# Name:Rebecca Dias Class: BECMPNA2 Rollno:19 PID:182027

**Experiment No. 5:**

**Fuzzy Set Properties and Operations**

**Aim:**

To learn the properties and operations on a Fuzzy set.

**Theory:**

A fuzzy set, then, is a set containing elements that have varying degrees of

membership in the set. If X is a collection of objects denoted by x, then fuzzy set Ã

in X is a set of ordered pairs.

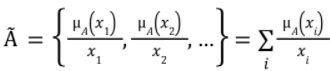


where X is the Universe of Discourse

 is the membership of element x in set Ã and 0 ≤ μ ≤ 1

**Fuzzy Sets Notation**

When the universe is discrete and finite



When the universe is continuous and infinite



**Example**

Create a fuzzy set for integers close to 6 where X = the set of integers

Solution: Ã = {(3, 0. 1), (4, 0. 4), (5, 0. 8), (6, 1), (7, 0. 8), (8, 0. 3), (9, 0. 1)}

**Fuzzy Set Operations**

Consider two fuzzy sets A and B on the universe X. For a given element x of the universe, the set theoretic operations union, intersection and complement are defined as follows:

**Union:** The membership function  of union of two fuzzy sets A and B is defined as:



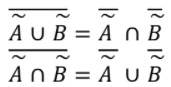
**Intersection:** The membership function  of intersection of two fuzzy sets A and B is defined as:



**Complement:** The membership function of complement of a normalized fuzzy set A, is defined as:



**De Morgan’s laws:** De Morgan’s laws stated for fuzzy sets, as denoted by these expressions.



**Properties of Fuzzy Set**

1. **Support of Fuzzy Set**

The support of the fuzzy set A is S(A), which is a crisp set of all x ∈ X such that

μÃ(x) > 0. The element x in X at which μÃ(x) = 0.5 is called crossover point.

1. **Core of Fuzzy Set**

The core of the fuzzy set A is C(A), which is a crisp set of all x ∈ X such that μÃ(x)=1.

1. **α-level set and Strong α-level set**

α-level set is a crisp set of elements that belong to fuzzy set A atleast to degree α

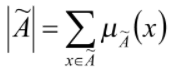
Ã α = {x ∈ X |μÃ(x) ≥ α}

Strong α-level set is defined as

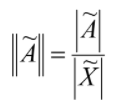
Ã’ α = {x ∈ X |μÃ(x) > α}

1. **Cardinality**

The cardinality of fuzzy set A is defined as



Relative cardinality is



1. **Height**

The height of the fuzzy set A is the largest membership grade of an element in A

Height (A) = max (μÃ(x))

1. **Normality**

A fuzzy set X is called normal, if there exist atleast one element x ε X such that μÃ(x)=1. A fuzzy set that is not normal is called subnormal.

**Applications of Fuzzy Logic**

Fuzzy logic is used in various fields such as automotive systems, domestic goods, environment control, etc. Some of the common applications are:

* + It is used in the **aerospace field** for **altitude control** of spacecraft and satellites.
  + This controls the **speed and traffic** in the **automotive systems.**
  + It is used for **decision making support systems** and personal evaluation in the large company business.
  + It also controls the pH, drying, chemical distillation process in the **chemical industry**.
  + Fuzzy logic is used in **Natural language processing** and various intensive [applications in Artificial Intelligence](https://www.edureka.co/blog/artificial-intelligence-applications/).
  + It is extensively used in **modern control systems** such as expert systems.
  + Fuzzy Logic mimics how a person would make decisions, only much faster. Thus, you can use it with [Neural Networks](https://www.edureka.co/blog/what-is-a-neural-network/).

Code:

def union(A,B):

u = {}

for i in A:

if i in B:

u[i]=max(A[i],B[i])

else:

u[i]=A[i]

for i in B:

if i not in A:

u[i]=B[i]

return(u)

def intersection(A,B):

inter = {}

for i in A:

if i in B:

inter[i]=min(A[i],B[i])

else:

inter[i]=A[i]

for i in B:

if i not in A:

inter[i]=B[i]

return(inter)

def difference(A,B):

comp\_b = complement(B)

diff = intersection(A,comp\_b)

def complement(A):

comp\_a={}

for i in A:

comp\_a[i] = round((1-A[i]),1)

return(comp\_a)

def morgan(A,B):

p = intersection(A,B)

p\_bar = complement(p)

comp\_a = complement(A)

comp\_b = complement(B)

q = union(comp\_a,comp\_b)

if(p\_bar == q):

print("(A n B)': ",p\_bar)

print("A' u B': ",q)

print("Thus, (A n B)' = A' u B'")

print("Law 1 proved")

p = union(A,B)

p\_bar = complement(p)

comp\_a = complement(A)

comp\_b = complement(B)

q = intersection(comp\_a,comp\_b)

