

Prelucrarea unei matrici

Ce este o matrice pătratică?

O **matrice pătratică** este o matrice particulară, cu același număr de linii și de coloane — adică cu $n = m$. Pentru simplitate, vom menționa doar n -ul, deoarece m -ul are aceeași valoare. Matricea de acest tip are o serie de particularități care se folosesc în anumite probleme.

Diagonala principală și diagonala secundară

Diagonala principală a unei matrice este diagonala care începe din colțul stânga-sus și se termină în colțul dreapta-jos. Mai exact, un element $a[i][j]$ aparține diagonalei principale dacă $i = j$.

Similar, diagonala secundară este cealaltă diagonală, care începe din colțul dreapta-sus și se termină în cel stânga-jos. Un element $a[i][j]$ aparține diagonalei secundare dacă $i + j = n + 1$, sau cu alte cuvinte, $j = n - i + 1$ (unde matricea este indexată de la 1).

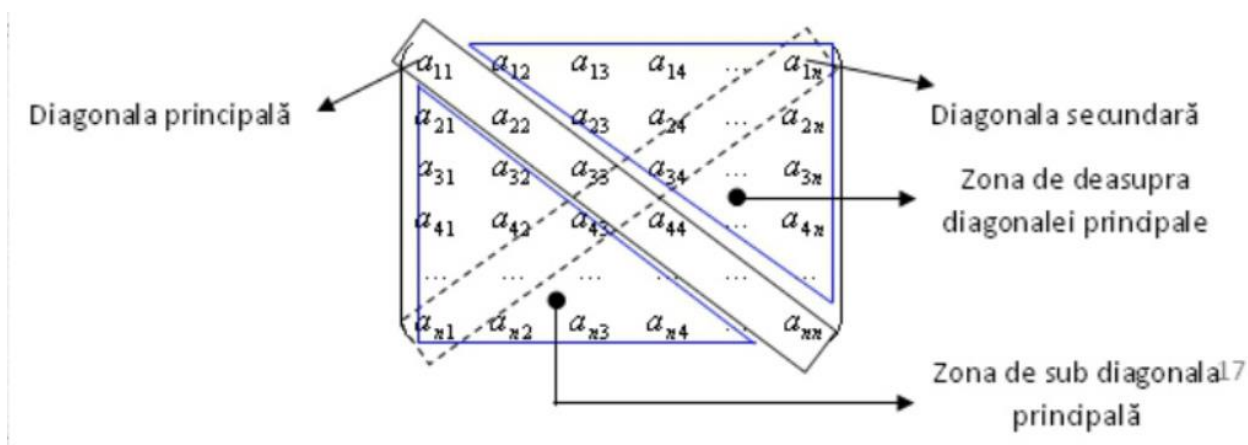


Figura 1. Structura unei matrici pătratice

1. Se cere implementarea unui program Java care să citească o matrice pătratică de dimensiune $n \times n$ de la tastatură, să afișeze elementele diagonalei secundare, și să calculeze sumele elementelor aflate deasupra și sub diagonala secundară, precum și deasupra și sub diagonala principală.

```
1 package laborator3;
2 import java.io.*;
3 public class PrelucrareMatrice {
4     public static int [][] citesteMatrice() throws IOException {
5         // Metoda pentru citirea unei matrice de la tastatură
6         InputStreamReader isr = new InputStreamReader(System.in);
7         BufferedReader br = new BufferedReader(isr);
8         System.out.println("introduceti nr. de linii");
9         String s = br.readLine(); // Citirea numărului de linii (și coloane) ale matricei
10        int n = Integer.parseInt(s);
11        int [][] matrice = new int[n][n];
12        // Declararea matricei de dimensiune n x n
13        System.out.println("introduceti elementele");
14        for(int i = 0; i < n; i++) {
15            for(int j = 0; j < n; j++) {
16                // Parcurgerea matricei pentru a citi elementele
17                s = br.readLine();
18                matrice[i][j] = Integer.parseInt(s);
19                // Conversia la numere întregi și atribuirea elementelor în matrice
20            }
21        }
22        return matrice;
23    }
24}
```

