661
p-value는 0.9342로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

```
# 연습문제 30 p279

# 15명의 남자 대학생을 기준으로 피로하지 않은 상태와 피로한 상태에서 외부자극에 반응을 나타내기까지 시간을 측정한 기록이 다음과 같을 때, 피로한 상태에서는 신체기능조절능력이 떨어진다고 할 수 있는지 검정하시오

# [ 조건 ] : 모집단은 정규분포를 따른다.

from scipy import stats

# 귀무가설 : 피로한 상태에서 신체기능조절능력이 떨어지지 않는다.
# 대립가설 : 피로한 상태에서 신체기능조절능력이 떨어진다.

non_fatigued = [158, 92, 65, 98, 33, 89, 148, 58, 142, 117, 74, 66, 109, 57, 85]
fatigued = [91, 59, 215, 226, 223, 91, 92, 177, 134, 116, 153, 219, 143, 164, 100]

t_statistic, p_value = stats.ttest_ind(non_fatigued,fatigued)
print(f"t-statistic : {t_statistic:.4f}")
print(f"p-value: {p_value:.4f}\n")

alpha = 0.05
```

```
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

t-statistic : -3.1498
p-value: 0.0039

p-value는 0.0039로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

연습문제 31 p280

특별한 윤활유 용기들의 용량은 분산 0.3L 의 분산을 가지고 정규분포를 따르는 것으로 알려져 있다. 모분산이 0.03인가를 검정하기 위하여 10개의 용기를 추출하여 다음 자료를 얻었을 때 검정하시오

[조건] : 유의수준 0.01로 검정하여라

```
from scipy import stats

# 귀무가설: 모분산이 0.03이다
# 대립가설: 모분산이 0.03이 아니다

l = [10.2, 9.7, 10.1, 10.3, 10.1, 9.8, 9.9, 10.4, 10.3, 9.8]
n = len(l)
s2 = sum((x - sum(l)/n)**2 for x in l) / (n-1)
chi2_stat = (n-1) * s2 / 0.03
p_value = stats.chi2.sf(chi2_stat, n-1)

print(f"검정통계량 : {chi2_stat:.4f}")
print(f"p-value : {p_value:.4f}\n")

alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무
```

가설을 기각할 수 없음")
검정통계량 : 18.1333
p-value : 0.0337

p-value는 0.0337로, 유의 수준 0.01보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

연습문제 32 p280

기계로 채워지는 음료수는 만일 함량의 분산이 1.15dl을 초과하면 관리하에 있지 않다고 한다. 이 기계로부터 25개의 음료수를 추출하여 함량을 조사한 결과 표본분산이 2.03dl로 나타났다. 기계가 관리하에 있지 않다는 가설을 유의수준 0.05로 검정하라

[조건] : 함량의 분포는 근사적으로 정규분포를 따른다.

```
from scipy import stats

# 귀무가설 : 기계가 관리하에 있다.
# 대립가설 : 기계가 관리하에 있지 않다.

n = 25
sample_variance = 2.03
population_variance = 1.15

chi_squared_statistic = (n - 1) * sample_variance / population_variance
p_value = 1 - stats.chi2.cdf(chi_squared_statistic, n - 1)

print(f"Chi-statistic: {chi_squared_statistic:.4f}")
print(f"p-value: {p_value:.4f}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

Chi-statistic: 42.3652
p-value: 0.0117

p-value는 0.0117로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.
연습문제 33 p280

병에 자동으로 음료를 채우는 시스템에서 채워지는 음료수 양의 분산이 1g이하일 때 시스템이 안정적이라고 할 수 있다. 품질관리 책임자가 음료수병 10개를 임의로 추출하여 음료수의 양을 측정한 결과 표본분산 s^2 = 0.16 이라면 이 시스템의 분산이 1g 이하로 시스템이 안정적인지 유의수준 5%에서 검정하시오.

```
from scipy import stats

# 귀무가설 : 시스템이 안정적이다. ( 분산이 1g 이하이다. )
# 대립가설 : 시스템이 안정적이지 않다. ( 분산이 1g 이하가 아니다. )

n = 10
s2 = 0.16
alpha = 0.05
chi2_stat = (n-1) * s2 / 1
p_value = stats.chi2.sf(chi2_stat, n-1)

print(f"statistics : {chi2_stat:.4f}")
print(f"p-value : {p_value:.4f}\n")

if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

statistics : 1.4400
p-value : 0.9976

p-value는 0.9976로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

연습문제 1 / 예제(9.2), p284

n = 46 중에서 고장이 난 기계들을 분석한 결과 전기적인 고장으로 인한 고장은 x1 = 9, 기계적 결함으로 인한 고장은 x2 = 24, 오작동으로 인한 고장이 x3 = 13일 때. 3 종류 고장의 확률은 각각 20%, 50%, 30% 라는 의견에 대해 유의수준 0.05로 카이제곱 적합도 검정을

시행하시오

```
from scipy import stats

# 귀무가설 : 3 종류 고장의 확률은 각각 20%, 50%, 30%
# 대립가설 : 3 종류 고장의 확률은 각각 20%, 50%, 30%가 아니다.

observed = [9, 24, 13]
expected = [0.2 * 46, 0.5 * 46, 0.3 * 46]
chi2_stat, p_value = stats.chisquare(observed, expected)

print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
검정통계량: 0.0942
p-value: 0.9539906110247998

p-value는 0.9540로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

연습문제 2 / 예제(9.4), p286

공장의 작업장에서 발생하는 사고를 줄이기 위해 새로운 안전운동을 도입하려고 한다. 공장 관리자가 그 주에 발생한 사고 기록을 조사한 결과 다음과 같은 결과를 얻었다. 일주일 동안 총 사고 횟수가 n = 270으로 이에 따르면 특히 월요일과 금요일에 사고가 더 많이 발생한 것으로 나타났다. 만약 이것이 사실이라면 월요일과 금요일에 특히 주의할 필요가 있으며, 이 날들의 사고를 줄이기 위한 조치를 취해야 할 것이다. 그렇다면 월요일과 금요일에 다른 요일보다 사고가 빈번하게 발생하는지 검정하여라

```
from scipy import stats

# 귀무가설 : 월요일과 금요일에 사고가 빈번하게 발생하지 않는다.
# 대립가설 : 월요일과 금요일에 사고가 빈번하게 발생한다.
```

```
data = {'월요일' : 65,  '화요일' : 43, '수요일' : 48, '목요일' : 41, '금요일' : 73}
observed = list(data.values())
total = sum(observed)
expected = [total/5 for _ in range(5)]
chi2_stat, p_value = stats.chisquare(observed, expected)

print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
검정통계량: 14.9630
p-value: 0.004778654086776808

p-value는 0.0048로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.
```

```
# 예제(9.6)

# 회사에서 학력과 회사에 대한 만족도 조사의 연관성이 있는지 알아보기 위해 회사원 300명을 랜덤하게 뽑아 조사한 결과 다음의 자료를 얻었다. 두 변수에 연관성이 있다고 할 수 있는가?

from scipy import stats

# 귀무가설 : 학력과 회사에 대한 만족도는 연관성이 없다.
# 대립가설 : 학력과 회사에 대한 만족도는 연관성이 있다.

dataA = ['고졸 이하', 40, 32, 10]
dataB = ['대졸 이하', 92, 50, 28]
dataC = ['대학원 이상', 16, 20, 12]

observed = [dataA[1:], dataB[1:], dataC[1:]]
chi2_stat, p_value, dof, expected = stats.chi2_contingency(observed)
```

```
print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
검정통계량: 8.7636
p-value: 0.06728899139887037
p-value는 0.0673로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

```
# 예제(9.7)

# 대도시 근교에서 출퇴근하며 혼자서만 승용차를 이용하는 사람들 중에서 250명을 무작위로 추출하여 승용차의 크기와 통근 거리 사이에 관계가 있다고 할 수 있는지를 유의수준 5%에서 검정하시오.

from scipy import stats

# 귀무가설 : 승용차의 크기와 통근 거리 사이에 관계가 없다.
# 대립가설 : 승용차의 크기와 통근 거리 사이에 관계가 있다.

경승용차   = [6, 27, 19]
소형승용차 = [8, 36, 17]
중형승용차 = [21, 45, 33]
대형승용차 = [14, 18, 6]

observed = [경승용차, 소형승용차, 중형승용차, 대형승용차]
chi2_stat, p_value, dof, expected = stats.chi2_contingency(observed)

print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
```

```
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량: 14.1584
p-value: 0.027916449953844118

p-value는 0.0279로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

```
# 예제(9.9)

# 정당에서는 지역에 따라 A, B, C 세 후보에 대한 지지도가 다른지를 알아보기 위해 각 도시에서 200명씩 조사한 결과 다음의 자료를 얻었다. 지역에 따라 지지도가 다른지 검정하여라

from scipy import stats

# 귀무가설 : 지역에 따라 지지도가 다르지 않다.
# 대립가설 : 지역에 따라 지지도가 다르다.

서울 = [73, 71, 56]
부산 = [102, 55, 43]
대구 = [73, 66, 61]
광주 = [62, 98, 40]

observed = [서울, 부산, 대구, 광주]
chi2_stat, p_value, dof, expected = stats.chi2_contingency(observed)

print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량: 31.2946
p-value: 2.226786050768775e-05

p-value는 0.0000로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

