Experiment No:3

**Aim**: To find Relation of two sets and min-max composition of two relations.

**Title:** For a speed control DC motor the membership function of series resistance armature current and speed find the cartesian product of R x I and I x N, then compute relation for Resistance and Speed.

**Theory:**

Fuzzy Cartesian Product

Cartesian product of fuzzy sets can be given by

A = 0.2/x1 + 0.5/x2 + 1/x3

B = 0.3/y1 + 0.9/y2

Y1 Y2

A x B = R = X1 0.2 0.2

X2 0.3 0.5

X3 0.3 0.9

The max-min composition of relations R1 and R2 denoted by MaxMin(R1, R2) is a fuzzy relation in UxW, such that for all (u, w) in UxW, MaxMin(R1, R2)(u, w) = Max(Min(R1(u, v), R2(v, w))) over all v in the set V.

**Program:**

import java.util.\*;

public class Relation {

public static void main(String []args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the number of element in Resistance set: ");

int n1 = sc.nextInt();

double a[][] = new double[n1][2];

System.out.println("Enter the elements of the Set: ");

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

a[i][0] = sc.nextDouble();

a[i][1] = sc.nextDouble();

}

System.out.print("Enter the number of element in Current set: ");

int n2 = sc.nextInt();

double b[][] = new double[n2][2];

System.out.println("Enter the elements of the Set: ");

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

b[i][0] = sc.nextDouble();

b[i][1] = sc.nextDouble();

}

System.out.print("Enter the number of element in Speed set: ");

int n3 = sc.nextInt();

double c[][] = new double[n3][2];

System.out.println("Enter the elements of the Set: ");

for (int i = 0; i < n3; i++) {

c[i][0] = sc.nextDouble();

c[i][1] = sc.nextDouble();

}

System.out.println("Cartesian Product of Resistance And Current: ");

double d[][] = new double[n1][n2];

d = cartesian(a, b, n1, n2);

print(d, n1, n2);

System.out.println("Cartesian Product of Current and Speed:");

double e[][] = new double[n2][n3];

e = cartesian(b, c, n2, n3);

print(e, n2, n3);

double f[][] = new double[n1][n3];

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

for(int j= 0;j<n3;j++){

double max =0,min=0;

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

min = (d[i][k]>e[k][j])? e[k][j]:d[i][k];

max = (max>min)?max:min;

}

f[i][j]=max;

}

}

System.out.println("The relation of resistance on speed: ");

print(f,n1,n3);

}

public static double[][] cartesian(double a[][], double b[][], int n1, int n2) {

double c[][] = new double[n1][n2];

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

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

c[i][j] = min(a[i][0], b[j][0]);

}

return c;

}

public static double min(double a, double b) {

if (a < b)

return a;

else

return b;

}

public static void print(double p[][], int n1, int n2) {

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

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

System.out.print(p[i][j] + " ");

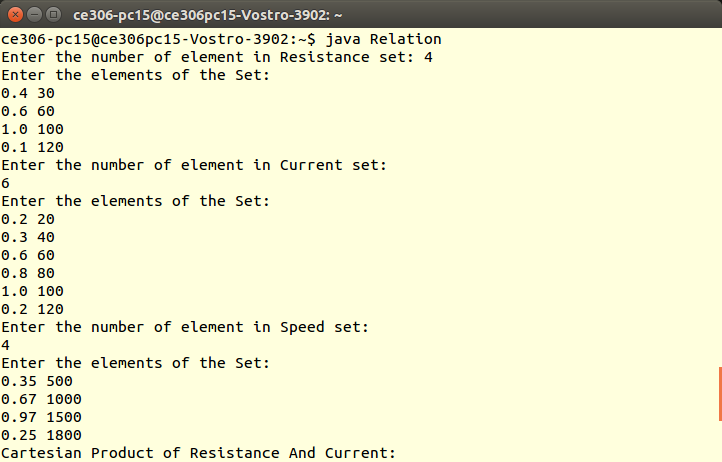
System.out.println();

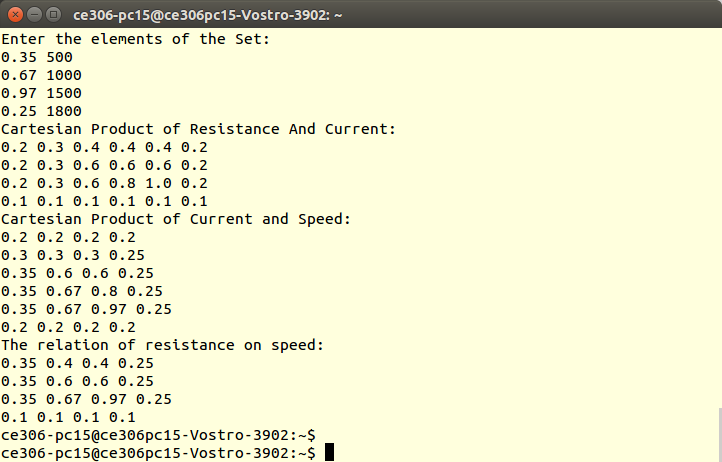
}

}

}

**Output:**





**Conclusion:**

Hence, relation and max-min composition of two relation has been studied and implemented.