Квадратни матрици

ПРЕПОДАВАТЕЛ: ИНЖ. ВЕСЕЛИНА МАРИНОВА

Видове матрици:
Квадратни матрици - представляват двумерни масиви с еднакъв брой редове и колони.

Симетрична матрица - квадратна матрица, чиито елементи са симетрично разположени относно главния диагонал и са равни.
Триъгълна матрица - квадратна матрица, при която всички елементи под или над главния диагонал са нули и тя е съответно горна или долна триъгълна матрица.
Диагонална матрица - квадратна матрица, чиито ненулеви елементи са само в главния диагонал.
Скаларна матрица - диагонална матрица, всички елементи от главния диагонал са равни.
Единична матрица - скаларна матрица с елементи от главния диагонал равни на единица.

Квадратни матрици

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]

Диагонали в квадратната матрица:
1.Главен диагонал

Задача 1 : Намиране на сбора на елементите по главния диагонал. Номера на колоната и номера на реда по диагонала са еднакви, т.е. в проверката ще проверяваме дали No.колона==No.ред.

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]

```
public static void main(String[] args) {
        Scanner kb=new Scanner(System.in);
        int arr[][]=new int[100][100];
        int i,j,n;
        do{     System.out.println("rows = columns :");
                n=kb.nextInt();
        }while(n<=0||n>100);
        for(i=0;i<n;i++)
            for(j=0;j<n;j++){  System.out.println("element arr["+i+"]["+j+"]=");
                               arr[i][j]=kb.nextInt();    }
        int sum=0;
        for(i=0;i<n;i++)
            for(j=0;j<n;j++)
                if( i == j )
                    sum+=arr[i][j];
        System.out.println("sum="+sum);
kb.close();
}
```

Задача 2 : Намиране на сбора на елементите НАД главния диалонал.

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]

Проверката ще е No.ред < No.колона.
Вземете кода от горната задача, като само промените проверката от:
if(i==j) На if(i<j)

//сума от елементите над главния диагонал:

```
int sum1=0;
for(i=0;i<n;i++)
    for(j=0;j<n;j++)

        if(i<j)
            sum1+=arr[i][j];

System.out.print("suma ="+sum1);
```

Задача 3 : Намиране на сбора на елементите Под главния диалонал.
Проверката ще е No.колона < No.реда.
Вземете кода от първата задача и променете проверката от if(i==j) на if(i>j)

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]

//сума от елементите под главния диагонал:

```
int sum2=0;
for(i=0;i<n;i++)
    for(j=0;j<n;j++)

        if(i>j)
            sum2+=arr[i][j];

System.out.print("suma ="+sum2);
```

2.Второстепенен(втори главен) диагонал
Примерни задачи по второстепенния диагонал:
Задача 4 : Намиране на сбора на елементите по второстепения диагонал.

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]

Както виждате в таблицата за втория главен диагонал по-горе сбора на No.колона и No.ред е равен на зададения брой колони и редове, тоест проверката ще е : (No.колона + No.реда) == (броя(n) – 1)
(защото започваме да броим от 0)

//сума от елементите по вторичния диагонал:

```
int sum3=0;
for(i=0;i<n;i++)
    for(j=0;j<n;j++)

        if((i+j)==(n-1))
            sum3+=arr[i][j];

System.out.println("suma ="+sum3);
```

Задача 5 : Намиране на сбора на елементите НАД второстепения диагонал.

Променете проверката от if((i+j)==(n-1)) на if((i+j)<(n-1))

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]

//сума на елементите над вторичния диагонал:

```
int sum4=0;
for(i=0;i<n;i++)
    for(j=0;j<n;j++)

        if((i+j)<(n-1))
            sum4+=arr[i][j];

System.out.println("suma ="+sum4);
```

Задача 6: Намиране на сбора на елементите ПОД второстепения диагонал.

Вземете кода от по-горната задача и променете проверката от if((i+j)==(n-1)) на if((i+j)>(n-1))

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]

//сума от елементите под вторичния диагонал:

```
int sum5=0;
for(i=0;i<n;i++)
    for(j=0;j<n;j++)

        if((i+j)>(n-1))
                sum5+=arr[i][j];

System.out.println("suma ="+sum5);
```

Благодаря за вниманието!
По диагоналите ☺

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]
A[1][0]	A[1][1]	A[1][2]	A[1][3]	A[1][4]
A[2][0]	A[2][1]	A[2][2]	A[2][3]	A[2][4]
A[3][0]	A[3][1]	A[3][2]	A[3][3]	A[3][4]
A[4][0]	A[4][1]	A[4][2]	A[4][3]	A[4][4]