# MODUL PRAKTIKUM KECERDASAN BUATAN

## PERTEMUAN 4 – LOGIKA FUZZY

Tools : Jupyter notebook

Bahasa Pemrograman : Python

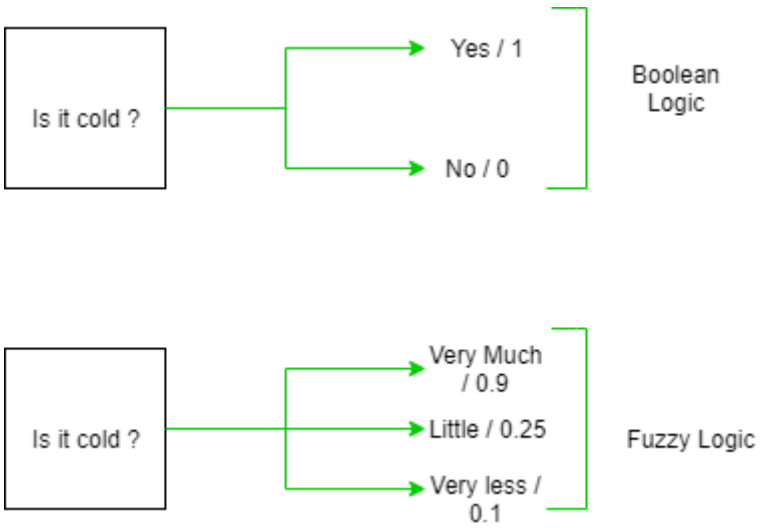
Dalam praktikum kali ini, kita akan belajar tentang bagaimana cara membangun Logika Fuzzy menggunakan Python.

### Logika Fuzzy

Fuzzy secara harfiah dapat diartikan sebagai hal-hal yang tidak jelas atau kabur. Di dunia nyata sering kali kita menghadapi situasi dimana kita tidak dapat menentukan apakah keadaan itu benar atau salah, oleh karena itu logika fuzzy dapat memberikan fleksibilitas yang sangat berharga dalam penalaran. Dengan cara ini, kita mendapatkan keleluasaan untuk mencari solusi terbaik dari suatu permasalahan.

Logika fuzzy pertama kali dikenalkan oleh Lotfi Zadeh pada tahun 1965 berdasarkan Teori Himpunan Fuzzy. Logika fuzzy mengandung beberapa nilai logika yang merupakan nilai kebenaran dari suatu variabel atau masalah antara 0 dan 1. Konsep ini memberikan kemungkinan-kemungkinan yang tidak diberikan oleh komputer, tetapi mirip dengan rentang kemungkinan yang dihasilkan oleh manusia.

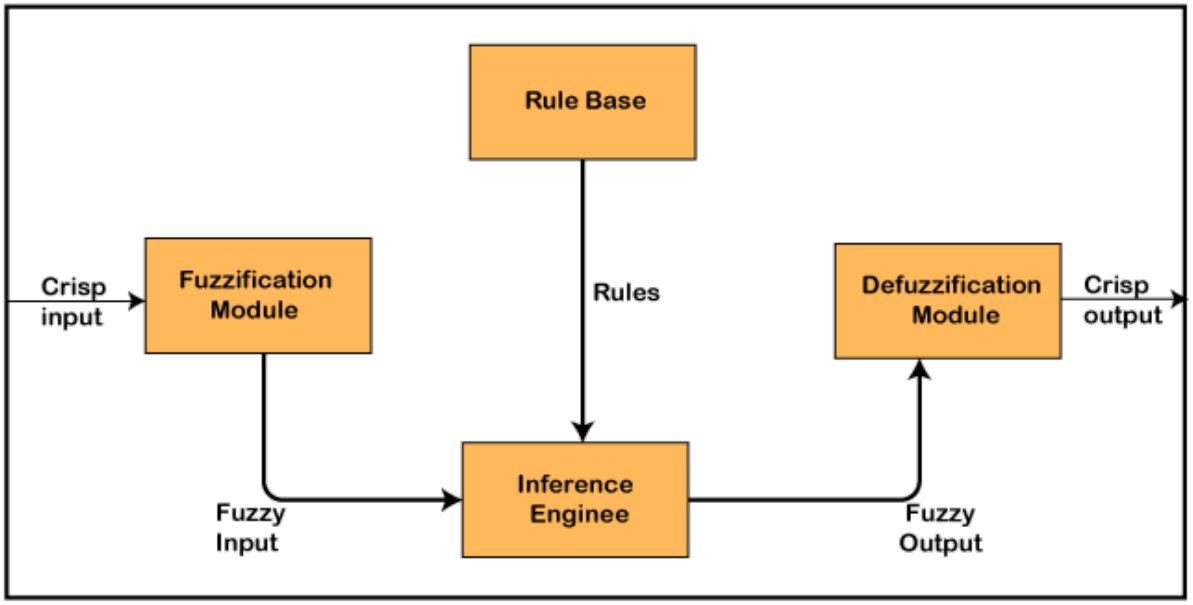
Dalam sistem Boolean, hanya ada dua kemungkinan (0 dan 1), di mana 1 menunjukkan nilai kebenaran mutlak dan 0 menunjukkan nilai salah mutlak. Tetapi dalam sistem Fuzzy, ada banyak kemungkinan yang berada diantara 0 dan 1, yang sebagian salah dan sebagian benar seperti yang terlihat pada gambar 4.1.




