MINISTERUL EDUCAȚIEI AL REPUBLICII MOLDOVA
UNIVERSITATEA DE STAT DIN TIRASPOL

CATEDRA

FIZICĂ, MATEMATICĂ ŞI TEHNOLOGII INFORMAŢIONALE

SPECIALITATEA

TEHNOLOGII INFORMAŢIONALE ÎN INSTRUIRE

Structuri algebrice pe calculator

Laboratorul Nr.6

Realizat: Cojucovschi Ion
Grupa: C1I1

Verificat: Chiriac Liubomir

Chişinău, 2018

Algoritmi privind obţinerea izotopilor unui grupoid

Problemă. Fie grupoidul () de dimensiunea n. Fie sunt trei permutări arbitrare ale mulţimii Q. Atunci aplicând permutarea elementelor de pe linia de bordare, permutarea elementelor de pe coloana de bordare şi permutarea a elementelor din interiorul tabelei, se obţine o nouă lege de compoziţie pe Q şi este clar că este izotop cu quasigrupul (). Se cere să se elaboreze un program care efecuează permutările date. Să se cerceteze proprietăţile algebrice ale quasigrupului obţinut.

Pentru fiecare din exemple, să se efectueze transformările .

Ex.1

·	1	2	3	4
1	4	1	2	3
2	3	4	1	2
3	2	3	4	1
4	1	2	3	4

α=(2 3 4 1)
β=(4 3 2 1)
=(2 1 4 3)

Ex.2

·	1	2	3	4
1	1	2	3	4
2	2	3	4	1
3	4	1	2	3
4	3	4	1	2

α=(2 3 1 4)
β=(3 1 4 2)
=(1 3 4 2)

Ex.3

·	1	2	3	4	5
1	1	2	3	4	5
2	2	3	1	5	4
3	4	1	5	2	3
4	3	5	4	1	2
5	5	4	2	3	1

α=(2 3 1 5 4)
β=(3 1 2 4 5)
=(5 3 2 4 1)

Ex.4

·	1	2	3
1	1	2	3
2	3	1	2
3	2	3	1

α=(1 3 2)
β=(3 2 1)
=(2 1 3)

Ex.5

·	1	2	3	4
1	1	2	3	4
2	2	1	4	3
3	3	4	1	2
4	4	3	2	1

α=(4 3 2 1)
β=(4 3 2 1)
=(4 3 2 1)

Ex.6

·	1	2	3	4
1	1	2	3	4
2	3	1	4	2
3	2	4	1	3
4	4	3	2	1

α=(4 3 2 1)
β=(3 4 1 2)
=(2 3 4 1)

Ex.7

·	1	2	3	4	5
1	1	2	3	4	5
2	2	3	4	5	1
3	3	4	5	1	2
4	4	5	1	2	3
5	5	1	2	3	4

α=(5 4 3 2 1)
β=(4 3 2 1 5)
=(3 2 1 5 4)

Elaborarea programului:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Laboratorul_numarul_6
{
    class Program
    {
        static int[,] a, tabel, opus, t1, p1, gr;
        static int[] b, c, f;
        static int r1, d1, d2, r, d, r2, n, m, t, i, j, p, l;


        static void Main(string[] args)
        {
            t1 = new int[30, 30];

            int n = 0;
            Console.Write("Dati ordinul matricei a, n=");
            n = Convert.ToInt32(Console.ReadLine());

            initializationmatrix(a,n);      ///initiate matrix

            afisare(a,n);
            b = new int[100];
            c = new int[100];
            E(a,n);
```

```
    afisare(a, n);
    asociativ(a, n);
    medial(a, n);
    paramedial(a, n);
    bicomutativ(a, n);
    ag_gr(a, n);
    ga_gr(a, n);
    ga_gr1(a, n);
    ad_gr(a, n);
    da_gr(a, n);
    hexagonal(a, n);
    dist_dr(a, n);
    dist_st(a, n);

    r = 0; r1 = 0;

    unitate_dreapta(a, n, out r);
    unitate_stanga(a, n, out r1);
    unitate(ref r, ref r1);
    ward(a, n);
    ward_invers(a, n);

    Console.ReadKey();




}
public static void initializationmatrix(out int[,] a, int n)
{
    ////initializam rarrayul
    a = new int[15, 15];
```

```
            for (int i = 1; i < n+1; i++)
            {

                for (int j = 1; j < n+1; j++)
                {
                    Console.WriteLine("a[" + i + "," + j+ "]=");
                    a[i, j] = Convert.ToInt32(Console.ReadLine());
                }

            }



        }

        public static void E(int[,]a,int n)
        {
            int j = 0;
            for (int i = 1; i < n + 1; i++)
            {
                if (a[i, i] == i & mm(i, n) != 0 & nn(i, n) != 0) {
Console.WriteLine("=E("+nn(i,n)+","+mm(i,n)+");");j++; }
            }
            if (j == 0) Console.WriteLine("NU sunt");
        }

        public static int mm(int g, int n)
        {
            int ii, jj, k, x, q;

            for (jj = 1; jj < n + 1; jj++) {
                k = 0;
                x = jj;
```

```
        do
        {
            x = a[x, g];
            k++;
        } while (x != jj || k < n+1);

        if (k > n) b[jj] = 0;
        else b[jj] = k;
    }

    q = b[1];
    for (jj = 1; jj < n + 1; jj++)
    {
        if (b[jj] > q)
        {
            q = b[jj];
        }

    }
    return q;
}

public static int nn(int gg,int n)
{

    int ii, jj, k, x, q;
    for (jj=1;jj<n+1;jj++)
    {
        k = 0;
        x = jj;

        do {
            x = a[gg, x];
```

```
                k++;
            } while (x!=jj || k<n+2);
        }
        q = c[1];
        for (jj = 1; jj < n + 1; jj++)
        {
            if (c[jj] > q) q = c[jj];
        }
        return q;
    }
        public static void Topus(int[,] ai, int n)
        {
            for (int i = 1; i < n + 1; i++)
                for (int j = 1; j < n + 1; j++)
                {
                    Console.Write("ai[" + i + "," + j + "]=");
                    int element = Convert.ToInt32(Console.ReadLine());
                    ai[i, j] = element;

                    if (element == 1) opus[i, 1] = j;
                }
            int k = 1;
            for (int i = 1; i < n + 1; i++)
                for (int j = 0; j < n + 1; j++)
                {
                    tabel[i, j] = k;
                    k++;
                }
        }


        public static void produs_cartezian(out int[,] masiv, int[,] a, int[,]
b, int n, int m)
```

```
{
    int k = 1;
    masiv = new int[30, 30];
    int[] c = new int[100];
    int[] f = new int[100];
    for (int i = 1; i < n + 1; i++)
        for (int j = 1; j < m + 1; j++)
        {
            f[k] = j;
            c[k] = i;
            t1[i, j] = k;
            k = k + 1;
        }
    for (int i = 1; i < m * n; i++)
        for (int j = 1; j < n * m; j++)
        {
            masiv[i, j] = t1[a[c[i], c[j]], b[f[i], f[j]]];
        }
}
public static void afisare(int[,] a, int n)
{

    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            Console.Write(a[i, j]);
        }
        Console.WriteLine();
    }
    Console.WriteLine();
}
```

```
public static void asociativ(int[,] masiv, int n)
{

    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[j, k];
                d1 = masiv[i, j];
                if (masiv[i, d] != masiv[d1, k]) l++;
            }

        }
    }
    if (l == 0) Console.WriteLine("ESTE ASOCIATIV");
    else Console.WriteLine("NU ESTE ASOCIATIV");

}

public static void medial(int[,] masiv, int n)
{
    int l = 0;

    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
```

```
            {
                for (int t = 1; t < n + 1; t++)
                {

                    d = masiv[i, j];
                    r = masiv[k, t];
                    d1 = masiv[i, k];
                    r1 = masiv[j, t];
                    if (masiv[d, r] != masiv[d1, r1])
                        l++;

                }
            }
        }
    }
    if (l == 0) Console.WriteLine("ESTE MEDIAL");
    else Console.WriteLine("NU ESTE MEDIAL");

}

public static void paramedial(int[,] masiv, int n)
{
    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                for (int t = 1; t < n + 1; t++)
                {

                    d = masiv[i, j];
```

```
                    r = masiv[k, t];
                    d1 = masiv[t, j];
                    r1 = masiv[k, i];
                    if (masiv[d, r] != masiv[d1, r1]) l++;

                }
            }
        }
    }
    if (l == 0) Console.WriteLine("ESTE PARAMEDIAL");
    else Console.WriteLine("NU ESTE PARAMEDIAL");
}

public static void bicomutativ(int[,] masiv, int n)
{

    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                for (int t = 1; t < n + 1; t++)
                {

                    d = masiv[i, j];
                    r = masiv[k, t];
                    d1 = masiv[t, k];
                    r1 = masiv[j, i];
                    if (masiv[d, r] != masiv[d1, r1]) l++;

                }
```

```
            }
        }
    }
    if (l == 0) Console.WriteLine("ESTE BICOMUTATIV");
    else Console.WriteLine("NU ESTE BICOMUTATIV");
}

public static void ag_gr(int[,] masiv, int n)
{

    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[i, j];
                d1 = masiv[k, j];
                if (masiv[d, k] != masiv[d1, i]) l++;
            }
        }
    }
    if (l == 0) Console.WriteLine("ESTE AG GRUPOID");
    else Console.WriteLine("NU ESTE AG GRUPOID");
}

public static void ga_gr(int[,] masiv, int n)
{

    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
```

```
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[i, j];
                d1 = masiv[j, i];
                if (masiv[d, k] != masiv[k, d1]) l++;
            }
        }
    }
    if (l == 0) Console.WriteLine("ESTE GA GRUPOID");
    else Console.WriteLine("NU ESTE GA GRUPOID");
}


public static void ga_gr1(int[,] masiv, int n)
{
    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[i, j];
                d1 = masiv[k, i];
                if (masiv[d, k] != masiv[d1, j]) l++;
            }
        }
    }
    if (l == 0) Console.WriteLine("ESTE GA1 GRUPOID");
    else Console.WriteLine("NU ESTE GA1 GRUpoid");
}
```

```
public static void ad_gr(int[,] masiv, int n)
{
    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[j, k];
                d1 = masiv[j, i];
                if (masiv[i, d] != masiv[k, d1]) l++;
            }
        }
    }
    if (l == 0) Console.WriteLine("ESTE AD GRUPOID");
    else Console.WriteLine("NU ESTE AD GRUpoid");
}


public static void da_gr(int[,] masiv, int n)
{
    int l = 0;
    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < n + 1; j++)
        {
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[j, k];
                d1 = masiv[i, j];
                if (masiv[i, d] != masiv[k, d1]) l++;
```

```
                    }
                }
            }
            if (l == 0) Console.WriteLine("ESTE DA GRUPOID");
            else Console.WriteLine("NU ESTE DA GRUpoid");
        }



        public static void hexagonal(int[,] masiv, int n)
        {
            int l = 0;
            for (int i = 1; i < n + 1; i++)
            {
                for (int j = 1; j < n + 1; j++)
                {
                    for (int k = 1; k < n + 1; k++)
                    {
                        for (int t = 1; t < n + 1; t++)
                        {

                            d = masiv[i, j];
                            r = masiv[k, t];
                            d1 = masiv[i, k];
                            r1 = masiv[j, t];
                            r2 = masiv[j, i];
                            if (masiv[i, i] != i || masiv[d, r] != masiv[d1,
r1] || masiv[i, r2] != masiv[d, i] && masiv[d, i] != j) l++;

                        }
                    }
                }
            }
```

```
    if (l == 0) Console.WriteLine("ESTE HEXAGONAL");
    else Console.WriteLine("NU ESTE HEXAGONAL");
}


public static void dist_dr(int[,] masiv, int n)
{
    int l = 0;

    for (int i = 1; i < n + 1; i++)
        for (int j = 1; j < n + 1; j++)
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[i, j];
                d1 = masiv[i, k];
                r1 = masiv[j, k];
                if (masiv[d, k] != masiv[d1, r1]) l++;
            }

    if (l == 0) Console.WriteLine("ESTE DISTRIBUTIV DE DREAPTA");
    else Console.WriteLine("NU ESTE DISTRIBUTIV DE DREAPTA");

}

public static void dist_st(int[,] masiv, int n)
{
    int l = 0;

    for (int i = 1; i < n + 1; i++)
        for (int j = 1; j < n + 1; j++)
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[i, j];
```

```
                d1 = masiv[k, i];
                r1 = masiv[k, j];
                if (masiv[k, d] != masiv[d1, r1]) l++;
            }

    if (l == 0) Console.WriteLine("ESTE DISTRIBUTIV DE STANGA");
    else Console.WriteLine("NU ESTE DISTRIBUTIV DE STANGA");

}


public static void unitate_dreapta(int[,] masiv, int n, out int r)
{
    int l;
    int j = 0; r = 0;
    for (int i = 1; i < n + 1; i++)
    {
        l = 0;
        j++;
        if (masiv[j, i] == i)
        {
            for (int k = 1; k < n + 1; k++)
            {
                if (masiv[k, j] == k) l++;
                if (l == n) r = j;
            }
        }
    }
    if (r != 0) Console.WriteLine("ESTE UNITATE DREAPTA  " + r);
    else Console.WriteLine("NU ESTE UNITATE STANGA");
}
```

```
public static void unitate_stanga(int[,] masiv, int n, out int r2)
{
    int l;
    int j = 0; r2 = 0;
    for (int i = 1; i < n + 1; i++)
    {
        l = 0;
        j++;
        if (masiv[i, j] == i)
        {
            for (int k = 1; k < n + 1; k++)
            {
                if (masiv[j, k] == k) l++;
                if (l == n) r2 = j;
            }
        }
    }
    if (r2 != 0) Console.WriteLine("ESTE UNITATE STANGA  " + r2);
    else Console.WriteLine("NU ESTE UNITATE STANGA");
}


public static void unitate(ref int r, ref int r2)
{

    if (r1 == r2 && r > 0) Console.WriteLine("Este unitate " + r);
    else Console.WriteLine("NU este unitate");

}

public static void ward(int[,] masiv, int n)
{
```

```
    int l = 0;
    for (int i = 1; i < n + 1; i++)
        for (int j = 1; j < n + 1; j++)
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[i, j];
                d1 = masiv[i, k];
                d2 = masiv[j, k];
                if (d != masiv[d1, d2]) l += 1;
            }

    if (l == 0) Console.WriteLine("ESTE WARD");
    else Console.WriteLine("NU ESTE WARD");
}


public static void ward_invers(int[,] masiv, int n)
{
    int l = 0;
    for (int i = 1; i < n + 1; i++)
        for (int j = 1; j < n + 1; j++)
            for (int k = 1; k < n + 1; k++)
            {
                d = masiv[i, j];
                d1 = masiv[k, i];
                d2 = masiv[k, j];
                if (d != masiv[d1, d2]) l += 1;
            }

    if (l == 0) Console.WriteLine("ESTE WARD INVERS");
    else Console.WriteLine("NU ESTE WARD INVERS");
}
```

```
        }
}
```

În urma compilării sau obţinut următoarele rezultate:

Ex. 1

```
    _1 _2 _3 _4
   | 4  1  2  3
   | 3  4  1  2
   | 2  3  4  1
   | 1  2  3  4

Introdu substitutia ALFA
2
3
4
1

Introdu substitutia BETA
4
3
2
1

Introdu substitutia GAMA
2
1
4
3

Iteratia ALFA
 3 4 1 2
 2 3 4 1
 1 2 3 4
 4 1 2 3

Iteratia BETA
 2 1 4 3
 1 4 3 2
 4 3 2 1
 3 2 1 4

Iteratia GAMA
 1 2 3 4
 2 3 4 1
 3 4 1 2
 4 1 2 3
```

```
Proprietatile algebrice grupoid initial
4 1 2 3
3 4 1 2
2 3 4 1
1 2 3 4
4=E(1,2)
Nu este asociativ
Este medial
Este paramedial
Este bicomutativ
Este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Este unitate de stanga 4
Nu este unitate
Nu este Ward
Este Ward invers

Proprietatile algebrice grupoid gama
1 2 3 4
2 3 4 1
3 4 1 2
4 1 2 3
1=E(1,1)
Este asociativ
Este medial
Este paramedial
Este AG grupoidv
Este GA grupoid
Este GA1 grupoid
Este AD grupoid
Este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Este unitate de dreapta 1
Este unitate de stanga 1
Este unitate 1
Nu este Ward
Nu este Ward invers
```

Ex. 2

```
    _1 _2 _3 _4
   | 1  2  3  4
   | 2  3  4  1
   | 4  1  2  3
   | 3  4  1  2

Introdu substitutia ALFA
2
3
1
4

Introdu substitutia BETA
3
1
4
2

Introdu substitutia GAMA
1
3
4
2_
Iteratia ALFA
 2 3 4 1
 4 1 2 3
 1 2 3 4
 3 4 1 2

Iteratia BETA
 4 2 1 3
 2 4 3 1
 1 3 2 4

Iteratia GAMA
 2 3 1 4
 3 2 4 1
 4 1 2 3
 1 4 3 2
```

```
Proprietatile algebrice grupoid initial
1 2 3 4
2 3 4 1
4 1 2 3
3 4 1 2
1=E(1,2)
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Este unitate de stanga 1
Nu este unitate
Nu este Ward
Nu este Ward invers

Proprietatile algebrice grupoid gama
2 3 1 4
3 2 4 1
4 1 2 3
1 4 3 2
2=E(3,2)
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward invers
```

Ex. 3

```
    _1 _2 _3 _4 _5
   | 1  2  3  4  5
   | 2  3  1  5  4
   | 4  1  5  2  3
   | 3  5  4  1  2
   | 5  4  2  3  1

Introdu substitutia ALFA
2
3
1
5
4

Introdu substitutia BETA
3
1
2
4
5

Introdu substitutia GAMA
5
3
2
4
1
Iteratia ALFA
 2 3 1 5 4
 4 1 5 2 3
 1 2 3 4 5
 5 4 2 3 1
 3 5 4 1 2

Iteratia BETA
 1 2 3 5 4
 5 4 1 2 3
 3 1 2 4 5
 2 5 4 3 1
 4 3 5 1 2
Iteratia GAMA
 5 3 2 1 4
 1 4 5 3 2
 2 5 3 4 1
 3 1 4 2 5
 4 2 1 5 3
```

```
Proprietatile algebrice grupoid initial
1  2  3  4  5
2 3 1 5 4
4 1 5 2 3
3 5 4 1 2
5 4 2 3 1
1=E(1,2)
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Este unitate de stanga 1
Nu este unitate
Nu este Ward
Nu este Ward invers

Proprietatile algebrice grupoid gama
5 3 2 1 4
1 4 5 3 2
2 5 3 4 1
3 1 4 2 5
4 2 1 5 3
3=E(3,3)
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward invers
```

Ex. 4

```
    _1 _2 _3
   | 1  2  3
   | 3  1  2
   | 2  3  1

Introdu substitutia ALFA
1
3
2

Introdu substitutia BETA
3
2
1

Introdu substitutia GAMA
2
1
3
Iteratia ALFA
 1 2 3
 2 3 1
 3 1 2

Iteratia BETA
 3 2 1
 1 3 2
 2 1 3

Iteratia GAMA
 3 1 2
 2 3 1
 1 2 3
```

```
Proprietatile algebrice grupoid initial
1 2 3
3 1 2
2 3 1
1=E(1,2)
Nu este asociativ
Este medial
Este paramedial
Este bicomutativ
Este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Este unitate de stanga 1
Nu este unitate
Nu este Ward
Este Ward invers

Proprietatile algebrice grupoid gama
3 1 2
2 3 1
1 2 3
3=E(1,2)
Nu este asociativ
Este medial
Este paramedial
Este bicomutativ
Este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Este unitate de stanga 3
Nu este unitate
Nu este Ward
Este Ward invers
```

Ex. 5

```
   | 1  2  3  4
   | 2  1  4  3
   | 3  4  1  2
   | 4  3  2  1

Introdu substitutia ALFA
4
3
2
1

Introdu substitutia BETA
4
3
2
1

Introdu substitutia GAMA
4
3
2
1
Iteratia ALFA
 4 3 2 1
 3 4 1 2
 2 1 4 3
 1 2 3 4

Iteratia BETA
 1 2 3 4
 2 1 4 3
 3 4 1 2
 4 3 2 1

Iteratia GAMA
 4 3 2 1
 3 4 1 2
 2 1 4 3
 1 2 3 4
```

```
Proprietatile algebrice grupoid initial
1 2 3 4
2 1 4 3
3 4 1 2
4 3 2 1
1=E(1,1)
Este asociativ
Este medial
Este paramedial
Este bicomutativ
Este GA grupoid
Este GA1 grupoid
Este AD grupoid
Este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Este unitate de dreapta 1
Este unitate de stanga 1
Este unitate 1
Este Ward
Este Ward invers

Proprietatile algebrice grupoid gama
4 3 2 1
3 4 1 2
2 1 4 3
1 2 3 4
4=E(1,1)
Este asociativ
Este medial
Este paramedial
Este bicomutativ
Este AG grupoidv
Este GA grupoid
Este GA1 grupoid
Este AD grupoid
Este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Este unitate de dreapta 4
Este unitate de stanga 4
Este unitate 4
Este Ward
Este Ward invers
```

Ex. 6

```
    _1 _2 _3 _4
   | 1  2  3  4
   | 3  1  4  2
   | 2  4  1  3
   | 4  3  2  1

Introdu substitutia ALFA
4
3
2
1

Introdu substitutia BETA
3
4
1
2

Introdu substitutia GAMA
2
3
4
1
Iteratia ALFA
 4 3 2 1
 2 4 1 3
 3 1 4 2
 1 2 3 4

Iteratia BETA
 2 1 4 3
 1 3 2 4
 4 2 3 1
 3 4 1 2

Iteratia GAMA
 3 2 1 4
 2 4 3 1
 1 3 4 2
 4 1 2 3
```

```
Proprietatile algebrice grupoid initial
1 2 3 4
3 1 4 2
2 4 1 3
4 3 2 1
1=E(1,2)
Nu este asociativ
Este medial
Este paramedial
Este bicomutativ
Este AG grupoid
Nu este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Este unitate de stanga 1
Nu este unitate
Nu este Ward
Este Ward invers

Proprietatile algebrice grupoid gama
3 2 1 4
2 4 3 1
1 3 4 2
4 1 2 3
Nu este asociativ
Nu este medial
Este paramedial
Este bicomutativ
Nu este AG grupoid
Este GA grupoid
Nu este GA1 grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward invers
```

Ex. 7

```
    _1 _2 _3 _4 _5
   | 1  2  3  4  5
   | 2  3  4  5  1
   | 3  4  5  1  2
   | 4  5  1  2  3
   | 5  1  2  3  4

Introdu substitutia ALFA
5
4
3
2
1

Introdu substitutia BETA
4
3
2
1
5

Introdu substitutia GAMA
3
2
1
5
4
Iteratia ALFA
 5 1 2 3 4
 4 5 1 2 3
 3 4 5 1 2
 2 3 4 5 1
 1 2 3 4 5

Iteratia BETA
 3 2 1 5 4
 2 1 5 4 3
 1 5 4 3 2
 5 4 3 2 1
 4 3 2 1 5

Iteratia GAMA
 1 2 3 4 5
 2 3 4 5 1
 3 4 5 1 2
 4 5 1 2 3
 5 1 2 3 4
```

```
Proprietatile algebrice grupoid initial
1 2 3 4 5
2 3 4 5 1
3 4 5 1 2
4 5 1 2 3
5 1 2 3 4
1=E(1,1)
Este asociativ
Este paramedial
Este bicomutativ
Este AG grupoid
Este GA grupoid
Este GA1 grupoid
Este AD grupoid
Este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Este unitate de dreapta 1
Este unitate de stanga 1
Este unitate 1
Nu este Ward
Nu este Ward invers
Proprietatile algebrice grupoid gama
1 2 3 4 5
2 3 4 5 1
3 4 5 1 2
4 5 1 2 3
5 1 2 3 4
1=E(1,1)
Este asociativ
Este medial
Este paramedial
Este bicomutativ
Este AG grupoid
Este GA grupoid
Este GA1 grupoid
Este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stinga
Este unitate de dreapta 1
Este unitate de stanga 1
Este unitate 1
Nu este Ward
Nu este Ward invers
```

Tabelul rezultatelor obţinute :

	Ex.1		Ex.2		Ex.3		Ex.4		Ex.5		Ex.6		Ex.7	
α	3 4 1 2 2 3 4 1 1 2 3 4 4 1 2 3		2 3 4 1 4 1 2 3 1 2 3 4 3 4 1 2		2 3 1 5 4 4 1 5 2 3 1 2 3 4 5 5 4 3 2 1		1 2 3 2 3 1 3 1 2		4 3 2 1 3 4 1 2 2 1 4 3 1 2 3 4		4 3 2 1 2 4 1 3 3 1 4 2 1 2 3 4		5 1 2 3 4 4 5 1 2 3 3 4 5 1 2 2 3 4 5 1	
β	2 1 4 3 1 4 3 2 4 3 2 1 3 2 1 4		4 2 1 3 2 4 3 1 3 1 4 2 1 3 2 4		1 2 3 5 4 5 4 1 2 3 3 1 2 4 5 2 5 4 3 1		3 2 1 1 3 2 3 1 2		1 2 3 4 2 1 4 3 3 4 1 2 4 3 2 1		2 1 4 3 1 3 2 4 4 2 3 1 3 4 1 2		3 2 1 5 4 2 1 5 4 3 1 5 4 3 2 5 4 3 2 1	
γ	1 2 3 4 2 3 4 1 3 4 1 2 4 1 2 3		3 4 1 2 2 3 4 1 1 2 3 4 4 1 2 3		5 3 2 1 4 1 4 5 3 2 2 5 3 4 1 3 1 4 2 5		3 1 2 2 3 1 1 2 3		4 3 2 1 3 4 1 2 2 1 4 3 1 2 3 4		3 2 1 4 2 4 1 3 1 3 4 2 4 1 2 3		1 2 3 4 5 2 3 4 5 1 3 4 5 1 2 4 5 1 2 3	
multiple	4=E(1,2)	1=E(1,1)	1=E(1,2)	2=E(3,2)	1=E(1,2)	3=E(3,3)	1=E(1,2)	3=E(1,2)	1=E(1,1)	4=E(1,1)	1=E(1,2)	X	1=E(1,1)	1=E(1,1)
tiv	-	+	-	-	-	-	-	-	+	+	-	-	+	+
	+	+	-	-	-	-	+	+	+	+	+	-	+	+
edial	+	+	-	-	-	-	+	+	+	+	+	+	+	+
utativ	+	+	-	-	-	-	+	+	+	+	+	-	+	+
upoid	+	+	-	-	-	-	+	+	+	+	+	+	+	+
upoid	-	+	-	-	-	-	-	-	+	+	-	-	+	+
upoid	-	+	-	-	-	-	-	-	+	+	-	-	+	+
upoid	-	+	-	-	-	-	-	-	+	+	-	-	+	+
upoid	-	+	-	-	-	-	-	-	+	+	-	-	+	+
onal	-	-	-	-	-	-	-	-	-	-	-	-	-	-
utiv la dreapta	-	-	-	-	-	-	-	-	-	-	-	-	-	-
utiv la stânga	-	-	-	-	-	-	-	-	-	-	-	-	-	-
de dreapta	-	1	-	-	-	-	-	-	1	4	-	-	1	1
de stânga	4	1	1	-	1	-	1	3	1	4	1	-	1	1
	-	1	-	-	-	-	-	-	1	4	-	-	1	1
	-	-	-	-	-	-	-	-	+	+	-	-	-	-
nvers	+	-	-	-	-	-	+	+	+	+	+	-	-	-