

CATEDRA

FIZICĂ, MATEMATICĂ ȘI TEHNOLOGII INFORMAȚIONALE

SPECIALITATEA

TEHNOLOGII INFORMAȚIONALE ÎN INSTRUIRE

Structuri algebrice pe calculator

Laboratorul Nr.4

Realizat: Cojucovschi Ion

Grupa: CIII

Verificat: Chiriac Liubomir

Chișinău, 2018

Algoritmi de realizare a produsului direct special dintre doi grupoizi

Problemă. Se dă grupul(), se cere de elaborat un algoritm care realizează produsul direct special(), conform următoarelor legi:

Legea 1

(

Legea 2

(

Pentru fiecare din exemplele de mai jos să se realizeze ()

Ex.1

+	1	2	3
1	1	2	3
2	2	3	1
3	3	1	2

Ex.2

+	1	2
1	1	2
2	2	1

Ex.3

+	1	2	3	4
1	1	2	3	4
2	2	3	4	1
3	3	4	1	2
4	4	1	2	3

Ex.4

+	1	2	3	4
1	1	2	3	4
2	2	1	4	3
3	3	4	1	2
4	4	3	2	3

Ex.5

+	1	2	3
1	1	2	3
2	3	1	2
3	2	3	1

Elaborarea programului:

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

namespace Laboratorul_numarul_4
{
    class Program
    {
        static int[, ] a, b, tabel, opus, t1, p1, gr;
        static int[] 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[, ] b = new int[7, 7] { { 0, 0, 0, 0, 0, 0, 0 }, { 0, 1, 2, 3, 4, 5, 6 }, { 0, 3, 1, 2, 6, 4, 5 }, { 0, 2, 3, 1, 5, 6, 4 }, { 0, 4, 5, 6, 1, 2, 3 }, { 0, 6, 4, 5, 3, 1, 2 }, { 0, 5, 6, 4, 2, 3, 1 } };
            int[, ] ai = new int[3, 3] { { 0, 0, 0 }, { 0, 1, 2 }, { 0, 2, 1 } };
```

```

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

        ////initializationmatrix(a,n);          ///initiate matrix
        produs_cartezian(out a, ai, b, n, m);

        r = 0; r1 = 0;

        afisare(a, n);
        asociativ(a, n);
        medial(a, n);
        paramedial(a, n);
        bicomutativ(a, n);
        ag_gr(a, n);
        ga_gr(a, n);
        ga_grl(a, n);
        ad_gr(a, n);
        da_gr(a, n);
        hexagonal(a, n);
        dist_dr(a, n);
        dist_st(a, n);
        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 = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                Console.WriteLine("a[" + i + 1 + "," + j + 1 + "]=");
                a[i, j] = Convert.ToInt32(Console.ReadLine());
            }
        }
    }

}

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];
                    dl = masiv[k, i];
                    r1 = masiv[k, j];
                    if (masiv[k, d] != masiv[dl, 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;
            }