Gambar 4.1. Contoh Perbandingan Logika Boolean dan Logika Fuzzy

(https://www.geeksforgeeks.org/fuzzy-logic-introduction/ )

Adapun dalam membangun sebuah sistem logika fuzzy, terdapat arsitektur yang perlu diperhatikan. Arsitektur sistem fuzzy dapat dilihat pada gambar 4.2.




Gambar 4.1. Arsitektur Sistem Logika Fuzzy (https://www.javatpoint.com/fuzzy-logic )

Arsitektur sistem logika fuzzy terdiri dari 4 komponen yaitu:

1. Basis Pengetahuan (*Rule Base*)

   Basis pengetahuan berisi semua aturan dan kondisi **jika-maka** yang ditawarkan oleh para ahli untuk mengontrol sistem pengambilan keputusan.

2. Fuzzifikasi (*Fuzzification*)

   Kompnen ini mengubah input atau angka tegas (*crips*) menjadi himpunan fuzzy. Misalkan mengelompokkan variabel usia ke dalam: muda (M), parobaya (P), dan tua (T).

3. Mesin Inferensi (*Inference Enginee*)

   Komponen ini merupakan komponen utama dalam sistem logika fuzzy, karena semua informasi diproses di Mesin Inferensi. Ini memungkinkan pengguna untuk menemukan tingkat kecocokan antara input fuzzy saat ini dan aturan. Setelah pencocokan derajat, sistem ini menentukan aturan mana yang akan ditambahkan sesuai dengan bidang input yang diberikan.

4. Defuzzifikasi (*Defuzzification*)

   Komponen ini mengambil input himpunan fuzzy yang dihasilkan oleh Mesin Inferensi, dan kemudian mengubahnya menjadi nilai tegas (*crips*). Ini adalah langkah terakhir dalam proses sistem logika fuzzy. Nilai tegas adalah jenis nilai yang dapat diterima oleh pengguna.

Hal lain yang tidak kalah penting dalam logika fuzzy adalah fungsi keanggotaan (*membership function*). Fungsi keanggotaan merupakan grafik yang mendefinisikan bagaimana setiap titik dalam ruang input dipetakan ke nilai keanggotaan antara 0 dan 1. Ruang input sering disebut sebagai semesta wacana atau himpunan universal (u) yang berisi semua elemen yang mungkin menjadi perhatian di setiap bagian tertentu. Beberapa jenis fungsi keanggotaan yang cukup populer antara lain: Singleton, Gaussian, Trapesium, dan Segitiga.

### *Library Python* yang Digunakan

Python menyediakan *library* yang dapat digunakan untuk membangun logika fuzzy. Dalam sesi ini kita akan menggunakan salah satu *library* tersebut, yaitu Scikit-Fuzzy.

Scikit-Fuzzy adalah kumpulan algoritma logika fuzzy yang dimaksudkan untuk digunakan dalam SciPy Stack dan ditulis dalam bahasa komputasi Python. Paket ini mengimplementasikan banyak *tool* yang berguna untuk proyek yang melibatkan logika fuzzy atau logika abu-abu.

Untuk dapat memanfaatkan *library* ini maka perlu dilakukan instalasi terlebih dahulu melalui *command-prompt* dengan perintah **`pip install scikit-fuzzy`**.