```
# 예제(9.11)

# 공단에 인접한 세 지역에서 공해를 느끼는 정도가 지역에 따라 차이가 있는가를 알아보고자 세 지역에서 97명, 95명 99명을 랜덤 추출하여 산업공해로 인한 악취를 느끼는 횟수에 대해 조사한 결과 다음과 같을 때, 범주1은 매일, 범주2는 적어도 일주일에 한번, 범주3은 적어도 한달에 한번, 범주4번은 한달에 한 번 보다는 적게, 범주 5는 전혀 악취를 느끼지 않는 경우를 뜻할 때, 이 자료로부터 지역에 따라 공해를 느끼는 정도가 다른지를 검정하시오.

# [ 조건 ] : 유의수준 1%에서 검정

from scipy import stats

# 귀무가설: 지역에 따라 공해를 느끼는 정도가 차이가 없다.
# 대립가설: 지역에 따라 공해를 느끼는 정도가 차이가 있다.

지역1 = [20, 28, 23, 14, 12]
지역2 = [14, 34, 21, 14, 12]
지역3 = [4, 12, 10, 20, 53]

observed = [지역1, 지역2, 지역3]
chi2_stat, p_value, dof, expected = stats.chi2_contingency(observed)

print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량: 70.6416
p-value: 3.661824689679792e-12

p-value는 0.0000로, 유의 수준 0.01보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 5

# 사상자에 따른 데이터가 시간에 따라 존재할 때 범주1이 아침, 범주2가 점심, 범주3이 저녁이라고 할 때 시간에 따라 사상자가 다르다는 것을 검정하시오

# [ 조건 ] : 유의수준 0.05에서 검정하시오

from scipy import stats

# 귀무가설 : 시간에 따라 사상자가 다르지 않다.
# 대립가설 : 시간에 따라 사상자가 다르다.

사상자 = [1372, 1578, 1686]
total = sum(사상자)
expected = [total/3 for _ in range(3)]
chi2_stat, p_value = stats.chisquare(사상자, expected)

print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량: 32.9370
p-value: 7.043980413550305e-08

p-value는 0.0000로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

연습문제 6

아침 시간에 도심에 있는 5개 다리를 이용하는 교통량의 비율이 2 : 3 : 3 : 4 : 6이라고 감독자가 주장할 때 6000대의 차량을 추출했더니 5개 다리를 건너는 차의 대수가 각각 720, 970, 1013, 1380, 1917이다. 감독자의 주장이 맞는지 유의수준 5%에서 검정하시오.

```
from scipy import stats

# 귀무가설 : 5개 다리를 이용하는 교통량의 비율이 2 : 3 : 3 : 4 : 6이다.
# 대립가설 : 5개 다리를 이용하는 교통량의 비율이 2 : 3 : 3 : 4 : 6이 아니다.

observed = [720, 970, 1013, 1380, 1917]
total = sum(observed)
expected = [total * (2/18), total * (3/18), total * (3/18), total * (4/18), total * (6/18)]
chi2_stat, p_value = stats.chisquare(observed, expected)

print(f"검정통계량: {chi2_stat:.4f}")
print(f"p-value: {p_value:.4f}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량: 10.4135
p-value: 0.0340

p-value는 0.0340로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

연습문제 7

스포츠잡지를 발행하는 회사에서 새로운 고객에게 3가지 경품 (티셔츠, 커피잔, 귀걸이) 중 하나를 선물하고 있다. 500명의 새로운 고객을 무작위로 추출하여 경품의 선호도에 대해 조사한 결과가 다음과 같을 때. 경품의 선호도에 차이가 있는지 검정하라

[조건] : 유의수준 5%에서 검정하라

```
from scipy import stats

# 귀무가설 : 경품의 선호도에 차이가 없다.
# 대립가설 : 경품의 선호도에 차이가 있다.

frequency = [183, 175, 142]
total = sum(frequency)
expected = [total/3 for _ in range(3)]
chi2_stat, p_value = stats.chisquare(frequency, expected)

print(f"검정통계량 : {chi2_stat:.4f}")
print(f"p-value : {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

# 연습문제 8

# 어떤 유형의 범죄발생건수가 대도시의 각 지역별로 다른지를 알아보기 위한 조사가 수행되었다. 특별히 조사대상으로 선정된 범죄는 강간, 강도, 철도 및 살인이다. 범죄발생건수가 대도시의 각 지역과는 무관한지를 유의수준 0.01로 검정하시오

from scipy import stats

# 귀무가설: 범죄발생건수가 대도시의 각 지역과는 무관하다.
# 대립가설: 범죄발생건수가 대도시의 각 지역과는 무관하지 않다.

지역1 = [162, 118, 451, 18]
지역2 = [310, 196, 996, 25]
지역3 = [258, 193, 458, 10]
지역4 = [280, 175, 390, 19]

observed = [지역1, 지역2, 지역3, 지역4]
chi2_stat, p_value, dof, expected = stats.chi2_contingency(observed)

print(f"검정통계량 : {chi2_stat:.4f}")
```

```
print(f"p-value : {p_value}\n")

alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량 : 124.5297
p-value : 1.576242682023537e-22

p-value는 0.0000로, 유의 수준 0.01보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 9

# 다음은 나이에 따라 자동차 A, B, C, D, E 에 대한 선호도를 조사한 결과이다. 나이와 선호하는 자동차 종류는 무관한지 유의수준 5%에서 검정하시오.

from scipy.stats import chi2_contingency

# 귀무가설: 나이와 선호하는 자동차 종류는 무관하다.
# 대립가설: 나이와 선호하는 자동차 종류는 무관하지 않다.

data = [[42, 29, 12, 58], [59, 34, 43, 19], [67, 42, 81, 7]]
chi2, p_value, dof, expected = chi2_contingency(data)

print(f"검정통계량: {chi2:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량: 104.3775
p-value: 3.058332817880553e-20

p-value는 0.0000로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 13

# x^2 = 87.6 일 때 운동 강도와 흡연습관에 상관관계가 있는지 검정하시오.

from scipy.stats import chi2_contingency

# 귀무가설: 운동 강도와 흡연습관은 독립이다.
# 대립가설: 운동 강도와 흡연습관은 독립이 아니다.

data = [[113, 113, 110, 159], [119, 135, 172, 190], [77, 91, 86, 65], [181, 152, 124, 73]]
chi2, p_value, dof, expected = chi2_contingency(data)

print(f"검정통계량: {chi2:.4f}")
print(f"p-value: {p_value}\n")

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량: 87.2727
p-value: 5.7306646048374425e-15

p-value는 0.0000로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 14

# 다음의 데이터는 어느 공장과정에서 3대의 기계로부터 얻어진 제품을 두 등급으로 분류한 분류표이다 자료에서 등급과 기계의 종류를 유의수준 1%에서 독립인지 검정하시오
```

```
from scipy.stats import chi2_contingency

# 귀무가설 : 등급과 기계의 종류는 독립이다.
# 대립가설 : 등급과 기계의 종류는 독립이 아니다.

data = [[78, 65, 68], [22, 8, 30]]
chi2, p_value, dof, expected = chi2_contingency(data)

print(f"검정통계량 : {chi2:.4f}")
print(f"p-value : {p_value}\n")

alpha = 0.01
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
검정통계량 : 9.3759
p-value : 0.009205414784649132

p-value는 0.0092로, 유의 수준 0.01보다 작다.
따라서 귀무 가설을 기각한다.
```

```
# 연습문제 15

#  x^2 = 4.18 일 때, 몸의 정도(건강도(1-30))과 머리 잃는 정도에 관계가 있는지 검정하시오

from scipy.stats import chi2_contingency

# 귀무가설 : 몸의 정도와 머리 잃는 정도는 관계가 없다.
# 대립가설 : 몸의 정도와 머리 잃는 정도는 관계가 있다.

data = [[137, 22, 40], [218, 34, 67], [153, 30, 68]]
chi2, p_value, dof, expected = chi2_contingency(data)

print(f"검정통계량 : {chi2:.4f}")
print(f"p-value : {p_value}\n")
```

```
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

검정통계량 : 4.8089
p-value : 0.30747607236753727

p-value는 0.3075로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

연습문제 9 p302, node (16)

