Central Department

of

Computer Science and Information Technology

Tribhuvan University

Lab Report

on

Implementation of Fuzzy Sets and Operations

Submitted to:

Jagdish Bhatta

CDCSIT

Tribhuvan University

Submitted By:

Karna Bahadur Shrestha

MSc. CSIT 2020

Second Semester

Rollno 14

Date: 27th Jan 2022

#2

Code

```
#Lab 2 Implementation of fuzzy set and operations
def enter(name):
    list={}
    n=int(input("Enter the number of elements in set"+name))
    for i in range(n):
        name=input("Enter the name: ")
        while 1:
            value=float(input("Enter the value: "))
            if(value>=0 and value<=1):
                list[name]=value
                break;
            else:
                print("Value must be >= 0 and <=1")
    return list
def enteralpha(name):
    while 1:
        alpha=float(input("Enter the value of "+name+":"))
        if(alpha>=0 and alpha<=1):
            return alpha
        else:
            print("Value must be >= 0 and <=1")
#putting the element of set to each other, making the elements of sets
equal
def makeequal(dict1,dict2):
    for dict2_key in dict2:
        for i in range(len(dict1)):
            if(dict2_key in dict1)!=True:
                dict1[dict2_key]=0
    return dict1

def sortdict(A):
    sorted_keys = sorted(A.keys())
    sorteddict = {key:A[key] for key in sorted_keys}
    return sorteddict


def Union(A,B):
    Y_set=dict()
    for A_key, B_key in zip(A, B):
        A_value = A[A_key]
        B_value = B[B_key]

        if A_value >= B_value :
            if A_value>0 or B_value>0:
                Y_set[A_key] = A_value
        elif B_value>A_value:
            Y_set[B_key] = B_value
```

```
    print('Union:', Y_set)


def Intersection(A,B):
    Y_set=dict()
    for A_key, B_key in zip(A, B):
        A_value = A[A_key]
        B_value = B[B_key]

        if A_value <= B_value :
            if A_value>0 and B_value>0:
                Y_set[A_key] = A_value
        elif B_value<=A_value:
            if A_value>0 and B_value>0:
                Y_set[B_key] = B_value
    print('Intersection:', Y_set)

def complement(A,setname):
    Y_set=dict()
    for A_key in A:
        A_value = A[A_key]


        if 1-A_value !=0:
            Y_set[A_key] = round(1-A_value,2)
    print("The complement of set",setname,"is:",Y_set)

def checksubset(A,B):
    countA=countB=0
    for A_key, B_key in zip(A,B):
        A_value = A[A_key]
        B_value = B[B_key]

        if B_value <= A_value :
            countA=countA+1
        if A_value<=B_value:
            countB=countB+1

    if(countA==len(B)):
        print("B is subset of A")
    if(countB==len(A)):
        print("A is subset of B")
    if((countA==len(B) or countB==len(A))!=True):
        print("Non are subset of each set")

def alphacut(A,alpha,setname):
    Y_set=dict()
```

```
    for A_key in A:
        A_value = A[A_key]
        if alpha<=A_value and A_value>0:
            Y_set[A_key] = A_value

    print("After alpha cut value ",alpha," the set
",setname,"is:",Y_set)

def strictalphacut(A,alpha,setname):
    Y_set=dict()
    for A_key in A:
        A_value = A[A_key]
        if alpha<A_value and A_value>0:
            Y_set[A_key] = A_value

    print("After strict alpha cut value ",alpha," the set
",setname,"is:",Y_set)

# A = {'a': 0.3, 'b': 0.6, 'c':0.5, 'd': 0.5}
# B= {'a': 0.5,  'b': 0.9, 'c': 0.6, 'd':0.5, 'e': 0.1}
A_origin=enter("A")
B_origin=enter("B")
print("Set A:",A)
print("Set B", B)
A=makeequal(A,B)
B=makeequal(B,A)

A=sortdict(A)
B=sortdict(B)
print("Set A after processed:",A)
print("Set B after processed:", B)
print("-----------------------------------")
Union(A,B)
print("-----------------------------------")
Intersection(A,B)
print("-----------------------------------")
complement(A_origin,"A")
complement(B_origin,"B")
print("-----------------------------------")
checksubset(A,B)
print("-----------------------------------")
alpha=enteralpha("alpha")
alphacut(A,alpha,"A")
alphacut(B,alpha,"B")
print("-----------------------------------")
strictalpha=enteralpha("strictalpha")
strictalphacut(A,strictalpha,"A")
strictalphacut(B,strictalpha,"B")
```

Output

```
Enter the number of elements in setA4
Enter the name: a
Enter the value: 0.3
Enter the name: b
Enter the value: -0.4
Value must be >= 0 and <=1
Enter the value: 4
Value must be >= 0 and <=1
Enter the value: 0.6
Enter the name: c
Enter the value: 0.5
Enter the name: d
Enter the value: 0.4
Enter the number of elements in setB5
Enter the name: a
Enter the value: 0.5
Enter the name: b
Enter the value: 0.9
Enter the name: c
Enter the value: 0.6
Enter the name: d
Enter the value: 0.5
Enter the name: e
Enter the value: 0.1
Set A: {'a': 0.3, 'b': 0.6, 'c': 0.5, 'd': 0.4, 'e': 0}
Set B {'a': 0.5, 'b': 0.9, 'c': 0.6, 'd': 0.5, 'e': 0.2}
Set A after processed: {'a': 0.3, 'b': 0.6, 'c': 0.5, 'd': 0.4, 'e': 0}
Set B after processed: {'a': 0.5, 'b': 0.9, 'c': 0.6, 'd': 0.5, 'e': 0.2}
----------------------------------
Union: {'a': 0.5, 'b': 0.9, 'c': 0.6, 'd': 0.5, 'e': 0.2}
----------------------------------
Intersection: {'a': 0.3, 'b': 0.6, 'c': 0.5, 'd': 0.4}
----------------------------------
The complement of set A is: {'a': 0.7, 'b': 0.4, 'c': 0.5, 'd': 0.6}
The complement of set B is: {'a': 0.5, 'b': 0.1, 'c': 0.4, 'd': 0.5, 'e':
0.9}
----------------------------------
A is subset of B
----------------------------------
Enter the value of alpha:0.5
After alpha cut value  0.5  the set  A is: {'b': 0.6, 'c': 0.5}
After alpha cut value  0.5  the set  B is: {'a': 0.5, 'b': 0.9, 'c': 0.6,
'd': 0.5}
----------------------------------
Enter the value of strictalpha:0.5
After strict alpha cut value  0.5  the set  A is: {'b': 0.6}
After strict alpha cut value  0.5  the set  B is: {'b': 0.9, 'c': 0.6}
```