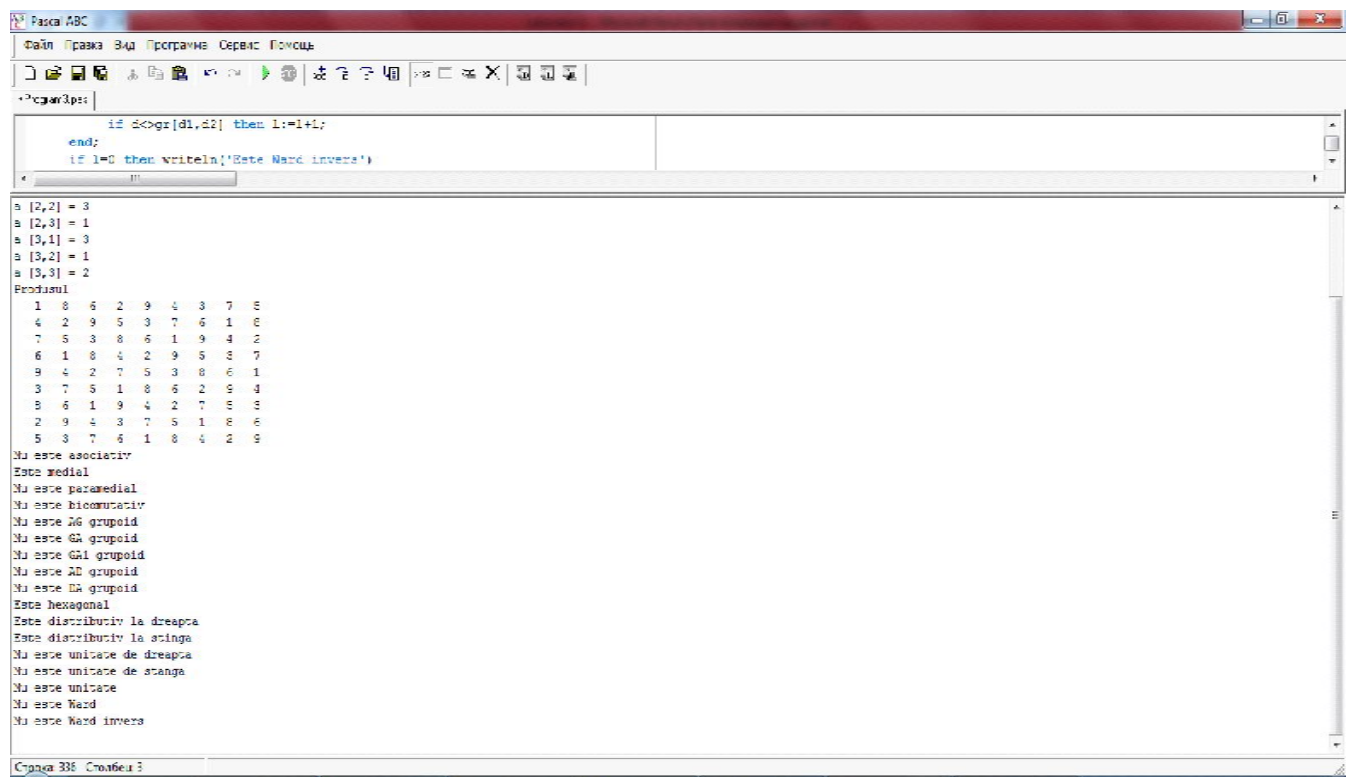
    if (l == 0) Console.WriteLine("ESTE WARD INVERS");
    else Console.WriteLine("NU ESTE WARD INVERS");
}

}
}

```

Rezultatele obținute în urma compilării conform legii 1:

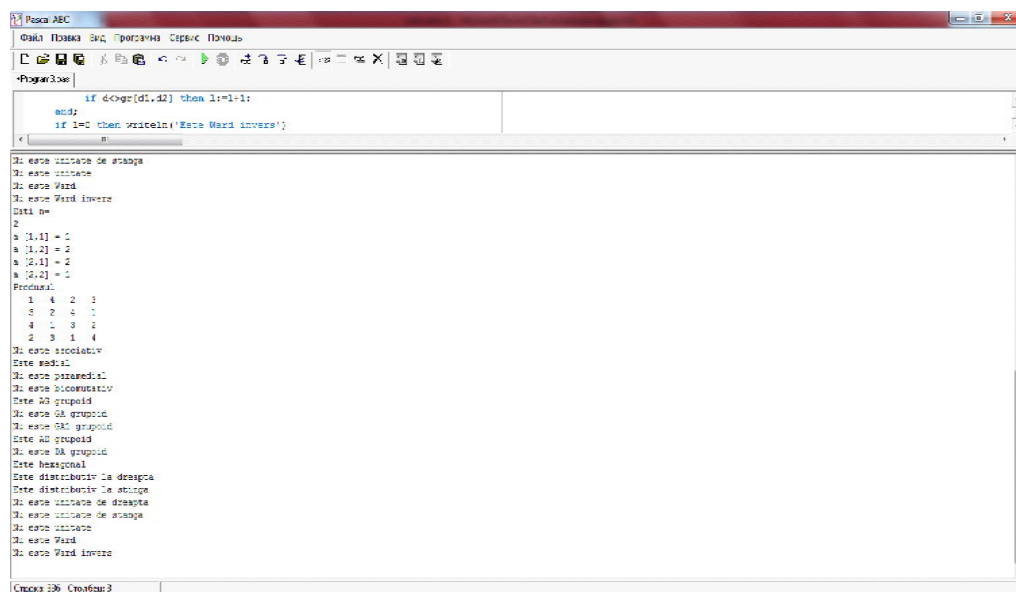
Ex.1



