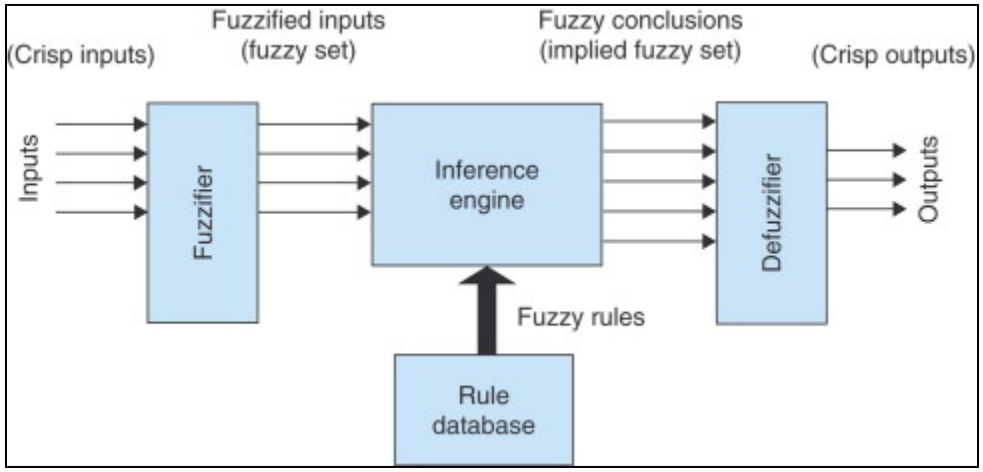
Aim - To design a Fuzzy control system using Fuzzy tool/library in Python.

Theory -

Explain the fuzzy control with an example.

Fuzzy control, also known as fuzzy logic control, is a method of control that uses fuzzy logic to make decisions based on imprecise or uncertain information. It's particularly useful in situations where traditional control methods may struggle to handle complex, nonlinear systems.



Let's take the example of a washing machine to understand how fuzzy control works:

Fuzzy Control in a Washing Machine:

Fuzzy control is a smart way to manage a washing machine using fuzzy logic, which deals with imprecise information. Imagine your washing machine needs to decide how much water to use and how long to wash based on how dirty the clothes are.

Instead of strict rules, fuzzy logic allows for gradual decisions. For instance, instead of just "dirty" or "clean," clothes can be "slightly dirty," "moderately dirty," or "very dirty."

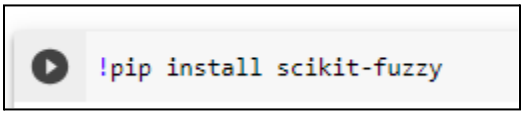
Fuzzy control involves:

1. Input Fuzzification: Turning crisp data like dirtiness levels into fuzzy categories, with defined membership functions to represent the degrees of membership.

2. Rule Creation: Making fuzzy rules like "If clothes are very dirty, use more water and wash longer." These rules are based on expert knowledge.

3. Inference: Combining rules and fuzzified inputs to decide on actions. Each rule contributes according to its level of fit.
4. Aggregation: Combining the actions suggested by different rules to get a final action plan.
5. Output Defuzzification: Translating the aggregated fuzzy plan back into crisp actions, like the specific wash time and water level.
So, fuzzy control lets your washing machine intelligently adjust its settings based on uncertain information, ensuring efficient cleaning while adapting to varying dirtiness levels and water amounts.

Code -
Installing the fuzzy library



Code 1 - Only output water level

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

# Create fuzzy variables
dirtiness = ctrl.Antecedent(np.arange(0, 11, 1), 'dirtiness')
water_level = ctrl.Consequent(np.arange(0, 11, 1), 'water_level')

# Create membership functions for input and output variables
dirtiness['very_low'] = fuzz.trimf(dirtiness.universe, [0, 0, 2.5])
dirtiness['low'] = fuzz.trimf(dirtiness.universe, [0, 2.5, 5])
dirtiness['medium'] = fuzz.trimf(dirtiness.universe, [2.5, 5, 7.5])
dirtiness['high'] = fuzz.trimf(dirtiness.universe, [5, 7.5, 10])
dirtiness['very_high'] = fuzz.trimf(dirtiness.universe, [7.5, 10, 10])

water_level.automf(5)  # Automatically create membership functions

# Create rules
rule1 = ctrl.Rule(dirtiness['very_low'], water_level['poor'])
rule2 = ctrl.Rule(dirtiness['low'], water_level['mediocre'])
rule3 = ctrl.Rule(dirtiness['medium'], water_level['average'])
rule4 = ctrl.Rule(dirtiness['high'], water_level['decent'])
```

```
rule5 = ctrl.Rule(dirtiness['very_high'], water_level['good'])

# Create control system
washing_ctrl = ctrl.ControlSystem([rule1, rule2, rule3, rule4, rule5])
washing_machine = ctrl.ControlSystemSimulation(washing_ctrl)

