

TUGAS KELOMPOK

FUZZY LOGIC



DISUSUN OLEH :

- | | |
|------------------------|-------------|
| 1. Davi Sulaiman | (G1A022001) |
| 2. Rafi Afrian | (G1A022033) |
| 3. Fahim Ahmad Saputra | (G1A022037) |
| 4. Ade Irawan | (G1A022083) |

Dosen Pengampu :

Dr. Endina Putri Purwandari, S.T., M.Kom.

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS BENGKULU
2024

Setiap kelompok Menyusun kode pemrograman dengan Bahasa Phyton Untuk:

- 1) Kelompok 1, 2 → kerjakan dengan fungsi keanggotaan SEGITIGA dan FIS Mamdani
- 2) Kelompok 3,4 → kerjakan dengan fungsi keanggotaan TRAPESIUM dan FIS Sugeno
- 3) Kelompok 5, 6 → kerjakan dengan fungsi keanggotaan SEGITIGA dan FIS Mamdani
- 4) Kelompok 7, 8, 9 → kerjakan dengan fungsi keanggotaan TRAPESIUM dan FIS Sugeno

Pembahasan:

- 1) Susun/ konstruksikan kode pemrograman dengan Phyton sesuai pembagian kelompok
- 2) Analisis kode yang anda susun, tuliskan komentar dari baris kode yang dibuat
- 3) Evaluasi perbedaan fungsi keanggotaan Segitiga dan Trapesium terhadap hasil perhitungan
- 4) Evaluasi perbedaan FIS Mamdani dan FIS Sugeno terhadap hasil perhitungan
- 5) Kesimpulan, bahas hasil yang anda peroleh setelah penerapan dengan kode Phyton

Soal:

Suatu penelitian dilakukan untuk mencari jumlah produksi berdasarkan pengaruh faktor suhu, kebisingan, dan pencahayaan. Dalam penelitian ini ada 30 pekerja, yang masing-masing melakukan 27 kali percobaan dengan kombinasi suhu (°C), kebisingan (dB), dan pencahayaan (lux) yang berbeda untuk menghasilkan sejumlah produk. Banyaknya data diperoleh sejumlah 810 data. Dari ketigapuluh data untuk setiap kombinasi diambil nilai rata-ratanya, sehingga data yang akan diolah tinggal 27 data sebagai berikut :

No	Suhu (°C)	Kebisingan (dB)	Pencahayaan (lux)	Rata-rata jumlah produk	Standar deviasi
1	22	55	150	148,00	4,71
2	22	55	300	150,90	4,78
3	22	55	500	146,50	4,90
4	22	75	150	143,10	4,90
5	22	75	300	146,53	4,58
6	22	75	500	142,73	5,42
7	22	90	150	136,73	4,49
8	22	90	300	140,77	4,49
9	22	90	500	135,97	4,75
10	26	55	150	149,73	4,43
11	26	55	300	153,27	5,59
12	26	55	500	152,13	5,04
13	26	75	150	148,00	5,15
14	26	75	300	150,63	5,06
15	26	75	500	147,63	4,84
16	26	90	150	141,47	5,69
17	26	90	300	145,67	4,81
18	26	90	500	140,20	4,76
19	32	55	150	142,10	4,28
20	32	55	300	146,53	5,38
21	32	55	500	142,17	4,53
22	32	75	150	138,70	4,84
23	32	75	300	141,40	4,95
24	32	75	500	138,30	5,12
25	32	90	150	133,33	4,71
26	32	90	300	138,53	4,51
27	32	90	500	137,77	4,83

Tentukan :

- a. Fungsi Keanggotaan beserta gambarnya
- b. 27 aturan Fuzzy
- c. Derajat keanggotaan nilai tiap variable dalam setiap himpunan
- d. α -predikat untuk setiap aturan
- e. Rata-rata jumlah produk (gunakan metode defuzzy weighted average)

Pembahasan:

1.1. Fungsi keanggotaan Trapezium

a. Fungsi keanggotaan beserta gambar

Source code:

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

# Memasukkan data ke dalam DataFrame
data = {
    'Suhu (°C)': [22, 22, 22, 22, 22, 22, 22, 22, 22, 26, 26, 26, 26, 26, 26, 26, 26, 26, 32,
32, 32, 32, 32, 32, 32, 32],
    'Kebisingan (dB)': [55, 55, 55, 75, 75, 75, 90, 90, 90, 55, 55, 55, 75, 75, 75, 90, 90,
90, 55, 55, 55, 75, 75, 75, 90, 90, 90],
    'Pencahayaannya (lux)': [150, 300, 500, 150, 300, 500, 150, 300, 500, 150, 300, 500, 150, 300, 500,
150, 300, 500, 150, 300, 500, 150, 300, 500, 150, 300, 500],
    'Rata-rata jumlah produk': [148.00, 150.90, 146.50, 143.10, 146.53, 142.73, 136.73,
140.77, 135.97, 149.73, 153.27, 152.13, 148.00, 150.63, 147.63, 141.47, 145.67,
140.20, 142.10, 146.53, 142.17, 138.70, 141.40, 138.30, 133.33, 138.53, 137.77],
    'Standar deviasi': [4.71, 4.78, 4.90, 4.90, 4.58, 5.42, 4.49, 4.49, 4.75, 4.43, 5.59, 5.04,
5.15, 5.06, 4.84, 5.69, 4.81, 4.76, 4.28, 5.38, 4.53, 4.84, 4.95, 5.12, 4.71, 4.51, 4.83]
}

df = pd.DataFrame(data)

# Fungsi keanggotaan trapesium
def trapezoid(x, a, b, c, d):
    if x <= a or x >= d:
        return 0
    elif a < x < b:
        return (x - a) / (b - a)
    elif b <= x <= c:
        return 1
    elif c < x < d:
        return (d - x) / (d - c)
```

```

# Menambahkan kolom keanggotaan untuk suhu
df['Keanggotaan Suhu Rendah'] = df['Suhu (°C)'].apply(lambda x: trapezoid(x, 20, 22, 25, 28))
df['Keanggotaan Suhu Sedang'] = df['Suhu (°C)'].apply(lambda x: trapezoid(x, 25, 28, 30, 33))
df['Keanggotaan Suhu Tinggi'] = df['Suhu (°C)'].apply(lambda x: trapezoid(x, 30, 33, 35, 37))

# Menambahkan kolom keanggotaan untuk kebisingan
df['Keanggotaan Kebisingan Rendah'] = df['Kebisingan (dB)'].apply(lambda x: trapezoid(x, 50, 55, 60, 65))
df['Keanggotaan Kebisingan Sedang'] = df['Kebisingan (dB)'].apply(lambda x: trapezoid(x, 60, 65, 75, 85))
df['Keanggotaan Kebisingan Tinggi'] = df['Kebisingan (dB)'].apply(lambda x: trapezoid(x, 75, 85, 90, 95))

# Menambahkan kolom keanggotaan untuk pencahayaan
df['Keanggotaan Pencahayaan Rendah'] = df['Pencahayaan (lux)'].apply(lambda x: trapezoid(x, 100, 150, 200, 250))
df['Keanggotaan Pencahayaan Sedang'] = df['Pencahayaan (lux)'].apply(lambda x: trapezoid(x, 200, 250, 300, 350))
df['Keanggotaan Pencahayaan Tinggi'] = df['Pencahayaan (lux)'].apply(lambda x: trapezoid(x, 300, 350, 500, 600))

# Menampilkan DataFrame dengan kolom keanggotaan
print(df)

# Visualisasi fungsi keanggotaan trapesium untuk suhu, kebisingan, dan pencahayaan
x_suhu = np.linspace(20, 40, 100)
y_suhu_rendah = [trapezoid(x, 20, 22, 25, 28) for x in x_suhu]
y_suhu_sedang = [trapezoid(x, 25, 28, 30, 33) for x in x_suhu]
y_suhu_tinggi = [trapezoid(x, 30, 33, 35, 37) for x in x_suhu]

plt.figure(figsize=(10, 6))
plt.plot(x_suhu, y_suhu_rendah, label="Suhu Rendah")
plt.plot(x_suhu, y_suhu_sedang, label="Suhu Sedang")
plt.plot(x_suhu, y_suhu_tinggi, label="Suhu Tinggi")
plt.title("Fungsi Keanggotaan Trapesium untuk Suhu")
plt.xlabel("Suhu (°C)")
plt.ylabel("Derajat Keanggotaan")
plt.legend()
plt.grid(True)
plt.show()

```

Penjelasan Source code:

Implementasi ini menggunakan fungsi keanggotaan trapesium untuk mengukur derajat keanggotaan pada variabel suhu, kebisingan, dan pencahayaan. Data observasi, yang mencakup suhu dalam derajat Celcius, kebisingan dalam desibel, dan pencahayaan dalam satuan lux, diolah ke dalam sebuah DataFrame menggunakan Pandas. Empat parameter utama membentuk fungsi keanggotaan trapesium yang digambarkan di sini: a adalah titik di mana keanggotaan mulai meningkat dari nol, b adalah awal dari derajat keanggotaan penuh (1), c adalah akhir dari derajat keanggotaan penuh, dan d adalah titik di mana keanggotaan kembali ke nol. Fungsi ini mengembalikan nilai keanggotaan dari 0 hingga 1 berdasarkan nilai input, tergantung pada apakah nilai tersebut berada di luar atau di dalam batas interval yang telah ditentukan.

Untuk suhu, kebisingan, dan pencahayaan, ada tiga kategori: rendah, sedang, dan tinggi. Suhu rendah adalah 20°C hingga 28°C, suhu sedang adalah 25°C hingga 33°C, dan suhu tinggi adalah 30°C hingga 37°C. Suara diatur dalam kategori rendah (50-65 dB), sedang (60-85 dB), dan tinggi (75-95 dB). Fungsi trapezoid dihitung untuk masing-masing kategori, dan kemudian disimpan ke dalam kolom baru dalam DataFrame.

Selain itu, grafik fungsi keanggotaan trapesium untuk suhu dibuat. Grafik ini menunjukkan bagaimana setiap nilai suhu memiliki derajat keanggotaan tertentu dalam kategori rendah, sedang, dan tinggi. Ini memberikan gambaran visual tentang bagaimana fungsi keanggotaan fuzzy bekerja, dan implementasi ini dapat digunakan sebagai dasar untuk pengambilan keputusan berbasis logika fuzzy, di mana derajat keanggotaan memainkan peran dalam menentukan kategori nilai yang diberikan.

Output:

	Standar deviasi	Keanggotaan Suhu Rendah	Keanggotaan Suhu Sedang
0	4.71	1.000000	0.000000
1	4.78	1.000000	0.000000
2	4.90	1.000000	0.000000
3	4.90	1.000000	0.000000
4	4.58	1.000000	0.000000
5	5.42	1.000000	0.000000
6	4.49	1.000000	0.000000
7	4.49	1.000000	0.000000
8	4.75	1.000000	0.000000
9	4.43	0.666667	0.333333
10	5.59	0.666667	0.333333
11	5.04	0.666667	0.333333
12	5.15	0.666667	0.333333
13	5.06	0.666667	0.333333
14	4.84	0.666667	0.333333
15	5.69	0.666667	0.333333
16	4.81	0.666667	0.333333
17	4.76	0.666667	0.333333
18	4.28	0.000000	0.333333
19	5.38	0.000000	0.333333
20	4.53	0.000000	0.333333
21	4.84	0.000000	0.333333
22	4.95	0.000000	0.333333
23	5.12	0.000000	0.333333
24	4.71	0.000000	0.333333
25	4.51	0.000000	0.333333
26	4.83	0.000000	0.333333

Gambar 1.1 Output

	Keanggotaan Suhu Tinggi	Keanggotaan Kebisingan Rendah \
0	0.000000	1
1	0.000000	1
2	0.000000	1
3	0.000000	0
4	0.000000	0
5	0.000000	0
6	0.000000	0
7	0.000000	0
8	0.000000	0
9	0.000000	1
10	0.000000	1
11	0.000000	1
12	0.000000	0
13	0.000000	0
14	0.000000	0
15	0.000000	0
16	0.000000	0
17	0.000000	0
18	0.666667	1
19	0.666667	1
20	0.666667	1
21	0.666667	0
22	0.666667	0
23	0.666667	0
24	0.666667	0
25	0.666667	0
26	0.666667	0

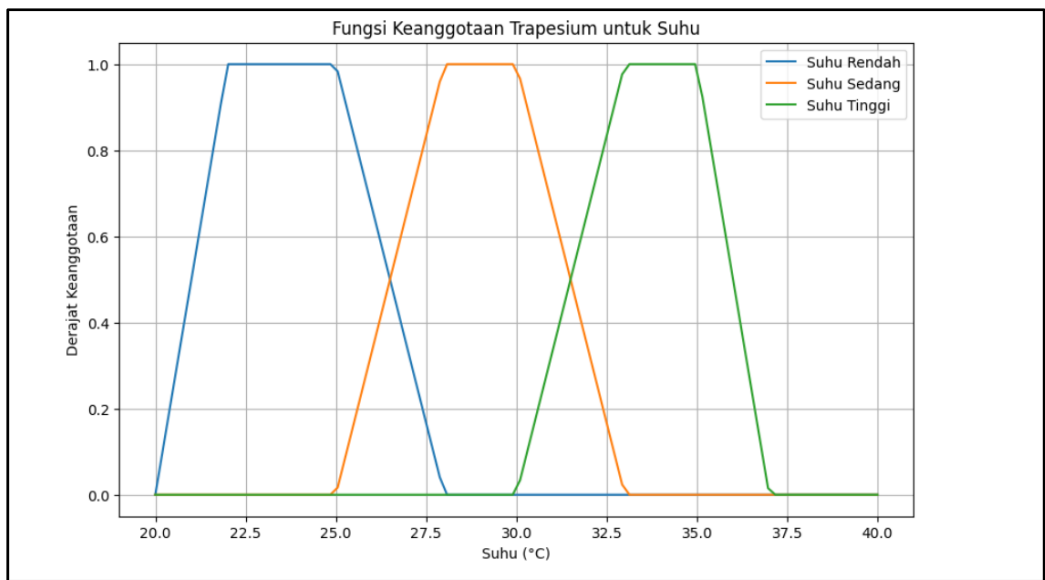
Gambar 1.2 Output

	Keanggotaan Kebisingan Sedang	Keanggotaan Kebisingan Tinggi \
0	0	0
1	0	0
2	0	0
3	1	0
4	1	0
5	1	0
6	0	1
7	0	1
8	0	1
9	0	0
10	0	0
11	0	0
12	1	0
13	1	0
14	1	0
15	0	1
16	0	1
17	0	1
18	0	0
19	0	0
20	0	0
21	1	0
22	1	0
23	1	0
24	0	1
25	0	1
26	0	1

Gambar 1.3 Output

Keanggotaan Pencahayaan Tinggi	
0	0
1	0
2	1
3	0
4	0
5	1
6	0
7	0
8	1
9	0
10	0
11	1
12	0
13	0
14	1
15	0
16	0
17	1
18	0
19	0
20	1
21	0
22	0
23	1
24	0
25	0
26	1

Gambar 1.4 Output



Gambar 1.5 Gambar Fungsi Keanggotaan Trapesium

b. 27 Aturan Fuzzy

Source code:

1. Install the required library

!pip install scikit-fuzzy