### Penerapan Logika Fuzzy pada Data Employee (employee.csv)

Pada sesi kali ini, kita akan menerapkan Logika Fuzzy menggunakan Python pada dataset Employee (employee.csv). Langkah-langkah yang perlu dilakukan adalah:

1. Membaca dataset
2. Menentukan fungsi keanggotaan
3. Menghitung derajat keanggotaan
4. Menentukan status keanggotaan
5. Membuat basis pengetahuan fuzzy (*Fuzzy Rule-Based*)

### Langkah 1: Membaca Dataset

Langkah awal dalam sesi ini adalah melakukan pembacaan dataset. Dataset yang kita gunakan merupakan dataset employee dengan 5 atribut, yaitu: ID, Name, Age, Year of Service, dan Salary (dalam ribuan). Adapun proses pembacaan data set menggunakan python adalah sebagai berikut:

## Fuzzy Logic Implementation using Python

### Read data

In [3]:
```
import numpy as np
import pandas as pd
```

In [4]:
```
dataset = pd.read_csv("employee.csv")
dataset
```

*Output*

Out[4]:

| | ID | Name | Age | Years of Service | Salary |
|---|---|---|---|---|---|
| **0** | NO01 | Ani | 35 | 3 | 2940 |
| **1** | NO02 | Budi | 26 | 2 | 3440 |
| **2** | NO03 | Risa | 30 | 6 | 3000 |
| **3** | NO04 | Soni | 41 | 12 | 3800 |
| **4** | NO05 | Huda | 39 | 13 | 6400 |
| **5** | NO06 | Rani | 34 | 1 | 2200 |
| **6** | NO07 | Ria | 37 | 4 | 4160 |
| **7** | NO08 | Ridho | 49 | 17 | 6000 |
| **8** | NO09 | Rusdi | 37 | 5 | 5000 |
| **9** | NO10 | Burhan | 36 | 14 | 5020 |

Selanjutnya akan dilakukan pengambilan nilai dari masing-masing atribut.

1. Mendapatkan dan menampilkan nilai pada atribut “Age”

```
In [4]: #Get the Age attribute's value
        dage=np.array(dataset)
        dage=dage[:,[0,2]]
        print("Index 0 = ID, Index 1 = Age")
        pd.DataFrame(dage)
```

*Output*

```
Index 0 = ID, Index 1 = Age
```

Out[4]:

| | 0 | 1 |
|---|---|---|
| **0** | NO01 | 35 |
| **1** | NO02 | 26 |
| **2** | NO03 | 30 |
| **3** | NO04 | 41 |
| **4** | NO05 | 39 |
| **5** | NO06 | 34 |
| **6** | NO07 | 37 |
| **7** | NO08 | 49 |
| **8** | NO09 | 37 |
| **9** | NO10 | 36 |

In [8]:
```
age01=np.sum(dage[0:1,1],axis=0)
age02=np.sum(dage[1:2,1],axis=0)
age03=np.sum(dage[2:3,1],axis=0)
age04=np.sum(dage[3:4,1],axis=0)
age05=np.sum(dage[4:5,1],axis=0)
age06=np.sum(dage[5:6,1],axis=0)
age07=np.sum(dage[6:7,1],axis=0)
age08=np.sum(dage[7:8,1],axis=0)
age09=np.sum(dage[8:9,1],axis=0)
age10=np.sum(dage[9:10,1],axis=0)
print("age 01:",age01)
print("age 02:",age02)
print("age 03:",age03)
print("age 04:",age04)
print("age 05:",age05)
print("age 06:",age06)
print("age 07:",age07)
print("age 08:",age08)
print("age 09:",age09)
print("age 10:",age10)
```

*Output*

```
age 01: 35
age 02: 26
age 03: 30
age 04: 41
age 05: 39
age 06: 34
age 07: 37
age 08: 49
age 09: 37
age 10: 36
```

2. Mendapatkan dan menampilkan nilai pada atribut “Year of Service”

In [7]:
```
#Get the Year of Service attribute's value
dyos=np.array(dataset)
dyos=dyos[:,[0,3]]
print("Index 0 = ID, Index 1 = Year of Service")
pd.DataFrame(dyos)
```

*Output*

```
Index 0 = ID, Index 1 = Year of Service
```

Out[7]:

| | 0 | 1 |
|---|---|---|
| 0 | NO01 | 3 |
| 1 | NO02 | 2 |
| 2 | NO03 | 6 |
| 3 | NO04 | 12 |
| 4 | NO05 | 13 |
| 5 | NO06 | 1 |
| 6 | NO07 | 4 |
| 7 | NO08 | 17 |
| 8 | NO09 | 5 |
| 9 | NO10 | 14 |

In [10]:
```
yos01=np.sum(dyos[0:1,1],axis=0)
yos02=np.sum(dyos[1:2,1],axis=0)
yos03=np.sum(dyos[2:3,1],axis=0)
yos04=np.sum(dyos[3:4,1],axis=0)
yos05=np.sum(dyos[4:5,1],axis=0)
yos06=np.sum(dyos[5:6,1],axis=0)
yos07=np.sum(dyos[6:7,1],axis=0)
yos08=np.sum(dyos[7:8,1],axis=0)
yos09=np.sum(dyos[8:9,1],axis=0)
yos10=np.sum(dyos[9:10,1],axis=0)
print("YoS 01:",yos01)
print("YoS 02:",yos02)
print("YoS 03:",yos03)
print("YoS 04:",yos04)
print("YoS 05:",yos05)
print("YoS 06:",yos06)
print("YoS 07:",yos07)
print("YoS 08:",yos08)
print("YoS 09:",yos09)
print("YoS 10:",yos10)
```