if(p\_bar == q):

print("\n(A u B)': ",p\_bar)

print("A' n B': ",q)

print("Thus, (A u B)' = A' n B'")

print("Law 2 proved")

print("\nHence De Morgan's Law is proved")

def support(A):

supp = dict((k,v) for k,v in A.items() if v > 0)

supp\_ele = []

for i in supp.keys():

supp\_ele.append(i)

if(len(supp\_ele) == 0):

return("PHI")

else:

return(supp\_ele)

def core(A):

core = dict((k,v) for k,v in A.items() if v == 1)

core\_ele = []

for i in core.keys():

core\_ele.append(i)

if(len(core\_ele) == 0):

return("PHI")

else:

return(core\_ele)

def height(A):

av = A.values()

mx = max(av)

return(mx)

def cardinality(A):

av = A.values()

sm = sum(av)

return(sm)

def relative\_cardinality(A):

av = A.values()

rc = str(format(sum(av)/len(av),".2f"))

return(rc)

def alpha\_cut(A,alpha):

ac = dict((k,v) for k,v in A.items() if v >= alpha)

ac\_ele = []

for i in ac.keys():

ac\_ele.append(i)

if(len(ac\_ele) == 0):

return("PHI")

else:

return(ac\_ele)

def strong\_alpha\_cut(A,alpha):

sac = dict((k,v) for k,v in A.items() if v > alpha)

sac\_ele = []

for i in sac.keys():

sac\_ele.append(i)

if(len(sac\_ele) == 0):

return("PHI")

else:

return(sac\_ele)

def check\_normal\_or\_not(C,x):

if(1 in C.values()):

return(x,"is Normal")

else:

return(x,"is Subnormal")

def start():

print('-'\*93,'\n')

u = union(mem\_a,mem\_b)

print("Union: ",u)

print()

print('-'\*93,'\n')

inter = intersection(mem\_a,mem\_b)

print("Intersection: ",inter)

print()

print('-'\*93,'\n')

diff = difference(mem\_a,mem\_b)

print("Difference: ",diff)

print()

print('-'\*93,'\n')

comp\_a = complement(mem\_a)

comp\_b = complement(mem\_b)

print("A's complement: ",comp\_a)

print("B's complement: ",comp\_b)

print()

print('-'\*93,'\n')

morgan(mem\_a,mem\_b)

print()

print('-'\*93,'\n')

supp\_a = support(mem\_a)

supp\_b = support(mem\_b)

print("Support of A: ",supp\_a)

print("Support of B: ",supp\_b)

print()

print('-'\*93,'\n')

core\_a = core(mem\_a)

core\_b = core(mem\_b)

print("Core of A: ",core\_a)

print("Core of B: ",core\_b)

print()

print('-'\*93,'\n')

height\_a = height(mem\_a)

height\_b = height(mem\_b)

print("Height of A = ",height\_a)

print("Height of B = ",height\_b)

print()

print('-'\*93,'\n')

cardinality\_a = cardinality(mem\_a)

cardinality\_b = cardinality(mem\_b)

print("Cardinality of A: ",cardinality\_a)

print("Cardinality of B: ",cardinality\_b)

print()

print('-'\*93,'\n')

relative\_cardinality\_a = relative\_cardinality(mem\_a)

relative\_cardinality\_b = relative\_cardinality(mem\_b)

print("Relative Cardinality of A: ",relative\_cardinality\_a)

print("Relative Cardinality of B: ",relative\_cardinality\_b)

print()

print('-'\*93,'\n')

alpha\_cut\_a = alpha\_cut(mem\_a,alpha)

alpha\_cut\_b = alpha\_cut(mem\_b,alpha)

print("Alpha cut of A({}): {}".format(alpha,alpha\_cut\_a))

print("Alpha cut of B({}): {}".format(alpha,alpha\_cut\_b))

print()

print('-'\*93,'\n')

strong\_alpha\_cut\_a = strong\_alpha\_cut(mem\_a,alpha)

strong\_alpha\_cut\_b = strong\_alpha\_cut(mem\_b,alpha)

print("Strong Alpha cut of A({}): {}".format(alpha,strong\_alpha\_cut\_a))

print("Strong Alpha cut of B({}): {}".format(alpha,strong\_alpha\_cut\_b))

print()

print('-'\*93,'\n')

check\_normal\_or\_not\_a = check\_normal\_or\_not(mem\_a,'A')

check\_normal\_or\_not\_b = check\_normal\_or\_not(mem\_b,'B')

print(check\_normal\_or\_not\_a)

print(check\_normal\_or\_not\_b)

print()

def inputt():

global mem\_a, mem\_b, alpha

a=[]

print('-'\*40+' FUZZY\_SET\_A '+'-'\*40)

n=input("Enter the elements of set A: ")

a=n.split(' ')

print("Enter the membership value for each element: ")

mem\_a = {}

for i in a:

print(i,"= ",end='')

mem\_a[i] = float(input())

if not 0 <= mem\_a[i] <=1:

print("Enter membership value between [0,1]!!")