Fig 2. Declararea și parcurgerea matricei

```
24 public static int [] diagonalaSec(int [][] matrice) {
25     // Metoda pentru afișarea diagonalei secundare a matricei
26     int [] diags = new int[matrice.length];
27     for(int i = 0; i < matrice.length; i++) {
28         diags[i] = matrice[i][matrice.length - i - 1];
29         System.out.println("elem diag secundara" + diags[i]);
30         // Afișarea elementelor de pe diagonala secundară
31     }
32     return diags;
33 }
34
35
36 public static int sumDiagSecD(int [][] matrice) {
37     // Metoda pentru calcularea sumei elementelor de deasupra diagonalei secundare
38     int suma = 0;
39     for(int i = 0; i < matrice.length - 1; i++)
40         for(int j = 0; j < matrice.length - i - 1; j++)
41             suma += matrice[i][j];
42     return suma;
43 }
44
45
46 public static int sumDiagSecS(int [][] matrice) {
47     // Metoda pentru calcularea sumei elementelor de sub diagonala secundară
48     int suma = 0;
49     for(int i = 1; i < matrice.length; i++)
50         for(int j = matrice.length - i; j < matrice.length; j++)
51             suma += matrice[i][j];
52     return suma;
53 }
```

Fig 3. Calculare sume și afișare diagonală secundară

```
public static int sumDeasDiagprinc(int [][] matrice) {
    // Metoda pentru calcularea sumei elementelor de deasupra diagonalei principale
    int suma = 0;
    for(int i = 0; i < matrice.length; i++)
        for(int j = 0; j < matrice.length; j++)
            if(i < j)
                suma += matrice[i][j];
    return suma;
}

public static int sumSubDiagPric(int [][] matrice) {
    // Metoda pentru calcularea sumei elementelor de sub diagonala principală
    int suma = 0;
    for(int i = 0; i < matrice.length; i++)
        for(int j = 0; j < matrice.length; j++)
            if(i > j)
                suma += matrice[i][j];
    return suma;
}
```

Fig 4. Calculare sumă diagonală pricipală

```
74 public static void main (String [] args) throws IOException {
75     int a[][] = citesteMatrice();
76     // Apelul metodei de citire a matricei
77     int [] ds = diagonalaSec(a);
78     // Apelul metodei de afişare a diagonalei secundare
79     for(int i = 0; i < ds.length; i++)
80         System.out.println(" " + ds[i]);
81     System.out.println("Suma elem de deasupra diag. sec.: " + sumDiagSecD(a));
82     // Afişarea sumei elementelor de deasupra diagonalei secundare
83     System.out.println("Suma elem de sub diag. sec.: " + sumDiagSecS(a));
84     // Afişarea sumei elementelor de sub diagonala secundară
85 }
86 }
```

Fig 5. Main programului

2. Se cere implementarea unui program Java care să citească o matrice pătratică de dimensiune $n \times n$ de la tastatură, să verifice dacă matricea este **simetrică** față de diagonala principală, și să afișeze un mesaj corespunzător (“Matrice simetrică” / “Matrice nesimetrică”).

```
package lab3;
import java.util.Scanner;
public class simetric {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Introduceti dimensiunea n: ");
        int n = in.nextInt();
        int[][] a = new int[n][n];
        System.out.println("Introduceti elementele matricei:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                a[i][j] = in.nextInt();
            }
        }
        boolean simetrica = true;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (a[i][j] != a[j][i]) {
                    simetrica = false;
                    break;
                }
            }
        }
        if (simetrica)
            System.out.println("Matrice simetrica");
        else
            System.out.println("Matrice nesimetrica");
    }
}
```

3. Se cere implementarea unui program Java care să citească o matrice pătratică de dimensiune $n \times n$ de la tastatură și să afișeze **elementele aflate pe conturul matricei**, precum și **suma acestora**.

```
package lab3;
import java.util.Scanner;
public class contur {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Introduceti dimensiunea n: ");
        int n = in.nextInt();
        int[][] a = new int[n][n];
        System.out.println("Introduceti elementele matricei:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                a[i][j] = in.nextInt();
            }
        }

        int suma = 0;
        System.out.println("Elementele de pe conturul matricei sunt:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                // Conditie pentru contur:
                if (i == 0 || i == n - 1 || j == 0 || j == n - 1) {
                    System.out.print(a[i][j] + " ");
                    suma += a[i][j];
                } else {
                    System.out.print(" "); // spatiu pentru forma dreptunghiulara
                }
            }
            System.out.println();
        }

        System.out.println("Suma elementelor de pe contur este: " + suma);
    }
}
```