Name: **Harsh Pramod Padyal**

Roll No : **40**

Div: **D20B**

Experiment 07

Aim: To implement the properties of fuzzy sets along with fuzzification and defuzzification.

Theory:

Fuzzy set theory extends classical set theory to allow **partial membership**. In classical set theory, membership is binary (0 or 1). In fuzzy set theory, membership values range between **0 and 1**, representing degrees of belonging.

Fuzzy Set Operations

1. **Union**

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

2. **Intersection**

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

3. **Complement**

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

4. **Scalar Multiplication**

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

5. **Sum**

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

Fuzzification

Fuzzification is the process of **converting crisp (precise) values** into **fuzzy values** using membership functions. Example: A temperature of 30°C can be represented as **0.7** "**Warm**" and **0.3** "**Hot**".

Mathematically,

$$\mu_{\text{fuzzy set}}(x) = \text{MembershipFunction}(x)$$

Common membership functions:

- **Triangular**
- **Trapezoidal**
- **Gaussian**

Defuzzification

Defuzzification is the process of **converting fuzzy values** back into a **crisp output**. It is used when a system needs a real-world decision or control value.

One common method is **Centroid (Center of Gravity)**:

$$\text{Crisp Output} = \frac{\sum_i \mu(x_i) \cdot x_i}{\sum_i \mu(x_i)}$$

Code:

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

```
import numpy as np
```

```
# Elements
x = [1, 2, 3, 4, 5]
```

```
# Membership values for two fuzzy sets
A = [0.1, 0.4, 0.7, 0.9, 0.2]
B = [0.3, 0.6, 0.8, 0.5, 0.1]
```

```
# Fuzzy set operations
A_union_B = [max(a, b) for a, b in zip(A, B)]
```

```
A_intersection_B = [min(a, b) for a, b in zip(A, B)]
```

```
A_complement = [1 - a for a in A]
```

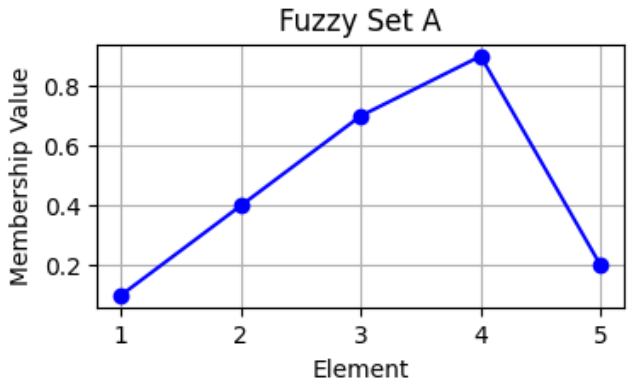
```
A_sum_B = [min(1, a + b) for a, b in zip(A, B)]  # capped at 1
```

```
# --- Fuzzification Example ---
# Triangular Membership Function
def triangular_mf(x, a, b, c):
    return max(min((x-a)/(b-a), (c-x)/(c-b)), 0)
```

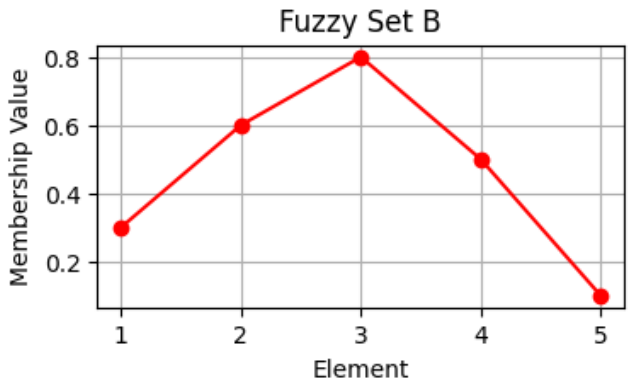
```
crisp_value = 3.5
fuzzy_value = triangular_mf(crisp_value, 2, 4, 6)
```

```
# --- Defuzzification Example (Centroid) ---
universe = np.array([0, 1, 2, 3, 4, 5, 6])
membership_values = [triangular_mf(val, 2, 4, 6) for val in universe]
centroid = sum(u * m for u, m in zip(universe, membership_values)) / sum(membership_values)
```

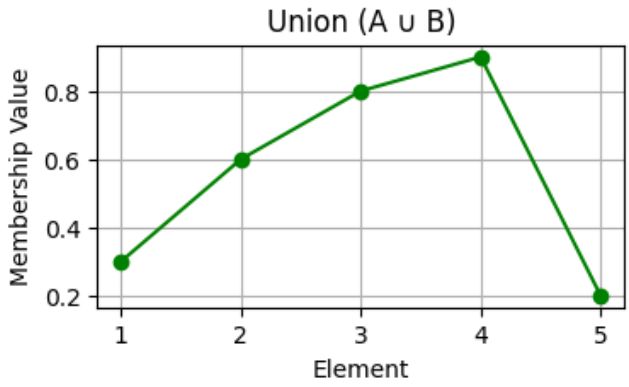
```
# Plot 1: Fuzzy Set A
plt.figure(figsize=(4, 2))
plt.plot(x, A, 'o-', color='blue')
plt.title("Fuzzy Set A")
plt.xlabel('Element')
plt.ylabel('Membership Value')
plt.grid(True)
plt.show()
```



