Aim: To implement fuzzy set Properties

Theory:

Fuzzy set theory is an extension of classical set theory that allows elements to have partial membership in a set.

In classical (crisp) sets, an element can only belong fully (membership = 1) or not belong at all (membership = 0).

However, in fuzzy sets, the membership value of an element can be any number between 0 and 1, representing degrees of belonging.

A fuzzy set A is represented as a collection of ordered pairs:
$A = \{(x, \mu_A(x)) \mid x \in X\}$
where:
- x is an element from the universal set X,
- $\mu_A(x)$ is the membership function of A, giving a value between 0 and 1.

1. **Union of Fuzzy Sets**

The union of two fuzzy sets A and B combines their memberships by taking the maximum value for each element:

$$\mu_{A \cup B}(x) = max(\mu_A(x), \mu_B(x))$$

This shows that the element belongs to at least one of the sets.

2. **Intersection of Fuzzy Sets**

The intersection of two fuzzy sets is defined by the minimum membership value for each element:

$$\mu_{A \cap B}(x) = min(\mu_A(x), \mu_B(x))$$

This shows the degree to which the element belongs to both sets.

3. **Complement of a Fuzzy Set**

The complement of a fuzzy set AAA indicates how much an element does not belong to the set:

$$\mu_{A'}(x) = 1 - \mu_A(x)$$

4. **Scalar Multiplication of a Fuzzy Set**

Scalar multiplication changes the degree of membership by multiplying it with a constant α, where 0 ≤ α ≤ 10 \leq \alpha \leq 10≤α≤1:

$$\mu_{\alpha A}(x) = \alpha . \mu_A(x)$$

5. **Sum of Fuzzy Sets**

The sum of two fuzzy sets adds their membership values but ensures the result does not exceed 1:

$$\mu_{A+B}(x) = min(1, \mu_A(x) + \mu_B(x))$$

Applications of Fuzzy Set Properties

Fuzzy sets are widely used in areas where human-like reasoning and approximate values are required, such as:

- Weather prediction (e.g., low, medium, high rainfall)
- Medical diagnosis
- Decision-making systems
- Control systems in appliances

Code:

```
# Import required library
import matplotlib.pyplot as plt
```

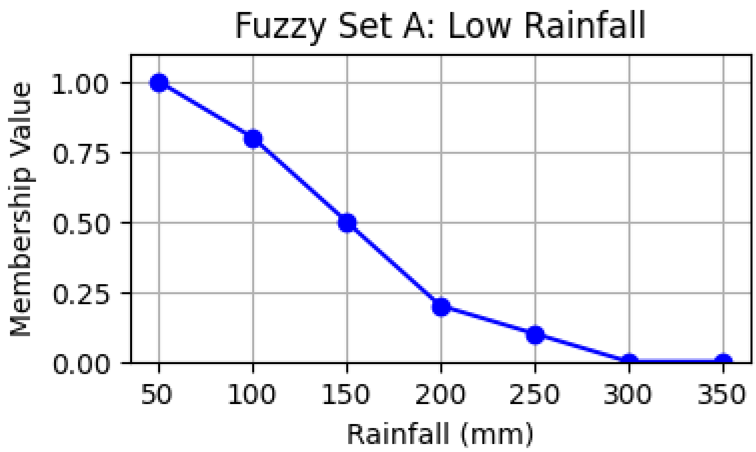
```
# Rainfall data points in mm
rainfall = [50, 100, 150, 200, 250, 300, 350]

# Fuzzy Set A: Low Rainfall (membership values)
low_rainfall = [1.0, 0.8, 0.5, 0.2, 0.1, 0.0, 0.0]

# Fuzzy Set B: High Rainfall (membership values)
high_rainfall = [0.0, 0.1, 0.3, 0.6, 0.8, 0.9, 1.0]
```

```
# 1. Union (A ∪ B)
union_AB = [max(a, b) for a, b in zip(low_rainfall, high_rainfall)]

# 2. Intersection (A ∩ B)
intersection_AB = [min(a, b) for a, b in zip(low_rainfall, high_rainfall)]

# 3. Complement of A
complement_A = [1 - a for a in low_rainfall]

# 4. Scalar Multiplication of A (α = 0.5)
alpha = 0.5
scalar_mult_A = [alpha * a for a in low_rainfall]

# 5. Sum of Fuzzy Sets (A + B)
sum_AB = [min(1, a + b) for a, b in zip(low_rainfall, high_rainfall)]
```