```
if <ogr[d1,d2] then l:=l+1;
end;
if l=0 then writeln('Este Ward inversa')
```

a [2,2] = 3
a [2,3] = 1
a [3,1] = 3
a [3,2] = 1
a [3,3] = 2
Produsul
1 8 6 2 9 4 3 7 5
4 2 9 5 3 7 6 1 8
7 5 3 8 6 1 9 4 2
6 1 8 4 2 9 5 3 7
9 4 2 7 5 3 8 6 1
3 7 5 1 8 6 2 9 4
8 6 1 9 4 2 7 5 3
2 9 4 3 7 5 1 8 6
5 3 7 4 1 8 4 2 9
Nu este asociativ
Este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GAI grupoid
Nu este AI grupoid
Nu este LA grupoid
Este hexagonal
Este distributiv la dreapta
Este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward inversa

Ex.2



```
if <ogr[d1,d2] then l:=l+1;
end;
if l=0 then writeln('Este Ward inversa')
```

Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward inversa
Data n=
2
a [1,1] = 1
a [1,2] = 2
a [2,1] = 2
a [2,2] = 1
Produsul
1 4 2 3
3 2 4 1
4 1 3 2
2 3 1 4
Nu este asociativ
Este medial
Nu este paramedial
Nu este bicomutativ
Este AG grupoid
Nu este GA grupoid
Nu este GAI grupoid
Nu este AI grupoid
Nu este LA grupoid
Este hexagonal
Este distributiv la dreapta
Este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward inversa

Ex.3

