In [1]:
```python
# Sample Fuzzy Sets
A = {"a": 0.2, "b": 0.5, "c": 0.7}
B = {"b": 0.6, "c": 0.3, "d": 0.9}
```

In [2]:
```python
 # ------------------------------
# 1. Fuzzy Set Operations
 # ------------------------------
# Union (A ∪ B): max(A(x), B(x))
 def fuzzy_union(A, B):
    result = {}
    universe = set(A.keys()).union(B.keys())
    for x in universe:
        result[x] = max(A.get(x, 0), B.get(x, 0))
    return result

print("Union A ∪ B:", fuzzy_union(A, B))
```

```
Union A ∪ B: {'b': 0.6, 'c': 0.7, 'a': 0.2, 'd': 0.9}
```

In [3]:
```python
# Intersection (A ∩ B): min(A(x), B(x))
def fuzzy_intersection(A, B):
    result = {}
    universe = set(A.keys()).union(B.keys())
    for x in universe:
        result[x] = min(A.get(x, 0), B.get(x, 0))
    return result

print("Intersection A ∩ B:", fuzzy_intersection(A, B))
```

```
Intersection A ∩ B: {'b': 0.5, 'c': 0.3, 'a': 0, 'd': 0}
```

In [4]:
```python
# Complement (A'): 1 - A(x)
# def fuzzy_complement(A):
#     return {x: 1 - val for x, val in A.items()}

def fuzzy_complement(A):
    result = {}
    for x in A:
        result[x] = 1- A[x]
    return result

print("Complement of A:", fuzzy_complement(A))
```

```
Complement of A: {'a': 0.8, 'b': 0.5, 'c': 0.30000000000000004}
```

In [5]:
```python
# Difference (A - B): max(0, A(x) - B(x))
def fuzzy_difference(A, B):
    result = {}
    universe = set(A.keys()).union(B.keys())
    for x in universe:
        result[x] = max(0, A.get(x, 0) - B.get(x, 0))
    return result

print("Difference A - B:", fuzzy_difference(A, B))
```

```
Difference A - B: {'b': 0, 'c': 0.39999999999999997, 'a': 0.2, 'd': 0}
```

In [6]:
```python
# 2. Fuzzy Cartesian Product (Relation)
# ------------------------------
# Cartesian Product R = A × B with min(A(x), B(y))
def cartesian_product(A, B):
    relation = {}
    for a_key, a_val in A.items():
        for b_key, b_val in B.items():
            relation[(a_key, b_key)] = min(a_val, b_val)
    return relation
```

In [7]:
```python

print("\n--- Fuzzy Relation R = A × B ---")
R = cartesian_product(A, B)
for pair in sorted(R):
    print(f"R{pair} = {R[pair]}")
```

```
--- Fuzzy Relation R = A × B ---
R('a', 'b') = 0.2
R('a', 'c') = 0.2
R('a', 'd') = 0.2
R('b', 'b') = 0.5
R('b', 'c') = 0.3
R('b', 'd') = 0.5
R('c', 'b') = 0.6
R('c', 'c') = 0.3
R('c', 'd') = 0.7
```

In [8]:
```
print("\n--- Fuzzy Relation S = B × A ---")
S = cartesian_product(B, A)
for pair in sorted(S):
    print(f"S{pair} = {S[pair]}")
```

```
--- Fuzzy Relation S = B × A ---
S('b', 'a') = 0.2
S('b', 'b') = 0.5
S('b', 'c') = 0.6
S('c', 'a') = 0.2
S('c', 'b') = 0.3
S('c', 'c') = 0.3
S('d', 'a') = 0.2
S('d', 'b') = 0.5
S('d', 'c') = 0.7
```