*Output*

```
YoS 01: 3
YoS 02: 2
YoS 03: 6
YoS 04: 12
YoS 05: 13
YoS 06: 1
YoS 07: 4
YoS 08: 17
YoS 09: 5
YoS 10: 14
```

3. Mendapatkan dan menampilkan nilai pada atribut “Salary”

In [8]:
```
#Get the Salary attribute's value
dsalary=np.array(dataset)
dsalary=dsalary[:,[0,4]]
print("Index 0 = ID, Index 1 = Salary")
pd.DataFrame(dsalary)
```

*Output*

```
Index 0 = ID, Index 1 = Salary
```

Out[8]:

| | 0 | 1 |
|---|---|---|
| **0** | NO01 | 2940 |
| **1** | NO02 | 3440 |
| **2** | NO03 | 3000 |
| **3** | NO04 | 3800 |
| **4** | NO05 | 6400 |
| **5** | NO06 | 2200 |
| **6** | NO07 | 4160 |
| **7** | NO08 | 6000 |
| **8** | NO09 | 5000 |
| **9** | NO10 | 5020 |

In [11]:
```
salary01=np.sum(dsalary[0:1,1],axis=0)
salary02=np.sum(dsalary[1:2,1],axis=0)
salary03=np.sum(dsalary[2:3,1],axis=0)
salary04=np.sum(dsalary[3:4,1],axis=0)
salary05=np.sum(dsalary[4:5,1],axis=0)
salary06=np.sum(dsalary[5:6,1],axis=0)
salary07=np.sum(dsalary[6:7,1],axis=0)
salary08=np.sum(dsalary[7:8,1],axis=0)
salary09=np.sum(dsalary[8:9,1],axis=0)
salary10=np.sum(dsalary[9:10,1],axis=0)
print("salary 01:",salary01)
print("salary 02:",salary02)
print("salary 03:",salary03)
print("salary 04:",salary04)
print("salary 05:",salary05)
print("salary 06:",salary06)
print("salary 07:",salary07)
print("salary 08:",salary08)
print("salary 09:",salary09)
print("salary 10:",salary10)
```

*Output*

```
salary 01: 2940
salary 02: 3440
salary 03: 3000
salary 04: 3800
salary 05: 6400
salary 06: 2200
salary 07: 4160
salary 08: 6000
salary 09: 5000
salary 10: 5020
```

## Langkah 2: Menentukan Fungsi Keanggotaan (*Membership Function*)

Pada tahap ini, kita akan menentukan jenis fungsi keanggotaan dan rentang nilai yang akan digunakan berdasarkan 3 atribut (Age, Year of Servioce, dan Salary). Kali ini kita akan menggunakan fungsi keanggotaan segitiga (*Triangular Membership Function*) dimana fungsi tersebut sudah disediakan oleh library scikit-fuzzy (trimf).

**Create Fuzzy Set**

In [12]:
```
import skfuzzy as fuz
import matplotlib.pyplot as plt
```

***Generate Triangular Membership Function***

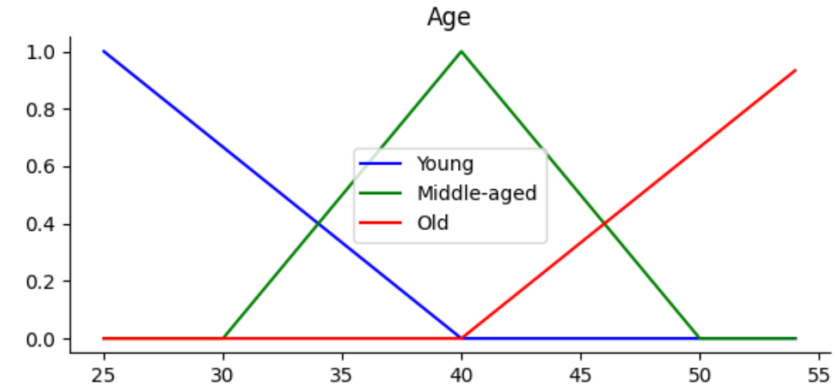
skfuzzy.membership.trimf(x, abc)