어떤 회사에서 생산되는 제품의 불량품과 양품의 비율이 낮, 저녁, 밤에 따라 다른지를 검정하기 위해 다음과 같은 자료를 얻었다. 낮, 저녁, 밤에 만들어진 제품의 불량품과 양품의 비율이 서로 같다고 할 수 있는가를 검정하시오

```
from scipy.stats import chi2_contingency
import numpy as np

# 귀무가설: 낮, 저녁, 밤에 만들어진 제품의 불량품과 양품의 비율이 서로 같다.
# 대립가설: 낮, 저녁, 밤에 만들어진 제품의 불량품과 양품의 비율이 서로 다르다.

defective = [80, 70, 80]
non_defective = [1120, 930, 720]

obs = np.array([defective, non_defective])
chi2, p_value, dof, expected = chi2_contingency(obs)
print(f'p-value: {p_value:.4f}')

alpha = 0.025
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

p-value: 0.0144
p-value는 0.0144로, 유의 수준 0.025보다 작다.
따라서 귀무 가설을 기각한다.

```
# 연습문제 17

# 다음 자료는 1000명의 주부를 대상으로 4 기간 동안 조사한 생활수준이다. 각 생활수준의 범주 내에서 주분의 비율이 각 기간마다 동일한지 P값을 이용하여 검정하라

from scipy.stats import chi2_contingency
import numpy as np

# 귀무가설: 각 기간마다 주부의 생활수준의 비율은 동일하다.
# 대립가설: 각 기간마다 주부의 생활수준의 비율은 동일하지 않다.

improved = [72, 63, 47, 40]
same     = [144, 135, 100, 105]
worsened = [84, 102, 53, 55]

obs = np.array([improved, same, worsened])
chi2, p_value, dof, expected = chi2_contingency(obs)
print(f'p-value: {p_value}')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

p-value: 0.43170620277662486
p-value는 0.4317로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

```
# 연습문제 11 p302, node (18)

# 다가오는 선거에서 두 명의 도지사후보에 대한 유권자들의 성향을 알아보기 위한 조사를 두 도시에 대해 500명의 유권자를 대상으로 조사한 결과는 다음과 같다. 유권자들의 후보 A 지지율 , 후보 B 지지율 및 미결정 비율은 각 도시마다 동일할 때, 귀무가설을 유의수준 5% 에서 검정하시오.
```

```
from scipy.stats import chi2_contingency
import numpy as np

# 귀무가설: 두 도시의 유권자들의 성향은 동일하다.
# 대립가설: 두 도시의 유권자들의 성향은 동일하지 않다.

candidate_A = [204, 225]
candidate_B = [211, 198]
undecided   = [85, 77]

obs = np.array([candidate_A, candidate_B, undecided])
chi2, p_value, dof, expected = chi2_contingency(obs)
print(f'p-value: {p_value}')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
p-value: 0.39926962230647095
p-value는 0.3993로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

연습문제 20

액화천연가스(LNG)의 저장 기지 후보지로 고려되는 세 지역의 여론을 알아보기 위해 세 지역에서 각각 400명, 350명, 350명을 랜덤 추출하여 기지의 건설에 대한 찬성여부를 물은 결과 다음과 같을 때, 지역에 따라 찬성률에 차이가 있다고 할 수 있는지 검정하시오

```
from scipy.stats import chi2_contingency
import numpy as np

# 귀무가설: 지역에 따라 찬성률에 차이가 없다.
# 대립가설: 지역에 따라 찬성률에 차이가 있다.

region1 = [198, 202]
region2 = [140, 210]
```

```
region3 = [133, 217]
# 표본 크기 (n(x))는 계산에 영향을 주지 않아 삭제!

obs = np.array([region1[:2], region2[:2], region3[:2]])
chi2, p_value, dof, expected = chi2_contingency(obs)
print(f'p-value: {p_value}')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

p-value: 0.002811845136348792
p-value는 0.0028로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

# 예제(10.2)

# 8개의 다른 종류의 토양에 대한 땅콩의 산출량 y와 옥수수의 산출량 x에 관한 자료를 제시할 때 두 변수간 표본상관계수 r을 구하시오

import numpy as np

x = [2.4, 3.4, 4.6, 3.7, 2.2, 3.3, 4.0, 2.1]
y = [1.33, 2.12, 1.80, 1.65, 2.00, 1.76, 2.11, 1.63]

x_mean = np.mean(x)
y_mean = np.mean(y)
numerator = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y))
denominator = np.sqrt(sum((x_i - x_mean) ** 2 for x_i in x)) * np.sqrt(sum((y_i - y_mean) ** 2 for y_i in y))
r = numerator / denominator

print(f'표본상관계수 r: {r:.3f}')
표본상관계수 r: 0.347

# 예제(10.3)
```

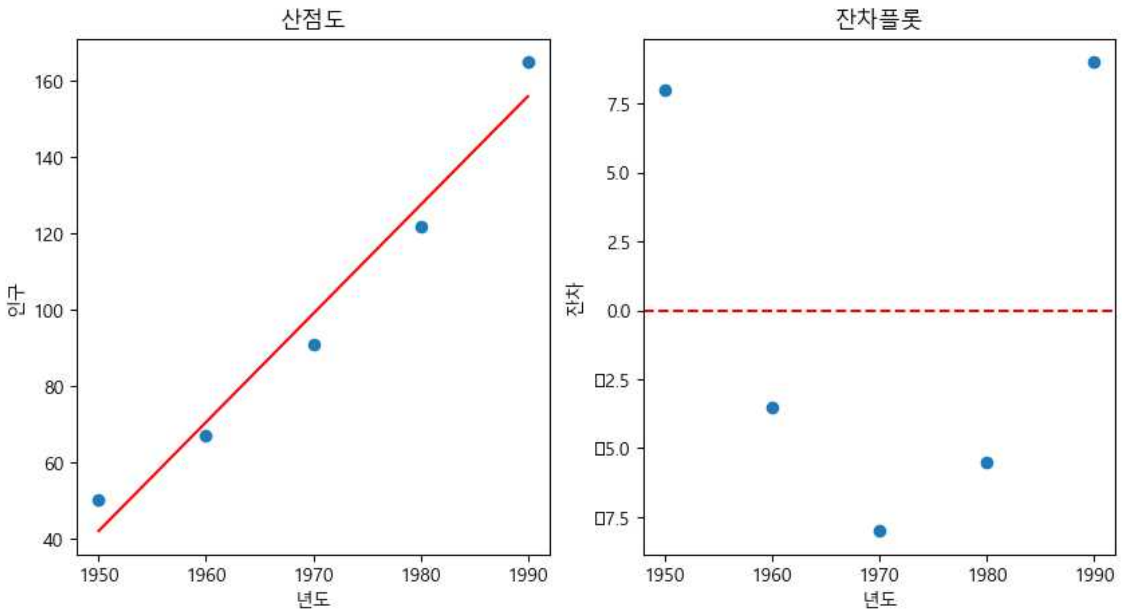
```
# 다음과 같은 표를 이용하여 문제에 답하여라
# 1. 상관계수 (r) 구하기
# 2. 강우량에서 대기오염의 제거정도를 예측하기 위한 회귀직선의 방정식 구하기
# 3. 강우량이 x = 5.8 일 때, 대기오염 제거정도를 추정하라 ( 머신러닝 )

import numpy as np

x = [4.3, 4.5, 5.9, 5.6, 6.1, 5.2, 3.8, 2.1, 7.5]
y = [126, 121, 116, 118, 114, 118, 132, 141, 108]

x_mean = np.mean(x)
y_mean = np.mean(y)
numerator = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y))
denominator = np.sqrt(sum((x_i - x_mean) ** 2 for x_i in x)) * np.sqrt(sum((y_i - y_mean) ** 2 for y_i in y))
r = numerator / denominator

print(f'1. 상관계수 r : {r:.4f}')
b1 = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y)) / sum((x_i - x_mean) ** 2 for x_i in x)
b0 = y_mean - b1 * x_mean
print(f'2. 회귀직선의 방정식: y = {b0:.4f} ± {b1:.4f}x')
x = 5.8 # 강우량 할당
y_hat = b0 + b1 * x
print(f'3. 강우량이 {x} 일 때 대기오염 제거정도: {y_hat:.3f}')

