

**Lab no.:- 4 Date:** 2079-09-22

**Title: Write a program for the operations of fuzzy set.**

Fuzzy Set

A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership (characteristic) function which assigns to each object a grade of membership ranging between zero and one.

Fuzzy operations are performed on fuzzy sets, whereas crisp operations are performed on crisp sets. Fuzzy operations are very useful in the design of a Fuzzy Logic Controller. It allows the manipulation of fuzzy variables by different means.

Union:

In the case of the union of crisp sets, we simply have to select repeated elements only once. In the case of fuzzy sets, when there are common elements in both the fuzzy sets, we should select the element with the maximum membership value.

Intersection:

In the case of the intersection of crisp sets, we simply have to select common elements from both sets. In the case of fuzzy sets, when there are common elements in both the fuzzy sets, we should select the element with minimum membership value.

Complement:

Fuzzy complement is identical to crisp complement operation. Membership value of every element in the fuzzy set is complemented with respect to 1, i.e. it is subtracted from 1.

**IDE** : Dev-C++

**Language** : C++

**Source Code**

//wap for operations of fuzzy sets of two sets

#include<iostream>

using namespace std;

*//function for finding minimum membership*

double min(double x, double y) {

if (x < y)

return x;

else

return y;

}

*//function for displaying set*

void display(int s[], double m[], int l) {

cout << "{ ( ";

for (int i = 0; i < l; i++) {

if (i != l - 1)

cout << s[i] << " , " << m[i] << " ) ,( ";

if (i == l - 1)

cout << s[i] << " , " << m[i] << " ) }" << endl;

}

}

int main() {

*//defining variables*

int a[100],b[100];

double m[100],n[100],cm[100],cn[100],me[100],te[100], mem[100];

int c[100],p[100], q[100];

int n1, n2;

int f = 0, d = 0, e = 0, k = 0;

*// input of set datas*

cout << "Enter the size of set a" << endl;

cin >> n1;

cout << "Enter the size of set b" << endl;

cin >> n2;

*//input of set a*

cout << "Enter the members of set a with its members" << endl;

for (int i = 0; i < n1; i++) {

cin >> a[i];

cin >> m[i];

}

cout << "A = ";

display(a, m, n1);

*//input of set b*

cout << "Enter the members of set b" << endl;

for (int i = 0; i < n2; i++) {

cin >> b[i];

cin >> n[i];

}

cout << "B = ";

display(b, n, n2);

*//finding the intersection of set a and b*

for (int i = 0; i < n1; i++) {

for (int J = 0; J < n2; J++)

if (a[i] == b[J]) {

c[k] = a[i];

me[k] = min(m[i], n[J]);

k = k + 1;

d = k;

}

}

*//no of members of union*

e = n1 + n2 - d;

*//printing intersection of set*

cout << endl << "INTERSECTION =";

display(c, me, d);

for (int i = 0; i < n1; i++) {

p[i] = a[i];

te[i] = m[i];

}

*//p is the set of whole set a and set b*

k = 0;

for (int i = n1; i < n1 + n2; i++) {

p[i] = b[k];

te[i] = n[k];

k++;

}

*//q is the set of union of a and b*

k = 0;

for (int i = 0; i < n1 + n2; i++) {

f = 0;

for (int j = 0; j < n1 + n2 - e; j++) {

if (p[i] == c[j] && te[i] == me[j]) {

f = 1;

}

}

if (f == 0) {

q[k] = p[i];

mem[k] = te[i];

k++;

}

}

*// displaying union set*

cout << endl << "UNION = ";

display(q, mem, e);

*//complement*

for (int i = 0; i < n2; i++) {

cm[i] = 1 - m[i];

cn[i] = 1 - n[i];

}

cout << endl << "COMPLEMENT OF A = ";

display(a, cm, n1);

cout << endl << "COMPLEMENT OF B = ";

display(b, cn, n2);

return 0;

}

**Output**

