Tugas Besar 1 IF 2123 Aljabar Linier dan Geometri

13520016 - Gagas Praharsa Bahar
13520044 - Adiyansa Prasetya Wicaksana
13520081 - Andhika Arta Aryanto

BAB 1
Deskripsi Masalah

Sistem persamaan linier (SPL) adalah sekumpulan persamaan linear yang dikorelasikan untuk membentuk suatu sistem. Kata “sistem” di sini menunjukkan bahwa persamaan - persamaan tersebut perlu dipertimbangkan bersamaan dan tidak dapat berdiri sendiri.

Terdapat tiga bentuk umum dari sistem persamaan linier, untuk SPL dengan m dan n yang tidak diketahui, dapat ditulis seperti ini

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m, \end{aligned}$$

Bisa juga ditulis dalam bentuk persamaan vektor :

$$x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Dan yang terakhir dalam bentuk persamaan matriks Ax = B, dengan A merupakan matriks m x n, **x** adalah vektor kolom dengan entri n dan **B** vektor kolom dengan entri m

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Matriks diatas bisa diubah menjadi matrik augmented dengan bentuk :

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_n \end{bmatrix}$$

SPL dapat diselesaikan dengan beberapa metode, yakni metode eliminasi Gauss, metode eliminasi Gauss-Jordan, metode matriks balikan ($x = A^{-1}b$), dan kaidah Cramer (khusus SPL dengan n peubah dan n persamaan). Ada tiga kemungkinan solusi yang dapat terjadi pada sebuah SPL :

1. Mempunyai solusi unik/tunggal,
2. Mempunyai banyak solusi/tidak berhingga, atau
3. Tidak ada solusi sama sekali.

Pada Tugas Besar 1 ini, kami diminta untuk membuat satu atau lebih library aljabar linier dalam Bahasa Java. Library tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, dan kaidah Cramer. Library tersebut akan digunakan untuk beberapa hal, diantaranya : menyelesaikan berbagai persoalan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi.

BAB 2
TEORI SINGKAT

2.1 **Operasi Baris Elementer**

Operasi Baris Elementer (OBE) adalah suatu operasi yang diterapkan pada baris suatu matriks, biasanya digunakan untuk menemukan invers suatu matriks atau menyelesaikan SPL. Ada tiga OBE yang bisa diterapkan pada matriks *augmented , yaitu:*

1. Kalikan sebuah baris dengan konstanta tidak nol.
2. Pertukaran 2 buah baris
3. Tambahkan 1 baris dengan kelipatan baris lainnya

Solusi SPL bisa diperoleh setelah menerapkan OBE pada baris sampai terbentuk matriks eselon baris atau matriks eselon baris tereduksi, setelah itu dilakukan eliminasi Gauss atau eliminasi Gauss - Jordan tergantung dengan matriks eselon akhir yang kita dapatkan.

2.2 **Metode Eliminasi Gauss**

Eliminasi Gauss merupakan salah satu algoritma yang dapat digunakan untuk menyelesaikan sistem persamaan linier. Metode ini dinamai dari matematikawan Carl Friedrich Gauss (1777–1855). Metode Gauss dilakukan pada SPL yang berbentuk matrik *augmented*, menggunakan metode OBE sampai terbentuk matrik eselon baris (matriks yang memiliki 1 utama pada setiap baris kecuali baris yang berisi 0 semua, dan setiap kolom semakin menjorok ke dalam), berikut ilustrasi matriks setelah dilakukan OBE :

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_n \end{bmatrix} \sim \text{OBE} \sim \begin{bmatrix} 1 & * & * & \dots & * & * \\ 0 & 1 & * & \dots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

Setelah sudah dalam bentuk matriks eselon baris, persamaan pada matriks eselon baris bisa dipecahkan dengan teknik penyulihan mundur (*backward substitution)*.

2.3 **Metode Eliminasi Gauss Jordan**

Metode ini merupakan pengembangan dari metode eliminasi Gauss. Perbedaan dengan metode Gauss adalah pada metode ini, OBE diterapkan sampai didapat matriks eselon baris tereduksi.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} & b_m \end{bmatrix} \sim \text{OBE} \sim \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & * \\ 0 & 1 & 0 & \dots & 0 & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

Dapat dilihat bahwa pada matriks eselon baris tereduksi, elemen di atas dan di bawah 1 utama harus bernilai 0.

Setelah matriks eselon terbentuk, tidak lagi diperlukan substitusi mundur karena nilai variabel langsung didapatkan dari matrik *augmented* akhir. Metode Gauss - Jordan terdiri dari dua fase, yaitu fase maju yang akan menghasilkan nilai 0 di bawah 1 utama

$$\begin{bmatrix} 2 & 3 & -1 & 5 \\ 4 & 4 & -3 & 3 \\ -2 & 3 & -1 & 1 \end{bmatrix} \sim \overset{\text{OBE}}{\dots} \sim \begin{bmatrix} 1 & 3/2 & -1/2 & 5/2 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

Lalu fase mundur untuk menghasilkan nilai 0 di atas satu utama

$$\begin{bmatrix} 1 & 3/2 & -1/2 & 5/2 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{bmatrix} \overset{\text{R1} - (3/2)\text{R2}}{\sim} \begin{bmatrix} 1 & 0 & -5/4 & -11/4 \\ 0 & 1 & 1/2 & 7/2 \\ 0 & 0 & 1 & 3 \end{bmatrix} \overset{\substack{\text{R1} + (5/4)\text{R3} \\ \sim \\ \text{R2} - (1/2)\text{R3}}}{} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

Matriks eselon baris tereduksi

2.4 **Determinan**

Determinan adalah nilai yang dapat dihitung dari unsur suatu matriks persegi. Determinan matriks A dapat dilambangkan dengan det(A). Pada tubes ini, terdapat 2 metode yang digunakan untuk menghitung determinan. Metode yang pertama adalah reduksi baris dengan melakukan OBE pada suatu matriks sampai diperoleh matriks segitiga (segitiga bawah atau atas).

Untuk contoh misalnya dilakukan OBE pada matriks A sampai diperoleh matriks segitiga bawah

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \overset{\text{OBE}}{\sim} \begin{bmatrix} a'_{11} & a'_{12} & \dots & a'_{1n} \\ 0 & a'_{22} & \dots & a'_{2n} \\ \vdots & \vdots & \dots & a'_{3n} \\ 0 & 0 & 0 & a'_{nn} \end{bmatrix}$$

Untuk det (A) didapatkan dari perkalian semua elemen pada diagonal matriks segitiga yang terbentuk. Rumusnya :

$$\det(A) = (-1)^p \, a'_{11} a'_{22} \ldots a'_{nn}$$

Dengan p merupakan banyaknya pertukaran baris yang dilakukan pada OBE

Metode yang kedua adalah menghitung determinan dengan ekspansi kofaktor. Misalnya ada suatu matriks yang berukuran n x n , dapat kita berikan definisi bahwa M_{ij} merupakan minor untuk entri a_i (determinan submatrix yang elemennya tidak berapa pada baris i dan kolom j), lalu kofaktor entri a_i dapat ditemukan dengan rumus $C_i = (-1)^{(i+j)} M_i$. Terlihat bahwa nilai kofaktor akan berbeda tangga bergantung dengan nilai i dan j, untuk mengingat hal ini bisa diperhatikan pola berikut :

$$\begin{bmatrix} + & - & + & - & \ldots \\ - & + & - & + & \ldots \\ + & - & + & - & \ldots \\ - & + & - & + & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \end{bmatrix}$$

Dengan menggunakan kofaktor, maka kita dapat mencari determinan dengan salah satu persamaan berikut :

$$\det(A) = a_{11}C_{11} + a_{12}C_{12} + \ldots + a_{1n}C_{1n}$$
$$\det(A) = a_{21}C_{21} + a_{22}C_{22} + \ldots + a_{2n}C_{2n}$$
$$\vdots$$
$$\det(A) = a_{n1}C_{n1} + a_{n2}C_{n2} + \ldots + a_{nn}C_{nn}$$

Secara baris

$$\det(A) = a_{11}C_{11} + a_{21}C_{21} + \ldots + a_{n1}C_{n1}$$
$$\det(A) = a_{12}C_{12} + a_{22}C_{22} + \ldots + a_{n2}C_{n2}$$
$$\vdots$$
$$\det(A) = a_{1n}C_{1n} + a_{2n}C_{2n} + \ldots + a_{nn}C_{nn}$$

Secara kolom

2.5 **Matriks Balikan**

Misal matriks A merupakan matriks persegi dengan ukuran n x n, maka balikan(inverse) dari matriks A merupakan A^{-1} sedemikian sehingga $AA^{-1} = A^{-1}A = I$. , I merupakan matriks identitas berukuran n x n. Salah satu cara yang bisa digunakan untuk mencari matriks balikan ada Metode Eliminasi Gauss-Jordan

$$[A|I] \overset{\text{G-J}}{\sim} [I|A^{-1}]$$

Singkatnya, identitas di $\begin{bmatrix} 12 & 4 & 12 \\ 6 & 2 & -10 \\ -16 & 16 & 16 \end{bmatrix}$ dibentuk sebuah matriks dan ditempatkan matriks sebelah matriks tersebut, lalu dilakukan Eliminasi Gauss Jordan pada kedua matriks sampai matriks A berubah menjadi matriks identitas. Bentuk akhir dari matriks identitas tersebut merupakan **matriks balikan** dari A.

$$\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & -40 & 16 & 9 \\ 0 & 1 & 0 & 13 & -5 & -3 \\ 0 & 0 & 1 & 5 & -2 & -1 \end{array}\right) = (I|A^{-1})$$

Jadi, balikan matriks A adalah

$$A^{-1} = \begin{bmatrix} -40 & 16 & 9 \\ 13 & -5 & -3 \\ 5 & -2 & -1 \end{bmatrix}$$

Apabila saat dilakukan OBE ditemukan baris yang seluruh elemennya bernilai 0 , dapat disimpulkan bahwa A tidak memiliki balikan.

Cara lain yang bisa digunakan untuk mencari matriks balikan adalah menggunakan adjoin dengan rumus :

$$A^{-1} = \frac{1}{\det(A)} \text{adj}(A)$$

2.6 **Matriks Kofaktor**

Untuk suatu matriks A berukuran n x n, C_i merupakan kofaktor dari entri a_i . Jadi, matriks kofaktor merupakan matriks yang terdiri dari kofaktor matriks itu sendiri. Susunan dari matriks kofaktor mengikuti susunan entri a_i , matriks kofaktor dari A :

$$\begin{bmatrix} C_{11} & C_{12} & \dots & C_{1n} \\ C_{21} & C_{22} & \dots & C_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ C_{n1} & C_{n2} & \dots & C_{nn} \end{bmatrix}$$

2.7 **Matriks Adjoin**

Misal A merupakan sebuah matriks n x n, adjoin dari A merupakan suatu matriks yang terbentuk dari pertukaran tempat baris dan kolom dari matriks kofaktor A. Dengan kata lain adjoin A merupakan **transpose** dari matriks kofaktor A.