1. 상관계수 r : -0.9787
2. 회귀직선의 방정식: y = 153.1755 ± -6.3240x
3. 강우량이 5.8 일 때 대기오염 제거정도: 116.496

# 예제(10.6)

# 예제 10.3에서 강우량 자료의 결정계수 R^2 구하기

import numpy as np

x = [4.3, 4.5, 5.9, 5.6, 6.1, 5.2, 3.8, 2.1, 7.5]
y = [126, 121, 116, 118, 114, 118, 132, 141, 108]

x_mean = np.mean(x)
y_mean = np.mean(y)
```

```
b1 = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y)) / sum((x_i - x_mean) ** 2 for x_i in x)
b0 = y_mean - b1 * x_mean
y_mean = np.mean(y)
ss_tot = sum((y_i - y_mean) ** 2 for y_i in y)
ss_res = sum((y_i - (b0 + b1 * x_i)) ** 2 for x_i, y_i in zip(x, y))
r2 = 1 - (ss_res / ss_tot)

print(f'결정계수 R^2: {r2:.5f}')
```

결정계수 R^2: 0.95777

```
# 예제(10.12)

# 다음의 그래프를 보고 산점도와 잔차플롯은 비선형관계가 더욱 강한 모형인지 판단하시오

import matplotlib.pyplot as plt
import numpy as np

plt.rc('font', family='Malgun Gothic')

x = [1950, 1960, 1970, 1980, 1990]
y = [50, 67, 91, 122, 165]

x_mean = np.mean(x)
y_mean = np.mean(y)
b1 = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y)) / sum((x_i - x_mean) ** 2 for x_i in x)
b0 = y_mean - b1 * x_mean

y_hat = [b0 + b1 * x_i for x_i in x]
residuals = [y_i - y_hat_i for y_i, y_hat_i in zip(y, y_hat)]

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 5))

ax1.scatter(x, y)
ax1.plot(x, y_hat, color='r')
ax1.set_xlabel('년도')
ax1.set_ylabel('인구')
ax1.set_title('산점도')
```

```
ax2.scatter(x, residuals)
ax2.axhline(y=0, color='r', linestyle='--')
ax2.set_xlabel('년도')
ax2.set_ylabel('잔차')
ax2.set_title('잔차플롯')
plt.show()
```



예제(10.15)

10명의 고등학생으로부터 TV 시청 시간과 성적을 조사한 결과를 다음과 같이 나타내었다. 최소 제곱법 선을 구하고 실제 기울기의 95% 신뢰구간을 구하여라

```
import numpy as np
from scipy import stats

x = [12, 21, 8, 20, 16, 16, 24, 0, 11, 18]
y = [3.1, 2.3, 3.5, 2.5, 3.0, 2.6, 2.1, 3.8, 2.9, 2.6]
alpha  = 0.05

x_mean = np.mean(x)
y_mean = np.mean(y)
b1 = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y)) / sum((x_i - x_mean) ** 2 for x_i in x)
b0 = y_mean - b1 * x_mean
```

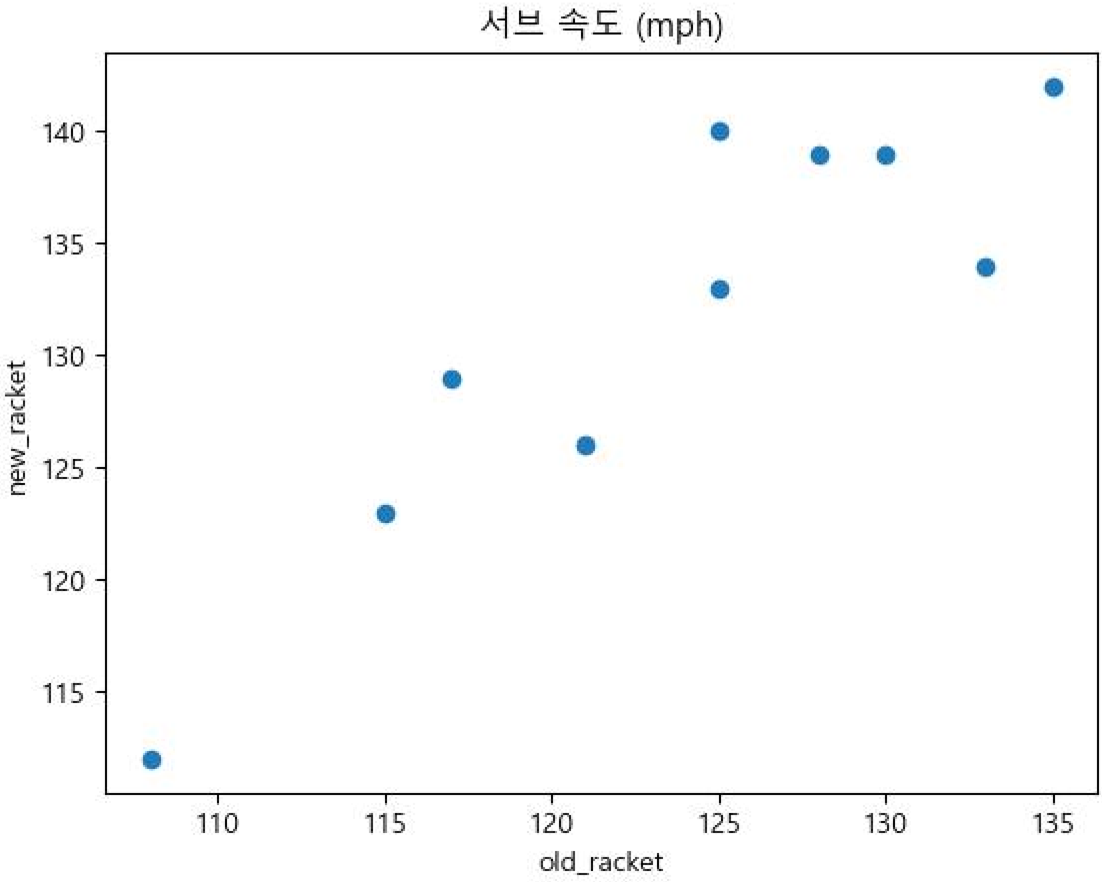
```
n = len(x)
df = n - 2
t = stats.t.ppf(1 - alpha / 2, df)
sse = sum((y_i - (b0 + b1 * x_i)) ** 2 for x_i, y_i in zip(x, y))
s2 = sse / df
x_mean = np.mean(x)
s_b1 = np.sqrt(s2 / sum((x_i - x_mean) ** 2 for x_i in x))
lower = abs(b1 + t * s_b1)
upper = abs(b1 - t * s_b1)

print(f'최소 제곱법 선: y = {b0:.4f} + x = {b1:.4f}')
print(f'실제 기울기의 95% 신뢰구간: {b1:.4f} ± {t * s_b1:.4f}')
```

최소 제곱법 선: y = 3.8916 + x = -0.0720
실제 기울기의 95% 신뢰구간: -0.0720 ± 0.0166

```
# 예제(10.18)

# 무작위로 뽑힌 프로 테니스 선수의 새로 개발된 테니스 라켓과 기존 라켓의 서브 속도를 mph로 나타낸 것이다. 오래된 라켓과 새로운 라켓을 사용한 속도 사이에 플러스 기울기를 가진 관계가 있는지 확인하는 증거를 찾아라.

from scipy import stats
import numpy as np
import matplotlib.pyplot as plt

old_racket = [125, 133, 108, 128, 115, 135, 125, 117, 130, 121]
new_racket = [133, 134, 112, 139, 123, 142, 140, 129, 139, 126]

x_mean = np.mean(old_racket)
y_mean = np.mean(new_racket)
numerator = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(old_racket, new_racket))
denominator = np.sqrt(sum((x_i - x_mean) ** 2 for x_i in old_racket)) * np.sqrt(sum((y_i - y_mean) ** 2 for y_i in new_racket))
r = numerator / denominator

print(f'상관 계수 r: {r:.4f}')

x, y = old_racket, new_racket
```

```
n = len(x)
df = n - 2
r = np.corrcoef(x, y)[0, 1]
t = r * np.sqrt(df / (1 - r ** 2))
p_value = 2 * (1 - stats.t.cdf(abs(t), df))

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.5f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.5f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

plt.rc('font', family='Malgun Gothic')
plt.scatter(x, y)
plt.xlabel('old_racket')
plt.ylabel('new_racket')
plt.title('서브 속도 (mph)')
plt.show()
```

```
상관 계수 r: 0.9004
p-value는 0.00038로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.
```



연습문제 1

임의로 추출한 컴퓨터 공학과 학생 6명의 통계학 점수와 프로그램 언어 점수에서 상관계수를 구하시오.