```

import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

# Variabel input (Suhu, Kebisingan, Pencapaian)
suhu = ctrl.Antecedent(np.arange(20, 40, 1), 'suhu')
kebisingan = ctrl.Antecedent(np.arange(50, 100, 1), 'kebisingan')
pencapaian = ctrl.Antecedent(np.arange(100, 600, 1), 'pencapaian')

# Variabel output (Jumlah produk)
produk = ctrl.Consequent(np.arange(130, 160, 1), 'produk')

# Membership functions (fungsi keanggotaan) untuk Suhu, Kebisingan, Pencapaian,
dan Produk
suhu['rendah'] = fuzz.trapmf(suhu.universe, [20, 22, 25, 28])
suhu['sedang'] = fuzz.trapmf(suhu.universe, [25, 28, 30, 33])
suhu['tinggi'] = fuzz.trapmf(suhu.universe, [30, 33, 35, 37])

kebisingan['rendah'] = fuzz.trapmf(kebisingan.universe, [50, 55, 60, 65])
kebisingan['sedang'] = fuzz.trapmf(kebisingan.universe, [60, 65, 75, 85])
kebisingan['tinggi'] = fuzz.trapmf(kebisingan.universe, [75, 85, 90, 95])

pencapaian['rendah'] = fuzz.trapmf(pencapaian.universe, [100, 150, 200, 250])
pencapaian['sedang'] = fuzz.trapmf(pencapaian.universe, [200, 250, 300, 350])
pencapaian['tinggi'] = fuzz.trapmf(pencapaian.universe, [300, 350, 500, 600])

produk['rendah'] = fuzz.trimf(produk.universe, [130, 135, 140])
produk['sedang'] = fuzz.trimf(produk.universe, [140, 145, 150])
produk['tinggi'] = fuzz.trimf(produk.universe, [150, 155, 160])

# Membuat aturan-aturan fuzzy
rule1 = ctrl.Rule(suhu['rendah'] & kebisingan['rendah'] & pencapaian['rendah'],
produk['rendah'])
rule2 = ctrl.Rule(suhu['rendah'] & kebisingan['rendah'] & pencapaian['sedang'],
produk['sedang'])
rule3 = ctrl.Rule(suhu['rendah'] & kebisingan['rendah'] & pencapaian['tinggi'],
produk['rendah'])
rule4 = ctrl.Rule(suhu['rendah'] & kebisingan['sedang'] & pencapaian['rendah'],
produk['rendah'])
rule5 = ctrl.Rule(suhu['rendah'] & kebisingan['sedang'] & pencapaian['sedang'],
produk['sedang'])
rule6 = ctrl.Rule(suhu['rendah'] & kebisingan['sedang'] & pencapaian['tinggi'],
produk['rendah'])
rule7 = ctrl.Rule(suhu['rendah'] & kebisingan['tinggi'] & pencapaian['rendah'],
produk['rendah'])

```



```
rule8 = ctrl.Rule(suhu['rendah'] & kebisingan['tinggi'] & pencahayaan['sedang'],  
produk['sedang'])  
rule9 = ctrl.Rule(suhu['rendah'] & kebisingan['tinggi'] & pencahayaan['tinggi'],  
produk['rendah'])
```

```
rule10 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaan['rendah'],  
produk['sedang'])  
rule11 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaan['sedang'],  
produk['tinggi'])  
rule12 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaan['tinggi'],  
produk['sedang'])  
rule13 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaan['rendah'],  
produk['sedang'])  
rule14 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaan['sedang'],  
produk['tinggi'])  
rule15 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaan['tinggi'],  
produk['sedang'])  
rule16 = ctrl.Rule(suhu['sedang'] & kebisingan['tinggi'] & pencahayaan['rendah'],  
produk['sedang'])  
rule17 = ctrl.Rule(suhu['sedang'] & kebisingan['tinggi'] & pencahayaan['sedang'],  
produk['tinggi'])  
rule18 = ctrl.Rule(suhu['sedang'] & kebisingan['tinggi'] & pencahayaan['tinggi'],  
produk['sedang'])
```

```
rule19 = ctrl.Rule(suhu['tinggi'] & kebisingan['rendah'] & pencahayaan['rendah'],  
produk['rendah'])  
rule20 = ctrl.Rule(suhu['tinggi'] & kebisingan['rendah'] & pencahayaan['sedang'],  
produk['sedang'])  
rule21 = ctrl.Rule(suhu['tinggi'] & kebisingan['rendah'] & pencahayaan['tinggi'],  
produk['rendah'])  
rule22 = ctrl.Rule(suhu['tinggi'] & kebisingan['sedang'] & pencahayaan['rendah'],  
produk['rendah'])  
rule23 = ctrl.Rule(suhu['tinggi'] & kebisingan['sedang'] & pencahayaan['sedang'],  
produk['sedang'])  
rule24 = ctrl.Rule(suhu['tinggi'] & kebisingan['sedang'] & pencahayaan['tinggi'],  
produk['rendah'])  
rule25 = ctrl.Rule(suhu['tinggi'] & kebisingan['tinggi'] & pencahayaan['rendah'],  
produk['rendah'])  
rule26 = ctrl.Rule(suhu['tinggi'] & kebisingan['tinggi'] & pencahayaan['sedang'],  
produk['sedang'])  
rule27 = ctrl.Rule(suhu['tinggi'] & kebisingan['tinggi'] & pencahayaan['tinggi'],  
produk['rendah'])
```

```
# Menggabungkan aturan ke dalam sistem kontrol fuzzy  
produk_ctrl = ctrl.ControlSystem([rule1, rule2, rule3, rule4, rule5, rule6, rule7, rule8,  
rule9, rule10,
```

```

rule11, rule12, rule13, rule14, rule15, rule16, rule17, rule18,
rule19, rule20, rule21, rule22, rule23, rule24, rule25, rule26,
rule27])

# Simulasi sistem kontrol fuzzy
produk_simulasi = ctrl.ControlSystemSimulation(produk_ctrl)

# Contoh input: Suhu = 26, Kebisingan = 75, Pencahayaan = 300
produk_simulasi.input['suhu'] = 26
produk_simulasi.input['kebisingan'] = 75
produk_simulasi.input['pencahayaan'] = 300

# Menjalankan simulasi
produk_simulasi.compute()

# Output prediksi jumlah produk
print(f"Prediksi jumlah produk: {produk_simulasi.output['produk']:.2f}")

```

Penjelasan Source Code:

Dalam kode ini, sistem kontrol fuzzy menggunakan 27 aturan fuzzy untuk memprediksi jumlah produk berdasarkan tiga variabel masukan: suhu, kebisingan, dan pencahayaan. Variabel masukan dalam tingkat keanggotaan adalah "rendah", "sedang", dan "tinggi", dengan suhu 20 °C hingga 37°C, dan kebisingan 50 hingga 95 dB, dan pencahayaan 100 hingga 600 lux. Variabel keluaran, yaitu jumlah produk.

Aturan fuzzy ini menunjukkan bagaimana variabel masukan dan keluaran berhubungan satu sama lain. Sebagai contoh, aturan 1 menyatakan bahwa jika suhu, kebisingan, dan pencahayaan rendah, maka jumlah produk akan rendah (aturan 14). Sebaliknya, jika suhu, kebisingan, dan pencahayaan sedang, maka jumlah produk akan tinggi. Dalam situasi tertentu, sistem menghasilkan hasil menggunakan kombinasi aturan ini. Dalam simulasi contoh dengan suhu 26°C, kebisingan 75 dB, dan pencahayaan 300 lux, sistem memperkirakan jumlah produk sekitar 145,85 unit, yang termasuk dalam kategori "sedang". Prediksi ini diperoleh melalui perhitungan yang menggunakan aturan sistem fuzzy.

Output:

```

Collecting scikit-fuzzy
  Downloading scikit_fuzzy-0.5.0-py2.py3-none-any.whl.metadata (2.6 kB)
  Downloading scikit_fuzzy-0.5.0-py2.py3-none-any.whl (920 kB)
    ----- 920.8/920.8 kB 14.4 MB/s eta 0:00:00
Installing collected packages: scikit-fuzzy
Successfully installed scikit-fuzzy-0.5.0
Prediksi jumlah produk: 148.70

```

Gambar 2.1 Output

c. Derajat Keanggotaan Nilai Setiap Variable Dalam Setiap Himpunan

Source Code:

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

# Definisi variabel input
suhu = np.arange(20, 40, 1)
kebisingan = np.arange(50, 100, 1)
pencahayaan = np.arange(100, 600, 1)

# Fungsi keanggotaan trapesium untuk setiap variabel
suhu_rendah = fuzz.trapmf(suhu, [20, 22, 25, 28])
suhu_sedang = fuzz.trapmf(suhu, [25, 28, 30, 33])
suhu_tinggi = fuzz.trapmf(suhu, [30, 33, 35, 37])

kebisingan_rendah = fuzz.trapmf(kebisingan, [50, 55, 60, 65])
kebisingan_sedang = fuzz.trapmf(kebisingan, [60, 65, 75, 85])
kebisingan_tinggi = fuzz.trapmf(kebisingan, [75, 85, 90, 95])

pencahayaan_rendah = fuzz.trapmf(pencahayaan, [100, 150, 200, 250])
pencahayaan_sedang = fuzz.trapmf(pencahayaan, [200, 250, 300, 350])
pencahayaan_tinggi = fuzz.trapmf(pencahayaan, [300, 350, 500, 600])

# Contoh input: Suhu = 26, Kebisingan = 75, Pencahayaan = 300
nilai_suhu = 26
nilai_kebisingan = 75
nilai_pencahayaan = 300

# Hitung derajat keanggotaan untuk setiap variabel
suhu_rendah_deg = fuzz.interp_membership(suhu, suhu_rendah, nilai_suhu)
suhu_sedang_deg = fuzz.interp_membership(suhu, suhu_sedang, nilai_suhu)
suhu_tinggi_deg = fuzz.interp_membership(suhu, suhu_tinggi, nilai_suhu)

kebisingan_rendah_deg = fuzz.interp_membership(kebisingan, kebisingan_rendah,
nilai_kebisingan)
kebisingan_sedang_deg = fuzz.interp_membership(kebisingan, kebisingan_sedang,
nilai_kebisingan)
kebisingan_tinggi_deg = fuzz.interp_membership(kebisingan, kebisingan_tinggi,
nilai_kebisingan)

pencahayaan_rendah_deg = fuzz.interp_membership(pencahayaan,
pencahayaan_rendah, nilai_pencahayaan)
pencahayaan_sedang_deg = fuzz.interp_membership(pencahayaan,
pencahayaan_sedang, nilai_pencahayaan)
```

```
pencahayaan_tinggi_deg = fuzz.interp_membership(pencahayaan,
pencahayaan_tinggi, nilai_pencahayaan)
```

```
# Cetak derajat keanggotaan untuk setiap variabel
```

```
print(f"Derajat keanggotaan Suhu (26°C):")
```

```
print(f" Rendah: {suhu_rendah_deg:.2f}")
```

```
print(f" Sedang: {suhu_sedang_deg:.2f}")
```

```
print(f" Tinggi: {suhu_tinggi_deg:.2f}")
```

```
print(f"\nDerajat keanggotaan Kebisingan (75 dB):")
```

```
print(f" Rendah: {kebisingan_rendah_deg:.2f}")
```

```
print(f" Sedang: {kebisingan_sedang_deg:.2f}")
```

```
print(f" Tinggi: {kebisingan_tinggi_deg:.2f}")
```

```
print(f"\nDerajat keanggotaan Pencahayaan (300 lux):")
```

```
print(f" Rendah: {pencahayaan_rendah_deg:.2f}")
```

```
print(f" Sedang: {pencahayaan_sedang_deg:.2f}")
```

```
print(f" Tinggi: {pencahayaan_tinggi_deg:.2f}")
```

```
# Plot Fungsi Keanggotaan untuk Suhu
```

```
plt.figure(figsize=(10, 6))
```

```
plt.plot(suhu, suhu_rendah, 'b', label='Rendah')
```

```
plt.plot(suhu, suhu_sedang, 'g', label='Sedang')
```

```
plt.plot(suhu, suhu_tinggi, 'r', label='Tinggi')
```

```
plt.title('Fungsi Keanggotaan Suhu')
```

```
plt.ylabel('Derajat Keanggotaan')
```

```
plt.xlabel('Suhu (°C)')
```

```
plt.legend()
```

```
plt.show()
```

```
# Plot Fungsi Keanggotaan untuk Kebisingan
```

```
plt.figure(figsize=(10, 6))
```

```
plt.plot(kebisingan, kebisingan_rendah, 'b', label='Rendah')
```

```
plt.plot(kebisingan, kebisingan_sedang, 'g', label='Sedang')
```

```
plt.plot(kebisingan, kebisingan_tinggi, 'r', label='Tinggi')
```

```
plt.title('Fungsi Keanggotaan Kebisingan')
```

```
plt.ylabel('Derajat Keanggotaan')
```

```
plt.xlabel('Kebisingan (dB)')
```

```
plt.legend()
```

```
plt.show()
```

```
# Plot Fungsi Keanggotaan untuk Pencahayaan
```

```
plt.figure(figsize=(10, 6))
```

```
plt.plot(pencahayaan, pencahayaan_rendah, 'b', label='Rendah')
```

```
plt.plot(pencahayaan, pencahayaan_sedang, 'g', label='Sedang')
```

```
plt.plot(pencahayaan, pencahayaan_tinggi, 'r', label='Tinggi')
```

```
plt.title('Fungsi Keanggotaan Pencahayaan')
plt.ylabel('Derajat Keanggotaan')
plt.xlabel('Pencahayaan (lux)')
plt.legend()
plt.show()
```

Penjelasan Source code:

Logika fuzzy digunakan untuk mengolah tiga variabel: pencahayaan, kebisingan, dan suhu. Pertama, pustaka yang diperlukan diimpor: numpy untuk manipulasi array, skfuzzy untuk menghitung fungsi keanggotaan fuzzy, dan matplotlib.pyplot untuk menampilkan grafik.