In [13]:
```
def OurRange (_lo,_hi,_step):
    ourr=np.arange(_lo,_hi,_step)
    return ourr

def MembershipFunc1 (_rule, _our_range, _title, _label0, _label1, _label2):
    low=fuz.trimf(_our_range, _rule[0])
    middle=fuz.trimf(_our_range, _rule[1])
    high=fuz.trimf(_our_range, _rule[2])

    fig, ax=plt.subplots(nrows=1,figsize=(6,3))
    ax.plot(_our_range, low, 'b', linewidth=1.5, label=_label0)
    ax.plot(_our_range, middle, 'g', linewidth=1.5, label=_label1)
    ax.plot(_our_range, high, 'r', linewidth=1.5, label=_label2)

    ax.set_title(_title)
    ax.legend()

    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.get_xaxis().tick_bottom()
    ax.get_yaxis().tick_left()

    plt.tight_layout()
    plt.show()

    return low, middle, high

def MembershipFunc2 (_rule, _our_range, _title, _label0, _label1):
    low=fuz.trimf(_our_range, _rule[0])
    high=fuz.trimf(_our_range, _rule[1])

    fig, ax=plt.subplots(nrows=1,figsize=(6,3))
    ax.plot(_our_range, low, 'b', linewidth=1.5, label=_label0)
    ax.plot(_our_range, high, 'r', linewidth=1.5, label=_label1)


    ax.set_title(_title)
    ax.legend()

    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.get_xaxis().tick_bottom()
    ax.get_yaxis().tick_left()

    plt.tight_layout()
    plt.show()

    return low, high
```

1. Mendefinisikan fungsi keanggotaan untuk atribut “Age”

In [16]:
```
xage = OurRange(25,55,1)
rage = np.array([
    [0,25,40],
    [30,40,50],
    [40,55,55]
])
lo_age,mi_age,hi_age=MembershipFunc1(rage,xage,'Age','Young','Middle-aged','Old')
```

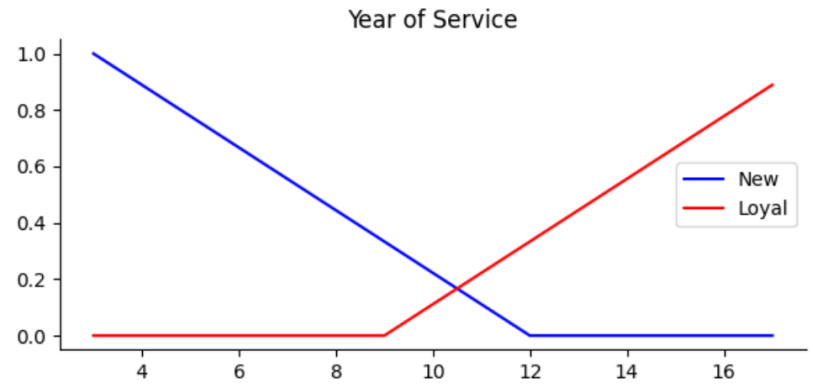
*Output*




2. Mendefinisikan fungsi keanggotaan untuk atribut “Year of Service”

```
In [40]: xyos = OurRange(3,18,1)
ryos = np.array([
    [0,3,12],
    [9,18,18]
])
lo_yos,hi_yos=MembershipFunc2(ryos,xyos,'Year of Service','New','Loyal')
```

*Output*




3. Mendefinisikan fungsi keanggotaan untuk atribut “Salary”

```
In [41]: xsalary = OurRange(1000,7000,1)
rsalary = np.array([
    [0,1000,3500],
    [2000,4000,6000],
    [4500,7000,7000]
])
lo_salary,mi_salary,hi_salary=MembershipFunc1(rsalary,xsalary,'Salary','Low','Middle','High')
```

*Output*


## Langkah 3 & 4: Menghitung Derajat Keanggotaan dan Menentukan Status Keanggotaan

Sebelumnya kita telah mendefinisikan fungsi keanggotaan dari atribut Age, Year of Service, dan Salary. Berikutnya, kita akan menghitung derajat keanggotaan dari masing-masing subyek/ *record* pada dataset employee.

**Calculate Membership Degrees**

In [42]:
```
def MembershipDeg1 (_range, _lo, _mi, _hi, _val):
    low=fuz.interp_membership(_range,_lo,_val)
    middle=fuz.interp_membership(_range,_mi,_val)
    high=fuz.interp_membership(_range,_hi,_val)

    return low, middle, high

def MembershipDeg2 (_range, _lo, _hi, _val):
    low=fuz.interp_membership(_range,_lo,_val)
    high=fuz.interp_membership(_range,_hi,_val)

    return low, high
```