```
import numpy as np

x = [70, 90, 80, 74, 65, 83]
y = [74, 84, 63, 87, 78, 90]

x_mean = np.mean(x)
y_mean = np.mean(y)
numerator = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y))
denominator = np.sqrt(sum((x_i - x_mean) ** 2 for x_i in x)) * np.sqrt(sum((y_i - y_mean) ** 2 for y_i in y))
r = numerator / denominator

print(f'상관 계수: {r:.4f}')
상관 계수: 0.2345
```

연습문제 3

어떤 공장에서 여러 수준의 온도 변화에 따른 당분으로 변환된 양을 측정한 데이터가 있을 때, 회귀직선을 추정하고 온도가 1.75일 때, 당분으로 변환된 양을 추정하시오

```
import numpy as np

x = [1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0]
y = [8.1, 7, 8.5, 9.8, 9.5, 8.9, 8.6, 10.2, 9.3, 9.2, 10.5]

x_new = 1.75

x_mean = np.mean(x)
y_mean = np.mean(y)
b1 = sum((x_i - x_mean) * (y_i - y_mean) for x_i, y_i in zip(x, y)) / sum((x_i - x_mean) ** 2 for x_i in x)
b0 = y_mean - b1 * x_mean

y_hat = b0 + b1 * x_new

print(f'1. 회귀직선: y = {b0:.4f} + {b1:.4f}x')
print(f'2. 온도가 {x_new} 일 때 당분으로 변환된 양: {y_hat:.2f}')
```

온도가 1.75 일 때 당분으로 변환된 양: 9.58
회귀직선: y = 5.9045 + 2.1000x

연습문제 5

다음과 같은 자료로 물음에 답하여라

[조건]
1. 선형회귀모델을 구하라
2. SE^2을 구하라
3. H0 : Beta1 = 0, Ha : beta ! 0을 alpha = 0.05로 검정하라
4. 유의수준 5%인 beta의 신뢰구간을 구하라

```
import numpy as np
from scipy import stats

x = np.array([1, 2, 3, 4, 5])
```

```
y = np.array([3, 3, 2, 6, 5])

# 선형 회귀 모델
slope, intercept, r_value, p_value, std_err = stats.linregress(x,y)
print("1. 선형 회귀 모델을 구하시오")
print(f'선형 회귀 모델 : b0 = {intercept:.2f} y = {intercept:.2f} + {slope:.2f}x')

# SE^2
SE_2 = (std_err**2) * 10
print("\n2. SE^2을 구하시오")
print(f'SE^2: {SE_2:.5f}')

# H0 : Beta1 = 0, Ha : beta ! 0을 alpha = 0.05로 검정
print("\n3. H0 : Beta1 = 0, Ha : beta ! 0을 alpha = 0.05로 검정하시오")
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

# 유의 수준 5%인 beta의 신뢰 구간
n = len(x)
t_critical = stats.t.ppf(1 - alpha/2, n-2)
lower_bound = abs(slope - t_critical * std_err)
upper_bound = slope + t_critical * std_err
print("\n4. 유의수준 5%인 beta의 신뢰구간을 구하시오")
print(f'유의 수준 5%인 beta의 신뢰 구간: [{lower_bound:.4f}, {upper_bound:.4f}]')
print(f'beta의 신뢰구간 {lower_bound:.4f} < beta < {upper_bound:.4f}')
```

```
1. 선형 회귀 모델을 구하시오
선형 회귀 모델 : b0 = 1.70 y = 1.70 + 0.70x

2. SE^2을 구하시오
SE^2: 1.96667

3. H0 : Beta1 = 0, Ha : beta ! 0을 alpha = 0.05로 검정하시오
p-value는 0.2126로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

4. 유의수준 5%인 beta의 신뢰구간을 구하시오
유의 수준 5%인 beta의 신뢰 구간: [0.7113, 2.1113]
beta의 신뢰구간 0.7113 < beta < 2.1113

```
# 연습문제 7

# 어떤 공장에서 여러 수준의 온도 변화에 따른 당분으로 변환된 양을 측정한 데이터가 있을 때, 회귀직선을 추정하고 온도가 1.75일 때, 당분으로 변환된 양을 추정하시오

# [ 조건 ]
# 1. 회귀직선을 추정하라
# 2. 온도가 1.75일 때, 당분으로 변환된 양을 추정하라
# 3. s(2)^2을 구하라
# 4. beta의 95% 신뢰구간을 구하라

import numpy as np
from scipy import stats

x = np.array([1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0])
y = np.array([8.1, 7.8, 8.5, 9.8, 9.5, 8.9, 8.6, 10.2, 9.3, 9.2, 10.5])

# 회귀직선 추정
print('1. 회귀직선을 추정하라')
slope, intercept, r_value, p_value, std_err = stats.linregress(x,y)
print(f'회귀직선: y = {intercept:.2f} + {slope:.2f}x')

# 온도가 1.75일 때 당분으로 변환된 양 추정
print('\n2. 온도가 1.75일 때, 당분으로 변환된 양을 추정하라')
y_pred = intercept + slope * 1.75
print(f'온도가 1.75일 때 당분으로 변환된 양: {y_pred:.2f}')

# s(2)^2 계산
SE_2 = std_err**2
print('\n3. s(e)^2을 구하라')
print(f's(2)^2: {SE_2:.4f}')

# beta의 95% 신뢰구간 계산
n = len(x)
t_critical = stats.t.ppf(0.975, n-2)
lower_bound = slope - t_critical * std_err
```

```
upper_bound = slope + t_critical * std_err
print(f'\nβ의 95% 신뢰구간: [{lower_bound:.3f} < β < {upper_bound:.3f}]')
```

1. 회귀직선을 추정하라
회귀직선: y = 6.41 + 1.81x

2. 온도가 1.75일 때, 당분으로 변환된 양을 추정하라
온도가 1.75일 때 당분으로 변환된 양: 9.58

3. s(e)^2을 구하라
s(2)^2: 0.3638

β의 95% 신뢰구간: [0.445 < β < 3.174]

```
# 연습문제 11

# 어떤 공장에서 여러 수준의 온도 변화에 따른 당분으로 변환된 양을 측정한 데이터가 있을 때, 회귀직선을 추정하고 온도가 1.75일 때, 당분으로 변환된 양을 추정하시오

# [ 조건 ]
# 1. 최소제곱 회귀직선식을 구하라
# 2. 90% 구간추정치를 구하라

import numpy as np
from scipy import stats

x = np.array([2, 3, 4, 4, 5, 6, 6, 7, 8, 8, 9, 10])
y = np.array([3, 7, 6, 8, 10, 8, 13, 16, 15, 21, 23, 24])

# 최소제곱 회귀직선식
slope, intercept, r_value, p_value, std_err = stats.linregress(x,y)
print(f'최소제곱 회귀직선식: y = {intercept:.2f} + {slope:.2f}x')

# 90% 구간 추정
n = len(x)
t_critical = stats.t.ppf(0.95, n-2)
lower_bound = slope - t_critical * std_err
margin_of_error = t_critical * std_err
print(f'90% 구간 추정: {slope:.2f} +- {margin_of_error:.3f}')
```