quit()

print("Fuzzy Set A: ",mem\_a)

print()

b=[]

print('-'\*40+' FUZZY\_SET\_B '+'-'\*40)

n=input("Enter the elements of set B: ")

b=n.split(' ')

print("Enter the membership value for each element: ")

mem\_b = {}

for i in b:

print(i,"= ",end='')

mem\_b[i] = float(input())

if not 0 <= mem\_a[i] <=1:

print("Enter membership value between [0,1]!!")

quit()

print("Fuzzy Set B: ",mem\_b)

print()

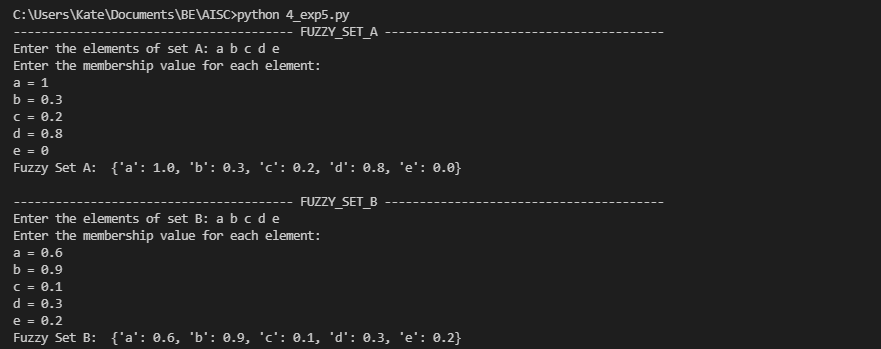
alpha = float(input("Enter the Alpha value between [0,1] for the alpha cut and strong alpha cut: "))

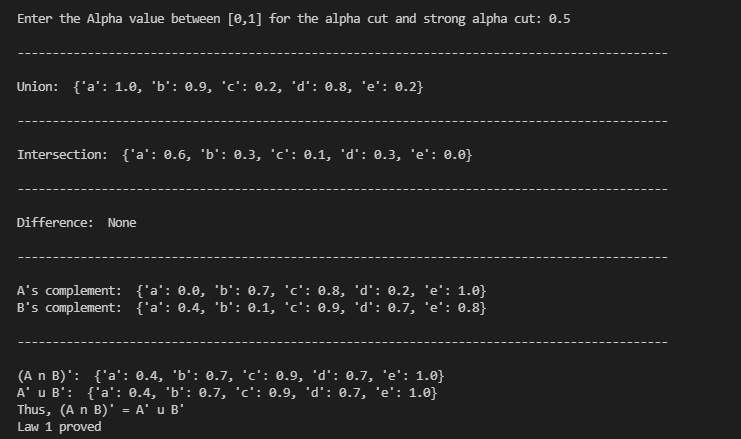
print()

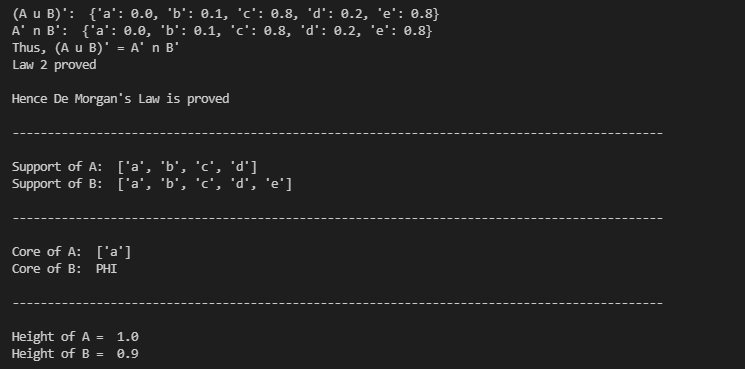
inputt()

start()

# Output:

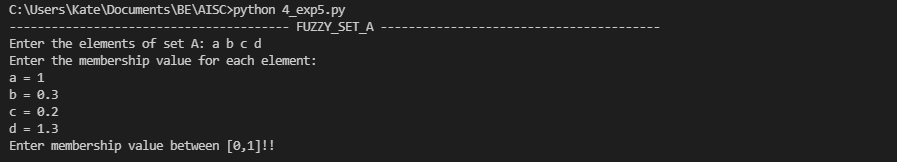


****

****

****

**When membership value is greater than 1**



# Conclusion:

We successfully implemented Fuzzy Sets and learnt about the different properties of Fuzzy Sets such as union, intersection, difference etc and also the operations of Fuzzy Sets and its Applications. We successfully implemented Fuzzy Sets in python.