```

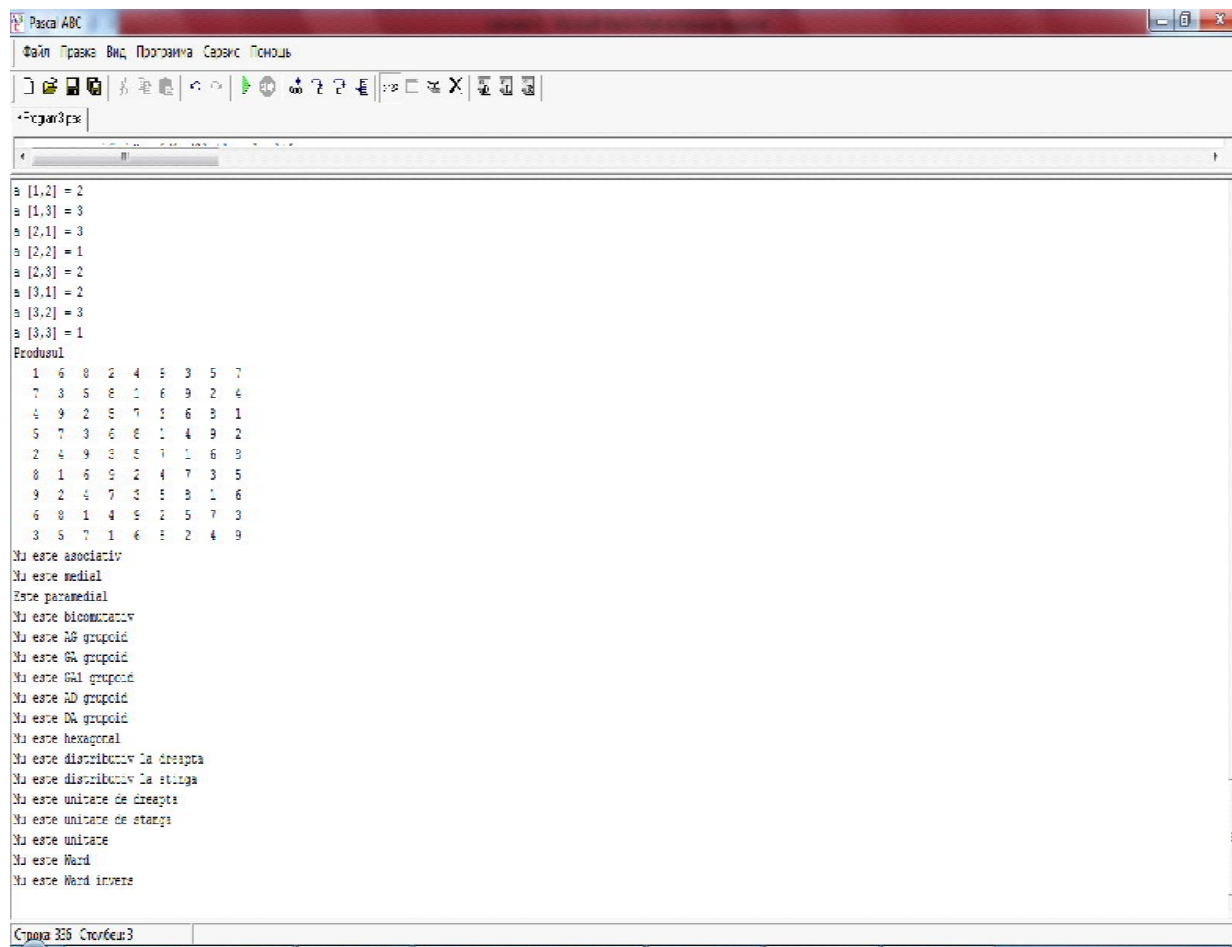
Pascal ABC
Файл  Правка  Вид  Программа  Сервис  Помощь
[Icons]
*Пример3.pas
[Code Editor]
a [4,4] = 3
Produsul
1 14 11 8 2 15 12 5 3 16 9 6 4 13 10 7
5 2 15 12 6 3 16 9 7 4 13 10 8 1 14 11
9 6 3 16 10 7 4 13 11 8 1 14 12 5 2 15
13 10 7 4 14 11 9 1 15 12 5 2 16 9 6 3
8 1 14 11 5 2 15 12 6 3 16 9 7 4 13 10
12 5 2 15 9 6 3 16 10 7 4 13 11 8 1 14
16 9 6 3 13 10 7 4 14 11 8 1 15 12 5 2
4 13 10 7 1 14 11 9 2 15 12 5 3 16 9 6
11 8 1 14 12 5 2 15 9 6 3 16 10 7 4 13
15 12 5 2 16 9 6 3 13 10 7 4 14 11 8 1
3 16 9 6 4 13 10 7 1 14 11 8 2 15 12 5
7 4 13 10 8 1 14 11 5 2 15 12 6 3 16 9
14 11 8 1 15 12 5 2 16 9 6 3 13 10 7 4
2 15 12 5 3 16 9 6 4 13 10 7 1 14 11 8
6 3 16 9 7 4 13 10 9 1 14 11 5 2 15 12
10 7 4 13 11 8 1 14 12 5 2 15 9 6 3 16
Nu este asociativ
Este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este CA grupoid
Nu este GA grupoid
Nu este AD grupoid
Nu este DA grupoid
Este hexagonal
Este distributiv la dreapta
Este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward invers
[Status Bar]
Строка:335  Столбец:3
  
```

Ex.4