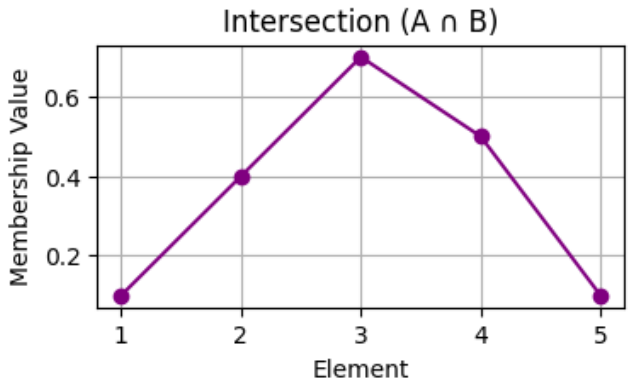

```
# Plot 2: Fuzzy Set B
plt.figure(figsize=(4, 2))
plt.plot(x, B, 'o-', color='red')
plt.title("Fuzzy Set B")
plt.xlabel('Element')
plt.ylabel('Membership Value')
plt.grid(True)
plt.show()
```



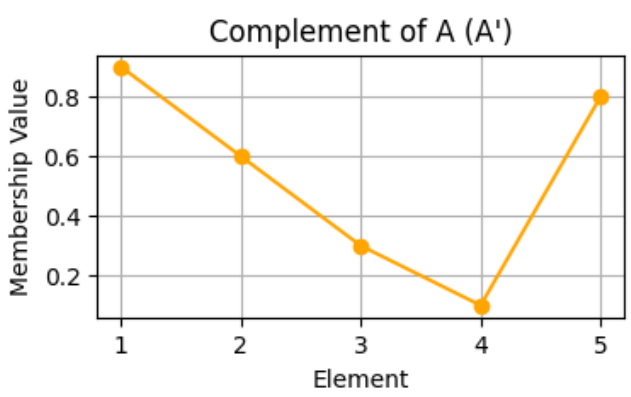

```
# Plot 3: Union (A ∪ B)
plt.figure(figsize=(4, 2))
plt.plot(x, A_union_B, 'o-', color='green')
plt.title("Union (A ∪ B)")
plt.xlabel('Element')
plt.ylabel('Membership Value')
plt.grid(True)
plt.show()
```



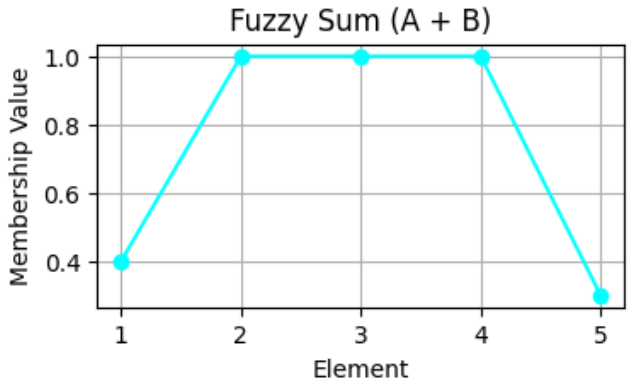
```
# Plot 4: Intersection (A ∩ B)
plt.figure(figsize=(4, 2))
plt.plot(x, A_intersection_B, 'o-', color='purple')
plt.title("Intersection (A ∩ B)")
plt.xlabel('Element')
plt.ylabel('Membership Value')
plt.grid(True)
plt.show()
```




```
# Plot 5: Complement of A
plt.figure(figsize=(4, 2))
plt.plot(x, A_complement, 'o-', color='orange')
plt.title("Complement of A (A')")
plt.xlabel('Element')
plt.ylabel('Membership Value')
plt.grid(True)
plt.show()
```




```
# Plot 6: Fuzzy Sum (A + B)
plt.figure(figsize=(4, 2))
plt.plot(x, A_sum_B, 'o-', color='cyan')
plt.title("Fuzzy Sum (A + B)")
plt.xlabel('Element')
plt.ylabel('Membership Value')
plt.grid(True)
plt.show()
```




```
print(f"Fuzzification example: Crisp value {crisp_value} → Fuzzy value {round(fuzzy_value, 2)}")
```

```
Fuzzification example: Crisp value 3.5 → Fuzzy value 0.75
```

```
print(f"Defuzzification (Centroid method): {round(centroid, 2)}")
```

```
Defuzzification (Centroid method): 4.0
```

Conclusion:

We implemented fuzzy set properties (union, intersection, complement, scalar multiplication, and sum) and extended the experiment to include **fuzzification** (crisp-to-fuzzy conversion) and **defuzzification** (fuzzy-to-crisp conversion). This demonstrates the complete fuzzy logic workflow used in real-world decision-making systems.