***Indicate membership status***

In [43]:
```
def Status1 (_membership,_label0,_label1,_label2):
    status=''
    if _membership[0]>_membership[1] and _membership[0]>_membership[2]:
        status=_label0
    elif _membership[1]>_membership[0] and _membership[1]>_membership[2]:
        status=_label1
    elif _membership[2]>_membership[0] and _membership[2]>_membership[1]:
        status=_label2
    return status

def Status2 (_membership,_label0,_label1):
    status=''
    if _membership[0]>_membership[1]:
        status=_label0
    elif _membership[1]>_membership[0]:
        status=_label1
    return status
```

***Show membership degrees and membership status***

In [45]:
```
age_member_01=MembershipDeg1(xage,lo_age,mi_age,hi_age,age01)
age_member_02=MembershipDeg1(xage,lo_age,mi_age,hi_age,age02)
age_member_03=MembershipDeg1(xage,lo_age,mi_age,hi_age,age03)
age_member_04=MembershipDeg1(xage,lo_age,mi_age,hi_age,age04)
age_member_05=MembershipDeg1(xage,lo_age,mi_age,hi_age,age05)
age_member_06=MembershipDeg1(xage,lo_age,mi_age,hi_age,age06)
age_member_07=MembershipDeg1(xage,lo_age,mi_age,hi_age,age07)
age_member_08=MembershipDeg1(xage,lo_age,mi_age,hi_age,age08)
age_member_09=MembershipDeg1(xage,lo_age,mi_age,hi_age,age09)
age_member_10=MembershipDeg1(xage,lo_age,mi_age,hi_age,age10)

print("age_member_01: ",age_member_01)
print("age_member_02: ",age_member_02)
print("age_member_03: ",age_member_03)
print("age_member_04: ",age_member_04)
print("age_member_05: ",age_member_05)
print("age_member_06: ",age_member_06)
print("age_member_07: ",age_member_07)
print("age_member_08: ",age_member_08)
print("age_member_09: ",age_member_09)
print("age_member_10: ",age_member_10)

age_status_01=Status1(age_member_01,'young','middle-aged','old')
age_status_02=Status1(age_member_02,'young','middle-aged','old')
age_status_03=Status1(age_member_03,'young','middle-aged','old')
age_status_04=Status1(age_member_04,'young','middle-aged','old')
age_status_05=Status1(age_member_05,'young','middle-aged','old')
```

```
age_status_06=Status1(age_member_06,'young','middle-aged','old')
age_status_07=Status1(age_member_07,'young','middle-aged','old')
age_status_08=Status1(age_member_08,'young','middle-aged','old')
age_status_09=Status1(age_member_09,'young','middle-aged','old')
age_status_10=Status1(age_member_10,'young','middle-aged','old')

print("age_status_01: ",age_status_01)
print("age_status_02: ",age_status_02)
print("age_status_03: ",age_status_03)
print("age_status_04: ",age_status_04)
print("age_status_05: ",age_status_05)
print("age_status_06: ",age_status_06)
print("age_status_07: ",age_status_07)
print("age_status_08: ",age_status_08)
print("age_status_09: ",age_status_09)
print("age_status_10: ",age_status_10)
```

*Output*

```
age_member_01:  (0.3333333333333333, 0.5, 0.0)
age_member_02:  (0.9333333333333333, 0.0, 0.0)
age_member_03:  (0.6666666666666666, 0.0, 0.0)
age_member_04:  (0.0, 0.9, 0.06666666666666667)
age_member_05:  (0.06666666666666667, 0.9, 0.0)
age_member_06:  (0.4, 0.4, 0.0)
age_member_07:  (0.2, 0.7, 0.0)
age_member_08:  (0.0, 0.1, 0.6)
age_member_09:  (0.2, 0.7, 0.0)
age_member_10:  (0.26666666666666666, 0.6, 0.0)
age_status_01:  middle-aged
age_status_02:  young
age_status_03:  young
age_status_04:  middle-aged
age_status_05:  middle-aged
age_status_06:
age_status_07:  middle-aged
age_status_08:  old
age_status_09:  middle-aged
age_status_10:  middle-aged
```