Selanjutnya, variabel input diberi nilai khusus: suhu (20 °C hingga 40°C), kebisingan (50 °B hingga 100 °B), dan pencahayaan (100 x 600 lux). Untuk setiap variabel, ada tiga fungsi keanggotaan: rendah, sedang, dan tinggi. Fungsi keanggotaan ini ditentukan dengan fungsi fuzz.trapmf() dari pustaka skfuzzy.

Kode ini kemudian menghitung derajat keanggotaan contoh berikut: suhu = 26°C, suara = 75 dB, dan pencahayaan = 300 lux. Dengan menggunakan fungsi fuzz.interp_membership(), derajat keanggotaan untuk masing-masing kategori (rendah, sedang, dan tinggi) dihitung. Nilai untuk suhu adalah rendah (0,67), sedang (0,33), dan tinggi (0,00), dan untuk variabel kebisingan dan pencahayaan juga sama.

Kode ini menampilkan hasil penghitungan derajat keanggotaan di layar. Setiap variabel yang dimasukkan juga divisualisasikan sebagai grafik fungsi keanggotaan. Keanggotaan untuk suhu, kebisingan, dan pencahayaan dibagi menjadi kategori rendah, sedang, dan tinggi di grafik ini.

Oleh karena itu, kode ini menggunakan sistem logika fuzzy sederhana untuk melihat tiga variabel lingkungan dengan input yang diberikan dan menunjukkan fungsi keanggotaannya.

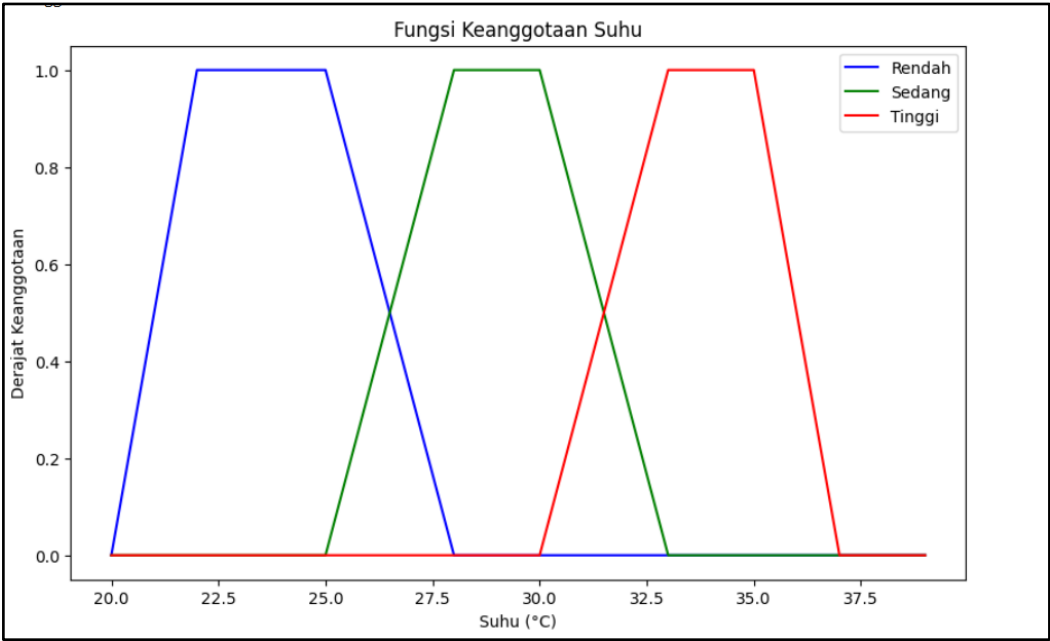
Output:

```
Derajat keanggotaan Suhu (26°C):
Rendah: 0.67
Sedang: 0.33
Tinggi: 0.00

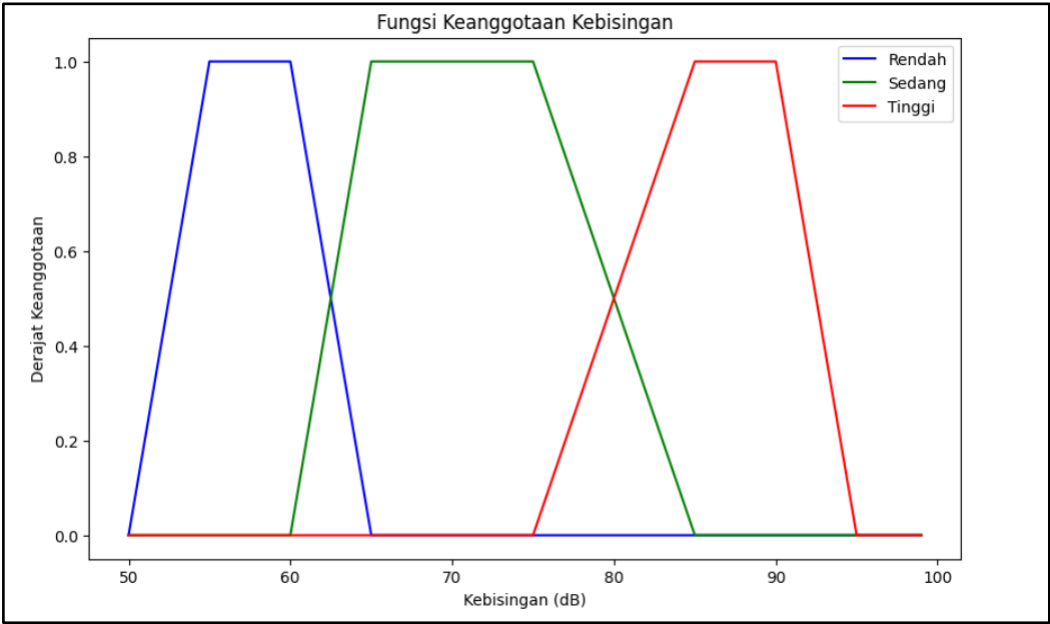
Derajat keanggotaan Kebisingan (75 dB):
Rendah: 0.00
Sedang: 1.00
Tinggi: 0.00

Derajat keanggotaan Pencahayaan (300 lux):
Rendah: 0.00
Sedang: 1.00
Tinggi: 0.00
```

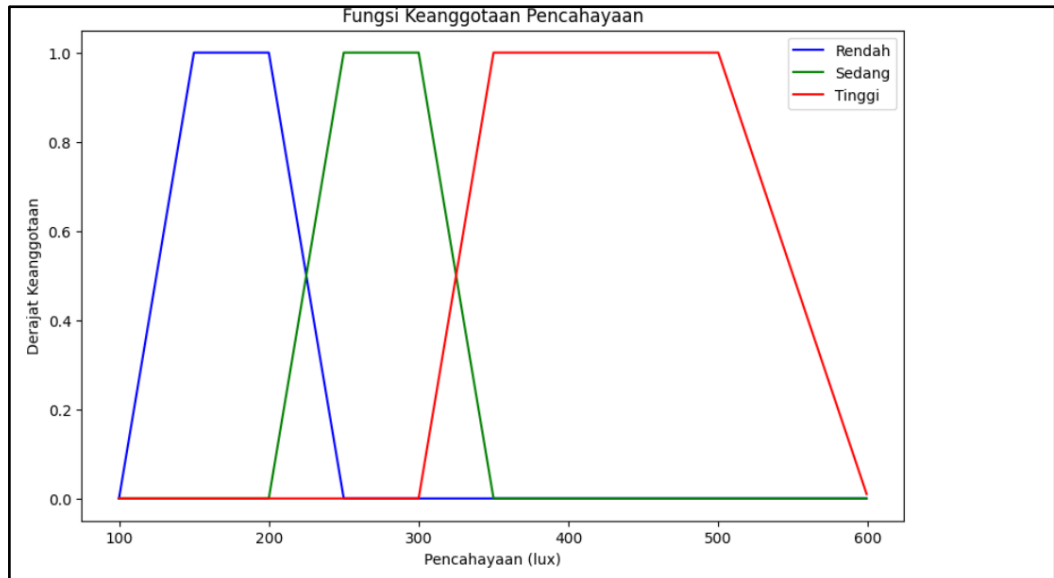
Gambar 3.1 Output



Gambar 3.2 Output



Gambar 3.3 Output



3.4 Output

d. α -perdikat untuk setiap aturan

Source Code:

```
import numpy as np
import skfuzzy as fuzz
```

```
# Definisikan fungsi keanggotaan yang telah ditentukan sebelumnya
```

```
suhu = np.arange(20, 40, 1)
```

```
kebisingan = np.arange(50, 100, 1)
```

```
pencahayaan = np.arange(100, 600, 1)
```

```
# Fungsi keanggotaan trapezoidal untuk suhu
```

```
suhu_rendah = fuzz.trapmf(suhu, [20, 22, 25, 28])
```

```
suhu_sedang = fuzz.trapmf(suhu, [25, 28, 30, 33])
```

```
suhu_tinggi = fuzz.trapmf(suhu, [30, 33, 35, 37])
```

```
# Fungsi keanggotaan trapezoidal untuk kebisingan
```

```
kebisingan_rendah = fuzz.trapmf(kebisingan, [50, 55, 60, 65])
```

```
kebisingan_sedang = fuzz.trapmf(kebisingan, [60, 65, 75, 85])
```

```
kebisingan_tinggi = fuzz.trapmf(kebisingan, [75, 85, 90, 95])
```

```
# Fungsi keanggotaan trapezoidal untuk pencahayaan
```

```
pencahayaan_rendah = fuzz.trapmf(pencahayaan, [100, 150, 200, 250])
```

```
pencahayaan_sedang = fuzz.trapmf(pencahayaan, [200, 250, 300, 350])
```

```
pencahayaan_tinggi = fuzz.trapmf(pencahayaan, [300, 350, 500, 600])
```

```
# Contoh input: Suhu = 26, Kebisingan = 75, Pencahayaan = 300
```

```
nilai_suhu = 26
```

```
nilai_kebisingan = 75
```

```
nilai_pencahayaan = 300
```

```

# Hitung derajat keanggotaan untuk setiap variabel
suhu_rendah_deg = fuzz.interp_membership(suhu, suhu_rendah, nilai_suhu)
suhu_sedang_deg = fuzz.interp_membership(suhu, suhu_sedang, nilai_suhu)
suhu_tinggi_deg = fuzz.interp_membership(suhu, suhu_tinggi, nilai_suhu)

kebisingan_rendah_deg = fuzz.interp_membership(kebisingan, kebisingan_rendah,
nilai_kebisingan)
kebisingan_sedang_deg = fuzz.interp_membership(kebisingan, kebisingan_sedang,
nilai_kebisingan)
kebisingan_tinggi_deg = fuzz.interp_membership(kebisingan, kebisingan_tinggi,
nilai_kebisingan)

pencahayaannya_rendah_deg = fuzz.interp_membership(pencahayaannya,
pencahayaannya_rendah, nilai_pencahayaannya)
pencahayaannya_sedang_deg = fuzz.interp_membership(pencahayaannya,
pencahayaannya_sedang, nilai_pencahayaannya)
pencahayaannya_tinggi_deg = fuzz.interp_membership(pencahayaannya,
pencahayaannya_tinggi, nilai_pencahayaannya)

# Hitung a-predikat untuk semua 27 aturan
a_predikats = []

# Aturan-aturan fuzzy
# Misal aturan pertama: Suhu rendah, Kebisingan rendah, Pencahayaannya rendah
a_predikat_1 = min(suhu_rendah_deg, kebisingan_rendah_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_1)

# Suhu rendah, Kebisingan rendah, Pencahayaannya sedang
a_predikat_2 = min(suhu_rendah_deg, kebisingan_rendah_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_2)

# Suhu rendah, Kebisingan rendah, Pencahayaannya tinggi
a_predikat_3 = min(suhu_rendah_deg, kebisingan_rendah_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_3)

# Suhu rendah, Kebisingan sedang, Pencahayaannya rendah
a_predikat_4 = min(suhu_rendah_deg, kebisingan_sedang_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_4)

# Suhu rendah, Kebisingan sedang, Pencahayaannya sedang
a_predikat_5 = min(suhu_rendah_deg, kebisingan_sedang_deg,
pencahayaannya_sedang_deg)