최소제곱 회귀직선식: y = -3.23 + 2.68x
90% 구간 추정: 2.68 +- 0.513

```
# 연습문제 13

# 데이터를 보고 물음에 답하라!

# [ 조건 ]
# 1. 회귀직선식을 구하시오
# 2. 95%의 beta의 구간추정치를 구하시오
# 3. 회귀직선의 기울기에 대한 유의성 검정을 수행하시오

import numpy as np
from scipy import stats
from sklearn.linear_model import LinearRegression

X = np.array([4, 2, 9, 8, 14, 2, 11, 14, 7, 4, 1, 9, 9, 10, 5]).reshape(-1, 1)
Y = np.array([423, 520, 550, 309, 690, 401, 470, 582, 284, 440, 452, 568, 339, 355,
472])

model = LinearRegression()
model.fit(X,Y)

print('1. 회귀직선식을 구하시오')
print("Intercept: ", round((model.intercept_), 3))
print("Coefficient: ", round((model.coef_[0]), 3))

Y_pred = model.predict(X)
residuals = Y - Y_pred
residual_sum_of_squares = np.sum(residuals**2)
s2 = residual_sum_of_squares / (len(Y) -2)

standard_error = np.sqrt(s2) * np.sqrt(np.sum((X - np.mean(X))**2))
t_critical = stats.t.ppf(1 -0.05/2 , df=len(Y)-2)

lower_bound = model.coef_[0] - t_critical * standard_error
upper_bound = model.coef_[0] + t_critical * standard_error

print('\n2. beta의 95% 구간추정치를 구하시오')
print("beta의 95% 구간추정치 : (", round((lower_bound), 3),",", round((upper_bound),
3) ,")")

t_statistic = model.coef_[0] / (np.sqrt(s2) * np.sqrt(np.sum((X - np.mean(X))**2)))
p_value = 2 * (1 - stats.t.cdf(abs(t_statistic), df=len(Y)-2))
```

```
print('\n3. 회귀직선의 기울기에 대한 유의성 검정을 수행하시오')
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

1. 회귀직선식을 구하시오
Intercept: 385.39
Coefficient: 9.855

2. beta의 95% 구간추정치를 구하시오
beta의 95% 구간추정치 : (-3636.206 , 3655.915)

3. 회귀직선의 기울기에 대한 유의성 검정을 수행하시오
p-value는 0.9954로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

```
# 연습문제 15

# 데이터를 보고 물음에 답하라!

# [ 조건 ]
# 1. 회귀직선식과 상관계수를 구하여라
# 2. 비선형 모델을 찾기 위해, log(인구수) 대 년도(1900~)을 그리고 회귀직선과 상관계수를 구하여라
# 3. 각각의 모델을 이용해 2100년의 인구수를 예측하고 인구수가 300(millions)에 이르는 시기를 예측하라
# 4. (3)의 정답에 대한 정확성을 논평하라(???)


import numpy as np
from sklearn.linear_model import LinearRegression

X = np.array([1900, 1905, 1910, 1915, 1920, 1925, 1930, 1935, 1940, 1945, 1950, 1955, 1960, 1965, 1970, 1975, 1980, 1985, 1990]).reshape(-1,1)
Y = np.array([76.1, 83.8, 92.4, 100.5, 106.5, 115.8, 123.1, 127.3, 132.5, 133.4, 151.9, 165.1, 180.0, 193.5, 204.0,215.5 ,227.2 ,237.9 ,249.4])
```

```
log_Y = np.log(Y)

model = LinearRegression()
model.fit(X,Y)

print('1. 회귀직선식과 상관계수를 구하여라')
print("Intercept: ", round((model.intercept_), 4))
print("Coefficient: ", round((model.coef_[0]), 4))
correlation_coefficient = np.corrcoef(X.reshape(1,-1), Y)[0][1]
print("상관계수 : ", round((correlation_coefficient), 4))

population_2100_modelA = model.predict(np.array([[2100]]))[0]
year_modelA = (300 - model.intercept_) / model.coef_[0]

model.fit(X,log_Y)
print('\n2. 비선형 모델을 찾기 위해, log(인구수) 대 년도(1900~)을 그리고 회귀직선과 상관계수를 구하여라')
print("Intercept: ", round((model.intercept_), 4))
print("Coefficient: ", round((model.coef_[0]), 4))
correlation_coefficient = np.corrcoef(X.reshape(1,-1), Y)[0][1]
print("상관계수 : ", round((correlation_coefficient), 4))

population_2100_modelB = model.predict(np.array([[2100]]))[0]
year_modelB = (300 - model.intercept_) / model.coef_[0]

print('\n3. 각각의 모델을 이용해 2100년의 인구수를 예측하고 인구수가 300(millions)에 이르는 시기를 예측하라')
print(f"2100년의 인구수 예측 (모델A) {population_2100_modelA}")
print(f"300 millian에 인구수가 도달하는 시점 (모델A) : {year_modelA}\n")

print(f"2100년의 인구수 예측 (모델B) {population_2100_modelB}")
print(f"300 millian에 인구수가 도달하는 시점 (모델B) : {year_modelB}")
```

```
1. 회귀직선식과 상관계수를 구하여라
Intercept:  -3593.284
Coefficient:  1.9264
상관계수 :  0.989

2. 비선형 모델을 찾기 위해, log(인구수) 대 년도(1900~)을 그리고 회귀직선과 상관계수를 구하여라
Intercept:  -20.2833
```

Coefficient: 0.013
상관계수 : 0.989

3. 각각의 모델을 이용해 2100년의 인구수를 예측하고 인구수가 300(millions)에 이르는 시기를 예측하라
2100년의 인구수 예측 (모델A) 452.0528070175437
300 millian에 인구수가 도달하는 시점 (모델A) : 2021.0669204568223

2100년의 인구수 예측 (모델B) 6.983727241844431
300 millian에 인구수가 도달하는 시점 (모델B) : 24666.983716788473

연습문제 1 / 예제(11.1), p351

토양의 박테리아 분량을 측정하기 위해 대상 지역을 4개 구로 나누고, 각 지역에서 15개의 소의 토양을 채취하여 박테리아의 분량을 측정하였다. 각 지역에 있어서 박테리아의 균집수에는 유의적인 차이가 있는지 분석하시오

```
import pandas as pd
from scipy import stats

# 귀무가설: 각 지역에 있어서 박테리아의 균집수에는 유의적인 차이가 없다.
# 대립가설: 각 지역에 있어서 박테리아의 균집수에는 유의적인 차이가 있다.

# 다중 데이터프레임으로 묶을 수 있으나 위와 같이 다중 리스트의 길이가 다른 경우 각 리스트별 저장 후 하나로 합치는 과정이 필요함. (zip 함수 사용)
A = [72, 69, 63, 53, 51]
B = [47, 52, 45, 30]
C = [56, 58, 56]
D = [69, 67, 62]

df = pd.DataFrame(zip(A, B, C, D), columns=['A', 'B', 'C', 'D'])

h_value, p_value = stats.kruskal(df['A'].dropna(), df['B'].dropna(), df['C'].dropna(), df['D'].dropna())

print(f'H-value: {round((h_value), 4)}')
print(f'P-value: {round((p_value), 4)}\n')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을
```

```
기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

H-value: 9.6185
P-value: 0.0221

p-value는 0.0221로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.
```

```
# 예제(11.1)

# 3종류의 온도계로 어느 날의 온도를 4회 측정한 결과 다음과 같았다. 이때, 각 온도계에 위한 온도에 차이가 있는지를 검정하시오

# [ 조건 ] : 표의 수치는 각 측정값에서 67.0을 뺀 것이다.


import pandas as pd
from scipy import stats

# 귀무가설 : 온도계에 따른 온도의 차이가 없다.
# 대립가설 : 온도계에 따른 온도의 차이가 있다.

# 11.1 과 다르게 한번에 데이터프레임으로 변환하는 경우.
data = {'A': [18, -18, -4, 8],
        'B': [24, 20, 1, 10],
        'C': [5, -24, -8, -17]}

df = pd.DataFrame(data)

f_value, p_value = stats.f_oneway(df['A'], df['B'], df['C'])

print(f'F-value: {round((f_value), 3)}')
print(f'P-value: {round((p_value), 3)}\n')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.3f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
```

```
else:
    print(f"p-value는 {p_value:.3f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

# 예제(11.3)

# 공원1 , 공원2, 공원3이 기계(A~D)를 사용하여 하루에 생산하는 제품의 수는 다음과 같았을 때, 기계의 차에 의한 영향을 배제하고 개인차를 검정하여라 또한 개인차를 배제한 기계에 의한 제품수의 평균차(기계의 우열)을 검정하라

# [ 조건 ]
# 1. 평균값 사이의 차를 조사하면 유의차는 인정되지 않는다.
# 2. 데이터는 기계의 차에 의한 변동 때문에 개인차에 의한 변동이 나타나지 않을 수도 있다.
# 3. 데이터의 각 제품 수는 제조한 제품 개수에서 [35]개를 제외한 것이다.

import pandas as pd
from scipy import stats

# 귀무가설A : 개인차가 없다.
# 대립가설A : 개인차가 있다.

# 귀무가설B : 기계에 의한 차이가 없다.
# 대립가설B : 기계에 의한 차이가 있다.

data = {'1': [1, 3, 2, 0],
        '2': [0, 2, 3, -2],
        '3': [-1, 1, 0, -3]}

df = pd.DataFrame(data, columns=['1', '2', '3'], index=['A', 'B', 'C', 'D'])