Untuk matriks kofaktor :

adj (A) =

$$\begin{bmatrix} 12 & 6 & -16 \\ 4 & 2 & 16 \\ 12 & -10 & 16 \end{bmatrix}$$

2.8 **Kaidah Cramer**

Kaidah Cramer merupakan suatu rumus yang dapat digunakan untuk menyelesaikan sembarang SPL Ax = b dengan n peubah dan n persamaan dan det(A) ≠0. Untuk matriks A berbentuk

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

m = n

Dapat dirumuskan bahwa X_n = ,

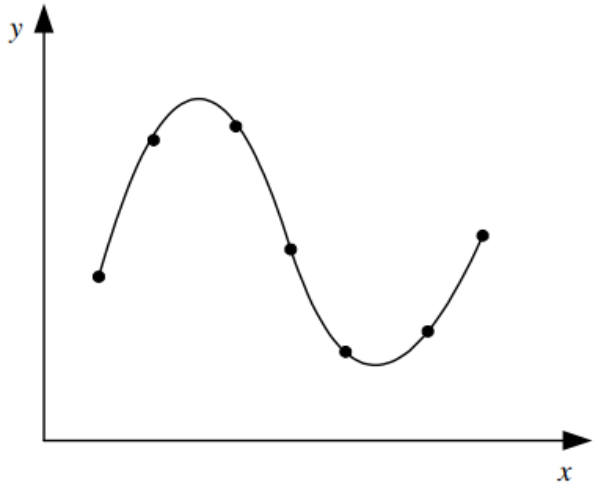
dalam hal ini A_n merupakan matriks baru yang diperoleh dari mengganti entri pada kolom ke-n dari A dengan matriks

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

$$\frac{\det(A_n)}{\det(A)}$$

2.9 **Interpolasi polinom**

Bila ada suatu data yang diketahui memiliki tingkat ketelitian sangat tinggi, kita dapat membuat kurva cocokan (fungsi yang mencocokan titik" data di dalam tabel tabel) dibuat melewati setiap titik. Hal ini disebut juga bahwa kita **menginterpolasi** titik - titik data dengan suatu fungsi. Apabila fungsi yang dicocokan berbentuk polinom, maka disebut dengan interpolasi polinom. Berikut gambar interpolasi :



Contoh persoalan yang diselesaikan dengan interpolasi ini adalah, diberikan n+1 buah titik berbeda $(x_o,y_o),(x_1,y_1),...(x_n,y_n)$. Tentukan suatu polinom $p_n(x)$ yang menginterpolasi semua titik tersebut sedemikian rupa sehingga $y_i - p_n(x_i)$ untuk i = 0,1,2…,n. Apabila sebuah fungsi interpolasi sudah ditemukan, kita dapat menggunakan fungsi tersebut untuk menghitung perkiraan nilai y di x = a, dengan $y = p_n(a)$. Beberapa metode interpolasi yang bisa kita gunakan adalah polinom lanjar, polinom kuadratik, polinom kubik, dan polinom dari derajat yang lebih tinggi bergantung dari data yang tersedia.

2.10 **Regresi Linear Berganda**

Mirip dengan interpolasi polinom, regresi linear berganda merupakan salah satu metode untuk memprediksi nilai dari beberapa data yang ada. Regresi linear berganda adalah model regresi linear yang melibatkan lebih dari satu variabel bebas atau prediktor. Jadi, regresi linear berganda mirip dengan regresi linear sederhana yang biasa kita gunakan, hanya dengan jumlah variabel yang banyak. Rumus umum yang biasa digunakan untuk regresi linear berganda, yaitu :

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \epsilon_i$$

dengan keterangan ,

y_i = variabel terikat yang akan diprediksi nilainya

$\boldsymbol{\beta}$ = koefisien regresi (nilai peningkatan atau penurunan)

x_i = variabel dependen

$\boldsymbol{\beta}_0$ merupakan nilai y saat i = 0 atau disebut juga *intercept* dan ε adalah *error term* atau nilai penganggu.

BAB 3
IMPLEMENTASI PUSTAKA DAN PROGRAM DALAM JAVA

Pada program Java kami, terdapat 12 class yang dibagi ke 4 folder utama source code, yaitu :

1. **Folder Matrix**

   Folder matrix berisi 2 class, Matrix.java dan Matrixinput.java

   1.1 Matrix.java

   Class ini berisi deskripsi dan representasi Matriks yang akan dibuat, disini kami menggunakan array dua dimensi yang berisi data bertipe double. Untuk isi dari class sebagai berikut

   - **Attributes**

Attribute	Description
private int Row	Berisi data integer yang menampung jumlah baris dari matriks
private int Col	Berisi data integer yang menampung jumlah kolom dari matriks
Private double[][] Contents	Menampung data angka double, merepresentasikan isi dari matriks

   - **Constructors**

     Ada 3 konstruktor matriks yang kami buat :

Constructor (Parameter)	Description
public Matrix(int row,int col)	Konstruktor akan membuat suatu matriks dengan baris sebanyak row dan kolom sebanyak col dengan isi angka ‘0’
public Matrix(int n)	Membentuk matriks identitas dengan jumlah baris dan kolom sebanyak n

Public Matrix(Matrix m)	Menerima parameter berupa matriks dan akan membuat copy dari matriks tersebut. Pada dasarnya seperti fungsi copyMatrix

- **Methods**

  Method berisi fungsi - fungsi dan prosedur yang bisa dilakukan pada object matriks yang telah dibuat untuk mendapat info tentang matriks.

Methods(Parameter)	Description
public double getElmt(int i, int j)	Mengembalikan isi dari matriks pada indeks i,j
public int getRowLength()	Mengembalikan banyak baris dari matriks
public int getColLength()	Mengembalikan banyak kolom dari matriks
public void setElmt(double x , int row, int col)	Mengubah nilai matriks pada indeks row,col dengan nilai x
public boolean isSquare()	Mengembalikan true jika dan hanya jika matriks memiliki jumlah baris dan kolom yang sama
public Matrix cofactor (int p , int q)	Mengembalikan matriks kofaktor dari entri matriks a_{pq}
public double determinantCofactor()	Mengembalikan nilai determinan dari matriks menggunakan metode kofaktor
public double determinantbyOBE	Mengembalikan nilai determinan dari matriks menggunakan metode OBE
public Matrix transpose()	Mengembalikan matriks hasil transpose
public Matrix countRowZero()	Mengembalikan jumlah row yang setiap kolomnya bernilai 0
public boolean isRowZero(int i)	Mengembalikan nilai true jika dan hanya jika elemen baris ke - i dari matriks bernilai 0 semua.

1.2 Matrixinput.java

Class ini digunakan untuk mengatur cara program menerima input dari user. Ada 2 cara yang bisa dilakukan, yaitu input menggunakan console dan input dari file.

- **Attributes**

  Tidak ada attribute pada class ini.

- **Constructors**

  Tidak ada konstruktor pada class ini.

- **Methods**

  Method berisi fungsi - fungsi yang akan digunakan saat akan menerima input dari user.

Methods(Parameter)	Description
public static Matrix input()	Method ini akan menampilkan menu input dan pilihannya (keyboard & file)
public static Matrix consoleInput()	Method mengeluarkan prompt di console dan akan menerima input berupa matriks augmented
public static Matrix interpolateInput()	Method yang digunakan saat user ingin melakukan input interpolasi. Akan memberikan pilihan antara input keyboard atau file.
Public static Matrix interpolateInt()	Method dipanggil untuk membantu input interpolasi, seperti $\{x_0,y_o\},\{x_1,y_1\},\{x_2,y_2\}...,\{x_n,y_n\}$
public static Matrix SPLInput()	Method digunakan untuk input SPL dalam bentuk Ax = b, dan mengubah input tersebut menjadi dalam bentuk matriks augmented
public static Matrix fileInput()	Method digunakan untuk menerima input dari file

2. Folder Utility

Folder Utility berisi Menu.java dan Output.java

2.1 UI.java

Class berisi method - method yang akan digunakan untuk inisialisasi menu pada program utama. Menggunakan method dari class Matrixinput.java untuk membantu penerimaan input matriks dari user. Menu menggunakan GUI dengan Swing.

- **Attributes**

  Tidak ada attribute pada class ini.

- **Constructors**

  Tidak ada konstruktor pada class ini.

- **Methods**

  Method berisi fungsi - fungsi dan prosedur yang bisa dilakukan pada object matriks yang telah dibuat.

Methods (Parameter)	Description
public static void mainMenu()	Method akan mengeluarkan prompt pilihan menu utama untuk user dan menerima input dari user.
public static void SPLMenu()	Method dipanggil saat user memilih menu SPL, akan mengeluarkan pilihan submenu untuk metode yang ingin digunakan untuk menyelesaikan SPL.
public static void determinantMenu()	Method dipanggil saat user memilih menu determinan, mengeluarkan 2 pilihan metode mencari determinan, yaitu dengan OBE atau dengan ekspansi kofaktor
public static void inverseMenu()	Method dipanggil saat user memilih menu invers, akan meminta input user untuk metode apa yang akan digunakan untuk mencari matriks balikan. Metode yang tersedia yaitu dengan eliminasi gauss dan dengan matriks adjoin
public static void interpolateMenu()	Method dipanggil saat user memilih menu interpolasi, method akan mengeluarkan hasil polinom dan hasil taksiran y dari x yang diinput user.
public static void regressionMenu()	Method dipanggil saat user memilih menu regresi, dan akan mengeluarkan output berupa hasil dari regresi.
public static void prompt()	Method dipanggil untuk meminta input

	user terkait apakah user ingin menggunakan kalkulator lagi atau tidak.
public static void exit()	Method untuk keluar dari program.

2.2 Output.java

Berisi class untuk mengoutput hasil dari algoritma ke dalam file txt dan method-method yang mendukung.

- **Attributes**

Attribute	Description
private double det	Berisi double yang menampung data determinan yang akan di output
private Matrix mat	Berisi matrix yang menampung data matrix yang akan di output
private String out	Berisi string yang akan di output
private String function	Berisi fungsi yang terbentuk dari Interpolasi dan regresi yang akan di output
private double interpolateRes	Berisi double yang berisi hasil interpolasi
private double interpolateGuess	Berisi double yang berisi masukan terhadap fungsi interpolasi
private double regressionRes	Berisi double yang berisi hasil regresi
private double[] regressionGuess	Berisi kumpulan double yang merupakan masukan terhadap hasil regresi
private String dir	Berisi string yang menunjukkan directory untuk disimpan file
private String path	Berisi string yang akan diisi dengan path dari file
private SimpleDateFormat formatter	Berisi formatter tanggal

- **Constructors**

Ada 3 konstruktor matriks yang kami buat :

Constructor (Parameter)	Description
public Output(double n)	Membentuk object Output dengan atribut determinan
public Output(Matrix m)	Membentuk object output dengan atribut matriks
public Output(String s)	Membentuk object Output dengan atribut out, string yang akan disimpan dalam file
public Output(String s, double x, double guess)	Membentuk object Output dengan atribut sesuai yang dibutuhkan oleh keluaran interpolasi
public Output(String s, double x, double[] guess)	Membentuk object Output dengan atribut sesuai yang dibutuhkan oleh keluaran regresi.

- **Methods**

  Method berisi fungsi - fungsi dan prosedur yang bisa dilakukan pada object matriks yang telah dibuat untuk mendapat info tentang matriks.

Methods(Parameter)	Description
public void createFile()	Membuat file baru dengan nama “/Tanggal/ /Jam/”.txt
public void detToFile()	Mengolah bentuk determinan untuk kemudian dijadikan output ke dalam file
public void inverseToFile()	Mengolah bentuk inverse untuk kemudian dijadikan output ke dalam file
public void interpolateToFile()	Mengolah bentuk keluaran interpolasi untuk kemudian dijadikan output ke dalam file
public void SPLtoFile()	Mengolah bentuk keluaran SPL untuk kemudian dijadikan output ke dalam file
public void regressionToFile()	Mengolah bentuk keluaran regresi untuk kemudian dijadikan output ke dalam file

public String matrixToString()	Mengolah bentuk matrix menjadi yang ekivalen dalam bentuk string.
public void pathMaker()	Membentuk path directory untuk output.

3. Folder Algoritma

Berisi implementasi dari algoritma - algoritma yang akan dilakukan pada matriks, terdiri dari 7 class, sebagai berikut :

3.1 Operation.java

Class berisi operasi eksternal yang bisa dilakukan pada matriks.

- **Attributes**

  Tidak ada attribute pada class ini.

- **Constructors**

  Tidak ada konstruktor pada class ini.

- **Methods**

  Method berisi operasi - operasi yang bisa dilakukan pada object matriks yang telah dibuat.

Methods (Parameter)	Description
public static Matrix OBE(Matriks M)	Method melakukan OBE pada matriks dan mengembalikan matriks eselon baris dari matriks tersebut
public static Matrix OBETereduksi(Matriks M)	Method melakukan OBE sampai terbentuk matriks eselon baris tereduksi dan mengembalikan matriks tersebut
public static Matrix extendMatrix(Matriks M1, Matriks M2)	Method akan mengembalikan gabungan matriks M1 dan M2, penggabungan dalam bentuk M2 menjadi kolom baru dari M1.
public static void pMultiplyConst(Matrix M,	Method mengalikan variabel k terhadap

double k)	seluruh elemen matriks.
public static Matrix multiplyMatrix(Matrix m1, Matrix m2)	Method melakukan perkalian matriks M1 x M2
public static Matrix cutRight(Matrix m)	Method digunakan untuk mengambil nilai b untuk SPL Ax = b yang sudah diubah menjadi matriks augmented

3.2 SPL.java

Class berisi attribute, method yang berhubungan dengan penyelesaian SPL.

- **Attributes**

Attribute	Description
private boolean noSolutions	Atribut yang bernilai true jika dan hanya jika SPL tidak memiliki solusi
private boolean manySolutions	Atribut yang bernilai true jika dan hanya jika SPL memiliki banyak solusi
private boolean singleSolution	Atribut yang bernilai true jika dan hanya jika SPL memiliki solusi singular
private String[] solution	Berisi solusi dari SPL dalam bentuk array of string
private Matrix m	Atribut matrix dari objek SPL

- **Constructors**

Constructor (Parameter)	Description
public SPL(Matrix M)	Konstruktor akan membuat sebuah objek spl yang berisikan matrix M

- **Methods**

Method berisi operasi - operasi yang bisa dilakukan pada object matriks yang telah dibuat.

Methods (Parameter)	Description
public String[] solve (int choice)	Method akan meminta pilihan user untuk metode penyelesaian SPL yang ingin digunakan, lalu mengembalikan solusi berupa array of string
Public String consoleOut()	Method digunakan saat ingin menampilkan solusi SPL pada layar
public void toNoSolutions()	Method yang menandai SPL bahwa dia tidak memiliki solusi
public void toManySolution()	Method yang menandai SPL dia memiliki solusi banyak(parametrik)
public void toSingleSolution()	Method menandai SPL yang memiliki solusi singular
public boolean manySolutions()	Method memanggil boolean yang mengecek apakah SPL memiliki solusi banyak atau tidak
public boolean noSolutions()	Method memanggil boolean yang mengecek apakah SPL memiliki solusi banyak atau tidak
public boolean singleSolutions()	Method memanggil boolean yang mengecek apakah SPL memiliki solusi banyak atau tidak
public String[] arrayDoubleToString(double[] d)	Method yang mengubah tipe array solusi dari double menjadi string.
Public String[] parametric (Matrix M)	Method yang digunakan saat SPL memiliki solusi banyak/parametrik
public String[] gaussEquation()	Method melakukan backward substitution pada matriks eselon baris dan mengembalikan solusinya
public string[] gaussJordanEquation	Method melakukan backward substitution pada matriks eselon baris tereduksi dan mengembalikan solusinya
public String[] cramerAlgo()	Method untuk menyelesaikan SPL dengan kaidah cramer
public static double[] inversSPL(Matrix M)	Method digunakan untuk menyelesaikan SPL dengan metode

	matriks balikan. Solusi SPL dikembalikan dalam bentuk array of double

3.3 Invers.java

Class Invers.java berisi method - method yang digunakan saat membutuhkan algoritma dengan menggunakan matriks balikan di dalamnya.

- **Attributes**

  Tidak ada attribute pada class ini

- **Constructors**

  Tidak ada konstruktor pada class ini.

- **Methods**

  Implementasi metode - metode yang digunakan untuk mencari matriks balikan dan algoritma menyelesaikan SPL dengan menggunakan matriks balikan.

Methods (Parameter)	Description
public static Matrix inversOBE(Matrix M)	Method mengembalikan matriks balikan dari M, matriks balikan dicari menggunakan metode OBE
public static Matrix inversCofactor(Matriks M)	Method mengembalikan matriks balikan dari M, matriks balikan dicari menggunakan metode kofaktor

3.4 Interpolate.java

Class berisi algoritma - algoritma yang dibutuhkan saat melakukan interpolasi polinom

- **Attributes**

  Tidak ada attribute pada class ini.

- **Constructors**

  Tidak ada konstruktor pada class ini.

- **Methods**

  Berisi metode yang dibutuhkan untuk melakukan interpolasi

Methods	Description
public static double[] interpolateAlg(Matrix M)	Mengembalikan array of double yang bersifat sebagai koefisien dari polinomial derajat
public static double functionInterpolate(double[] ar, double x)	Mengembalikan f(x) yang merupakan hasil interpolasi dari x, x merupakan input user.

3.5 Regression.java

Class berisi algoritma untuk melakukan regresi linear berganda

- **Attributes**

  Tidak ada attribute pada class ini.

- **Constructors**

  Tidak ada konstruktor pada class ini.

- **Methods**

  Berisi metode yang dibutuhkan untuk melakukan interpolasi

Methods	Description
public static double[] regressionAlgo(Matrix M)	Mengembalikan fungsi hasil regresi dalam bentuk array double
public static double functionRegression(double[] arrX, double[] inputX)	Menghitung taksiran hasil fungsi dari fungsi yang dihasilkan regressionAlgo

4. Folder Main

Folder ini hanya berisi 1 class dan merupakan driver dari library java yang sudah kami buat. Menggunakan class Menu.java dari utility untuk menampilkan menu utama serta menerima input dari user. Program akan menggunakan algoritma - algoritma yang sudah dibuat untuk menyelesaikan suatu permasalahan, beberapa contoh berupa menyelesaikan SPL, mencari determinan dari suatu matriks, dan masih banyak lagi.

- **Attributes**

  Tidak ada attribute pada class ini.

- **Constructors**

  Tidak ada konstruktor pada class ini.

- **Methods**

  Berisi metode main yang akan dijalankan

Methods	Description
public static void main (String[] args)	Metode utama dari program yang akan dijalankan dan bertindak sebagai driver dari program

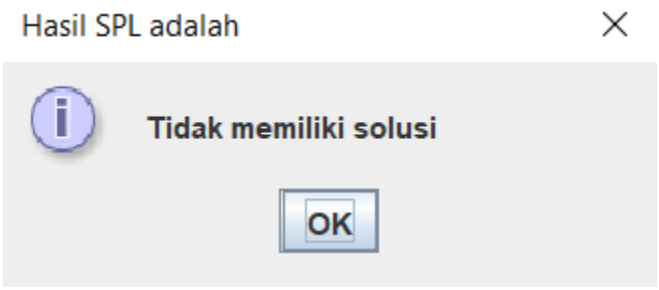
BAB 4
Eksperimen

4.1 Temukan solusi SPL ax = b, berikut :

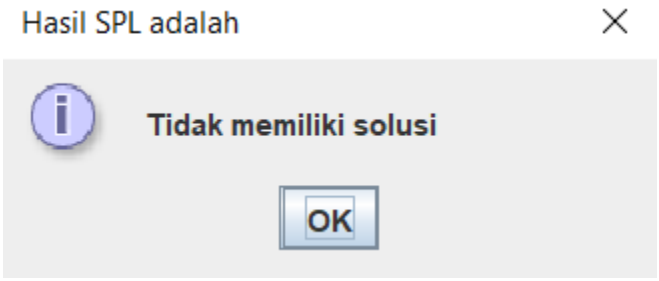
a.

$$A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix}$$

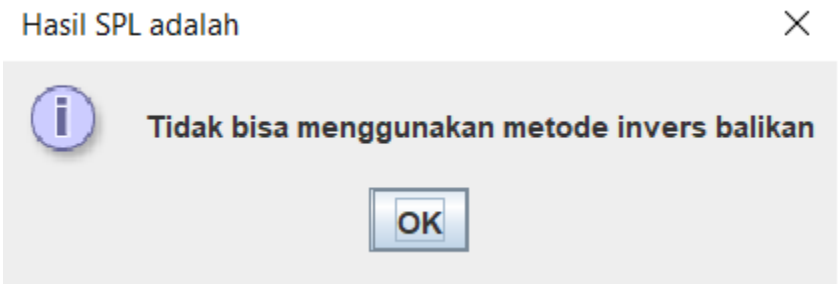
- **Dengan metode Gauss :**



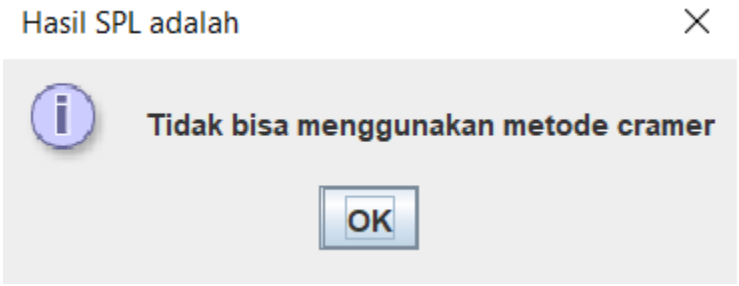
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**

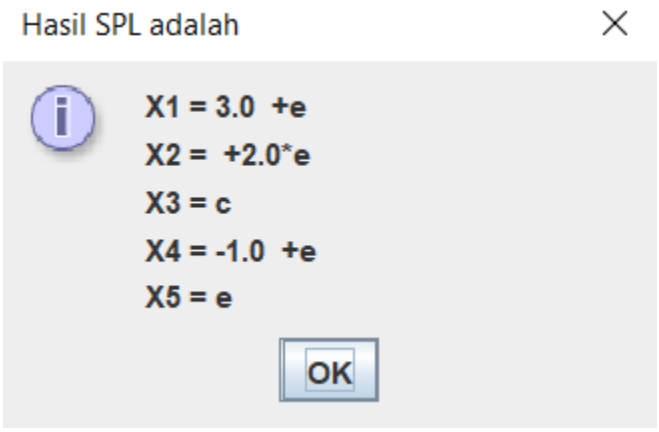


Analisis tambahan: Saat menggunakan metode gauss dan gauss-jordan, matrix dapat diproses namun ternyata SPL tidak memiliki hasil. Selain itu, persamaan ini tidak dapat dipecahkan menggunakan invers dan cramer karena memiliki determinan 0.

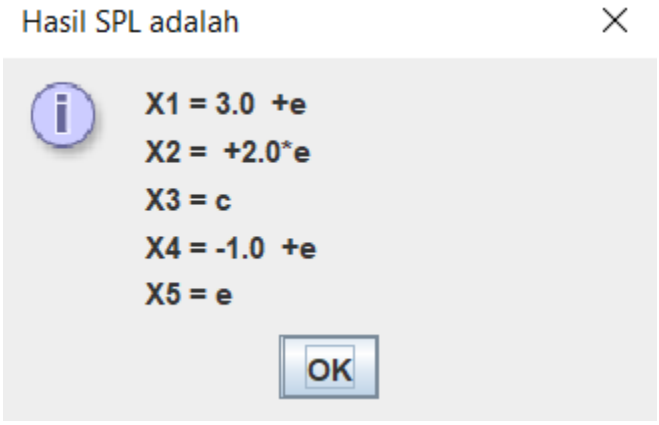
b.

$$A = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -3 & 0 \\ 2 & -1 & 0 & 1 & -1 \\ -1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 6 \\ 5 \\ -1 \end{bmatrix}$$

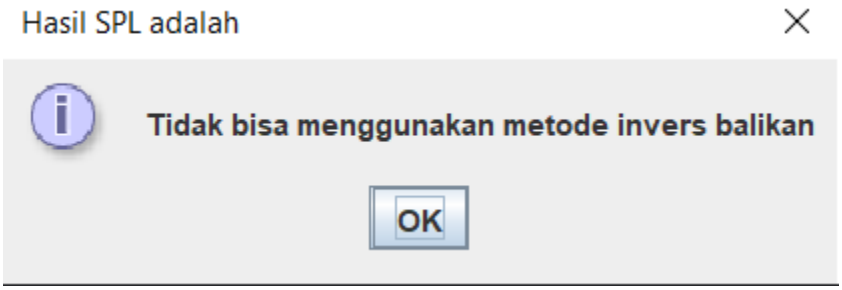
- **Dengan metode Gauss :**



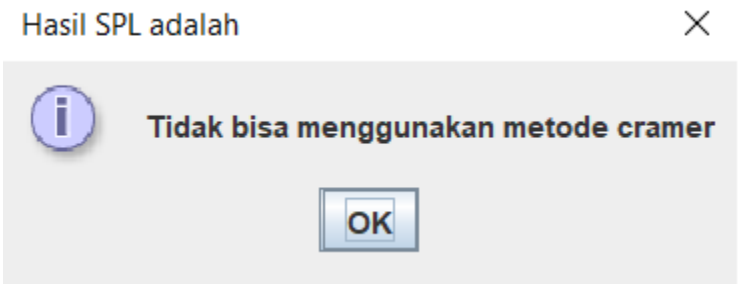
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**

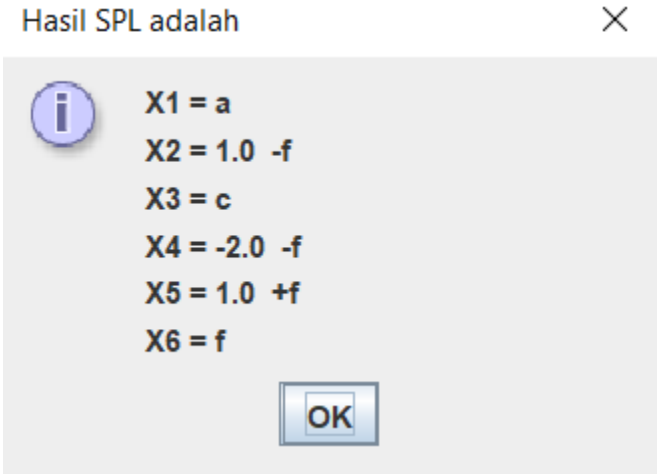


Analisis tambahan: SPL menghasilkan persamaan parametrik pada gauss dan gauss-jordan, sedangkan invers dan cramer tidak dapat memecahkan SPL ini karena matriks tidak berbentuk persegi.

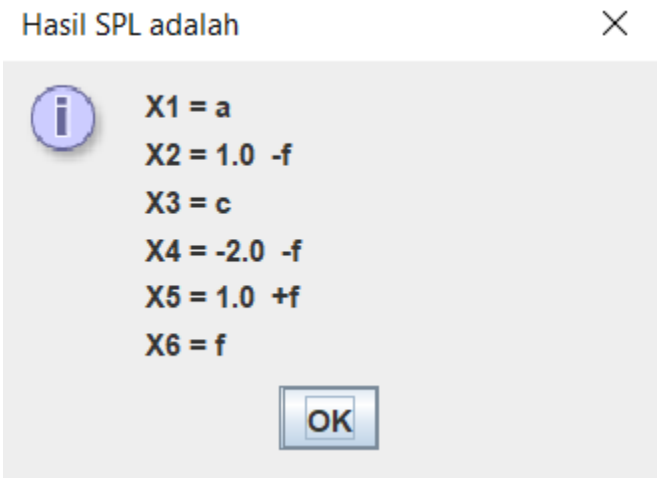
c.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$

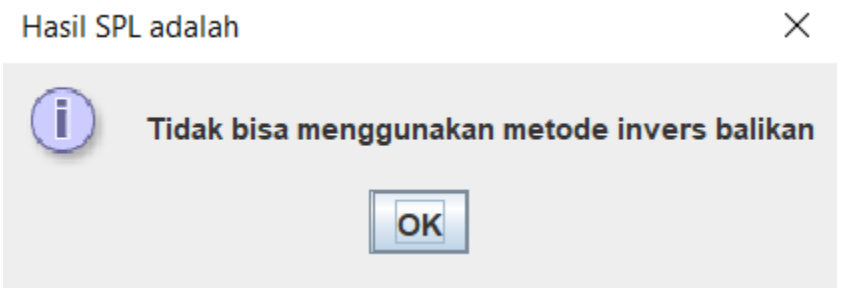
- **Dengan metode Gauss :**



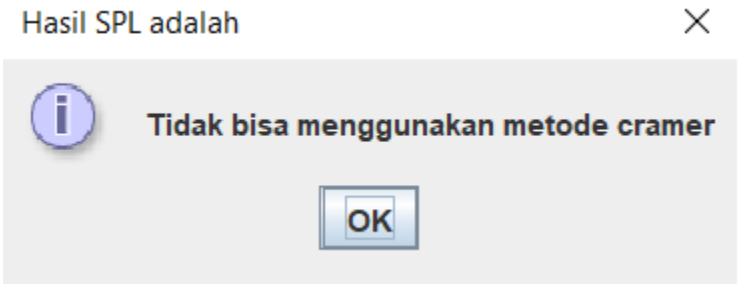
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**



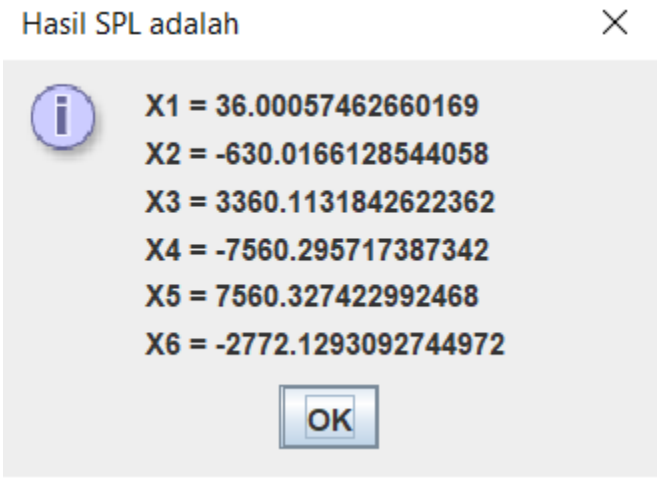
Analisis tambahan: SPL menghasilkan persamaan parametrik pada gauss dan gauss-jordan, sedangkan invers dan cramer tidak dapat memecahkan SPL ini karena matriks tidak berbentuk persegi.

d.

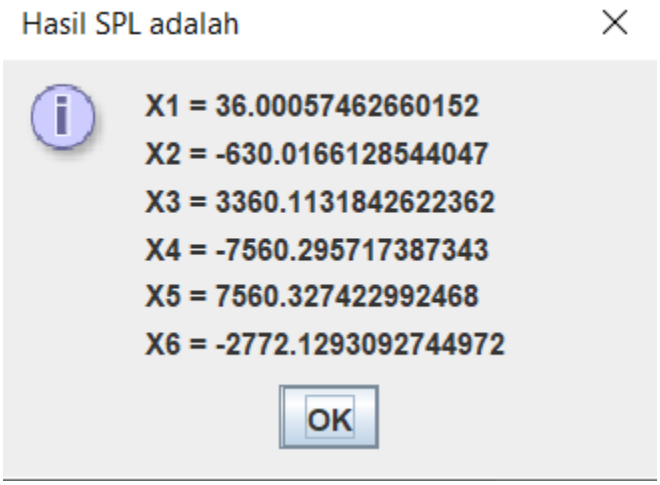
N = 6

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{n+2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n+1} \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

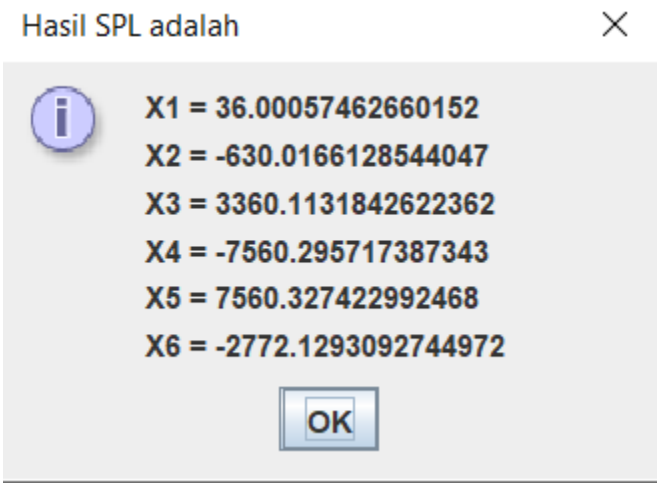
- **Dengan metode Gauss :**



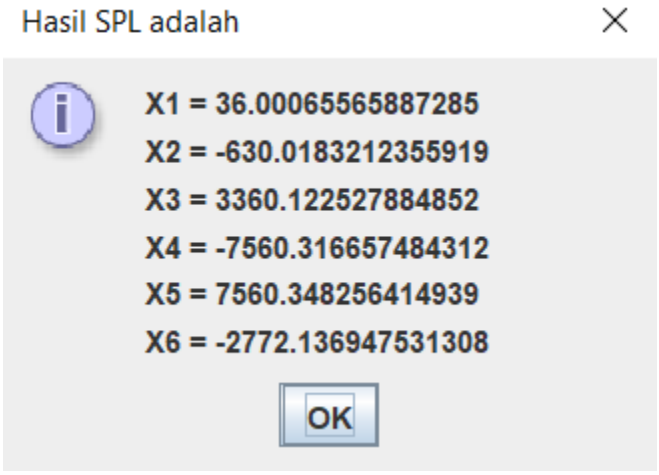
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**

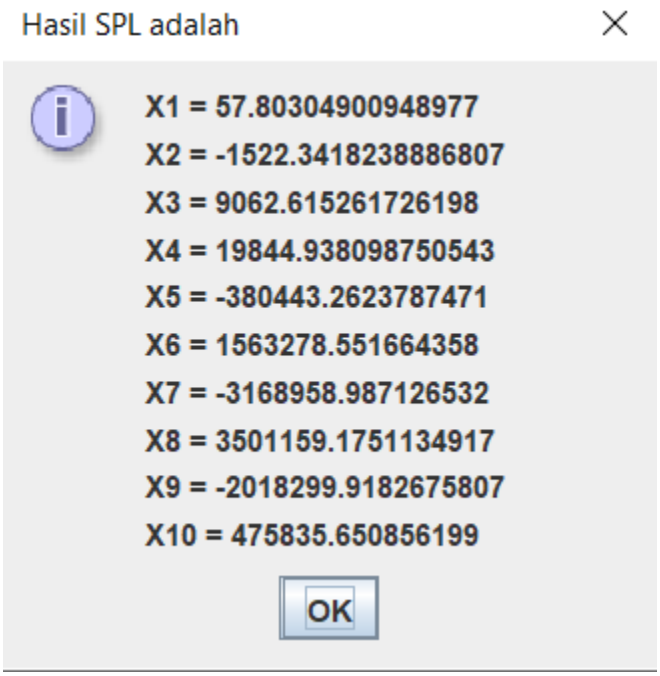


- **Dengan kaidah cramer :**

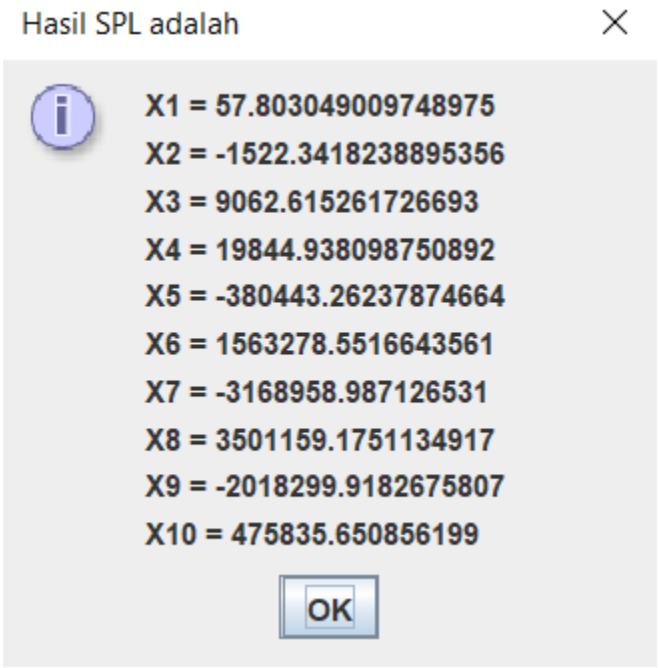


N = 10

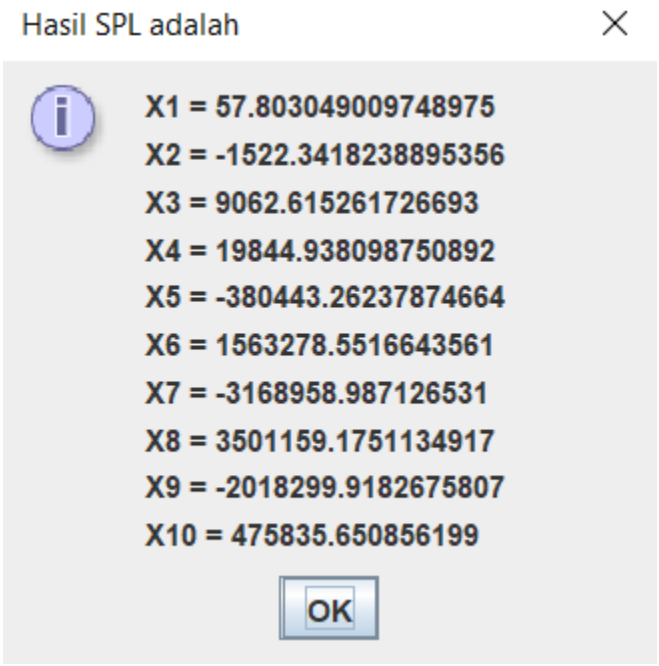
- **Dengan metode Gauss :**



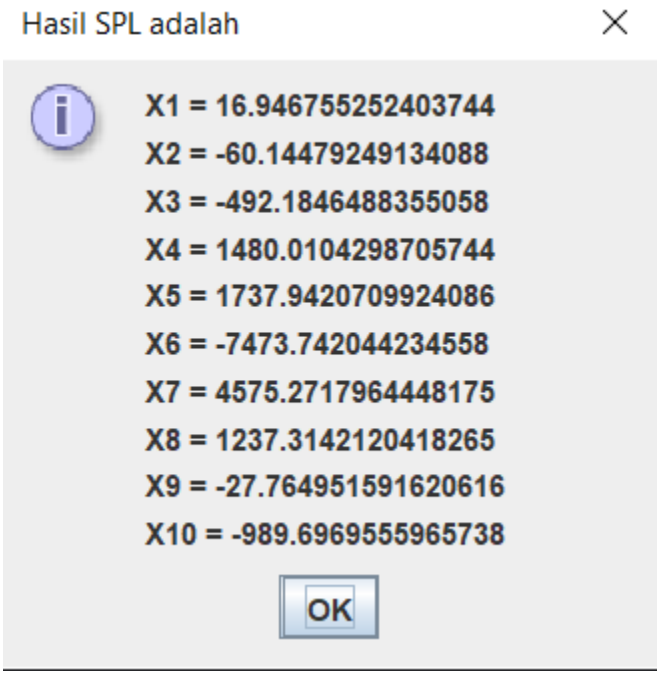
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**



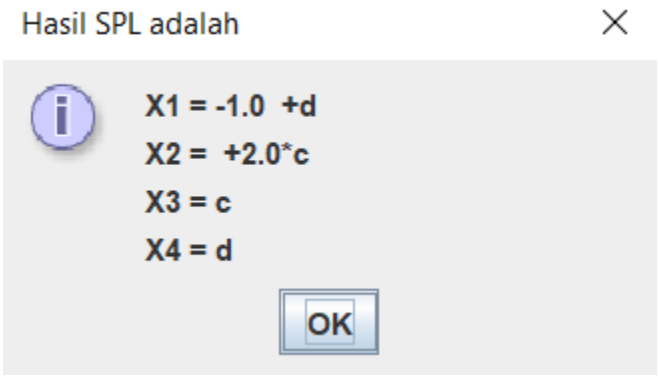
Analisa tambahan: Pada matrix hilbert dengan n = 10, terjadi perbedaan hasil dikarenakan ketidaktelitian floating point yang terbawa terus-menerus. Dengan kaidah cramer, matrix 10x10 akan memanggil matrix 9x9 dan seterusnya secara rekursif, sehingga jumlah operasi perkalian (yang menambahkan ketidaktelitian) berjumlah sangat banyak.

4.2 Menyelesaikan SPL berbentuk matriks *augmented*

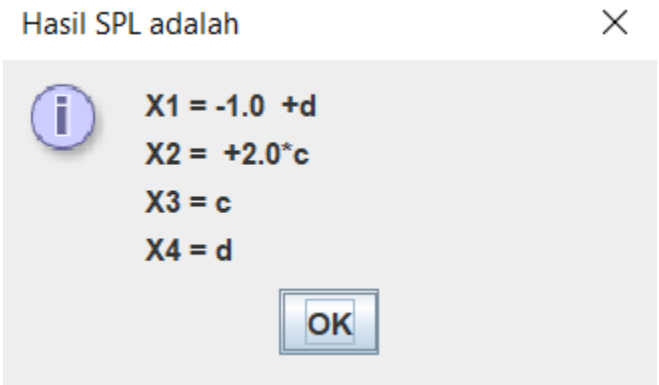
a.

$$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}.$$

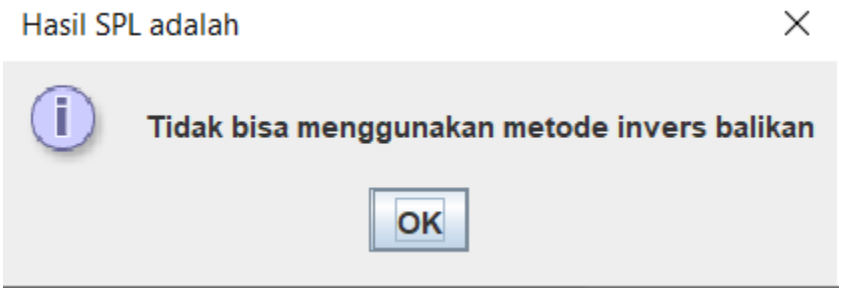
- **Dengan metode Gauss :**



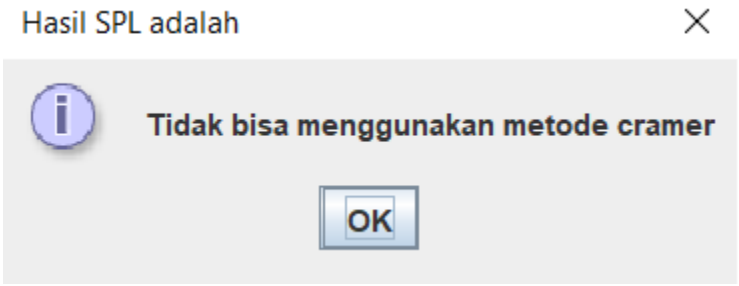
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**

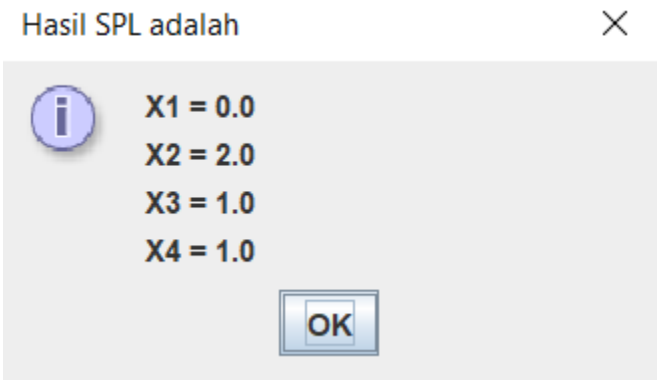


ANALISIS : Pada soal 2a metode cramer dan matriks balikan tidak bisa digunakan karena matriks bukan matriks persegi. Didapatkan hasil berupa parametrik dari metode gauss dan gauss jordan.

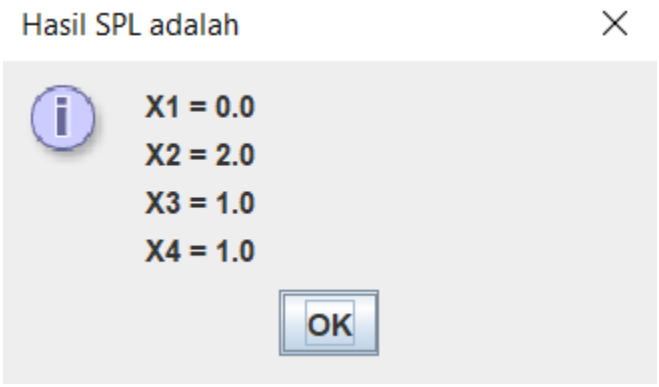
b.

$$\begin{bmatrix} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{bmatrix}.$$

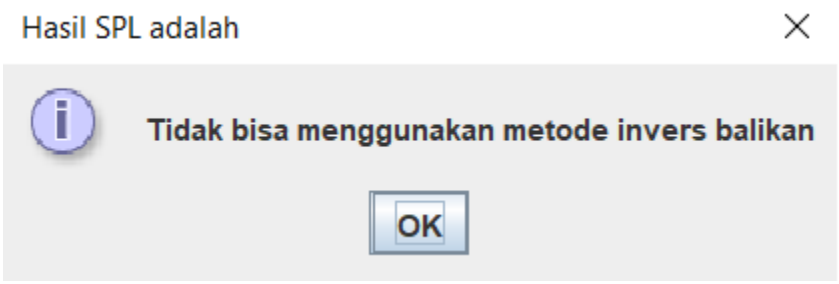
- **Dengan metode Gauss :**



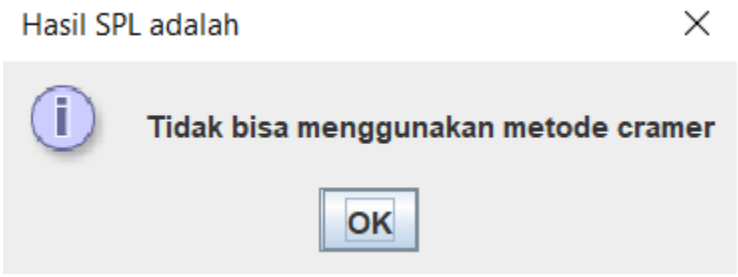
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**



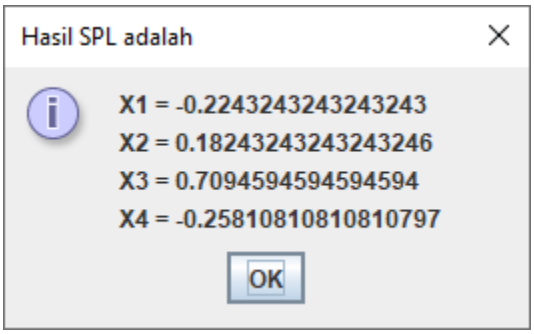
ANALISIS : Pada soal 2b metode cramer dan matriks balikan tidak bisa digunakan karena matriks bukan matriks persegi. SPL memiliki solusi singular, didapat dengan metode Gauss dan Gauss Jordan

4.3 Menyelesaikan SPL berbentuk :

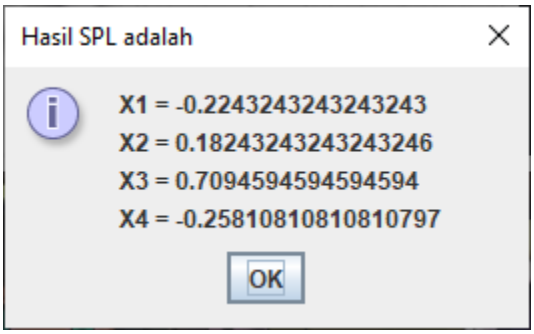
a.

$$
\begin{aligned}
8x_1 + x_2 + 3x_3 + 2x_4 &= 0 \\
2x_1 + 9x_2 - x_3 - 2x_4 &= 1 \\
x_1 + 3x_2 + 2x_3 - x_4 &= 2 \\
x_1 + 6x_3 + 4x_4 &= 3
\end{aligned}
$$

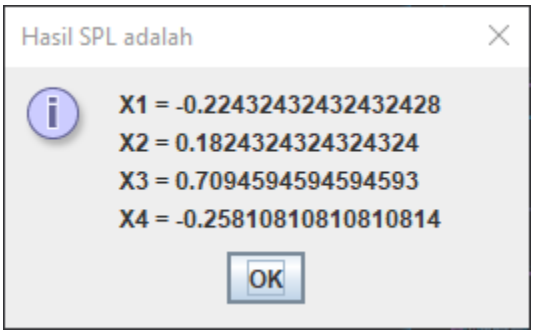
- **Dengan metode Gauss :**



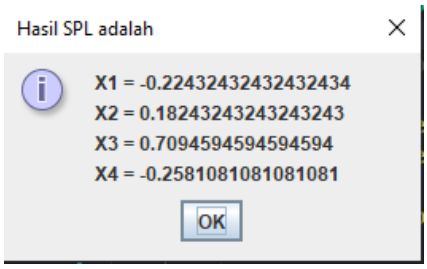
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**

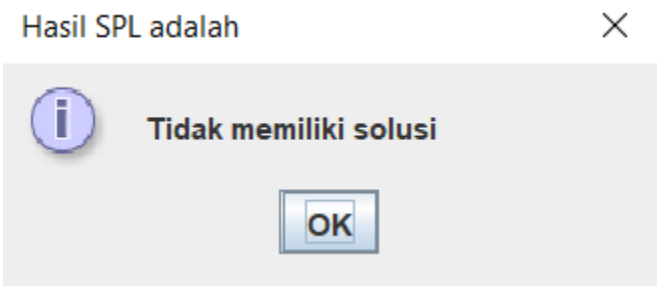


Analisis tambahan : Pada soal ini, semua metode dapat menyelesaikan SPL dengan menghasilkan solusi yang sama.

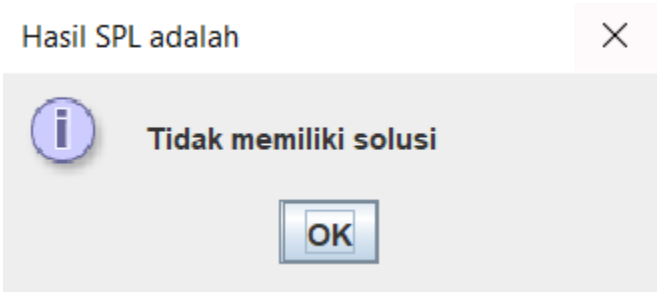
b.

$$
\begin{aligned}
x_7 + x_8 + x_9 &= 13.00 \\
x_4 + x_5 + x_6 &= 15.00 \\
x_1 + x_2 + x_3 &= 8.00 \\
0.04289(x_3 + x_5 + x_7) + 0.75(x_6 + x_8) + 0.61396x_9 &= 14.79 \\
0.91421(x_3 + x_5 + x_7) + 0.25(x_2 + x_4 + x_6 + x_8) &= 14.31 \\
0.04289(x_3 + x_5 + x_7) + 0.75(x_2 + x_4) + 0.61396x_1 &= 3.81 \\
x_3 + x_6 + x_9 &= 18.00 \\
x_2 + x_5 + x_8 &= 12.00 \\
x_1 + x_4 + x_7 &= 6.00 \\
0.04289(x_1 + x_5 + x_9) + 0.75(x_2 + x_6) + 0.61396x_3 &= 10.51 \\
0.91421(x_1 + x_5 + x_9) + 0.25(x_2 + x_4 + x_6 + x_8) &= 16.13 \\
0.04289(x_1 + x_5 + x_9) + 0.75(x_4 + x_8) + 0.61396x_7 &= 7.04
\end{aligned}
$$

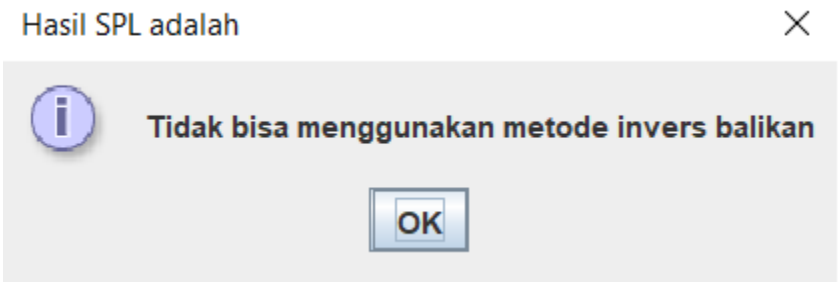
- **Dengan metode Gauss :**



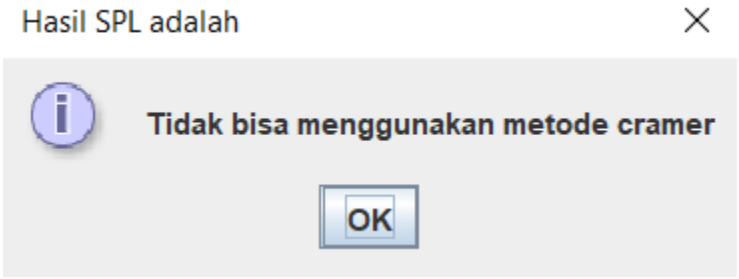
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**

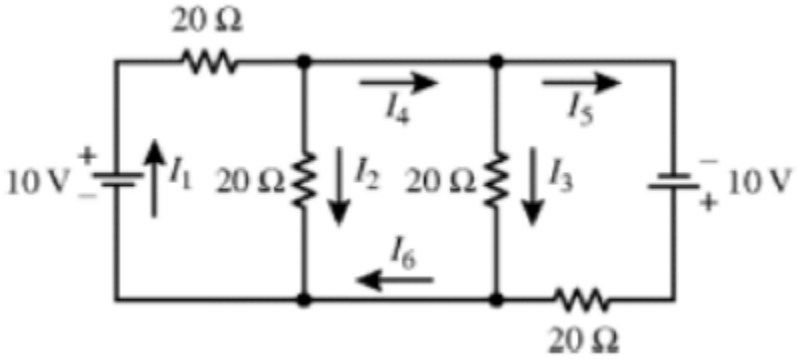


- **Dengan kaidah cramer :**



Analisis tambahan : Soal ini tidak bisa diselesaikan dengan matriks balikan dan kaidah cramer karena matriks memiliki ukuran 12 x 9 (bukan matriks persegi). Dengan menggunakan Gauss dan Gauss Jordan, didapat bahwa SPL tersebut tidak memiliki solusi.

4.4 Tentukan arus yang mengalir pada rangkaian listrik di bawah ini :

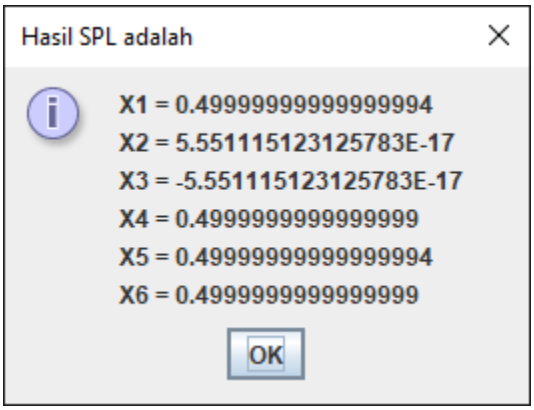


Bila rangkaian diubah menjadi matriks :

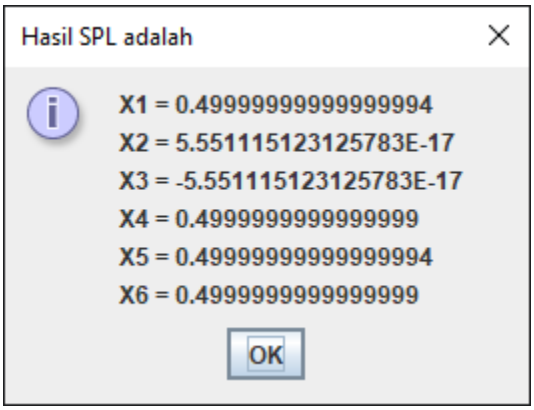
	X_1	X_2	X_3	X_4	X_5	X_6	b
1	4	0	0	-2	0	0	1
2	-2	0	0	4	-2	0	0
3	0	0	0	-2	4	0	1
4	1	-1	0	-1	0	0	0
5	0	0	-1	1	-1	0	0
6	0	0	0	1	0	-1	0

dengan $X_1 = I_1$, $X_2 = I_2$,... $X_n = I_n$

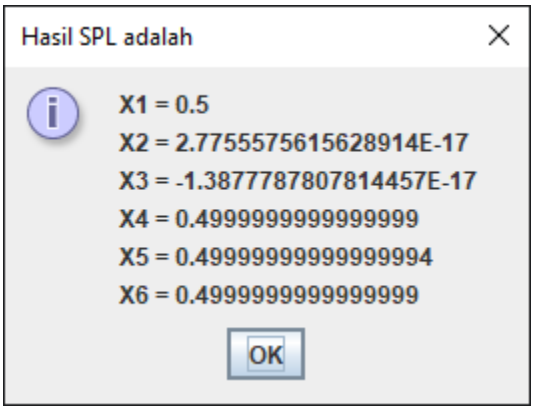
- **Dengan metode Gauss :**



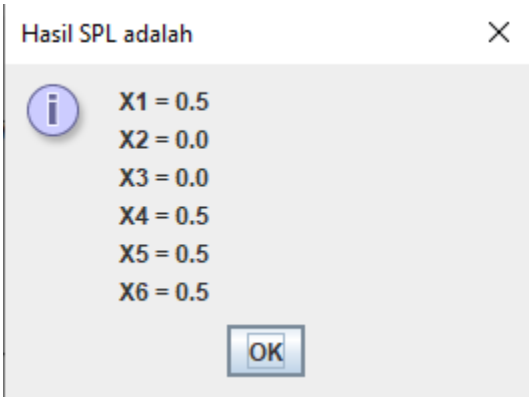
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



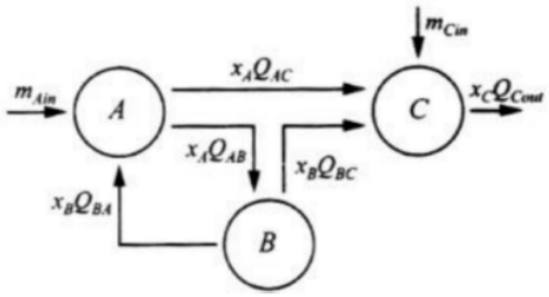
- **Dengan kaidah cramer :**



Analisis tambahan : Soal ini diselesaikan dengan mengubah persamaan dari rangkaian listrik menjadi matriks terlebih dahulu, lalu matriks diselesaikan dengan kalkulator matriks. Hasil dari ke 4 metode sangat mirip, terdapat sedikit perbedaan karena adanya ketidak-akuratan saat mengolah *double*

4.5

Lihatlah sistem reaktor pada gambar berikut



Dengan laju volume Q dalam m^3/s dan input massa m_{in} dalam mg/s. Konservasi massa pada tiap inti reaktor adalah sebagai berikut:

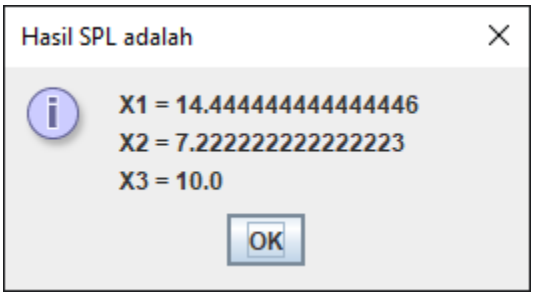
$$\text{A:} \quad m_{A_{in}} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0$$

$$\text{B:} \quad Q_{AB}x_A - Q_{BA}x_B - Q_{BC}x_B = 0$$

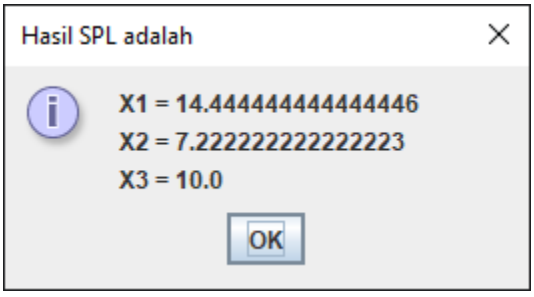
$$\text{C:} \quad m_{C_{in}} + Q_{AC}x_A + Q_{BC}x_B - Q_{C_{out}}x_C = 0$$

Tentukan solusi x_A, x_B, x_C dengan menggunakan parameter berikut : $Q_{AB} = 40$, $Q_{AC} = 80$, $Q_{BA} = 60$, $Q_{BC} = 20$ dan $Q_{Cout} = 150$ m^3/s dan $m_{Ain} = 1300$ dan $m_{Cin} = 200$ mg/s.