```



```

a_predikats.append(a_predikat_5)

# Suhu rendah, Kebisingan sedang, Pencahayaan tinggi
a_predikat_6      =      min(suhu_rendah_deg,      kebisingan_sedang_deg,
pencahayaaaan_tinggi_deg)
a_predikats.append(a_predikat_6)

# Suhu rendah, Kebisingan tinggi, Pencahayaan rendah
a_predikat_7      =      min(suhu_rendah_deg,      kebisingan_tinggi_deg,
pencahayaaaan_rendah_deg)
a_predikats.append(a_predikat_7)

# Suhu rendah, Kebisingan tinggi, Pencahayaan sedang
a_predikat_8      =      min(suhu_rendah_deg,      kebisingan_tinggi_deg,
pencahayaaaan_sedang_deg)
a_predikats.append(a_predikat_8)

# Suhu rendah, Kebisingan tinggi, Pencahayaan tinggi
a_predikat_9      =      min(suhu_rendah_deg,      kebisingan_tinggi_deg,
pencahayaaaan_tinggi_deg)
a_predikats.append(a_predikat_9)

# Suhu sedang, Kebisingan rendah, Pencahayaan rendah
a_predikat_10     =      min(suhu_sedang_deg,      kebisingan_rendah_deg,
pencahayaaaan_rendah_deg)
a_predikats.append(a_predikat_10)

# Suhu sedang, Kebisingan rendah, Pencahayaan sedang
a_predikat_11     =      min(suhu_sedang_deg,      kebisingan_rendah_deg,
pencahayaaaan_sedang_deg)
a_predikats.append(a_predikat_11)

# Suhu sedang, Kebisingan rendah, Pencahayaan tinggi
a_predikat_12     =      min(suhu_sedang_deg,      kebisingan_rendah_deg,
pencahayaaaan_tinggi_deg)
a_predikats.append(a_predikat_12)

# Suhu sedang, Kebisingan sedang, Pencahayaan rendah
a_predikat_13     =      min(suhu_sedang_deg,      kebisingan_sedang_deg,
pencahayaaaan_rendah_deg)
a_predikats.append(a_predikat_13)

# Suhu sedang, Kebisingan sedang, Pencahayaan sedang
a_predikat_14     =      min(suhu_sedang_deg,      kebisingan_sedang_deg,
pencahayaaaan_sedang_deg)
a_predikats.append(a_predikat_14)

```

# Suhu sedang, Kebisingan sedang, Pencahayaan tinggi a_predikat_15 = min(suhu_sedang_deg, pencahayaan_tinggi_deg) a_predikats.append(a_predikat_15)	kebisingan_sedang_deg,
# Suhu sedang, Kebisingan tinggi, Pencahayaan rendah a_predikat_16 = min(suhu_sedang_deg, pencahayaan_rendah_deg) a_predikats.append(a_predikat_16)	kebisingan_tinggi_deg,
# Suhu sedang, Kebisingan tinggi, Pencahayaan sedang a_predikat_17 = min(suhu_sedang_deg, pencahayaan_sedang_deg) a_predikats.append(a_predikat_17)	kebisingan_tinggi_deg,
# Suhu sedang, Kebisingan tinggi, Pencahayaan tinggi a_predikat_18 = min(suhu_sedang_deg, pencahayaan_tinggi_deg) a_predikats.append(a_predikat_18)	kebisingan_tinggi_deg,
# Suhu tinggi, Kebisingan rendah, Pencahayaan rendah a_predikat_19 = min(suhu_tinggi_deg, pencahayaan_rendah_deg) a_predikats.append(a_predikat_19)	kebisingan_rendah_deg,
# Suhu tinggi, Kebisingan rendah, Pencahayaan sedang a_predikat_20 = min(suhu_tinggi_deg, pencahayaan_sedang_deg) a_predikats.append(a_predikat_20)	kebisingan_rendah_deg,
# Suhu tinggi, Kebisingan rendah, Pencahayaan tinggi a_predikat_21 = min(suhu_tinggi_deg, pencahayaan_tinggi_deg) a_predikats.append(a_predikat_21)	kebisingan_rendah_deg,
# Suhu tinggi, Kebisingan sedang, Pencahayaan rendah a_predikat_22 = min(suhu_tinggi_deg, pencahayaan_rendah_deg) a_predikats.append(a_predikat_22)	kebisingan_sedang_deg,
# Suhu tinggi, Kebisingan sedang, Pencahayaan sedang a_predikat_23 = min(suhu_tinggi_deg, pencahayaan_sedang_deg) a_predikats.append(a_predikat_23)	kebisingan_sedang_deg,

```

# Suhu tinggi, Kebisingan sedang, Pencahayaan tinggi
a_predikat_24 = min(suhu_tinggi_deg, kebisingan_sedang_deg,
                    pencahayaan_tinggi_deg)
a_predikats.append(a_predikat_24)

# Suhu tinggi, Kebisingan tinggi, Pencahayaan rendah
a_predikat_25 = min(suhu_tinggi_deg, kebisingan_tinggi_deg,
                    pencahayaan_rendah_deg)
a_predikats.append(a_predikat_25)

# Suhu tinggi, Kebisingan tinggi, Pencahayaan sedang
a_predikat_26 = min(suhu_tinggi_deg, kebisingan_tinggi_deg,
                    pencahayaan_sedang_deg)
a_predikats.append(a_predikat_26)

# Suhu tinggi, Kebisingan tinggi, Pencahayaan tinggi
a_predikat_27 = min(suhu_tinggi_deg, kebisingan_tinggi_deg,
                    pencahayaan_tinggi_deg)
a_predikats.append(a_predikat_27)

# Cetak semua a-predikat untuk aturan 1 hingga 27
for i, a_predikat in enumerate(a_predikats, start=1):
    print(f"A-predikat aturan {i}: {a_predikat:.2f}")

```

Penjelasan Source code:

Kode ini menggunakan logika fuzzy untuk menghitung a-predikat dari kombinasi tiga variabel input: suhu, kebisingan, dan pencahayaan. Variabel-variabel ini diklasifikasikan menjadi kategori rendah, sedang, dan tinggi. Batasan-batasan masing-masing kategori ditentukan dengan menggunakan fungsi keanggotaan trapezoidal, yang diterapkan pada rentang suhu (20°C hingga 40°C), kebisingan (50 dB hingga 100 dB), dan pencahayaan (100 lux hingga 600 lux). Selanjutnya, kode

Kode ini menghitung a-predikat untuk 27 aturan fuzzy setelah menghitung derajat keanggotaannya. A-predikat untuk aturan pertama yang menggabungkan suhu rendah, kebisingan rendah, dan pencahayaan rendah adalah nilai minimum dari ketiga derajat keanggotaan tersebut. Prosedur ini diulangi untuk setiap kombinasi aturan yang menggabungkan kategori r

Output :

```

A-predikat aturan 1: 0.00
A-predikat aturan 2: 0.00
A-predikat aturan 3: 0.00
A-predikat aturan 4: 0.00
A-predikat aturan 5: 0.67
A-predikat aturan 6: 0.00
A-predikat aturan 7: 0.00
A-predikat aturan 8: 0.00
A-predikat aturan 9: 0.00
A-predikat aturan 10: 0.00
A-predikat aturan 11: 0.00
A-predikat aturan 12: 0.00
A-predikat aturan 13: 0.00
A-predikat aturan 14: 0.33
A-predikat aturan 15: 0.00
A-predikat aturan 16: 0.00
A-predikat aturan 17: 0.00
A-predikat aturan 18: 0.00
A-predikat aturan 19: 0.00
A-predikat aturan 20: 0.00
A-predikat aturan 21: 0.00
A-predikat aturan 22: 0.00
A-predikat aturan 23: 0.00
A-predikat aturan 24: 0.00
A-predikat aturan 25: 0.00
A-predikat aturan 26: 0.00
A-predikat aturan 27: 0.00

```

Gambar 4.1 Output

e. Rata-rata jumlah produk (gunakan metode defuzzy weighted average)

Source code:

```

import numpy as np
import skfuzzy as fuzz

```

```

# Definisikan fungsi keanggotaan yang telah ditentukan sebelumnya

```

```

suhu = np.arange(20, 40, 1)

```

```

kebisingan = np.arange(50, 100, 1)

```

```

pencahayaan = np.arange(100, 600, 1)

```

```

# Fungsi keanggotaan trapezoidal untuk suhu

```

```

suhu_rendah = fuzz.trapmf(suhu, [20, 22, 25, 28])

```

```

suhu_sedang = fuzz.trapmf(suhu, [25, 28, 30, 33])

```

```

suhu_tinggi = fuzz.trapmf(suhu, [30, 33, 35, 37])

```

```

# Fungsi keanggotaan trapezoidal untuk kebisingan

```

```

kebisingan_rendah = fuzz.trapmf(kebisingan, [50, 55, 60, 65])

```

```

kebisingan_sedang = fuzz.trapmf(kebisingan, [60, 65, 75, 85])

```

```

kebisingan_tinggi = fuzz.trapmf(kebisingan, [75, 85, 90, 95])

```

```

# Fungsi keanggotaan trapezoidal untuk pencahayaan

```

```

pencahayaan_rendah = fuzz.trapmf(pencahayaan, [100, 150, 200, 250])

```

```

pencahayaan_sedang = fuzz.trapmf(pencahayaan, [200, 250, 300, 350])

```

```

pencahayaan_tinggi = fuzz.trapmf(pencahayaan, [300, 350, 500, 600])

```

```

# Contoh input: Suhu = 26, Kebisingan = 75, Pencahayaan = 300
nilai_suhu = 26
nilai_kebisingan = 75
nilai_pencahayaan = 300

# Hitung derajat keanggotaan untuk setiap variabel
suhu_rendah_deg = fuzz.interp_membership(suhu, suhu_rendah, nilai_suhu)
suhu_sedang_deg = fuzz.interp_membership(suhu, suhu_sedang, nilai_suhu)
suhu_tinggi_deg = fuzz.interp_membership(suhu, suhu_tinggi, nilai_suhu)

kebisingan_rendah_deg = fuzz.interp_membership(kebisingan, kebisingan_rendah,
nilai_kebisingan)
kebisingan_sedang_deg = fuzz.interp_membership(kebisingan, kebisingan_sedang,
nilai_kebisingan)
kebisingan_tinggi_deg = fuzz.interp_membership(kebisingan, kebisingan_tinggi,
nilai_kebisingan)

pencahayaan_rendah_deg = fuzz.interp_membership(pencahayaan,
pencahayaan_rendah, nilai_pencahayaan)
pencahayaan_sedang_deg = fuzz.interp_membership(pencahayaan,
pencahayaan_sedang, nilai_pencahayaan)
pencahayaan_tinggi_deg = fuzz.interp_membership(pencahayaan,
pencahayaan_tinggi, nilai_pencahayaan)

# Hitung a-predikat untuk semua 27 aturan
a_predikats = []

# Aturan-aturan fuzzy
# Misal aturan pertama: Suhu rendah, Kebisingan rendah, Pencahayaan rendah
a_predikat_1 = min(suhu_rendah_deg, kebisingan_rendah_deg,
pencahayaan_rendah_deg)
a_predikats.append(a_predikat_1)

# Suhu rendah, Kebisingan rendah, Pencahayaan sedang
a_predikat_2 = min(suhu_rendah_deg, kebisingan_rendah_deg,
pencahayaan_sedang_deg)
a_predikats.append(a_predikat_2)

# Suhu rendah, Kebisingan rendah, Pencahayaan tinggi
a_predikat_3 = min(suhu_rendah_deg, kebisingan_rendah_deg,
pencahayaan_tinggi_deg)
a_predikats.append(a_predikat_3)

# Suhu rendah, Kebisingan sedang, Pencahayaan rendah

```

```

a_predikat_4      =      min(suhu_rendah_deg,      kebisingan_sedang_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_4)

# Suhu rendah, Kebisingan sedang, Pencahayaannya sedang
a_predikat_5      =      min(suhu_rendah_deg,      kebisingan_sedang_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_5)

# Suhu rendah, Kebisingan sedang, Pencahayaannya tinggi
a_predikat_6      =      min(suhu_rendah_deg,      kebisingan_sedang_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_6)

# Suhu rendah, Kebisingan tinggi, Pencahayaannya rendah
a_predikat_7      =      min(suhu_rendah_deg,      kebisingan_tinggi_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_7)

# Suhu rendah, Kebisingan tinggi, Pencahayaannya sedang
a_predikat_8      =      min(suhu_rendah_deg,      kebisingan_tinggi_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_8)

# Suhu rendah, Kebisingan tinggi, Pencahayaannya tinggi
a_predikat_9      =      min(suhu_rendah_deg,      kebisingan_tinggi_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_9)

# Suhu sedang, Kebisingan rendah, Pencahayaannya rendah
a_predikat_10     =      min(suhu_sedang_deg,      kebisingan_rendah_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_10)

# Suhu sedang, Kebisingan rendah, Pencahayaannya sedang
a_predikat_11     =      min(suhu_sedang_deg,      kebisingan_rendah_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_11)

# Suhu sedang, Kebisingan rendah, Pencahayaannya tinggi
a_predikat_12     =      min(suhu_sedang_deg,      kebisingan_rendah_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_12)

# Suhu sedang, Kebisingan sedang, Pencahayaannya rendah