# 개인차 검정
f_value, p_value = stats.f_oneway(df['1'], df['2'], df['3'])

print(f'F-value (개인차 검정) : {round((f_value), 2)}')
print(f'P-value (개인차 검정) : {round((p_value), 2)}\n')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
```

```
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

print("\n---------------------------------------------")
# 기계 차이 검정
f_value, p_value = stats.f_oneway(df.loc['A'], df.loc['B'], df.loc['C'], df.loc['D'])

print(f'\nF-value (기계차 검정) : {round((f_value), 2)}')
print(f'P-value (기계차 검정) : {round((p_value), 2)}\n')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
F-value (개인차 검정) : 1.66
P-value (개인차 검정) : 0.24

p-value는 0.2438로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

---------------------------------------------

F-value (기계차 검정) : 5.13
P-value (기계차 검정) : 0.03

p-value는 0.0286로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.
```

예제(11.4)

무연탄에서 코크스를 제조하는데 10% 첨가하는 역청탄 (Y(n))을 5종류 선택하고 타르피티(A1~A4)을 A1: 4%, A2: 6%, A3: 8%, A4: 10%로 첨가하여 5종류의 혼합탄을 제조하고 코크스의 내압강도 (kg/com^2)을 측정한 결과가 다음과 같을 때, 이 자료에 대해 이원배치법의 모형을 적용한다면 역청탄의 종류와 타르피치의 첨가량이 코크스의 내압강도에 미치는 영향을 검정하시오.

[조건] : 유의수준 5%에서 검정하시오.

```
import pandas as pd
from scipy import stats

# 귀무가설: 역청탄의 종류와 타르피치의 첨가량이 코크스의 내압강도에 영향을 미치지 않는
다.
# 대립가설: 역청탄의 종류와 타르피치의 첨가량이 코크스의 내압강도에 영향을 미친다.

data = {'Y1': [79, 75, 65, 65],
        'Y2': [72, 66, 62, 62],
        'Y3': [51, 48, 41, 41],
        'Y4': [58, 56, 45, 45],
        'Y5': [68, 62, 58, 58]}

df = pd.DataFrame(data, columns=['Y1', 'Y2', 'Y3', 'Y4', 'Y5'], index=['A1', 'A2',
'A3', 'A4'])

# 역청탄 종류 영향 검정
f_value1, p_value1 = stats.f_oneway(df['Y1'], df['Y2'], df['Y3'], df['Y4'], df['Y5'])

print(f'F-value : {f_value1}')
print(f'P-value : {p_value1}')

alpha = 0.05
if p_value1 < alpha:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을
기각한다.")
else:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무
가설을 기각할 수 없음")

# 타르피치 첨가량 영향 검정
f_value2, p_value2 = stats.f_oneway(df.loc['A1'], df.loc['A2'], df.loc['A3'], df.loc['A4'])

print(f'\nF-value2 : {f_value2}')
print(f'P-value2 : {p_value2}')

alpha = 0.05
if p_value2 < alpha:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을
기각한다.")
```

else:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

F-value : 13.13366093366094
P-value : 8.588588287858503e-05
p-value는 0.0001로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

F-value2 : 1.4011025358324143
P-value2 : 0.2788614720327024
p-value는 0.2789로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

예제(11.5)

세 종류의 호르몬 처리와 성별에 따라 혈액 칼슘값에 차이가 있는지 알아보기 위해 남녀 각 15명씩 선정하여 세 집단으로 나누어 세 가지 호르몬 처리를 한 후 혈액 칼슘을 측정한 결과가 다음과 같을 때, 조건을 따라 검정하시오

[조건]
1. 남녀 간의 혈액칼슘값에 차이가 있는가?
2. 처리 1,2,3 간의 혈액칼슘값에 차이가 있는가?
3. 성별과 처리 간의 상호작용(교호작용)이 있는가?

import pandas as pd
from scipy import stats

data = {'x1': [16.87, 16.18, 17.12, 16.83, 17.19, 15.86, 14.92, 15.63, 15.24, 14.80],
        'x2': [19.07, 18.77, 17.63, 16.99, 18.04, 17.20, 17.64, 17.89, 16.78, 16.72],
        'x3': [32.45, 28.71, 34.65, 28.79, 24.46, 30.54, 32.41, 28.97, 28.46, 29.65]}

df = pd.DataFrame(data)

f_value1, p_value1 = stats.f_oneway(df.loc[:4].mean(axis=1), df.loc[5:].mean(axis=1))

귀무가설: 남녀 간의 혈액칼슘값에 차이가 없다.
대립가설: 남녀 간의 혈액칼슘값에 차이가 있다.

print('1. 남녀 간의 혈액칼슘값에 차이가 있는가?')
print(f'F-value (gender): {f_value1}')

```
print(f'P-value (gender): {p_value1}')

alpha = 0.05
if p_value1 < alpha:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

f_value2, p_value2 = stats.f_oneway(df['x1'], df['x2'], df['x3'])

# 귀무가설: 처리 1,2,3 간의 혈액칼슘값에 차이가 없다.
# 대립가설: 처리 1,2,3 간의 혈액칼슘값에 차이가 있다.

print('\n2. 처리 1,2,3 간의 혈액칼슘값에 차이가 있는가?')
print(f'F-value (hormone treatment): {f_value2}')
print(f'P-value (hormone treatment): {p_value2}')

alpha = 0.05
if p_value2 < alpha:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

_, p_value3 = stats.f_oneway(df.loc[:4].mean(axis=1), df.loc[5:].mean(axis=1), df['x1'], df['x2'], df['x3'])
f_value3 = stats.f_oneway(df.loc[:4].mean(axis=1), df.loc[5:].mean(axis=1), df['x1'], df['x2'], df['x3']).statistic
# 귀무가설: 성별과 처리 간의 상호작용(교호작용)이 없다.
# 대립가설: 성별과 처리 간의 상호작용(교호작용)이 있다.

print('\n3. 성별과 처리 간의 상호작용(교호작용)이 있는가?')
print(f'P-value (상호작용(p)): {p_value3}')

alpha = 0.05
if p_value3 < alpha:
    print(f"p-value는 {p_value3:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
```

else:
    print(f"p-value는 {p_value3:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

1. 남녀 간의 혈액칼슘값에 차이가 있는가?
F-value (gender): 1.2281053891636358
P-value (gender): 0.2999781499107576
p-value는 0.3000로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

2. 처리 1,2,3 간의 혈액칼슘값에 차이가 있는가?
F-value (hormone treatment): 183.84284815750473
P-value (hormone treatment): 1.8793907468359085e-16
p-value는 0.0000로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

3. 성별과 처리 간의 상호작용(교호작용)이 있는가?
P-value (상호작용(p)): 3.5024986934065393e-19
p-value는 0.0000로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

예제(11.6)

시멘트 분쇄공정에서 시멘트 강도에 영향을 주는 여러 요인 중에서 우선적으로 석고의 종류(A) 와 석고첨가량으로 사용되는 SO(3)함량 (B)가 어떤 영향을 주는 지 실험한 결과가 다음과 같았을 때, 석고 종류의 효과, 첨가량에 대한 효과가 있는지와 석고의 종류와 첨가량 사이에 교호작용의 효과가 있는지를 검정하시오

[조건]
1. 유의수준은 0.05로 한다.

```
import pandas as pd
from scipy import stats

data = {'A1': [607, 672, 730, 746, 749, 698],
        'A2': [647, 698, 650, 660, 657, 618],
        'A3': [642, 686, 674, 696, 700, 658]}

df = pd.DataFrame(data, columns=['A1', 'A2', 'A3'], index=['B1', 'B2', 'B3', 'B4', 'B5', 'B6'])
```

```
f_value1, p_value1 = stats.f_oneway(df['A1'], df['A2'], df['A3'])

# 귀무가설: 석고의 종류에 따른 시멘트 강도의 차이가 없다.
# 대립가설: 석고의 종류에 따른 시멘트 강도의 차이가 있다.

print(f'P-value : {p_value1}')

alpha = 0.05
if p_value1 < alpha:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

f_value2, p_value2 = stats.f_oneway(df.loc['B1'], df.loc['B2'], df.loc['B3'], df.loc['B4'], df.loc['B5'], df.loc['B6'])

# 귀무가설: 첨가량에 따른 시멘트 강도의 차이가 없다.
# 대립가설: 첨가량에 따른 시멘트 강도의 차이가 있다.

print(f'P-value SO(3)): {p_value2}')

alpha = 0.05
if p_value2 < alpha:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