# Input dirtiness value
washing_machine.input['dirtiness'] = 6.231

# Perform fuzzy inference
washing_machine.compute()

# Output water level
print("Output Water Level:", washing_machine.output['water_level'])
```

Code 2 - Output water level and graphs

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

# Create fuzzy variables
dirtiness = np.arange(0, 11, 1)
water_level = np.arange(0, 11, 1)

# Create membership functions
dirtiness_membership = {
    'very_low': fuzz.trimf(dirtiness, [0, 0, 2.5]),
    'low': fuzz.trimf(dirtiness, [0, 2.5, 5]),
    'medium': fuzz.trimf(dirtiness, [2.5, 5, 7.5]),
    'high': fuzz.trimf(dirtiness, [5, 7.5, 10]),
    'very_high': fuzz.trimf(dirtiness, [7.5, 10, 10])
}

water_level_membership = {
    'very_low': fuzz.trimf(water_level, [0, 0, 2.5]),
    'low': fuzz.trimf(water_level, [0, 2.5, 5]),
    'medium': fuzz.trimf(water_level, [2.5, 5, 7.5]),
    'high': fuzz.trimf(water_level, [5, 7.5, 10]),
    'very_high': fuzz.trimf(water_level, [7.5, 10, 10])
}
```

```
# Plot membership functions
plt.figure(figsize=(4, 4))

for membership_name, membership_func in dirtiness_membership.items():
    plt.plot(dirtiness, membership_func, label=membership_name)

plt.title('Dirtiness Membership Functions')
plt.xlabel('Dirtiness')
plt.ylabel('Membership')
plt.legend()
plt.show()

plt.figure(figsize=(4, 4))

for membership_name, membership_func in water_level_membership.items():
    plt.plot(water_level, membership_func, label=membership_name)

plt.title('Water Level Membership Functions')
plt.xlabel('Water Level')
plt.ylabel('Membership')
plt.legend()
plt.show()

# Input dirtiness value
input_dirtiness = 6.89

# Calculate output water level using membership functions
output_water_level = np.zeros_like(water_level)

for membership_name, membership_func in dirtiness_membership.items():
    membership_degree = fuzz.interp_membership(dirtiness, membership_func,
input_dirtiness)
                output_water_level    =    np.maximum(output_water_level,
np.fmin(membership_degree, water_level_membership[membership_name]))

# Plot input/output relationship
plt.figure(figsize=(4, 4))
plt.plot(water_level, output_water_level, label='Output Water Level')
plt.title('Water Level Control Based on Dirtiness')
```

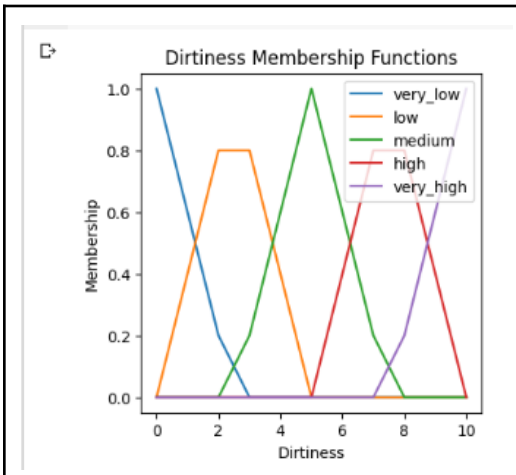
```
plt.xlabel('Water Level')
plt.ylabel('Membership')
plt.legend()
plt.show()

# Output water level
print("Output Water Level:", washing_machine.output['water_level'])
```

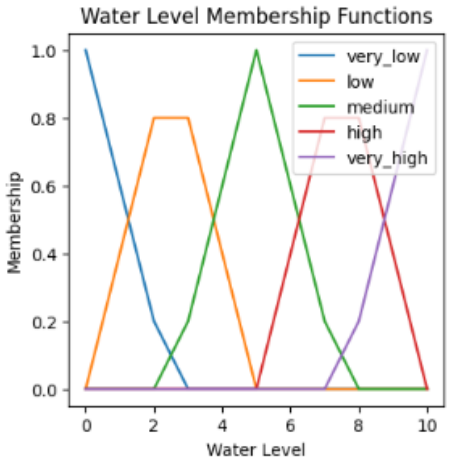
Output -

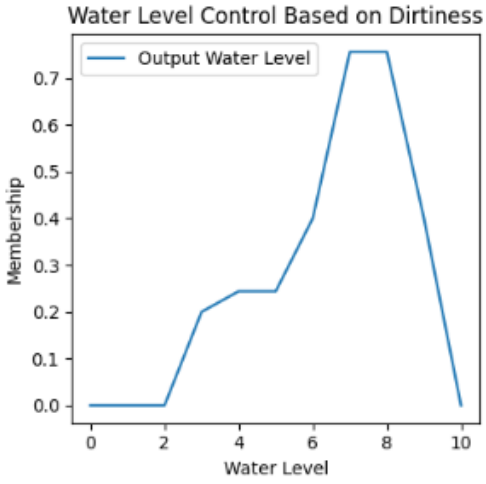
Output 1 -

```
Output Water Level: 6.176546888025046
```

Output 2 -








```
Output Water Level: 6.176546888025046
```

Conclusion - Thus, we have successfully understood how to build a fuzzy control system. We have considered the washing machine, where the factors under consideration are dirtiness and water level.