```

Pascal ABC
Файл  Правка  Вид  Программа  Сервис  Помощь
[Icons]
*Пример3.pas
[Code Editor]
a [4,4] = 1
Produsul
1 6 11 16 2 5 12 15 3 8 9 14 4 7 10 13
5 2 15 12 6 1 16 11 7 4 13 10 9 3 14 9
9 14 3 8 10 13 4 7 11 16 1 6 12 15 2 5
13 10 7 4 14 9 8 3 15 12 5 2 16 11 6 1
6 1 16 11 5 2 15 12 8 3 14 9 7 4 13 10
2 5 12 15 1 6 11 16 4 7 10 13 3 9 14
14 9 8 3 13 10 7 4 16 11 6 1 15 12 5 2
10 13 4 7 9 14 5 8 12 15 2 5 11 16 1 6
11 16 1 6 12 15 2 5 9 14 3 9 10 13 4 7
15 12 5 2 16 11 6 1 13 10 7 4 14 9 8 3
3 8 9 14 4 7 10 13 1 6 11 16 2 5 12 15
7 4 13 10 8 3 14 9 6 2 15 12 6 1 16 11
16 11 6 1 15 12 5 2 14 9 8 3 13 10 7 4
12 15 2 5 11 16 1 6 10 13 4 7 9 14 3 8
8 3 14 9 7 4 13 10 6 1 16 11 5 2 15 12
4 7 10 13 3 8 9 14 2 5 12 15 1 6 11 16
Nu este asociativ
Este medial
Nu este paramedial
Nu este bicomutativ
Este AG grupoid
Nu este CA grupoid
Nu este GA grupoid
Este AD grupoid
Nu este DA grupoid
Este hexagonal
Este distributiv la dreapta
Este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward invers
[Status Bar]
Строка:336  Столбец:3
  
```

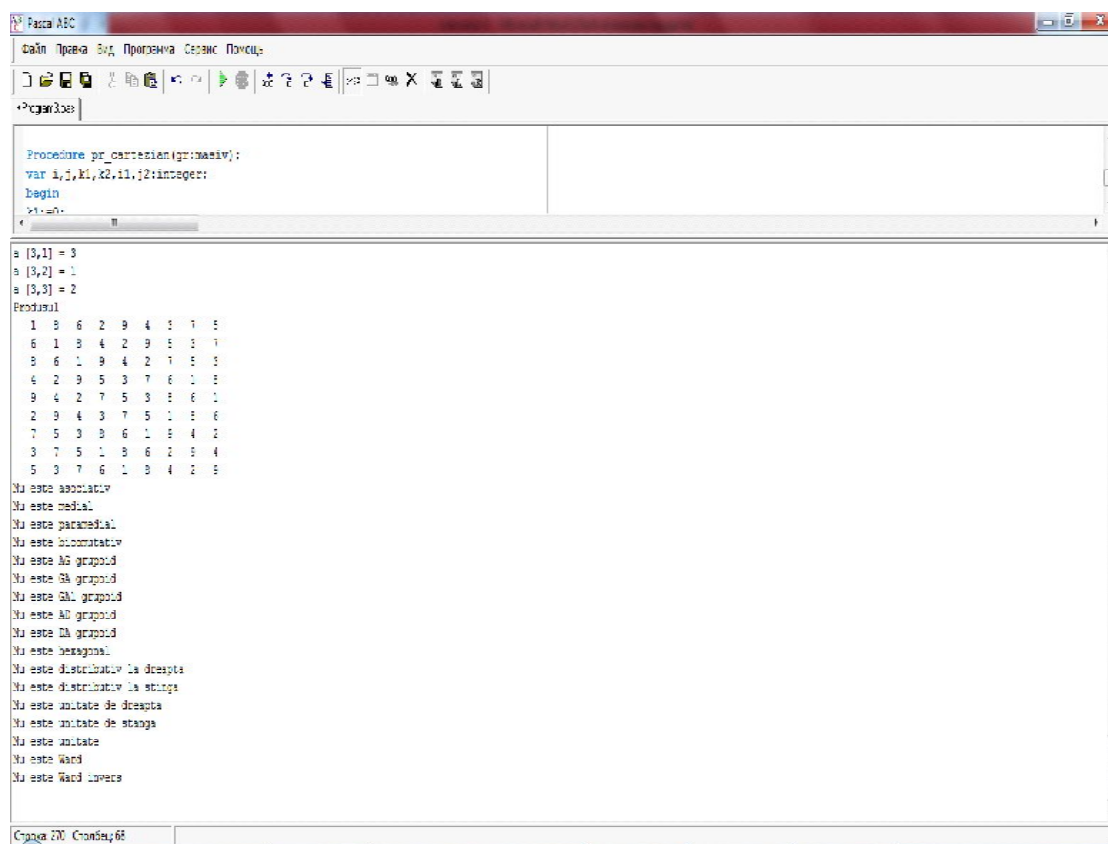
Ex.5