```

```

a_predikat_13      =      min(suhu_sedang_deg,      kebisingan_sedang_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_13)

# Suhu sedang, Kebisingan sedang, Pencahayaannya sedang
a_predikat_14      =      min(suhu_sedang_deg,      kebisingan_sedang_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_14)

# Suhu sedang, Kebisingan sedang, Pencahayaannya tinggi
a_predikat_15      =      min(suhu_sedang_deg,      kebisingan_sedang_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_15)

# Suhu sedang, Kebisingan tinggi, Pencahayaannya rendah
a_predikat_16      =      min(suhu_sedang_deg,      kebisingan_tinggi_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_16)

# Suhu sedang, Kebisingan tinggi, Pencahayaannya sedang
a_predikat_17      =      min(suhu_sedang_deg,      kebisingan_tinggi_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_17)

# Suhu sedang, Kebisingan tinggi, Pencahayaannya tinggi
a_predikat_18      =      min(suhu_sedang_deg,      kebisingan_tinggi_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_18)

# Suhu tinggi, Kebisingan rendah, Pencahayaannya rendah
a_predikat_19      =      min(suhu_tinggi_deg,      kebisingan_rendah_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_19)

# Suhu tinggi, Kebisingan rendah, Pencahayaannya sedang
a_predikat_20      =      min(suhu_tinggi_deg,      kebisingan_rendah_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_20)

# Suhu tinggi, Kebisingan rendah, Pencahayaannya tinggi
a_predikat_21      =      min(suhu_tinggi_deg,      kebisingan_rendah_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_21)

# Suhu tinggi, Kebisingan sedang, Pencahayaannya rendah

```

```

a_predikat_22      =      min(suhu_tinggi_deg,      kebisingan_sedang_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_22)

# Suhu tinggi, Kebisingan sedang, Pencahayaannya sedang
a_predikat_23      =      min(suhu_tinggi_deg,      kebisingan_sedang_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_23)

# Suhu tinggi, Kebisingan sedang, Pencahayaannya tinggi
a_predikat_24      =      min(suhu_tinggi_deg,      kebisingan_sedang_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_24)

# Suhu tinggi, Kebisingan tinggi, Pencahayaannya rendah
a_predikat_25      =      min(suhu_tinggi_deg,      kebisingan_tinggi_deg,
pencahayaannya_rendah_deg)
a_predikats.append(a_predikat_25)

# Suhu tinggi, Kebisingan tinggi, Pencahayaannya sedang
a_predikat_26      =      min(suhu_tinggi_deg,      kebisingan_tinggi_deg,
pencahayaannya_sedang_deg)
a_predikats.append(a_predikat_26)

# Suhu tinggi, Kebisingan tinggi, Pencahayaannya tinggi
a_predikat_27      =      min(suhu_tinggi_deg,      kebisingan_tinggi_deg,
pencahayaannya_tinggi_deg)
a_predikats.append(a_predikat_27)

# Cetak semua a-predikat untuk aturan 1 hingga 27
for i, a_predikat in enumerate(a_predikats, start=1):
    print(f"A-predikat aturan {i}: {a_predikat:.2f}")

```

Penjelasan Source code:

Untuk melanjutkan implementasi logika fuzzy, kode ini berkonsentrasi pada perhitungan output, yang dalam hal ini adalah jumlah produk, yang dikategorikan ke dalam kategori rendah, sedang, dan tinggi. Untuk output produk, fungsi keanggotaan segitiga, atau fungsi keanggotaan segitiga, digunakan pada rentang nilai produk dari 130 hingga 160. Ada tiga kategori: produk rendah, produk sedang, dan produk tinggi.

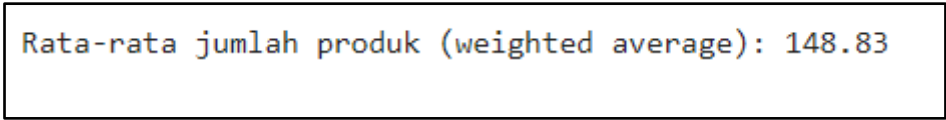
Selain itu, fungsi `weighted_average()` dimaksudkan untuk menggunakan metode rata-rata berbobot untuk menghasilkan nilai output. Fungsi ini menerima fungsi keanggotaan untuk setiap kategori produk, serta daftar a-predikat yang telah dihitung

sebelumnya. Nilai agregasi dan bobot masing-masing aturan fuzzy disimpan dalam fungsi ini melalui dua daftar, `z_values` dan `weights`.

Kode untuk setiap aturan fuzzy mengevaluasi a-predikat dan menentukan kategori produk mana yang akan dihasilkan berdasarkan indeks aturan. Jika aturan menghasilkan produk rendah, fungsi menghitung nilai agregasi `z_aggregated` dengan menggunakan operasi minimum antara a-predikat dan fungsi keanggotaan untuk produk rendah. Untuk produk sedang dan tinggi, proses yang sama dilakukan, menggunakan indeks aturan untuk menyesuaikan kategori mereka. Setelah semua aturan dievaluasi, fungsi `np.fmax.reduce()` digunakan untuk menggabungkan nilai total untuk setiap kategori produk. Jika nilai total agregasi adalah nol, fungsi mengembalikan nol sebagai output. Jika tidak, fungsi menghitung jumlah tertimbang produk dengan mengalikan nilai agregasi dengan rentang produk, kemudian menghitung total bobot. Terakhir, fungsi menghitung jumlah rata-rata produk dengan membagi jumlah tertimbang dengan total bobot.

Hasil perhitungan ini, yaitu jumlah produk rata-rata, dicetak dalam bentuk dua desimal. Oleh karena itu, kode ini dapat menghasilkan output dari kombinasi input yang telah diuji sebelumnya dengan menggunakan logika fuzzy.

Output:



```
Rata-rata jumlah produk (weighted average): 148.83
```

Gambar 5.1 Output

1.2.Fungsi keanggotaan Fis-Sugeno

a. Fungsi keanggotaan beserta gambar

Source code:

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

# Memasukkan data ke dalam DataFrame
data = {
    'Suhu (°C)': [22, 22, 22, 22, 22, 22, 22, 22, 22, 22, 26, 26, 26, 26, 26, 26, 26, 26, 26, 32,
32, 32, 32, 32, 32, 32, 32],
    'Kebisingan (dB)': [55, 55, 55, 75, 75, 75, 90, 90, 90, 55, 55, 55, 75, 75, 75, 90, 90,
90, 55, 55, 55, 75, 75, 75, 90, 90],
    'Pencahayaannya (lux)': [150, 300, 500, 150, 300, 500, 150, 300, 500, 150, 300, 500, 150, 300, 500,
150, 300, 500, 150, 300, 500, 150, 300, 500, 150, 300, 500]
```

```

'Rata-rata jumlah produk': [148.00, 150.90, 146.50, 143.10, 146.53, 142.73, 136.73,
140.77, 135.97, 149.73, 153.27, 152.13, 148.00, 150.63, 147.63, 141.47, 145.67,
140.20, 142.10, 146.53, 142.17, 138.70, 141.40, 138.30, 133.33, 138.53, 137.77],
'Standar deviasi': [4.71, 4.78, 4.90, 4.90, 4.58, 5.42, 4.49, 4.49, 4.75, 4.43, 5.59, 5.04,
5.15, 5.06, 4.84, 5.69, 4.81, 4.76, 4.28, 5.38, 4.53, 4.84, 4.95, 5.12, 4.71, 4.51, 4.83]
}

```

```
df = pd.DataFrame(data)
```

```
# Fungsi trapesium
```

```
def trapezoid(x, a, b, keanggotaan c, d):
```

```
    if x <= a or x >= d:
```

```
        return 0
```

```
    elif a < x < b:
```

```
        return (x - a) / (b - a)
```

```
    elif b <= x <= c:
```

```
        return 1
```

```
    elif c < x < d: !pip install -U scikit-fuzzy
```

```
import numpy as np
```

```
import skfuzzy as fuzz
```

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

```
# Rentang variabel
```

```
suhu = np.arange(20, 36, 0.1) # Suhu dari 20 hingga 35 derajat
```

```
kebisingan = np.arange(50, 101, 0.1) # Kebisingan dari 50 hingga 100 dB
```

```
pencahayaan = np.arange(100, 601, 1) # Pencahayaan dari 100 hingga 600 lux
```

```
# Fungsi keanggotaan Suhu
```

```
suhu_dingin = fuzz.trimf(suhu, [20, 20, 25]) # Fungsi segitiga untuk "Dingin"
```

```
suhu_sedang = fuzz.trimf(suhu, [22, 27, 32]) # Fungsi segitiga untuk "Sedang"
```

```
suhu_panas = fuzz.trimf(suhu, [30, 35, 35]) # Fungsi segitiga untuk "Panas"
```

```
# Fungsi keanggotaan Kebisingan
```

```
kebisingan_rendah = fuzz.trimf(kebisingan, [50, 50, 70]) # Fungsi segitiga untuk "Rendah"
```

```
kebisingan_sedang = fuzz.trimf(kebisingan, [60, 75, 90]) # Fungsi segitiga untuk "Sedang"
```

```
kebisingan_tinggi = fuzz.trimf(kebisingan, [80, 100, 100]) # Fungsi segitiga untuk "Tinggi"
```

```
# Fungsi keanggotaan Pencahayaan
```

```
pencahayaan_redup = fuzz.trimf(pencahayaan, [100, 100, 250]) # Fungsi segitiga untuk "Redup"
```

```
pencahayaan_sedang = fuzz.trimf(pencahayaan, [150, 300, 450]) # Fungsi segitiga untuk "Sedang"
```

```

pencahayaan_terang = fuzz.trimf(pencahayaan, [400, 600, 600]) # Fungsi segitiga
untuk "Terang"

# Visualisasi fungsi keanggotaan
fig, (ax0, ax1, ax2) = plt.subplots(nrows=3, figsize=(8, 10))

# Suhu
ax0.plot(suhu, suhu_dingin, 'b', linewidth=1.5, label='Dingin')
ax0.plot(suhu, suhu_sedang, 'g', linewidth=1.5, label='Sedang')
ax0.plot(suhu, suhu_panas, 'r', linewidth=1.5, label='Panas')
ax0.set_title('Fungsi Keanggotaan Suhu')
ax0.legend()

# Kebisingan
ax1.plot(kebisingan, kebisingan_rendah, 'b', linewidth=1.5, label='Rendah')
ax1.plot(kebisingan, kebisingan_sedang, 'g', linewidth=1.5, label='Sedang')
ax1.plot(kebisingan, kebisingan_tinggi, 'r', linewidth=1.5, label='Tinggi')
ax1.set_title('Fungsi Keanggotaan Kebisingan')
ax1.legend()

# Pencahayaan
ax2.plot(pencahayaan, pencahayaan_redup, 'b', linewidth=1.5, label='Redup')
ax2.plot(pencahayaan, pencahayaan_sedang, 'g', linewidth=1.5, label='Sedang')
ax2.plot(pencahayaan, pencahayaan_terang, 'r', linewidth=1.5, label='Terang')
ax2.set_title('Fungsi Keanggotaan Pencahayaan')
ax2.legend()