In [9]:
```
# ------------------------------
# 3. Max-Min Composition of Two Relations
# ------------------------------

def max_min_composition(R1, R2):
    result = {}

    # Get sets for X, Y, Z
    X = {x for x, _ in R1}
    Y = {y for _, y in R1}
    Y2 = {y for y, _ in R2}
    Z = {z for _, z in R2}

    # Ensure Y matches Y2 for valid composition
    if Y != Y2:
        raise ValueError("Mismatch in middle elements for composition!")

    # Perform max-min composition
    for x in X:
        for z in Z:
            min_vals = []
            for y in Y:
                val1 = R1.get((x, y), 0)
                val2 = R2.get((y, z), 0)
                min_vals.append(min(val1, val2))
            result[(x, z)] = max(min_vals)

    return result
```

In [10]:
```python
print("\n--- Max-Min Composition: R o S ---")
R_composed_S = max_min_composition(R, S)
for pair in sorted(R_composed_S):
    print(f"(R o S){pair} = {R_composed_S[pair]}")
```

```
--- Max-Min Composition: R o S ---
(R o S)('a', 'a') = 0.2
(R o S)('a', 'b') = 0.2
(R o S)('a', 'c') = 0.2
(R o S)('b', 'a') = 0.2
(R o S)('b', 'b') = 0.5
(R o S)('b', 'c') = 0.5
(R o S)('c', 'a') = 0.2
(R o S)('c', 'b') = 0.5
(R o S)('c', 'c') = 0.7
```

In [11]:

```
print("Fuzzy Set A:", A)
print("Fuzzy Set B:", B)

print("\n--- Fuzzy Set Operations ---")
print("Union A ∪ B:", fuzzy_union(A, B))
print("Intersection A ∩ B:", fuzzy_intersection(A, B))
print("Complement of A:", fuzzy_complement(A))
print("Difference A - B:", fuzzy_difference(A, B))

print("\n--- Fuzzy Relation R = A × B ---")
R = cartesian_product(A, B)
for pair in sorted(R):
    print(f"R{pair} = {R[pair]}")

print("\n--- Fuzzy Relation S = B × A ---")
S = cartesian_product(B, A)
for pair in sorted(S):
    print(f"S{pair} = {S[pair]}")

print("\n--- Max-Min Composition: R o S ---")
R_composed_S = max_min_composition(R, S)
for pair in sorted(R_composed_S):
    print(f"(R o S){pair} = {R_composed_S[pair]}")
```

```
Fuzzy Set A: {'a': 0.2, 'b': 0.5, 'c': 0.7}
Fuzzy Set B: {'b': 0.6, 'c': 0.3, 'd': 0.9}

--- Fuzzy Set Operations ---
Union A ∪ B: {'b': 0.6, 'c': 0.7, 'a': 0.2, 'd': 0.9}
Intersection A ∩ B: {'b': 0.5, 'c': 0.3, 'a': 0, 'd': 0}
Complement of A: {'a': 0.8, 'b': 0.5, 'c': 0.30000000000000004}
Difference A - B: {'b': 0, 'c': 0.39999999999999997, 'a': 0.2, 'd': 0}

--- Fuzzy Relation R = A × B ---
R('a', 'b') = 0.2
R('a', 'c') = 0.2
R('a', 'd') = 0.2
R('b', 'b') = 0.5
R('b', 'c') = 0.3
R('b', 'd') = 0.5
R('c', 'b') = 0.6
R('c', 'c') = 0.3
R('c', 'd') = 0.7

--- Fuzzy Relation S = B × A ---
S('b', 'a') = 0.2
S('b', 'b') = 0.5
S('b', 'c') = 0.6
S('c', 'a') = 0.2
S('c', 'b') = 0.3
S('c', 'c') = 0.3
S('d', 'a') = 0.2
S('d', 'b') = 0.5
S('d', 'c') = 0.7

--- Max-Min Composition: R o S ---
(R o S)('a', 'a') = 0.2
(R o S)('a', 'b') = 0.2
(R o S)('a', 'c') = 0.2
(R o S)('b', 'a') = 0.2
(R o S)('b', 'b') = 0.5
(R o S)('b', 'c') = 0.5
(R o S)('c', 'a') = 0.2
(R o S)('c', 'b') = 0.5
(R o S)('c', 'c') = 0.7
```

In [ ]: