Zadanie 1. Mateusz Laskowski

Zadanie polegalo na rozwiazaniu układu rownan i znalezieniu wartości własnych macierzy dla c=0 oraz c=1.

Przedstawiam rozwiązanie układu rownan za pomocą programu napisanego w jezyku java, oraz znalezione wartosci wlasne macierzy w czym pomogl program Mathematica.

Kod programu w jezyku java:

```
public class Matrix {
	private int nrows;
	private int ncols;
	private double[][] data;
	public Matrix(double[][] dat) {
		this.data = dat;
		this.nrows = dat.length;
		this.ncols = dat[0].length;
	}

	public Matrix(int nrow, int ncol) {
		this.nrows = nrow;
		this.ncols = ncol;
		data = new double[nrow][ncol];
	}

	public int getNrows() {
		return nrows;
	}

	public void setNrows(int nrows) {
		this.nrows = nrows;
	}

	public int getNcols() {
		return ncols;
	}

	public void setNcols(int ncols) {
    this.ncols = ncols;
}

public double[][] getValues() {
    return data;
}

public void setValues(double[][] values) {
    this.data = values;
}

public void setValueAt(int row, int col, double value) {
    data[row][col] = value;
}

public double getValueAt(int row, int col) {
    return data[row][col];
}

public boolean isSquare() {
    return nrows == ncols;
}

public int size() {
    if (isSquare())
        return nrows;
    return -1;
}

public Matrix multiplyByConstant(double constant) {
    Matrix mat = new Matrix(nrows, ncols);
    for (int i = 0; i < nrows; i++) {
        for (int j = 0; j < ncols; j++) {
            mat.setValueAt(i, j, data[i][j] * constant);
            }
        }
        return mat;
    }
    public Matrix insertColumnWithValue1() {
        Matrix X_ = new Matrix(this.getNrows(), this.getNcols()+1);
        for (int i=0;i<X_.getNrows();i++) {
            for (int j=0;j<X_.getNcols();j++) {
                if (j==0)
                    X_.setValueAt(i, j, 1.0);
                else
                    X_.setValueAt(i, j, this.getValueAt(i, j-1));

            }
        }
        return X_;
    }

        public void show(){
    for(int i=0;i<data.length;i++){
    System.out.println("");
    for(int j=0;j<data[i].length;j++){
    System.out.print("|"+data[i][j]+"|");}
        }
            System.out.print("\n");
            }
}

class MatrixMathematics {

    public MatrixMathematics(){}

    public static Matrix transpose(Matrix matrix) {
        Matrix transposedMatrix = new Matrix(matrix.getNcols(), matrix.getNrows());
```

```
		for (int i=0;i<matrix.getNrows();i++) {
			for (int j=0;j<matrix.getNcols();j++) {
				transposedMatrix.setValueAt(j, i, matrix.getValueAt(i, j));
			}
		}
		return transposedMatrix;
	}



	public static Matrix inverse(Matrix matrix)  {
		return (transpose(cofactor(matrix)).multiplyByConstant(1.0/determinant(matrix)));
	}



	public static double determinant(Matrix matrix)  {

		if (matrix.size()==2) {
			return (matrix.getValueAt(0, 0) * matrix.getValueAt(1, 1)) - ( matrix.getValueAt(0, 1) * 
matrix.getValueAt(1, 0));
		}
		double sum = 0.0;
		for (int i=0; i<matrix.getNcols(); i++) {
			sum += changeSign(i) * matrix.getValueAt(0, i) * determinant(createSubMatrix(matrix, 0, i));
		}
		return sum;
	}

	private static int changeSign(int i) {
		if (i%2==0)
			return 1;
		return -1;
	}

	public static Matrix createSubMatrix(Matrix matrix, int excluding_row, int excluding_col) {
        Matrix mat = new Matrix(matrix.getNrows()-1, matrix.getNcols()-1);
        int r = -1;
        for (int i=0;i<matrix.getNrows();i++) {
            if (i==excluding_row)
                continue;
                r++;
                int c = -1;
            for (int j=0;j<matrix.getNcols();j++) {
                if (j==excluding_col)
                    continue;
                mat.setValueAt(r, ++c, matrix.getValueAt(i, j));
            }
        }
        return mat;
    }



    public static Matrix cofactor(Matrix matrix)
    {
        Matrix mat = new Matrix(matrix.getNrows(), matrix.getNcols());
        for (int i=0;i<matrix.getNrows();i++) {
            for (int j=0; j<matrix.getNcols();j++) {
                mat.setValueAt(i, j, changeSign(i) * changeSign(j) *
determinant(createSubMatrix(matrix, i, j)));
            }
        }

        return mat;
    }



    public static Matrix add(Matrix matrix1, Matrix matrix2)  {

        Matrix sumMatrix = new Matrix(matrix1.getNrows(), matrix1.getNcols());
```

```
                for (int i=0; i<matrix1.getNrows();i++) {
                        for (int j=0;j<matrix1.getNcols();j++)
                                sumMatrix.setValueAt(i, j, matrix1.getValueAt(i, j) + matrix2.getValueAt(i,j));


                }
                return sumMatrix;
        }


        public static Matrix subtract(Matrix matrix1, Matrix matrix2)  {
                return add(matrix1,matrix2.multiplyByConstant(-1));
        }


        public static Matrix multiply(Matrix matrix1, Matrix matrix2)  {
                Matrix multipliedMatrix = new Matrix(matrix1.getNrows(), matrix2.getNcols());


                for (int i=0;i<multipliedMatrix.getNrows();i++) {
                        for (int j=0;j<multipliedMatrix.getNcols();j++) {
                                double sum = 0.0;
                                for (int k=0;k<matrix1.getNcols();k++) {
                                        sum += matrix1.getValueAt(i, k) * matrix2.getValueAt(k, j);
                                }
                                multipliedMatrix.setValueAt(i, j, sum);
                        }
                }
                return multipliedMatrix;
        }
}


class Start         {
public static void main (String[] args){
int x=7;
int y=7;
double [][]B={{1},{2},{3},{4},{5},{6},{7}};
Matrix o = new Matrix(x,y);
```

```
double [][]A={{4,1,0,0,0,0,1},
                         {1,4,1,0,0,0,0},
                         {0,1,4,1,0,0,0},
                         {0,0,1,4,1,0,0},
                         {0,0,0,1,4,1,0},
                         {0,0,0,0,1,4,1},
                         {1,0,0,0,0,1,4}};

double [][]C={{4,1,0,0,0,0,0},
                         {1,4,1,0,0,0,0},
                         {0,1,4,1,0,0,0},
                         {0,0,1,4,1,0,0},
                         {0,0,0,1,4,1,0},
                         {0,0,0,0,1,4,1},
                         {0,0,0,0,0,1,4}};
System.out.println("-----------------macierz A z c=1 -----------------------------");
Matrix tab=new Matrix(A);

tab.show();
System.out.println("-----------------macierz odwrotna----------------------------");
MatrixMathematics dzialaj=new MatrixMathematics();
System.out.println(dzialaj.determinant(tab));

o=dzialaj.inverse(tab);
o.show();
System.out.println("----------------- B-----------------------------");
Matrix tab2=new Matrix(B);
                tab2.show();

System.out.println("-----------------mnoze odwrotna do A przez B----------------------------");
Matrix wynik = new Matrix(x,y);
wynik=dzialaj.multiply(o,tab2);
wynik.show();
```

```
System.out.println("-----------------macierz A z c=1 ----------------------------");
Matrix macierz=new Matrix(C);
macierz.show();
System.out.println("-----------------macierz odwrotna---------------------------");
Matrix g = new Matrix(x,y);
MatrixMathematics odwroc = new MatrixMathematics();
System.out.println(odwroc.determinant(macierz));
g=odwroc.inverse(macierz);
g.show();
System.out.println("-----------------mnoze odwrotna do A przez B---------------------------");
Matrix wynik_2=new Matrix(x,y);
wynik_2= odwroc.multiply(g, tab2);
wynik_2.show();


                              }
                    }
```