```
Pascal ABC
Файл Правка Вид Программы Справка Помощь
[Icons]
+Результат:
a [1,2] = 2
a [1,3] = 3
a [2,1] = 3
a [2,2] = 1
a [2,3] = 2
a [3,1] = 2
a [3,2] = 3
a [3,3] = 1
Produsul
  1  6  8  2  4  5  3  5  7
  7  3  5  8  1  6  9  2  4
  4  9  2  5  7  1  6  8  1
  5  7  3  6  8  1  4  9  2
  2  4  9  3  5  7  1  6  8
  8  1  6  5  2  4  7  3  5
  9  2  4  7  3  5  8  1  6
  6  8  1  4  5  2  5  7  3
  3  5  7  1  6  5  2  4  9
Nu este asociativ
Nu este medial
Este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GHL grupoid
Nu este AG grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward invers
Строка 336 Столбец 3
```

Rezultatele obținute în urma compilării conform legii 2:

Ex. 1



```
Pascal ABC
Файл Правка Вид Программы Справка Помощь
[Icons]
+Результат:
Procedure pr_cartesian(grupoidiv);
var i,j,k1,x2,ii,j2:integer;
begin
  ii:=0;
a [3,1] = 3
a [3,2] = 1
a [3,3] = 2
Produsul
  1  8  6  2  9  4  5  7  1  5
  6  1  8  4  2  9  5  1  1
  8  6  1  9  4  2  7  1  1
  4  2  8  5  3  7  6  1  1
  8  4  2  7  5  3  1  6  1
  2  8  4  3  7  5  1  1  6
  7  5  3  8  6  1  9  4  2
  3  7  5  1  8  6  2  1  4
  5  3  7  6  1  8  4  2  1
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GHL grupoid
Nu este AG grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward invers
Строка 270 Столбец 66
```

Ex. 2

```

Pascal ABC
Файл Правка Вид Программа Сервис Помощь
[Icons]
Программа
Procedure pr_matrix(a:integer):
var i,j,k1,k2,i1,i2:integer;
begin
  writeln;
end;

Data: n=
• Программа: проверка свойств матрицы
Data: n=
2
a [1,1] = 1
a [1,2] = 2
a [2,1] = 2
a [2,2] = 1
Prodsul:
  1  4  2  8
  4  1  3  2
  3  2  4  1
  2  3  1  4
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GL grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward inversa
Строка 270 Столбец 68
  
```

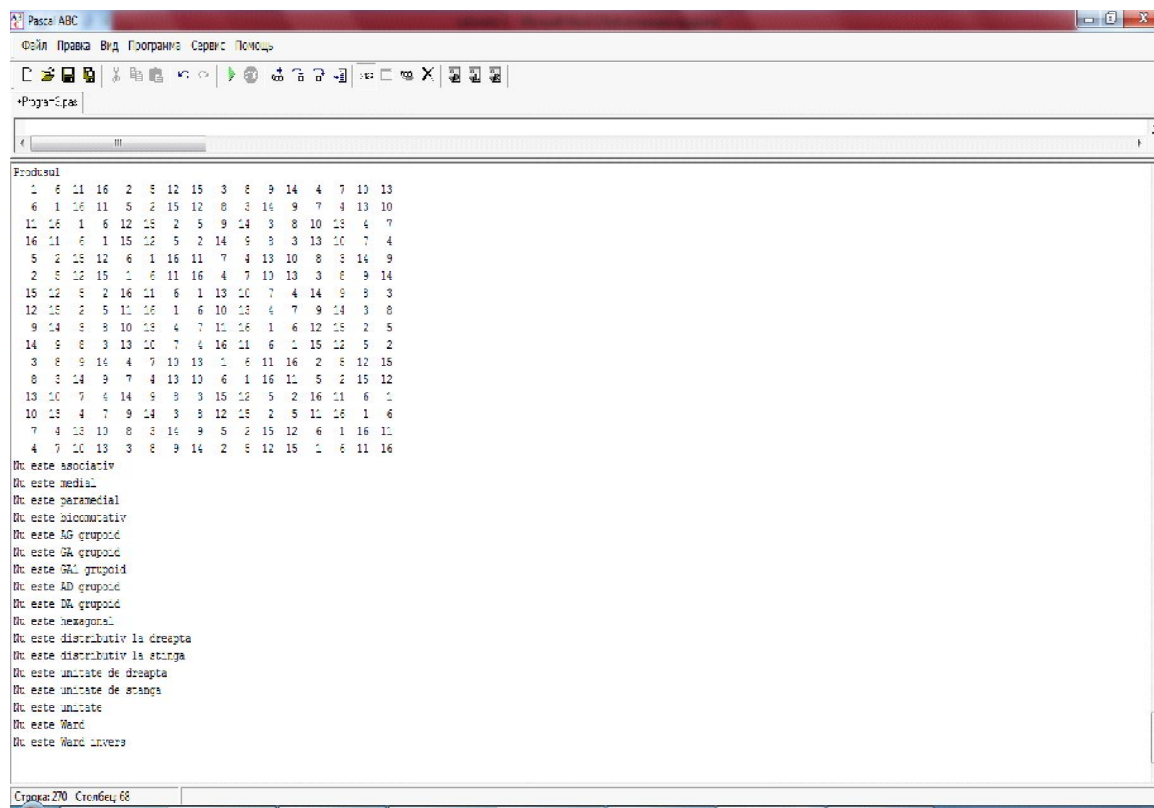
Ex. 3

```

Pascal ABC
Файл Правка Вид Программа Сервис Помощь
[Icons]
Программа
Procedure pr_matrix(a:integer):
var i,j,k1,k2,i1,i2:integer;
begin
  writeln;
end;

Data: n=
• Программа: проверка свойств матрицы
Data: n=
10
Prodsul:
  1  14  11  8  2  15  12  9  3  16  9  8  4  13  10  7
  8  1  14  11  5  2  15  12  6  3  16  9  7  4  13  10
  11  8  1  14  12  5  2  15  9  6  3  16  10  7  4  13
  14  11  8  1  15  12  5  2  16  9  6  3  13  10  7  4
  5  2  15  12  6  3  16  9  7  4  13  10  8  1  14  11
  12  5  2  15  9  6  3  16  10  7  4  13  11  8  1  14
  15  12  5  2  16  9  6  3  13  10  7  4  14  11  8  1
  2  15  12  5  3  16  9  6  4  13  10  7  1  14  11  8
  9  6  3  16  10  7  4  13  11  8  1  14  12  5  2  15
  16  9  6  3  13  10  7  4  14  11  8  1  15  12  5  2
  3  16  9  6  4  13  10  7  1  14  11  8  2  15  12  5
  6  3  16  9  7  4  13  10  8  1  14  11  5  2  15  12
  13  10  7  4  14  11  8  1  15  12  5  2  16  9  6  3
  4  13  10  7  1  14  11  8  2  15  12  5  3  16  9  6
  7  4  13  10  8  1  14  11  5  2  15  12  6  3  16  9
  10  7  4  13  11  8  1  14  12  5  2  15  9  6  3  16
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este bicomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GL grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward inversa
Строка 270 Столбец 68
  
```

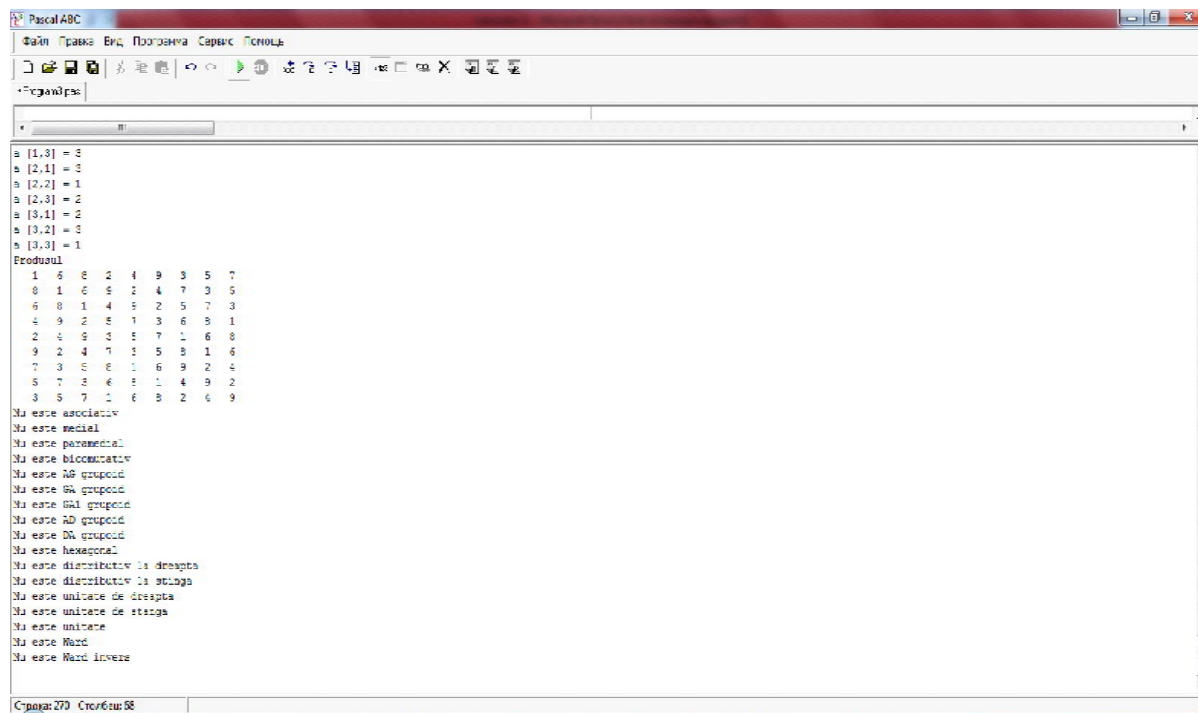
Ex. 4



```

Pascal ABC
Файл  Правка  Вид  Программы  Сервис  Помощь
[Icons]
+Program.pas
+
-----
Produsul
1  6  11  16  2  5  12  15  3  8  9  14  4  7  13  10
6  1  16  11  5  2  15  12  8  3  14  9  7  4  13  10
11  16  1  6  12  15  2  5  9  14  3  8  10  13  4  7
16  11  6  1  15  12  5  2  14  9  3  13  10  7  4
5  2  15  12  6  1  16  11  7  4  13  10  8  3  14  9
2  5  12  15  1  6  11  16  4  7  13  10  3  8  9  14
15  12  3  2  16  11  6  1  13  10  7  4  14  9  8  3
12  15  2  5  11  16  1  6  10  13  4  7  9  14  3  8
9  14  3  8  10  13  4  7  11  16  1  6  12  15  2  5
14  9  8  3  13  10  7  4  16  11  6  1  15  12  5  2
3  8  9  14  4  7  13  10  1  6  11  16  2  5  12  15
8  3  14  9  7  4  13  10  6  1  16  11  5  2  15  12
13  10  7  4  14  9  3  15  12  5  2  16  11  8  3
10  13  4  7  9  14  3  12  15  2  5  11  16  1  6
7  4  13  10  8  3  14  9  5  2  15  12  6  1  16  11
4  7  10  13  3  8  9  14  2  5  12  15  1  6  11  16
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este biocomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GAI grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward inversa
-----
Строка 270  Столбец 68
  
```

Ex. 5



```

Pascal ABC
Файл  Правка  Вид  Программы  Сервис  Помощь
[Icons]
+Program.pas
+
-----
a [1,1] = 3
a [2,1] = 3
a [2,2] = 1
a [2,3] = 2
a [3,1] = 2
a [3,2] = 3
a [3,3] = 1
Produsul
1  6  2  4  9  3  5  7
6  1  6  5  2  4  7  3  5
9  6  1  4  5  2  5  7  3
2  4  5  3  5  7  1  6  8
9  2  4  7  5  5  5  1  6
7  3  5  6  1  6  9  2  4
5  7  3  6  5  1  4  9  2
3  5  7  1  6  5  2  6  9
Nu este asociativ
Nu este medial
Nu este paramedial
Nu este biocomutativ
Nu este AG grupoid
Nu este GA grupoid
Nu este GAI grupoid
Nu este AD grupoid
Nu este DA grupoid
Nu este hexagonal
Nu este distributiv la dreapta
Nu este distributiv la stanga
Nu este unitate de dreapta
Nu este unitate de stanga
Nu este unitate
Nu este Ward
Nu este Ward inversa
-----
Строка 270  Столбец 68
  
```

Tabelul rezultatelor obținute:

[illegible]