In [46]:
```
yos_member_01=MembershipDeg2(xyos,lo_yos,hi_yos,yos01)
yos_member_02=MembershipDeg2(xyos,lo_yos,hi_yos,yos02)
yos_member_03=MembershipDeg2(xyos,lo_yos,hi_yos,yos03)
yos_member_04=MembershipDeg2(xyos,lo_yos,hi_yos,yos04)
yos_member_05=MembershipDeg2(xyos,lo_yos,hi_yos,yos05)
yos_member_06=MembershipDeg2(xyos,lo_yos,hi_yos,yos06)
yos_member_07=MembershipDeg2(xyos,lo_yos,hi_yos,yos07)
yos_member_08=MembershipDeg2(xyos,lo_yos,hi_yos,yos08)
yos_member_09=MembershipDeg2(xyos,lo_yos,hi_yos,yos09)
yos_member_10=MembershipDeg2(xyos,lo_yos,hi_yos,yos10)

print("yos_member_01: ",yos_member_01)
print("yos_member_02: ",yos_member_02)
print("yos_member_03: ",yos_member_03)
print("yos_member_04: ",yos_member_04)
print("yos_member_05: ",yos_member_05)
print("yos_member_06: ",yos_member_06)
print("yos_member_07: ",yos_member_07)
print("yos_member_08: ",yos_member_08)
print("yos_member_09: ",yos_member_09)
print("yos_member_10: ",yos_member_10)

yos_status_01=Status2(yos_member_01,'new','loyal')
yos_status_02=Status2(yos_member_02,'new','loyal')
yos_status_03=Status2(yos_member_03,'new','loyal')
yos_status_04=Status2(yos_member_04,'new','loyal')
yos_status_05=Status2(yos_member_05,'new','loyal')
yos_status_06=Status2(yos_member_06,'new','loyal')
yos_status_07=Status2(yos_member_07,'new','loyal')
yos_status_08=Status2(yos_member_08,'new','loyal')
yos_status_09=Status2(yos_member_09,'new','loyal')
yos_status_10=Status2(yos_member_10,'new','loyal')

print("yos_status_01: ",yos_status_01)
print("yos_status_02: ",yos_status_02)
print("yos_status_03: ",yos_status_03)
print("yos_status_04: ",yos_status_04)
print("yos_status_05: ",yos_status_05)
print("yos_status_06: ",yos_status_06)
print("yos_status_07: ",yos_status_07)
print("yos_status_08: ",yos_status_08)
print("yos_status_09: ",yos_status_09)
print("yos_status_10: ",yos_status_10)
```

*Output*

```
yos_member_01:  (1.0, 0.0)
yos_member_02:  (0.0, 0.0)
yos_member_03:  (0.6666666666666666, 0.0)
yos_member_04:  (0.0, 0.3333333333333333)
yos_member_05:  (0.0, 0.4444444444444444)
yos_member_06:  (0.0, 0.0)
yos_member_07:  (0.8888888888888888, 0.0)
yos_member_08:  (0.0, 0.8888888888888888)
yos_member_09:  (0.7777777777777778, 0.0)
yos_member_10:  (0.0, 0.5555555555555556)
yos_status_01:  new
yos_status_02:
yos_status_03:  new
yos_status_04:  loyal
yos_status_05:  loyal
yos_status_06:
yos_status_07:  new
yos_status_08:  loyal
yos_status_09:  new
yos_status_10:  loyal
```

In [47]:
```
salary_member_01=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary01)
salary_member_02=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary02)
salary_member_03=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary03)
salary_member_04=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary04)
salary_member_05=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary05)
salary_member_06=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary06)
salary_member_07=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary07)
salary_member_08=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary08)
salary_member_09=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary09)
salary_member_10=MembershipDeg1(xsalary,lo_salary,mi_salary,hi_salary,salary10)

print("salary_member_01: ",salary_member_01)
print("salary_member_02: ",salary_member_02)
print("salary_member_03: ",salary_member_03)
print("salary_member_04: ",salary_member_04)
print("salary_member_05: ",salary_member_05)
print("salary_member_06: ",salary_member_06)
print("salary_member_07: ",salary_member_07)
print("salary_member_08: ",salary_member_08)
print("salary_member_09: ",salary_member_09)
print("salary_member_10: ",salary_member_10)

salary_status_01=Status1(salary_member_01,'low','medium','high')
salary_status_02=Status1(salary_member_02,'low','medium','high')
salary_status_03=Status1(salary_member_03,'low','medium','high')
salary_status_04=Status1(salary_member_04,'low','medium','high')
salary_status_05=Status1(salary_member_05,'low','medium','high')
salary_status_06=Status1(salary_member_06,'low','medium','high')
salary_status_07=Status1(salary_member_07,'low','medium','high')
salary_status_08=Status1(salary_member_08,'low','medium','high')
salary_status_09=Status1(salary_member_09,'low','medium','high')
salary_status_10=Status1(salary_member_10,'low','medium','high')

print("salary_status_01: ",salary_status_01)
print("salary_status_02: ",salary_status_02)
print("salary_status_03: ",salary_status_03)
print("salary_status_04: ",salary_status_04)
print("salary_status_05: ",salary_status_05)
print("salary_status_06: ",salary_status_06)
print("salary_status_07: ",salary_status_07)
print("salary_status_08: ",salary_status_08)
print("salary_status_09: ",salary_status_09)
print("salary_status_10: ",salary_status_10)
```