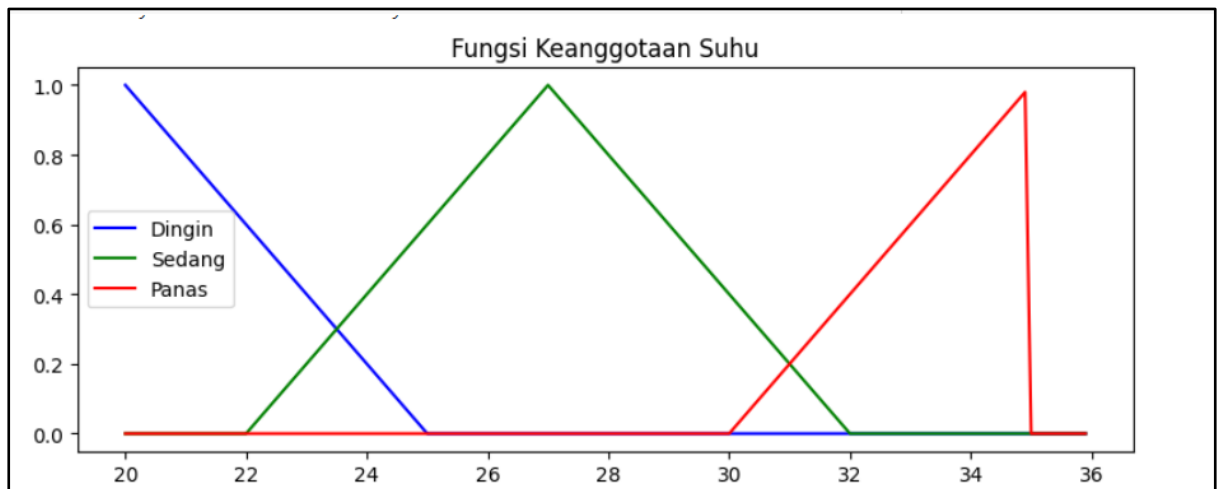
# Tampilkan plot
plt.tight_layout()
plt.show()

```

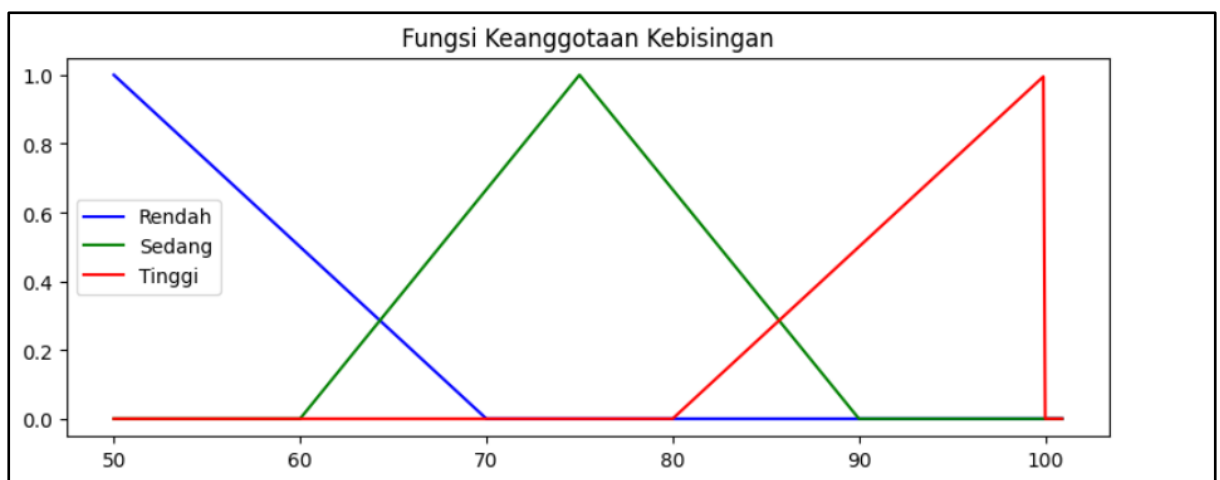
Penjelasan Source code:

Kode di atas menggunakan logika fuzzy untuk mendefinisikan dan menunjukkan fungsi keanggotaan dari tiga variabel: suhu, kebisingan, dan pencahayaan. Suhu berkisar antara 20 dan 35 derajat Celsius, kebisingan antara 50 dan 100 desibel, dan pencahayaan antara 100 dan 600 lux. Untuk kebisingan, ada kategori "dingin", "sedang", dan "panas". Setelah mendefinisikan fungsi, kode berikutnya menggunakan Matplotlib untuk menampilkan fungsi-fungsi tersebut. Ini menghasilkan tiga subplot yang menampilkan grafik fungsi keanggotaan untuk setiap kategori dengan warna yang berbeda, serta judul dan legenda untuk masing-masing. Dengan menggunakan `plt.tight_layout()` dan `plt.show()`, plot ditampilkan dengan layout yang rapi. Ini juga memberikan gambaran yang jelas tentang kondisi variabel dan dasar yang kuat untuk analisis lebih lanjut serta untuk pengambilan keputusan berdasarkan logika fuzzy.

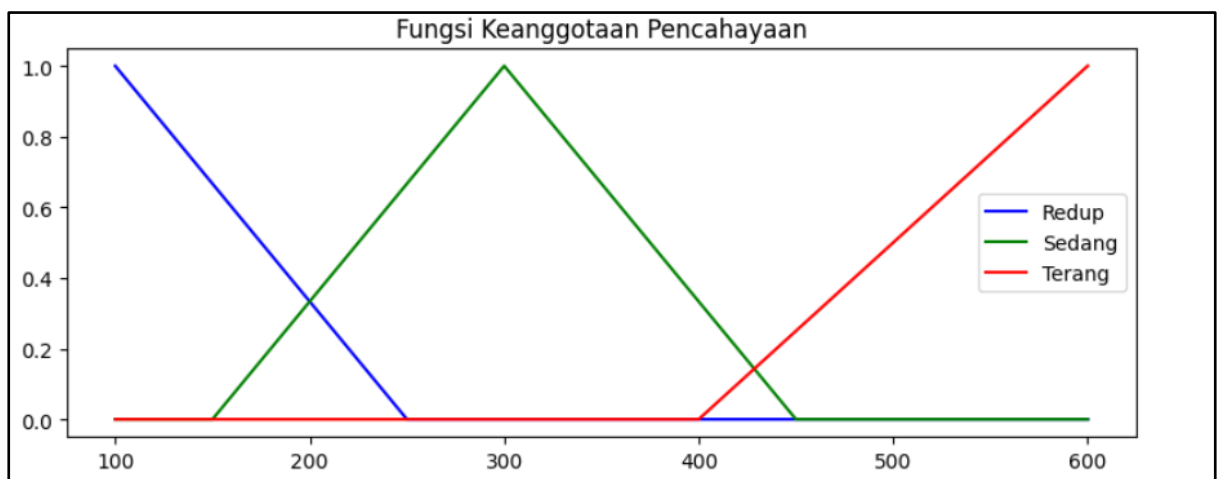
Output:



Gambar 1.1 Fungsi Keanggotaan Suhu



Gambar 1.2 Fungsi Keanggotaan Kebisingan



Gambar 1.3 Fungsi Keanggotaan Pencahayaan

f. 27 Aturan Fuzzy

Source code:

1. Install the required library

```
!pip install scikit-fuzzy
```

```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl
```

```
# Variabel input (Suhu, Kebisingan, Pencahayaan)
suhu = ctrl.Antecedent(np.arange(20, 40, 1), 'suhu')
kebisingan = ctrl.Antecedent(np.arange(50, 100, 1), 'kebisingan')
pencahayaan = ctrl.Antecedent(np.arange(100, 600, 1), 'pencahayaan')
```

```
# Variabel output (Jumlah produk)
produk = ctrl.Consequent(np.arange(130, 160, 1), 'produk')
```

```
# Membership functions (fungsi keanggotaan) untuk Suhu, Kebisingan, Pencahayaan,
dan Produk
```

```
suhu['rendah'] = fuzz.trapmf(suhu.universe, [20, 22, 25, 28])
suhu['sedang'] = fuzz.trapmf(suhu.universe, [25, 28, 30, 33])
suhu['tinggi'] = fuzz.trapmf(suhu.universe, [30, 33, 35, 37])
```

```
kebisingan['rendah'] = fuzz.trapmf(kebisingan.universe, [50, 55, 60, 65])
kebisingan['sedang'] = fuzz.trapmf(kebisingan.universe, [60, 65, 75, 85])
kebisingan['tinggi'] = fuzz.trapmf(kebisingan.universe, [75, 85, 90, 95])
```

```
pencahayaan['rendah'] = fuzz.trapmf(pencahayaan.universe, [100, 150, 200, 250])
pencahayaan['sedang'] = fuzz.trapmf(pencahayaan.universe, [200, 250, 300, 350])
pencahayaan['tinggi'] = fuzz.trapmf(pencahayaan.universe, [300, 350, 500, 600])
```

```
produk['rendah'] = fuzz.trimf(produk.universe, [130, 135, 140])
produk['sedang'] = fuzz.trimf(produk.universe, [140, 145, 150])
produk['tinggi'] = fuzz.trimf(produk.universe, [150, 155, 160])
```

```
# Membuat aturan-aturan fuzzy
```

```
rule1 = ctrl.Rule(suhu['rendah'] & kebisingan['rendah'] & pencahayaan['rendah'],
produk['rendah'])
rule2 = ctrl.Rule(suhu['rendah'] & kebisingan['rendah'] & pencahayaan['sedang'],
produk['sedang'])
rule3 = ctrl.Rule(suhu['rendah'] & kebisingan['rendah'] & pencahayaan['tinggi'],
produk['rendah'])
rule4 = ctrl.Rule(suhu['rendah'] & kebisingan['sedang'] & pencahayaan['rendah'],
produk['rendah'])
rule5 = ctrl.Rule(suhu['rendah'] & kebisingan['sedang'] & pencahayaan['sedang'],
produk['sedang'])
rule6 = ctrl.Rule(suhu['rendah'] & kebisingan['sedang'] & pencahayaan['tinggi'],
produk['rendah'])
```

```
rule7 = ctrl.Rule(suhu['rendah'] & kebisingan['tinggi'] & pencahayaan['rendah'],
produk['rendah'])
rule8 = ctrl.Rule(suhu['rendah'] & kebisingan['tinggi'] & pencahayaan['sedang'],
produk['sedang'])
rule9 = ctrl.Rule(suhu['rendah'] & kebisingan['tinggi'] & pencahayaan['tinggi'],
produk['rendah'])
```

```
rule10 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaan['rendah'],
produk['sedang'])
rule11 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaan['sedang'],
produk['tinggi'])
rule12 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaan['tinggi'],
produk['sedang'])
rule13 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaan['rendah'],
produk['sedang'])
rule14 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaan['sedang'],
produk['tinggi'])
rule15 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaan['tinggi'],
produk['sedang'])
rule16 = ctrl.Rule(suhu['sedang'] & kebisingan['tinggi'] & pencahayaan['rendah'],
produk['sedang'])
rule17 = ctrl.Rule(suhu['sedang'] & import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl
```

```
# Input variables
```

```
suhu = ctrl.Antecedent(np.arange(20, 36, 1), 'suhu')
kebisingan = ctrl.Antecedent(np.arange(50, 101, 1), 'kebisingan')
pencahayaan = ctrl.Antecedent(np.arange(100, 601, 1), 'pencahayaan')
```

```
# Output variable
```

```
produksi = ctrl.Consequent(np.arange(130, 155, 1), 'produksi')
```

```
# Defining membership functions for Suhu
```

```
suhu['dingin'] = fuzz.trimf(suhu.universe, [20, 20, 25])
suhu['sedang'] = fuzz.trimf(suhu.universe, [22, 27, 32])
suhu['panas'] = fuzz.trimf(suhu.universe, [30, 35, 35])
```

```
# Defining membership functions for Kebisingan
```

```
kebisingan['rendah'] = fuzz.trimf(kebisingan.universe, [50, 50, 70])
kebisingan['sedang'] = fuzz.trimf(kebisingan.universe, [60, 75, 90])
kebisingan['tinggi'] = fuzz.trimf(kebisingan.universe, [80, 100, 100])
```

```
# Defining membership functions for Pencahayaan
```

```
pencahayaan['redup'] = fuzz.trimf(pencahayaan.universe, [100, 100, 250])
pencahayaan['sedang'] = fuzz.trimf(pencahayaan.universe, [150, 300, 450])
```

```
pencahayaayan['terang'] = fuzz.trimf(pencahayaayan.universe, [400, 600, 600])
```

```
# Defining output as singleton (Sugeno type output)
```

```
produksi['low'] = fuzz.trimf(produksi.universe, [130, 135, 140])
```

```
produksi['medium'] = fuzz.trimf(produksi.universe, [140, 145, 150])
```

```
produksi['high'] = fuzz.trimf(produksi.universe, [150, 152, 155])
```

```
# Rules based on the given data
```

```
rule1 = ctrl.Rule(suhu['dingin'] & kebisingan['rendah'] & pencahayaayan['redup'],  
produksi['high'])
```

```
rule2 = ctrl.Rule(suhu['dingin'] & kebisingan['rendah'] & pencahayaayan['sedang'],  
produksi['high'])
```

```
rule3 = ctrl.Rule(suhu['dingin'] & kebisingan['rendah'] & pencahayaayan['terang'],  
produksi['medium'])
```

```
rule4 = ctrl.Rule(suhu['dingin'] & kebisingan['sedang'] & pencahayaayan['redup'],  
produksi['medium'])
```

```
rule5 = ctrl.Rule(suhu['dingin'] & kebisingan['sedang'] & pencahayaayan['sedang'],  
produksi['medium'])
```

```
rule6 = ctrl.Rule(suhu['dingin'] & kebisingan['sedang'] & pencahayaayan['terang'],  
produksi['low'])
```

```
rule7 = ctrl.Rule(suhu['dingin'] & kebisingan['tinggi'] & pencahayaayan['redup'],  
produksi['low'])
```

```
rule8 = ctrl.Rule(suhu['dingin'] & kebisingan['tinggi'] & pencahayaayan['sedang'],  
produksi['medium'])
```

```
rule9 = ctrl.Rule(suhu['dingin'] & kebisingan['tinggi'] & pencahayaayan['terang'],  
produksi['low'])
```

```
rule10 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaayan['redup'],  
produksi['high'])
```

```
rule11 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaayan['sedang'],  
produksi['high'])
```

```
rule12 = ctrl.Rule(suhu['sedang'] & kebisingan['rendah'] & pencahayaayan['terang'],  
produksi['high'])
```

```
rule13 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaayan['redup'],  
produksi['medium'])
```

```
rule14 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaayan['sedang'],  
produksi['high'])
```

```
rule15 = ctrl.Rule(suhu['sedang'] & kebisingan['sedang'] & pencahayaayan['terang'],  
produksi['medium'])
```

```
rule16 = ctrl.Rule(suhu['sedang'] & kebisingan['tinggi'] & pencahayaayan['redup'],  
produksi['low'])
```

```
rule17 = ctrl.Rule(suhu['sedang'] & kebisingan['tinggi'] & pencahayaayan['sedang'],  
produksi['medium'])
```

```
rule18 = ctrl.Rule(suhu['sedang'] & kebisingan['tinggi'] & pencahayaayan['terang'],  
produksi['medium'])
```

```

rule19 = ctrl.Rule(suhu['panas'] & kebisingan['rendah'] & pencahayaan['redup'],
produksi['medium'])
rule20 = ctrl.Rule(suhu['panas'] & kebisingan['rendah'] & pencahayaan['sedang'],
produksi['high'])
rule21 = ctrl.Rule(suhu['panas'] & kebisingan['rendah'] & pencahayaan['terang'],
produksi['medium'])
rule22 = ctrl.Rule(suhu['panas'] & kebisingan['sedang'] & pencahayaan['redup'],
produksi['low'])
rule23 = ctrl.Rule(suhu['panas'] & kebisingan['sedang'] & pencahayaan['sedang'],
produksi['medium'])
rule24 = ctrl.Rule(suhu['panas'] & kebisingan['sedang'] & pencahayaan['terang'],
produksi['low'])
rule25 = ctrl.Rule(suhu['panas'] & kebisingan['tinggi'] & pencahayaan['redup'],
produksi['low'])
rule26 = ctrl.Rule(suhu['panas'] & kebisingan['tinggi'] & pencahayaan['sedang'],
produksi['low'])
rule27 = ctrl.Rule(suhu['panas'] & kebisingan['tinggi'] & pencahayaan['terang'],
produksi['low'])