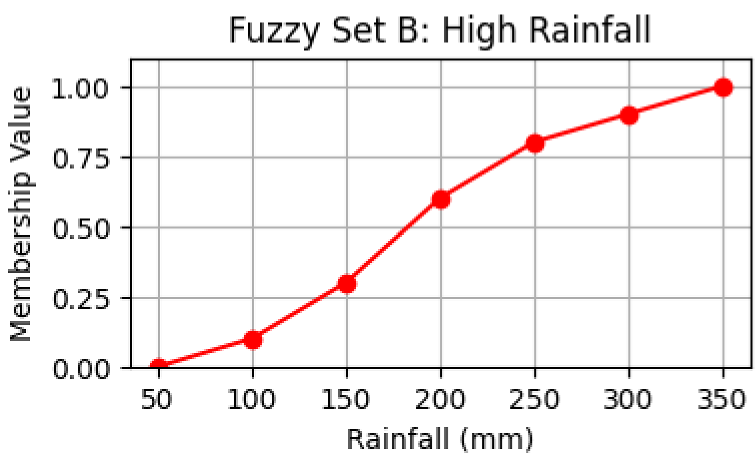
```
# Function to plot fuzzy sets
def plot_fuzzy_set(x, y, title, color):
    plt.figure(figsize=(4, 2))
    plt.plot(x, y, 'o-', color=color)
    plt.title(title)
    plt.xlabel('Rainfall (mm)')
    plt.ylabel('Membership Value')
```

```
    plt.grid(True)
    plt.ylim(0, 1.1)
    plt.show()
```

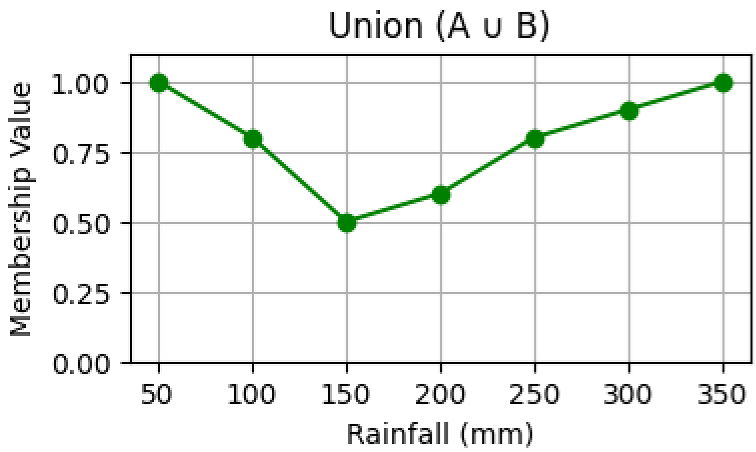
```
plot_fuzzy_set(rainfall, low_rainfall, "Fuzzy Set A: Low Rainfall",
'blue')
```



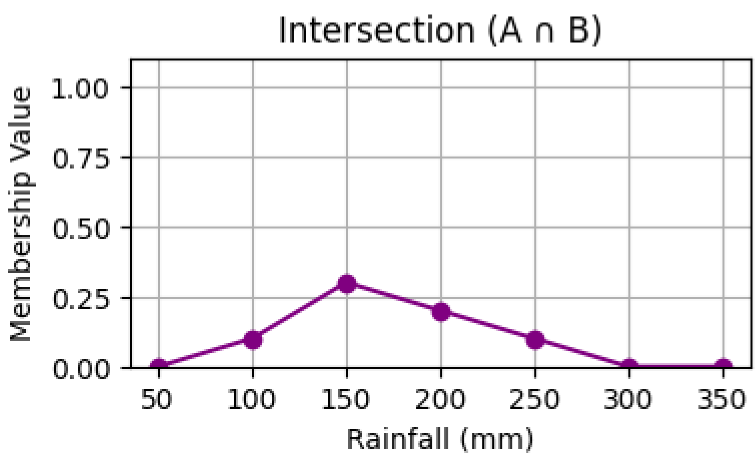

```
plot_fuzzy_set(rainfall, high_rainfall, "Fuzzy Set B: High Rainfall",
'red')
```