*Output*

```
salary_member_01:  (0.224, 0.47, 0.0)
salary_member_02:  (0.024, 0.72, 0.0)
salary_member_03:  (0.2, 0.5, 0.0)
salary_member_04:  (0.0, 0.9, 0.0)
salary_member_05:  (0.0, 0.0, 0.76)
salary_member_06:  (0.52, 0.1, 0.0)
salary_member_07:  (0.0, 0.92, 0.0)
salary_member_08:  (0.0, 0.0, 0.6)
salary_member_09:  (0.0, 0.5, 0.2)
salary_member_10:  (0.0, 0.49, 0.208)
salary_status_01:  medium
salary_status_02:  medium
salary_status_03:  medium
salary_status_04:  medium
salary_status_05:  high
salary_status_06:  low
salary_status_07:  medium
salary_status_08:  high
salary_status_09:  medium
salary_status_10:  medium
```

## Langkah 5: Membuat basis pengetahuan fuzzy (*Fuzzy Rule-Based*)

Pada langkah ini kita akan menentukan basis pengetahuan dengan aturan IF-THEN untuk menentukan seorang karyawan termasuk karyawan kontrak atau karyawan tetap.

Potongan kode 1

***Create Fuzzy Rule-based***

```
In [82]: def RuleBased (_age_status, _yos_status, _salary_status):
             emp_status=""
             if (_age_status=='young' or _age_status=='middle-aged') and _yos_status=='new'and (_salary_status==
                 emp_status="a contract employee"
             elif (_age_status=='middle-aged' or _age_status=='old') and _yos_status=='loyal' and (_salary_statu
                 emp_status="a permanent employee"
             else:
                 emp_status="unknown"
             return emp_status
```

Potongan kode 2

***Create Fuzzy Rule-based***

```
In [82]: , _salary_status):

atus=='middle-aged') and _yos_status=='new'and (_salary_status=='low' or _salary_status=='medium') :
"
 _age_status=='old') and _yos_status=='loyal' and (_salary_status=='medium' or _salary_status=='high'):
e"
```

***Show the outcome of the Fuzzy Rule-based***

In [81]:
```
employee_status01=RuleBased (age_status_01, yos_status_01, salary_status_01)
employee_status02=RuleBased (age_status_02, yos_status_02, salary_status_02)
employee_status03=RuleBased (age_status_03, yos_status_03, salary_status_03)
employee_status04=RuleBased (age_status_04, yos_status_04, salary_status_04)
employee_status05=RuleBased (age_status_05, yos_status_05, salary_status_05)
employee_status06=RuleBased (age_status_06, yos_status_06, salary_status_06)
employee_status07=RuleBased (age_status_07, yos_status_07, salary_status_07)
employee_status08=RuleBased (age_status_08, yos_status_08, salary_status_08)
employee_status09=RuleBased (age_status_09, yos_status_09, salary_status_09)
employee_status10=RuleBased (age_status_10, yos_status_10, salary_status_10)

print("employee_status01: ", employee_status01)
print("employee_status02: ", employee_status02)
print("employee_status03: ", employee_status03)
print("employee_status04: ", employee_status04)
print("employee_status05: ", employee_status05)
print("employee_status06: ", employee_status06)
print("employee_status07: ", employee_status07)
print("employee_status08: ", employee_status08)
print("employee_status09: ", employee_status09)
print("employee_status10: ", employee_status10)
```

*Output*

```
employee_status01:  a contract employee
employee_status02:  unknown
employee_status03:  a contract employee
employee_status04:  a permanent employee
employee_status05:  a permanent employee
employee_status06:  unknown
employee_status07:  a contract employee
employee_status08:  a permanent employee
employee_status09:  a contract employee
employee_status10:  a permanent employee
```

## Latihan: Uji Coba Menggunakan Berbagai Jenis Fungsi Keanggotaan (*Membership Function*)

1. Gunakan fungsi keanggotaan Trapesium dan Gaussian untuk data yang sama dan tunjukkan hasilnya!
2. Apakah terjadi perbedaan nilai derajat keanggotaan pada saat menggunakan jenis fungsi keanggotaan yang berbeda? Tunjukkan dan jelaskan!
3. Tampilkan hasil penentuan karyawan tetap atau karyawan kontrak beserta nama karyawannya!

## Referensi:


https://pythonhosted.org/scikit-fuzzy/overview.html

https://www.geeksforgeeks.org/fuzzy-logic-introduction/

https://www.edureka.co/blog/fuzzy-logic-ai/
https://www.javatpoint.com/fuzzy-logic