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).

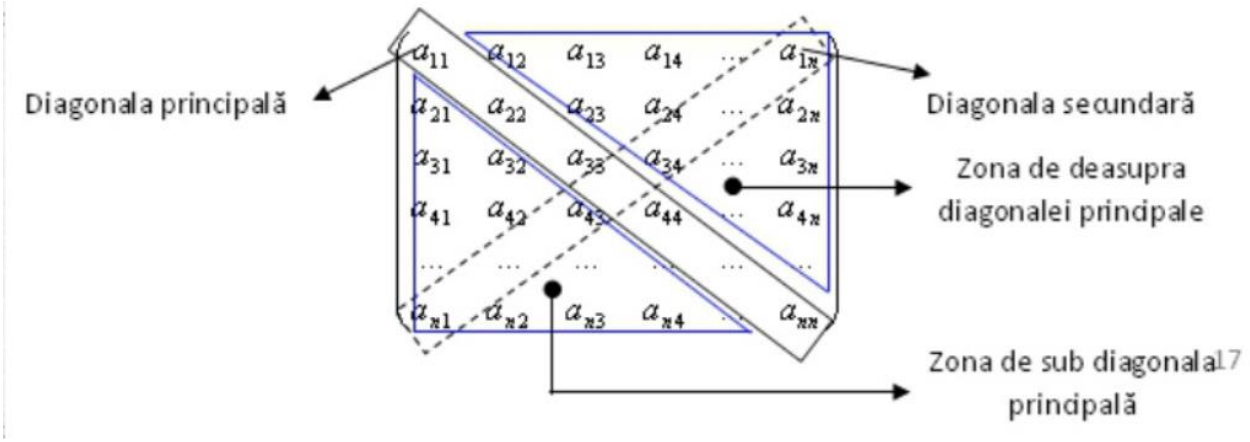

Figura 1. Structura unei matrici pătartice

1. Se cere implementarea unui program Java care să citească o matrice pătratică de dimensiune **n x 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.

```java
package laborator3;
import java.io.*;
public class PrelucrareMatrice {
    public static int [][] citesteMatrice() throws IOException {
        // Metoda pentru citirea unei matrice de la tastatură
        InputStreamReader isr = new InputStreamReader(System.in);
        BufferedReader br = new BufferedReader(isr);
        System.out.println("introduceti nr. de linii");
        String s = br.readLine(); // Citirea numărului de linii (și coloane) ale matricei
        int n = Integer.parseInt(s);
        int [][] matrice = new int[n][n];
        // Declararea matricei de dimensiune n x n
        System.out.println("introduceti elementele");
        for(int i = 0; i < n; i++) {
            for(int j = 0; j < n; j++) {
                // Parcurgerea matricei pentru a citi elementele
                s = br.readLine();
                matrice[i][j] = Integer.parseInt(s);
                // Conversia la numere întregi și atribuirea elementelor în matrice
            }
        }
        return matrice;
    }
```

Fig 2. Declararea și parcurgerea matricei

```java
    public static int [] diagonalaSec(int [][] matrice) {
        // Metoda pentru afișarea diagonalei secundare a matricei
        int [] diags = new int[matrice.length];
        for(int i = 0; i < matrice.length; i++) {
            diags[i] = matrice[i][matrice.length - i - 1];
            System.out.println("elem diag secundara" + diags[i]);
            // Afișarea elementelor de pe diagonala secundară
        }
        return diags;
    }


    public static int sumDiagSecD(int [][] matrice) {
        // Metoda pentru calcularea sumei elementelor de deasupra diagonalei secundare
        int suma = 0;
        for(int i = 0; i < matrice.length - 1; i++)
            for(int j = 0; j < matrice.length - i - 1; j++)
                suma += matrice[i][j];
        return suma;
    }


    public static int sumDiagSecS(int [][] matrice) {
        // Metoda pentru calcularea sumei elementelor de sub diagonala secundară
        int suma = 0;
        for(int i = 1; i < matrice.length; i++)
            for(int j = matrice.length - i; j < matrice.length; j++)
                suma += matrice[i][j];
        return suma;
    }
```

Fig 3. Calculare sume și afișare diagonală secundară

```java
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

```java
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 afișare 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        // Afișarea sumei elementelor de deasupra diagonalei secundare
83        System.out.println("Suma elem de sub diag. sec.: " + sumDiagSecS(a));
84        // Afișarea 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** × **n** de la tastatură, să verifice dacă matricea este **simetrică** față de diagonala principală, și să afișeze un mesaj corespunzător (“Matrice simetrică” / “Matrice nesimetrică”).