Zadanie nr 3: Rozklad LU macierzy 3x3

Autor: Mateusz Laskowski

Ponizej przedstawiam kod programu w jezyku java.

Wykonuje proste operacje :

-wczytania i wyświetlenia macierzy 3x3

-rozklad LU dla podanej macierzy (liczy macierze L oraz U, wyswietla je)

-wykonuje mnożenie L*U w celu sprawdzenia przez użytkownika czy rozkład został wykonany poprawnie

-niestety nie zdazylem zaimplementować możliwości wyboru elementu wiodącego w macierzy, wiec zablokowałem możliwość rozkładania niektórych macierzy przez swój program(sa wyświetlane stosowne komunikaty dla uzytkownika)

1) Macierz pierwsza:

1	2	3
4	5	6
7	8	9

```
C:\Users\Mateusz\Desktop\LU>java Start
podaj 9 elementow macierzy!(wierszami, moga byc wszystkie naraz )
1 2 3
4 5 6
7 8 9

|1.0||2.0||3.0|

|4.0||5.0||6.0|

|7.0||8.0||9.0|
 -39.0
macierz L:

|1.0||0.0||0.0|

|4.0||1.0||0.0|

|7.0||2.0||1.0|
 macierz U:

|1.0||2.0||3.0|

|0.0||-3.0||-6.0|

|0.0||0.0||0.0|
 macierz do porownaia z A (mnozenie L*U):

|1.0||2.0||3.0|

|4.0||5.0||6.0|

|7.0||8.0||9.0|
```

2)Macierz druga:

0	1	4
1	0	3
9	3	1

```
C:\Users\Mateusz\Desktop\LU>java Start
podaj 9 elementow macierzy!(wierszami, moga byc wszystkie naraz )
0 1 4
1 0 3
9 3 1

|0.0||1.0||4.0|

|1.0||0.0||3.0|

|9.0||3.0||1.0|
 2.0
Brak opcji wybierania elementu wiodacego. Podaj inna macierz!
Pokaz L: Opcja niedostepna
Pokaz U: Opcja niedostepna
L*U: Opcja niedostepna
```

Ponizej zamieszczam kod programu:

```
import java.util.Scanner;


class LU{

private boolean blok=false;

double [][]tab=new double[3][3];//macierz A


double [][] L=new double[3][3];//macierz L

double [][] U=new double[3][3];//macierz U

double [][]sprawdzRozklad=new double[3][3];//sprawdzacz

public void doA(){ //ustawiam rozmiar macierzy A


for(int i=0;i<tab.length; i++){

        for(int j=0;j<tab[i].length;j++){

                tab[i][j]=0;

                                }

                }
```

```
        }


public void Load(){ //wczytuje liczby do macierzy
System.out.println("podaj 9 elementow macierzy!(wierszami, moga byc wszystkie naraz )");
Scanner load=new Scanner(System.in);
double lap;
for(int i=0;i<tab.length; i++){
        for(int j=0;j<tab[i].length;j++){
tab[i][j]= load.nextDouble();
        if(tab[i][j]==0) blok=true;
                }
        }


                }

public void showA(){ //wyswietlam macierz A
for(int i=0;i<tab.length; i++){
System.out.println("");
        for(int j=0;j<tab[i].length;j++){
        System.out.print("|"+tab[i][j]+"|");
                                                                        }
System.out.print("\n ");
                                        }


                                }

public void showL(){ //wyswietlam macierz L
if(blok==false){
System.out.println("macierz L:");
for(int i=0;i<L.length; i++){
```

```
System.out.println("");
        for(int j=0;j<L[i].length;j++){
        System.out.print("|"+L[i][j]+"|");
                                                                                }
System.out.print("\n ");
                                        }
                                }
else System.out.println("Pokaz L: Opcja niedostepna");


}



public void showU(){ //wyswietlam macierz U
if(blok==false){
System.out.println("macierz U:");
for(int i=0;i<U.length; i++){
System.out.println("");
        for(int j=0;j<U[i].length;j++){
        System.out.print("|"+U[i][j]+"|");
                                                                                }
System.out.print("\n ");
                                        }
                                }
else System.out.println("Pokaz U: Opcja niedostepna");
                                }


        public void showS(){ //wyswietlam macierz do porownania
        if(blok==false){
System.out.println("macierz do porownaia z A (mnozenie L*U):");
for(int i=0;i<sprawdzRozklad.length; i++){
```

```
System.out.println("");
        for(int j=0;j<sprawdzRozklad[i].length;j++){
        System.out.print("|"+sprawdzRozklad[i][j]+"|");
                                                                                }
System.out.print("\n ");
                                        }
                                }
        else System.out.println("L*U: Opcja niedostepna");
}
public double determinant(){
System.out.print("Wyznacznik macierzy: ");
double det=0;
double q=0;
double w=0;
q=tab[0][0]*tab[1][1]*tab[2][2]+tab[0][1]*tab[1][2]*tab[2][0]+tab[0][2]*tab[1][0]*tab[2][1];
w=-tab[0][2]*tab[1][2]*tab[2][1]-tab[0][0]*tab[1][2]*tab[2][1]-tab[0][1]*tab[1][0]*tab[2][2];

det=w+q;
return det;

        }

public void LUd(){
if(tab[0][0]==0){ System.out.println("Brak opcji wybierania elementu wiodacego. Podaj inna
macierz!"); return; }
else{
U[0][0]=tab[0][0];
L[1][0]=tab[1][0]/U[0][0];
L[2][0]=tab[2][0]/U[0][0];
```

```
U[0][1]=tab[0][1];
U[1][1]=tab[1][1]-L[1][0]*U[0][1];
L[2][1]=(tab[2][1]-L[2][0]*U[0][1])/U[1][1];

if(L[2][1]<1&&L[2][1]!=0){
L[2][1]=Math.round(L[2][1]*10); //zaokraglam w GORE!
L[2][1]/=10; }


U[0][2]=tab[0][2];
U[1][2]=tab[1][2]-L[1][0]*U[0][2];
U[2][2]=tab[2][2]-L[2][0]*U[0][2]-L[2][1]*U[1][2];



L[0][0]=1;
L[1][1]=1;
L[2][2]=1;
 }


        }

public void multiply(){//mnozenie macierzy!

for(int i=0;i<sprawdzRozklad.length; i++){

        for(int j=0;j<sprawdzRozklad[i].length;j++){
        double sum=0.0;
        for(int k=0;k<L.length;k++){
        sum+=L[i][k]*U[k][j];
```

```
                        }
        sprawdzRozklad[i][j]=sum;
                }
        }
        return;
}

        }
```

```
class Start{
public static void main(String[] args){

LU A=new LU();

A.doA();
A.Load();
A.showA();
System.out.println(A.determinant());
A.LUd();

A.showL();

A.showU();

A.multiply();
A.showS();

        }
}
```