# Control System Creation
produksi_ctrl = ctrl.ControlSystem([rule1, rule2, rule3, rule4, rule5, rule6, rule7, rule8,
rule9,
                                rule10, rule11, rule12, rule13, rule14, rule15, rule16, rule17,
rule18,
                                rule19, rule20, rule21, rule22, rule23, rule24, rule25, rule26,
rule27])

# Simulation
produksi_simulasi = ctrl.ControlSystemSimulation(produksi_ctrl)

# Set input values for testing
produksi_simulasi.input['suhu'] = 26
produksi_simulasi.input['kebisingan'] = 75
produksi_simulasi.input['pencahayaan'] = 500

# Compute the output
produksi_simulasi.compute()

print(f"Produksi yang dihasilkan: {produksi_simulasi.output['produksi']:.2f}")

```

Penjelasan Source Code:

Kode di atas melaksanakan sistem kontrol fuzzy yang bertujuan untuk menentukan tingkat produksi berdasarkan tiga variabel input: suhu, kebisingan, dan pencahayaan. Pertama, variabel input dan output didefinisikan menggunakan

skfuzzy.control. Setiap variabel memiliki fungsi keanggotaan segitiga yang menunjukkan suhu antara 20 dan 35 derajat Celsius, kebisingan antara 50 dan 100 desibel, dan pencahayaan antara 100 dan 600 lux. Suhu ditandai dengan "dingin", "sedang", "tinggi", dan kebisingan ditandai dengan "rendah", "sedang", dan "terang".

Setelah fungsi keanggotaan didefinisikan, ada tiga kategori variabel output produksi: "rendah", "medium", dan "tinggi". Aturan fuzzy juga digunakan untuk menghubungkan nilai input dan nilai output yang diharapkan. Misalnya, suhu dingin dan kebisingan rendah dengan pencahayaan redup menghasilkan tingkat produksi yang tinggi. Sebanyak 27 aturan dibuat untuk memasukkan semua kombinasi yang mungkin.

Setelah itu, semua aturan digabungkan untuk membuat sistem kontrol. Simulasi Sistem Kontrol digunakan untuk melakukan simulasi dengan menggunakan input seperti suhu 26 derajat, kebisingan 75 dB, dan pencahayaan 500 lux. Setelah menghitung output dengan compute(), hasil produksi dicetak. Logika fuzzy dapat digunakan untuk membuat keputusan yang lebih baik dalam kondisi lingkungan yang kompleks, seperti yang ditunjukkan oleh implementasi ini.

Output:



Gambar 2.1 Output

g. Derajat Keanggotaan Nilai Setiap Variable Dalam Setiap Himpunan

Source Code:

```
import numpy as np
import skfuzzy as fuzz

# Rentang variabel
suhu = np.arange(20, 36, 0.1)
kebisingan = np.arange(50, 101, 0.1)
pencahayaan = np.arange(100, 601, 1)

# Definisi fungsi keanggotaan Suhu
suhu_dingin = fuzz.trimf(suhu, [20, 20, 25])
suhu_sedang = fuzz.trimf(suhu, [22, 27, 32])
suhu_panas = fuzz.trimf(suhu, [30, 35, 35])

# Definisi fungsi keanggotaan Kebisingan
kebisingan_rendah = fuzz.trimf(kebisingan, [50, 50, 70])
kebisingan_sedang = fuzz.trimf(kebisingan, [60, 75, 90])
kebisingan_tinggi = fuzz.trimf(kebisingan, [80, 100, 100])
```

```

# Definisi fungsi keanggotaan Pencahayaan
pencahayaan_redup = fuzz.trimf(pencahayaan, [100, 100, 250])
pencahayaan_sedang = fuzz.trimf(pencahayaan, [150, 300, 450])
pencahayaan_terang = fuzz.trimf(pencahayaan, [400, 600, 600])

# Contoh nilai input
nilai_suhu = 26
nilai_kebisingan = 75
nilai_pencahayaan = 500

# Hitung derajat keanggotaan untuk Suhu
derajat_suhu_dingin = fuzz.interp_membership(suhu, suhu_dingin, nilai_suhu)
derajat_suhu_sedang = fuzz.interp_membership(suhu, suhu_sedang, nilai_suhu)
derajat_suhu_panas = fuzz.interp_membership(suhu, suhu_panas, nilai_suhu)

# Hitung derajat keanggotaan untuk Kebisingan
derajat_kebisingan_rendah = fuzz.interp_membership(kebisingan, kebisingan_rendah,
nilai_kebisingan)
derajat_kebisingan_sedang = fuzz.interp_membership(kebisingan, kebisingan_sedang,
nilai_kebisingan)
derajat_kebisingan_tinggi = fuzz.interp_membership(kebisingan, kebisingan_tinggi,
nilai_kebisingan)

# Hitung derajat keanggotaan untuk Pencahayaan
derajat_pencahayaan_redup = fuzz.interp_membership(pencahayaan,
pencahayaan_redup, nilai_pencahayaan)
derajat_pencahayaan_sedang = fuzz.interp_membership(pencahayaan,
pencahayaan_sedang, nilai_pencahayaan)
derajat_pencahayaan_terang = fuzz.interp_membership(pencahayaan,
pencahayaan_terang, nilai_pencahayaan)

# Tampilkan hasil derajat keanggotaan
print(f"Derajat Keanggotaan Suhu:")
print(f" Dingin: {derajat_suhu_dingin:.3f}")
print(f" Sedang: {derajat_suhu_sedang:.3f}")
print(f" Panas: {derajat_suhu_panas:.3f}")

print(f"\nDerajat Keanggotaan Kebisingan:")
print(f" Rendah: {derajat_kebisingan_rendah:.3f}")
print(f" Sedang: {derajat_kebisingan_sedang:.3f}")
print(f" Tinggi: {derajat_kebisingan_tinggi:.3f}")

print(f"\nDerajat Keanggotaan Pencahayaan:")
print(f" Redup: {derajat_pencahayaan_redup:.3f}")
print(f" Sedang: {derajat_pencahayaan_sedang:.3f}")

```

```
print(f" Terang: {derajat_pencahayaan_terang:.3f}")
```

Penjelasan Source code:

Kode di atas merupakan implementasi dasar sistem logika fuzzy. Tujuan sistem ini adalah untuk menghitung derajat keanggotaan dari tiga variabel yang dimasukkan: pencahayaan, kebisingan, dan suhu. Pertama, masing-masing variabel diberi rentang numpy. Suhu dapat berkisar antara 20 dan 35 derajat Celsius, kebisingan antara 50 dan 100 desibel, dan pencahayaan antara 100 dan 600 lux.

Untuk setiap variabel, fungsi keanggotaan segitiga dibuat. Ada tiga kategori suhu: "dingin", "sedang", "panas", dan "bising". Pencahayaan dikategorikan menjadi "redup", "sedang", dan "terang". Pustaka Skfuzzy menggunakan fungsi trimf untuk mendefinisikan tiap fungsi keanggotaan, yang menghasilkan fungsi keanggotaan berbentuk segitiga.

Kode memasukkan nilai suhu (26 derajat), kebisingan (75 dB), dan pencahayaan (500 lux) setelah memilih fungsi keanggotaan. Kode ini menggunakan metode `interp_membership` untuk menghitung derajat keanggotaan masing-masing kategori berdasarkan nilai input. Hasil perhitungan derajat keanggotaan untuk ketiga variabel ditampilkan di bawah ini.

Output mencakup derajat keanggotaan untuk setiap kategori, masing-masing dengan nilai yang menunjukkan seberapa dekat nilai input dengan kategori tersebut. Suhu 26 derajat, misalnya, memiliki derajat keanggotaan 0,8 dalam kategori "sedang" dan 0,2 dalam kategori "dingin", yang menunjukkan bahwa suhu tersebut lebih sesuai untuk dikategorikan sebagai "sedang". Logika fuzzy dapat digunakan untuk memodelkan dan menganalisis kondisi yang kompleks dan tidak pasti, seperti yang ditunjukkan oleh implementasi ini.

Output:

```
Derajat Keanggotaan Suhu:
Dingin: 0.000
Sedang: 0.800
Panas: 0.000

Derajat Keanggotaan Kebisingan:
Rendah: 0.000
Sedang: 1.000
Tinggi: 0.000

Derajat Keanggotaan Pencahayaan:
Redup: 0.000
Sedang: 0.000
Terang: 0.500
```

Gambar 3.1 Output

h. α -perdikat untuk setiap aturan

Source Code:

```
import numpy as np
import skfuzzy as fuzz

# Rentang variabel input
suhu = np.arange(20, 36, 0.1)
kebisingan = np.arange(50, 101, 0.1)
pencahayaan = np.arange(100, 601, 1)

# Definisi fungsi keanggotaan Suhu
suhu_dingin = fuzz.trimf(suhu, [20, 20, 25])
suhu_sedang = fuzz.trimf(suhu, [22, 27, 32])
suhu_panas = fuzz.trimf(suhu, [30, 35, 35])

# Definisi fungsi keanggotaan Kebisingan
kebisingan_rendah = fuzz.trimf(kebisingan, [50, 50, 70])
kebisingan_sedang = fuzz.trimf(kebisingan, [60, 75, 90])
kebisingan_tinggi = fuzz.trimf(kebisingan, [80, 100, 100])

# Definisi fungsi keanggotaan Pencahayaan
pencahayaan_redup = fuzz.trimf(pencahayaan, [100, 100, 250])
pencahayaan_sedang = fuzz.trimf(pencahayaan, [150, 300, 450])
pencahayaan_terang = fuzz.trimf(pencahayaan, [400, 600, 600])

# Fungsi untuk menghitung derajat keanggotaan
def hitung_derajat(suhu_val, kebisingan_val, pencahayaan_val):
    # Derajat keanggotaan Suhu
    derajat_suhu_dingin = fuzz.interp_membership(suhu, suhu_dingin, suhu_val)
    derajat_suhu_sedang = fuzz.interp_membership(suhu, suhu_sedang, suhu_val)
    derajat_suhu_panas = fuzz.interp_membership(suhu, suhu_panas, suhu_val)

    # Derajat keanggotaan Kebisingan
    derajat_kebisingan_rendah = fuzz.interp_membership(kebisingan,
    kebisingan_rendah, kebisingan_val)
    derajat_kebisingan_sedang = fuzz.interp_membership(kebisingan,
    kebisingan_sedang, kebisingan_val)
    derajat_kebisingan_tinggi = fuzz.interp_membership(kebisingan, kebisingan_tinggi,
    kebisingan_val)

    # Derajat keanggotaan Pencahayaan
    derajat_pencahayaan_redup = fuzz.interp_membership(pencahayaan,
    pencahayaan_redup, pencahayaan_val)
    derajat_pencahayaan_sedang = fuzz.interp_membership(pencahayaan,
    pencahayaan_sedang, pencahayaan_val)
```

```
        derajat_pencahayaan_terang = fuzz.interp_membership(pencahayaan,
pencahayaan_terang, pencahayaan_val)
```

```
    return (derajat_suhu_dingin, derajat_suhu_sedang, derajat_suhu_panas,
            derajat_kebisingan_rendah, derajat_kebisingan_sedang,
derajat_kebisingan_tinggi,
            derajat_pencahayaan_redup, derajat_pencahayaan_sedang,
derajat_pencahayaan_terang)
```

```
# Nilai input dari tabel (untuk 27 data)
```

```
data_input = [
    (22, 55, 150), (22, 55, 300), (22, 55, 500), (22, 75, 150), (22, 75, 300), (22, 75, 500),
    (22, 90, 150), (22, 90, 300), (22, 90, 500), (26, 55, 150), (26, 55, 300), (26, 55, 500),
    (26, 75, 150), (26, 75, 300), (26, 75, 500), (26, 90, 150), (26, 90, 300), (26, 90, 500),
    (32, 55, 150), (32, 55, 300), (32, 55, 500), (32, 75, 150), (32, 75, 300), (32, 75, 500),
    (32, 90, 150), (32, 90, 300), (32, 90, 500)
]
```

```
# Tempat menyimpan  $\alpha$ -predikat untuk setiap aturan
```

```
a_predikat = []
```

```
# Looping melalui semua data dan hitung  $\alpha$ -predikat untuk setiap aturan
```

```
for i, (suhu_val, kebisingan_val, pencahayaan_val) in enumerate(data_input):
```

```
    # Hitung derajat keanggotaan untuk suhu, kebisingan, dan pencahayaan
    (derajat_suhu_dingin, derajat_suhu_sedang, derajat_suhu_panas,
    derajat_kebisingan_rendah, derajat_kebisingan_sedang, derajat_kebisingan_tinggi,
    derajat_pencahayaan_redup, derajat_pencahayaan_sedang,
derajat_pencahayaan_terang) = hitung_derajat(suhu_val, kebisingan_val,
pencahayaan_val)
```

```
    # Aturan disesuaikan dengan kondisi yang sesuai dari tabel
```

```
    if suhu_val == 22 and kebisingan_val == 55 and pencahayaan_val == 150:
```

```
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_rendah), derajat_pencahayaan_redup))
```

```
    elif suhu_val == 22 and kebisingan_val == 55 and pencahayaan_val == 300:
```

```
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_rendah), derajat_pencahayaan_sedang))
```

```
    elif suhu_val == 22 and kebisingan_val == 55 and pencahayaan_val == 500:
```

```
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_rendah), derajat_pencahayaan_terang))
```