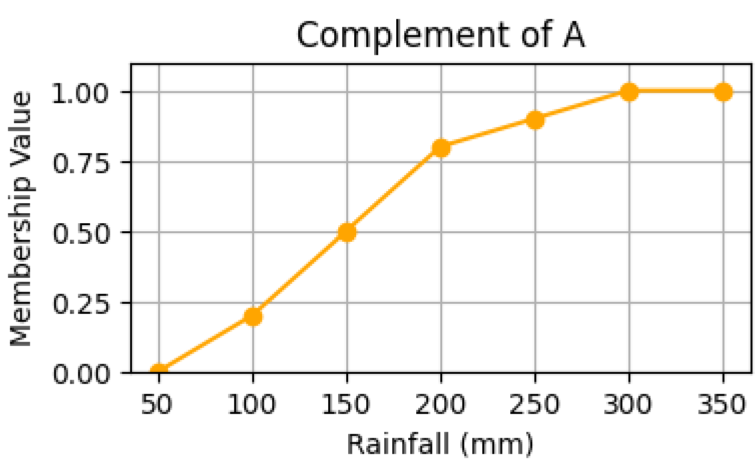

```
plot_fuzzy_set(rainfall, union_AB, "Union (A ∪ B)", 'green')
```



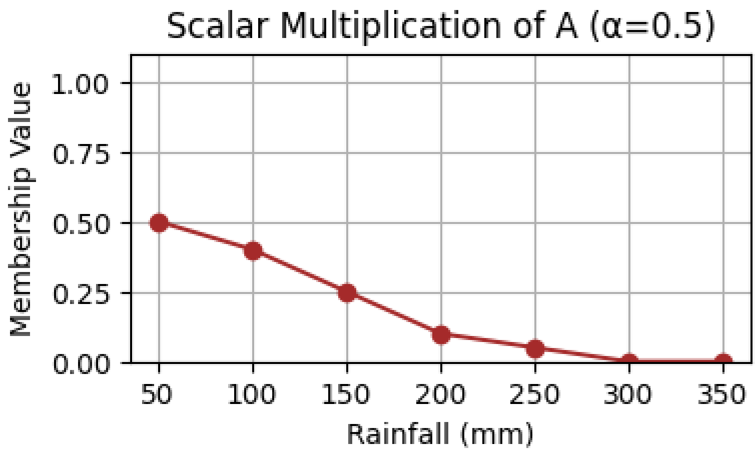

```
plot_fuzzy_set(rainfall, intersection_AB, "Intersection (A ∩ B)",
'purple')
```



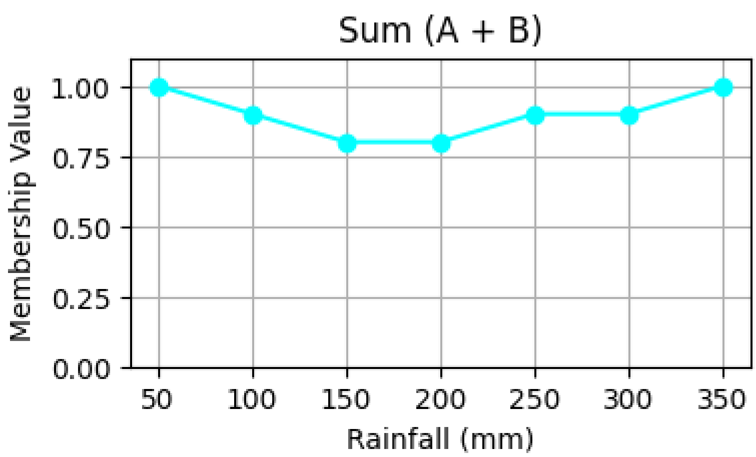

```
plot_fuzzy_set(rainfall, complement_A, "Complement of A", 'orange')
```




```
plot_fuzzy_set(rainfall, scalar_mult_A, "Scalar Multiplication of A
(α=0.5)", 'brown')
```




```
plot_fuzzy_set(rainfall, sum_AB, "Sum (A + B)", 'cyan')
```



Conclusion:

Thus, we have successfully implemented the basic properties of fuzzy sets including union, intersection, complement, scalar multiplication, and sum.
This experiment helps in understanding how fuzzy logic handles partial memberships, which is useful in real-world problems where boundaries are not clear-cut.

[AI&DS2_Expt_07]