_, p_value3 = stats.f_oneway(df['A1'], df['A2'], df['A3'], df.loc['B1'], df.loc['B2'], df.loc['B3'], df.loc['B4'], df.loc['B5'], df.loc['B6'])

# 귀무가설: 석고의 종류와 첨가량 사이에 교호작용의 효과가 없다.
# 대립가설: 석고의 종류와 첨가량 사이에 교호작용의 효과가 있다.

print(f'\nP-value : {p_value3}')

alpha = 0.05
if p_value3 < alpha:
```

```
    print(f"p-value는 {p_value3:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value3:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
F-value : 2.232521526796042
P-value : 0.14166715686164683
p-value는 0.1417로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

F-value SO(3)): 1.682756296857889
P-value SO(3)): 0.2130940316327734
p-value는 0.2131로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

P-value : 0.17321653654926122
p-value는 0.1732로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

연습문제 1

세 공정에서 생산된 철선의 인장강도 차이를 알아보기 위해 공정 1에서 4회, 공정 2에서 5회, 공정 3에서 6회 총 15회의 랜덤 측정을 진행한 후 얻은 인장강도 결과가 다음과 같을 때, 일원배치법의 모형을 적용해 공정에 따라 인장강도에 차가 있다고 할 수 있는지를 유의수준 5%에서 검정하시오.

```
import pandas as pd
from scipy import stats

process1 = [2, 3, 4, 5]
process2 = [4, 5, 6, 4, 3]
process3 = [6, 5, 7, 4, 6, 8]

df = pd.DataFrame(zip(process1, process2, process3), columns=['Process 1', 'Process 2', 'Process 3'])

f_value, p_value = stats.f_oneway(df['Process 1'].dropna(), df['Process 2'].dropna(), df['Process 3'].dropna())

print(f'F-value: {f_value}')
```

```
print(f'P-value: {p_value}\n')

alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

F-value: 2.882352941176471
P-value: 0.10779030282150491

p-value는 0.1078로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

```
# 연습문제 5

# 생산 공장에서 5명의 기능공 (b1~b5)가 4대의 기계 (a1~4)를 하루씩 이용하여 생산한 제품의 양을 조사한 결과이다. 제품을 생산하는데 기능공 사이에 효과가 다른지, 기계들의 효과가 다른지 검정하시오

import pandas as pd
from scipy import stats

data = {'b1': [90, 92, 95, 98],
        'b2': [98, 92, 93, 96],
        'b3': [99, 93, 91, 97],
        'b4': [100, 94, 96, 93],
        'b5': [96, 98, 90, 99]}

df = pd.DataFrame(data, columns=['b1', 'b2', 'b3', 'b4', 'b5'], index=['a1', 'a2', 'a3', 'a4'])

f_value1, p_value1 = stats.f_oneway(df['b1'], df['b2'], df['b3'], df['b4'], df['b5'])


print('1. 제품을 생산하는데 기능공 사이에 효과가 다른지 검정하시오')
print(f'F-value (workers): {f_value1}')
print(f'P-value (workers): {p_value1}')
```

```
alpha = 0.05
if p_value1 < alpha:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value1:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

f_value2, p_value2 = stats.f_oneway(df.loc['a2'], df.loc['a3'], df.loc['a4'])

print('\n2. 기계들의 효과가 다른지 검정하시오')
print(f'F-value (machines): {f_value2}')
print(f'P-value (machines): {p_value2}')

alpha = 0.05
if p_value2 < alpha:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value2:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
1. 제품을 생산하는데 기능공 사이에 효과가 다른지 검정하시오
F-value (workers): 0.2330508474576271
P-value (workers): 0.9154298052668637
p-value는 0.9154로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

2. 기계들의 효과가 다른지 검정하시오
F-value (machines): 2.9777777777777774
P-value (machines): 0.08910342424372679
p-value는 0.0891로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음
```

연습문제 7

접차된 레고(플라스틱) 조각들을 떼어놓는데 필요한 힘을 결정하기 위하여 연구가 이루어졌을 때, 네 가지의 습도를 사용하여 플라스틱의 세 가지 유형을 시험한 결과가 다음과 같다면. 이를 이용하여 분산분석표를 작성하고 결과를 설명하시오.

```
from scipy import stats
```

```
from math import sqrt

data = [[39.0, 33.1, 33.8, 33.0],
        [36.9, 27.2, 29.7, 28.5],
        [27.4, 29.2, 26.7, 30.9]]

n_rows = len(data)
n_cols = len(data[0])
n_total = n_rows * n_cols

grand_mean = sum([sum(row) for row in data]) / n_total
ss_humidity = sum([((sum(data[i]) / n_cols) - grand_mean) ** 2 for i in range(n_rows)]) * n_cols
ss_plastic_type = sum([((sum([data[i][j] for i in range(n_rows)]) / n_rows) - grand_mean) ** 2 for j in range(n_cols)]) * n_rows
ss_total = sum([(data[i][j] - grand_mean) ** 2 for i in range(n_rows) for j in range(n_cols)])
ss_error = ss_total - ss_humidity - ss_plastic_type
df_humidity = n_rows - 1
df_plastic_type = n_cols - 1
df_error = df_humidity * df_plastic_type
df_total = n_total - 1
ms_humidity = ss_humidity / df_humidity
ms_plastic_type = ss_plastic_type / df_plastic_type
ms_error = ss_error / df_error
f_humidity = ms_humidity / ms_error
f_plastic_type = ms_plastic_type / ms_error
p_humidity = stats.f.sf(f_humidity, df_humidity, df_error)
p_plastic_type = stats.f.sf(f_plastic_type, df_plastic_type, df_error)

print('요인\t\t제곱합\t\t자유도\t\t평균제곱\t\tF-통계량\t\tP-value')
print('A\t\t{:.3f}\t\t{}\t\t{:.3f}\t\t{:.3f}\t\t{:.3f}'.format(ss_humidity, df_humidity, ms_humidity, f_humidity, p_humidity))
print('B\t\t{:.3f}\t\t{}\t\t{:.3f}\t\t{:.3f}\t\t{:.3f}'.format(ss_plastic_type, df_plastic_type, ms_plastic_type, f_plastic_type, p_plastic_type))
print('오차\t\t{:.3f}\t\t{}\t\t{:.3f}'.format(ss_error, df_error, ms_error))
print('총합\t\t{:.3f}\t\t{}'.format(ss_total, df_total))
```

```
요인            제곱합          자유도          평균제곱                F-통계량
        P-value
A               79.272          2               39.636          4.692
```

0.059				
B	41.217	3	13.739	1.626
0.280				
오차	50.688	6	8.448	
총합	171.177	11		

연습문제 9

네 종류 기계와 세 사람의 기능공이 생산하는 제품의 생산량을 3회 반복하여 측정한 자료는 다음과 같다. 이때 [조건]을 유의수준 5%에서 검정하시오.

[조건]
1. 세 기능공의 능력은 같은가?
2. 네 종류의 기계의 성능은 같은가?
3. 네 종류의 기계와 세 기능공의 상호작용(교호작용)이 있는가?

```
import pandas as pd
from scipy import stats

data = {'1': [47, 58, 53], '2': [47, 64, 65], '3': [33, 42, 29], '4': [39, 52, 50]}
df = pd.DataFrame(data, index=['A', 'B', 'C'])

print('1. 세 기능공의 능력은 같은가?')
fvalue, p_value = stats.f_oneway(df.loc['A'], df.loc['B'], df.loc['C'])
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")

print('\n2 네 종류의 기계의 성능은 같은가?')
fvalue, p_value = stats.f_oneway(df['1'], df['2'], df['3'], df['4'])
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

```
print('\n3. 네 종류의 기계와 세 기능공의 상호작용(교호작용)이 있는가?')
_, p_value = stats.friedmanchisquare(df.loc['A'], df.loc['B'], df.loc['C'])
alpha = 0.05
if p_value < alpha:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 작다.\n따라서 귀무 가설을 기각한다.")
else:
    print(f"p-value는 {p_value:.4f}로, 유의 수준 {alpha}보다 크거나 같다.\n따라서 귀무 가설을 기각할 수 없음")
```

1. 세 기능공의 능력은 같은가?
p-value는 0.3113로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음

2 네 종류의 기계의 성능은 같은가?
p-value는 0.0234로, 유의 수준 0.05보다 작다.
따라서 귀무 가설을 기각한다.

3. 네 종류의 기계와 세 기능공의 상호작용(교호작용)이 있는가?
p-value는 0.1054로, 유의 수준 0.05보다 크거나 같다.
따라서 귀무 가설을 기각할 수 없음