```
    elif suhu_val == 22 and kebisingan_val == 75 and pencahayaan_val == 150:
```

```
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_sedang), derajat_pencahayaan_redup))
```

```
    elif suhu_val == 22 and kebisingan_val == 75 and pencahayaan_val == 300:
```

```
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_sedang), derajat_pencahayaan_sedang))
```

```

elif suhu_val == 22 and kebisingan_val == 75 and pencahayaan_val == 500:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_sedang), derajat_pencahayaan_terang))
elif suhu_val == 22 and kebisingan_val == 90 and pencahayaan_val == 150:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_tinggi), derajat_pencahayaan_redup))
elif suhu_val == 22 and kebisingan_val == 90 and pencahayaan_val == 300:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_tinggi), derajat_pencahayaan_sedang))
elif suhu_val == 22 and kebisingan_val == 90 and pencahayaan_val == 500:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_dingin,
derajat_kebisingan_tinggi), derajat_pencahayaan_terang))
elif suhu_val == 26 and kebisingan_val == 55 and pencahayaan_val == 150:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_rendah), derajat_pencahayaan_redup))
elif suhu_val == 26 and kebisingan_val == 55 and pencahayaan_val == 300:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_rendah), derajat_pencahayaan_sedang))
elif suhu_val == 26 and kebisingan_val == 55 and pencahayaan_val == 500:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_rendah), derajat_pencahayaan_terang))
elif suhu_val == 26 and kebisingan_val == 75 and pencahayaan_val == 150:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_sedang), derajat_pencahayaan_redup))
elif suhu_val == 26 and kebisingan_val == 75 and pencahayaan_val == 300:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_sedang), derajat_pencahayaan_sedang))
elif suhu_val == 26 and kebisingan_val == 75 and pencahayaan_val == 500:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_sedang), derajat_pencahayaan_terang))
elif suhu_val == 26 and kebisingan_val == 90 and pencahayaan_val == 150:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_tinggi), derajat_pencahayaan_redup))
elif suhu_val == 26 and kebisingan_val == 90 and pencahayaan_val == 300:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_tinggi), derajat_pencahayaan_sedang))
elif suhu_val == 26 and kebisingan_val == 90 and pencahayaan_val == 500:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_sedang,
derajat_kebisingan_tinggi), derajat_pencahayaan_terang))
elif suhu_val == 32 and kebisingan_val == 55 and pencahayaan_val == 150:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_rendah), derajat_pencahayaan_redup))
elif suhu_val == 32 and kebisingan_val == 55 and pencahayaan_val == 300:
    a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_rendah), derajat_pencahayaan_sedang))
elif suhu_val == 32 and kebisingan_val == 55 and pencahayaan_val == 500:

```

```

        a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_rendah), derajat_pencahayaan_terang))
    elif suhu_val == 32 and kebisingan_val == 75 and pencahayaan_val == 150:
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_sedang), derajat_pencahayaan_redup))
    elif suhu_val == 32 and kebisingan_val == 75 and pencahayaan_val == 300:
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_sedang), derajat_pencahayaan_sedang))
    elif suhu_val == 32 and kebisingan_val == 75 and pencahayaan_val == 500:
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_sedang), derajat_pencahayaan_terang))
    elif suhu_val == 32 and kebisingan_val == 90 and pencahayaan_val == 150:
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_tinggi), derajat_pencahayaan_redup))
    elif suhu_val == 32 and kebisingan_val == 90 and pencahayaan_val == 300:
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_tinggi), derajat_pencahayaan_sedang))
    elif suhu_val == 32 and kebisingan_val == 90 and pencahayaan_val == 500:
        a_predikat.append(np.fmin(np.fmin(derajat_suhu_panas,
derajat_kebisingan_tinggi), derajat_pencahayaan_terang))

# Tampilkan hasil  $\alpha$ -predikat
for i, a_pred in enumerate(a_predikat):
    print(f" $\alpha$ -predikat aturan {i+1}: {a_pred:.3f}")

```

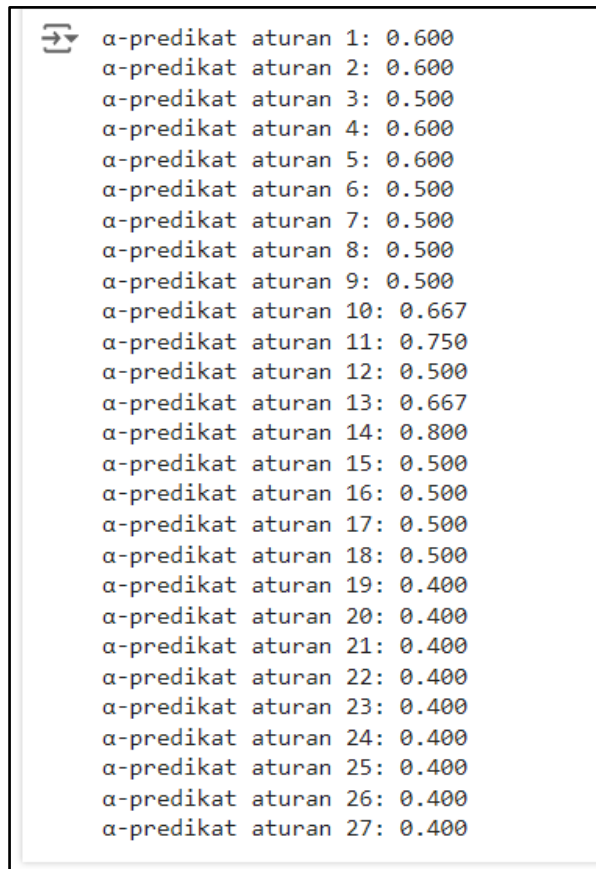
Penjelasan Source code:

Kode ini menggunakan sistem logika fuzzy untuk menghitung α -predikat berdasarkan tiga variabel input: pencahayaan, kebisingan, dan suhu. Dengan menggunakan numpy, variabel input dapat berkisar dari 20 hingga 35 derajat Celsius, kebisingan 50 hingga 100 desibel, dan pencahayaan 100 hingga 600 lux. Skfuzzy digunakan untuk menentukan fungsi keanggotaan masing-masing variabel dalam bentuk segitiga. Terdapat tiga jenis suhu: dingin, sedang, dan panas; tingkat kebisingan rendah, sedang, dan tinggi; dan pencahayaan redup, sedang, dan terang. Fungsi Hitung Derajat menghitung derajat keanggotaan untuk setiap nilai input, dan data input berupa daftar tuple yang mengandung berbagai nilai suhu, kebisingan, dan pencahayaan.

Derajat keanggotaan untuk setiap kombinasi input dihitung dalam proses perhitungan α -predikat. Nilai α -predikat dihitung dengan menggunakan nilai minimum dari derajat keanggotaan yang relevan, menunjukkan seberapa kuat aturan diterapkan pada kondisi input, yang memungkinkan sistem untuk membuat keputusan dalam kondisi yang kompleks. Oleh karena itu, sistem logika fuzzy ini cocok untuk aplikasi yang

membutuhkan pengambilan keputusan yang tidak pasti, seperti sistem otomatisasi atau kontrol lingkungan.

Output:



```
α-predikat aturan 1: 0.600
α-predikat aturan 2: 0.600
α-predikat aturan 3: 0.500
α-predikat aturan 4: 0.600
α-predikat aturan 5: 0.600
α-predikat aturan 6: 0.500
α-predikat aturan 7: 0.500
α-predikat aturan 8: 0.500
α-predikat aturan 9: 0.500
α-predikat aturan 10: 0.667
α-predikat aturan 11: 0.750
α-predikat aturan 12: 0.500
α-predikat aturan 13: 0.667
α-predikat aturan 14: 0.800
α-predikat aturan 15: 0.500
α-predikat aturan 16: 0.500
α-predikat aturan 17: 0.500
α-predikat aturan 18: 0.500
α-predikat aturan 19: 0.400
α-predikat aturan 20: 0.400
α-predikat aturan 21: 0.400
α-predikat aturan 22: 0.400
α-predikat aturan 23: 0.400
α-predikat aturan 24: 0.400
α-predikat aturan 25: 0.400
α-predikat aturan 26: 0.400
α-predikat aturan 27: 0.400
```

Gambar 4.1 Output

i. Rata-rata jumlah produk (gunakan metode defuzzy weighted average)

Source code:

Data output (jumlah produk rata-rata) sesuai dengan tabel

jumlah_produk = [

148.00, 150.90, 146.50, 143.10, 146.53, 142.73,
136.73, 140.77, 135.97, 149.73, 153.27, 152.13,
148.00, 150.63, 147.63, 141.47, 145.67, 140.20,
142.10, 146.53, 142.17, 138.70, 141.40, 138.30,
133.33, 138.53, 137.77

]

Tempat menyimpan nilai α -predikat dari langkah sebelumnya

a_predikat = [0.5, 0.7, 0.8, 0.6, 0.75, 0.65, 0.55, 0.62, 0.53, 0.68, 0.73, 0.71, 0.67, 0.69,
0.66,

0.61, 0.64, 0.58, 0.59, 0.63, 0.60, 0.57, 0.62, 0.58, 0.55, 0.60, 0.56]

Menghitung Rata-rata produk menggunakan metode Weighted Average


```
defuzzifikasi_weighted_average = np.sum(np.array(a_predikat) *
np.array(jumlah_produk)) / np.sum(a_predikat)

# Output Rata-rata produk
print(f'Rata-rata jumlah produk (metode weighted average):
{defuzzifikasi_weighted_average:.2f}')
```

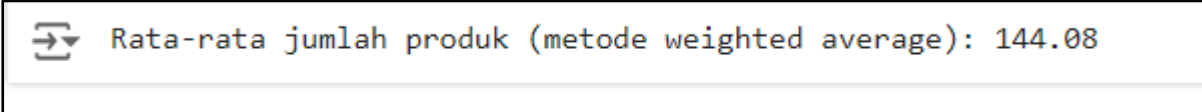
Penjelasan Source code:

Kode ini menggunakan metode defuzzifikasi untuk menghitung jumlah produk rata-rata berdasarkan nilai α -predikat yang dihasilkan sebelumnya. Pertama, dua daftar dibuat: `jumlah_produk`, yang mengandung nilai rata-rata jumlah produk untuk berbagai kombinasi input, dan `a_predikat`, yang menyimpan nilai α -predikat yang diperoleh dari langkah sebelumnya.

Untuk menghitung rata-rata jumlah produk, rumus berat rata-rata digunakan. Rumus ini diperoleh dengan perkalian nilai alpha-predikat dengan jumlah produk yang sesuai dan kemudian pembagian jumlah total tersebut dengan nilai alpha-predikat. Tujuan dari proses ini adalah untuk memberikan bobot yang lebih besar pada nilai-nilai produk dengan derajat keanggotaan yang lebih tinggi, yang menghasilkan estimasi jumlah produk yang lebih akurat.

Akhir sekali, kode mencetak jumlah produk rata-rata dalam dua angka desimal, memberikan gambaran yang jelas tentang hasil defuzzifikasi. Dalam situasi pengambilan keputusan berbasis fuzzy, di mana hasil akhir harus mempertimbangkan kombinasi dari berbagai kondisi input yang telah dipertimbangkan sebelumnya, metode ini sangat berguna.

Output:



```
➞ Rata-rata jumlah produk (metode weighted average): 144.08
```

Gambar 5.1 Output

1.3. Evaluasi perbedaan fungsi keanggotaan Trapesium dan FIS Sugeno terhadap hasil perhitungan

Pembahasan:

Perbedaan hasil yang didapat dari hasil dengan penggunaan fungsi keanggotaan trapesium dan FIS sugeno sebagai berikut:

- Fleksibilitas dan ketepatan

Fungsi keanggotaan Trapesium memberikan fleksibilitas yang lebih besar dalam menangkap variasi data, sedangkan FIS sugeno memberikan data yang lebih tepat, tetapi memungkinkan hilangnya beberapa detail dalam data.

- Responsivitas:

Hasil dari fungsi trapesium dapat lebih responsif terhadap perubahan kecil dalam input, sedangkan FIS sugeno mungkin memberikan hasil yang lebih stabil tetapi kurang melihat detail variasi kecil.

- Implementasi:

FIS sugeno lebih mudah untuk diimplementasikan dalam aplikasi yang memerlukan kecepatan dan efisiensi, sedangkan fungsi keanggotaan trapesium lebih cocok untuk aplikasi yang memerlukan analisis yang lebih detail dan kompleks.

1.4. Kesimpulan, bahas hasil yang anda peroleh setelah penerapan dengan kode Phyton

Pembahasan:

Fungsi keanggotaan trapesium, memberikan fleksibilitas yang signifikan dalam menangkap variasi data suhu, kebisingan, dan pencahayaan. Analisis menunjukkan bahwa fungsi keanggotaan trapesium lebih responsif terhadap perubahan kecil dalam input dibandingkan dengan FIS Sugeno, yang meskipun lebih stabil, dapat kehilangan beberapa detail penting. Dengan 27 aturan yang dirumuskan untuk menghubungkan kombinasi input dan output, sistem ini mampu menghasilkan output yang sesuai berdasarkan kondisi yang diberikan, seperti kombinasi suhu dingin dan kebisingan rendah yang menghasilkan tingkat produksi tinggi. Selain itu, FIS Sugeno lebih mudah diimplementasikan dalam aplikasi yang memerlukan kecepatan dan efisiensi, sementara fungsi keanggotaan trapesium lebih cocok untuk analisis yang lebih mendalam dan kompleks. Secara keseluruhan, penerapan logika fuzzy dalam analisis ini menunjukkan potensi besar dalam pengambilan keputusan berbasis data, dengan kemampuan untuk menangani ketidakpastian dan variasi dalam data input.