Wyniki:

Dla c=1:

x_1= -0.26016260162601623

x_2 = 0.44715447154471544

x_3 = 0.4715447154471545

x_4 = 0.6666666666666667

x_5 = 0.861788617881789

x_6 = 0.886178861788618

x_7 = 1.5934959349593498

dla c=0

x_1 = 0.1667893961708395

x_2 = 0.3328424153166421

x_3 = 0.5018409425625918

x_4 = 0.6597938144329897

x_5 = 0.8589837997054492

x_6 = 0.9042709867452134

x_7 = 1.5239322533136965

Wartosci własne macierzy:

Przypadek 1)

c=1

```
m:={{4,1,0,0,0,0,1},{1,4,1,0,0,0,0},{0,1,4,1,0,0,0},{0,0,1,4,1,0,0},{0,0,0,1,4,1,0},{0,0,0,0,1,4,1},{1,0,0,0,0,1,4}}
Eigenvalues[N[m]]
```

Wartosci wlasne:

```
{6.,5.24698,5.24698,3.55496,3.55496,2.19806,2.19806}
```

Przypadek 2)

c=0

```
m:={{4,1,0,0,0,0,0},{1,4,1,0,0,0,0},{0,1,4,1,0,0,0},{0,0,1,4,1,0,0},{0,0,0,1,4,1,0},{0,0,0,0,1,4,1},{0,0,0,0,0,1,4}}
Eigenvalues[N[m]]
```

Wartosci własne:

```
{6.,5.24698,5.24698,3.55496,3.55496,2.19806,2.19806}
```