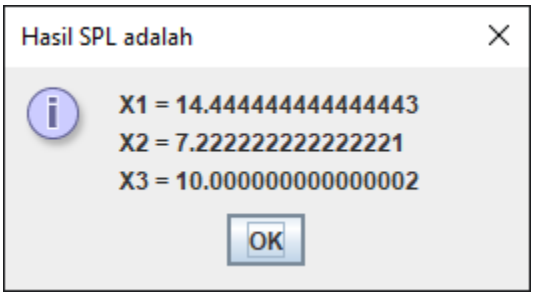
- **Dengan metode Gauss :**



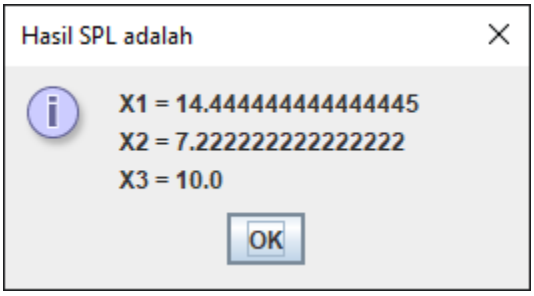
- **Dengan metode Gauss - Jordan :**



- **Dengan metode matriks balikan :**



- **Dengan kaidah cramer :**



Analisis tambahan: Persamaan konservasi massa diatas dapat dibentuk menjadi sistem persamaan linier dengan memasukkan beberapa parameter yang ada pada studi kasus. SPL ini dapat diselesaikan oleh semua metode penyelesaian SPL.

4.6

a.

Gunakan tabel di bawah ini untuk mencari polinom interpolasi dari pasangan titik-titik yang terdapat dalam tabel. Program menerima masukan nilai x yang akan dicari nilai fungsi $f(x)$.

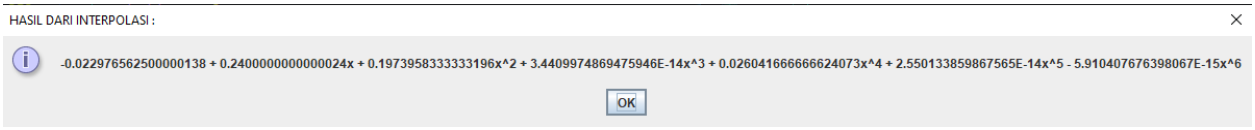
x	0.1	0.3	0.5	0.7	0.9	1.1	1.3
$f(x)$	0.003	0.067	0. 148	0.248	0.370	0.518	0.697

Lakukan pengujian pada nilai-nilai default berikut:

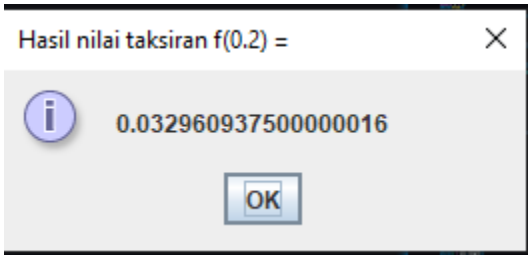
$x = 0.2$ f(x) = ?
$x = 0.55$ f(x) = ?
$x = 0.85$ f(x) = ?
$x = 1.28$ f(x) = ?

Hasil yang didapat :

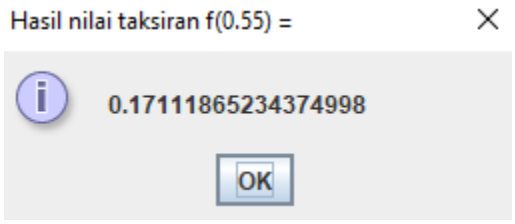
POLINOM INTERPOLASI YANG DIDAPAT :



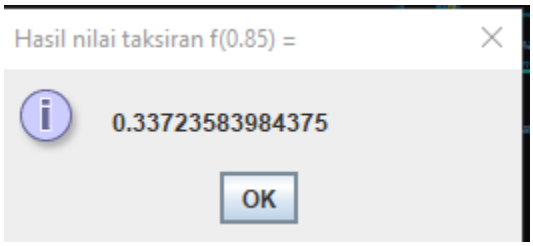
- **X = 0.2**
  HASIL TAKSIRAN :



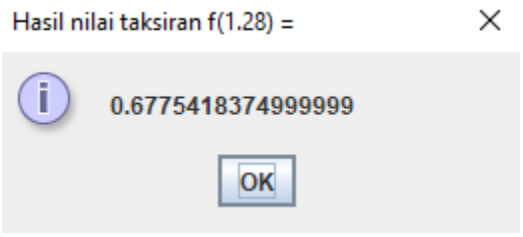
- **X = 0.55**
  HASIL TAKSIRAN :



- **X = 0.85**
  HASIL TAKSIRAN :



- **X = 1.28**
  HASIL TAKSIRAN :



Analisis tambahan: Keluaran dari fungsi akan berbentuk sebuah fungsi interpolasi dimana saat dimasukkan parameter fungsi, nilainya konsisten dengan apa yang ada di tabel (misal f(1.1) < f(1.28) < f(1.30))

b.

Jumlah kasus positif baru Covid-19 di Indonesia semakin fluktuatif dari hari ke hari. Di bawah ini diperlihatkan jumlah kasus baru Covid-19 di Indonesia mulai dari tanggal 17 Juni 2021 hingga 31 Agustus 2021:

Tanggal	Tanggal (desimal)	Jumlah Kasus Baru
17/06/2021	6,567	12.624
30/06/2021	7	21.807
08/07/2021	7,258	38.391
14/07/2021	7,451	54.517
17/07/2021	7,548	51.952
26/07/2021	7,839	28.228
05/08/2021	8,161	35.764
15/08/2021	8,484	20.813
22/08/2021	8,709	12.408
31/08/2021	9	10.534

Tanggal (desimal) adalah tanggal yang sudah diolah ke dalam bentuk desimal 3 angka di belakang koma dengan memanfaatkan perhitungan sebagai berikut:

tanggal(desimal) = bulan + (tanggal / jumlah hari pada bulan tersebut)

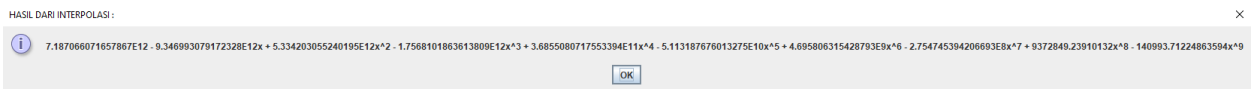
Sebagai **contoh**, untuk tanggal 17/06/2021 (dibaca: 17 Juni 2021) diperoleh tanggal(desimal) sebagai berikut:

Tanggal(desimal) = 6 + (17/30) = 6,567

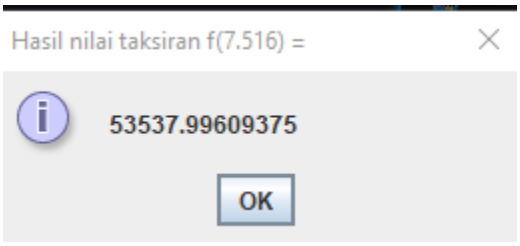
Gunakanlah data di atas dengan memanfaatkan **polinom interpolasi** untuk melakukan prediksi jumlah kasus baru Covid-19 pada tanggal-tanggal berikut:

a. 16/07/2021
b. 10/08/2021
c. 05/09/2021
d. beserta masukan user lainnya berupa **tanggal (desimal) yang sudah diolah** dengan asumsi prediksi selalu dilakukan untuk tahun 2021.

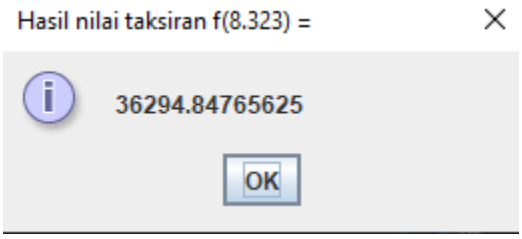
POLINOM INTERPOLASI



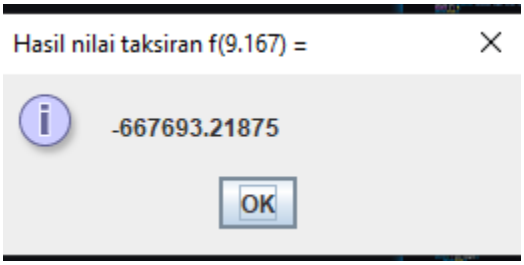
- **16/07/2021 (7.516)**
  HASIL TAKSIRAN :



- **10/08/2021 (8.323)**
  HASIL TAKSIRAN :



- **05/09/2021 (9.167)**
  HASIL TAKSIRAN :



Analisis tambahan: Keluaran dari fungsi akan berbentuk sebuah fungsi interpolasi dimana saat dimasukkan parameter fungsi, nilainya konsisten dengan apa yang ada pada data awal.

c. Sederhanakan fungsi

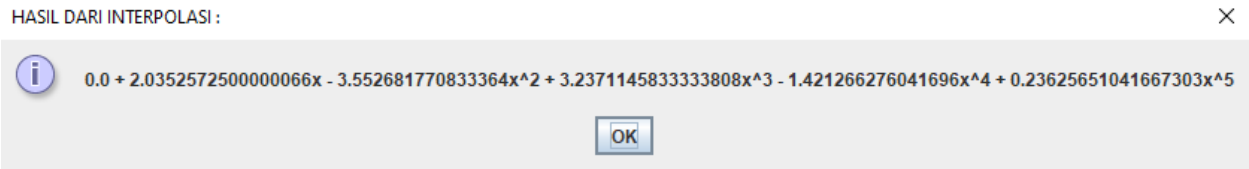
$$f(x) = \frac{x^2 + \sqrt{x}}{e^x + x}$$

dengan polinom interpolasi derajat n di dalam selang [0, 2]. Sebagai contoh, jika n = 5, maka titik-titik x yang diambil di dalam selang [0, 2] berjarak $h = (2 - 0)/5 = 0.4$.

Menggunakan polinom interpolasi derajat n di dalam selang [0,2].

- Untuk n = 5 (jarak h = 0.4) :

f(x) =



Analisis tambahan: Fungsi diatas dapat didekatkan dengan menggunakan interpolasi untuk mendapatkan sebuah fungsi polinomial yang merepresentasikan suatu fungsi dengan keakuratan tertentu.

4.7

7. Studi Kasus Regresi Linier Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3	Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3
0.90	72.4	76.3	29.18	1.07	23.2	76.8	29.38
0.91	41.6	70.3	29.35	0.94	47.4	86.6	29.35
0.96	34.3	77.1	29.24	1.10	31.5	76.9	29.63
0.89	35.1	68.0	29.27	1.10	10.6	86.3	29.56
1.00	10.7	79.0	29.78	1.10	11.2	86.0	29.48
1.10	12.9	67.4	29.39	0.91	73.3	76.3	29.40
1.15	8.3	66.8	29.69	0.87	75.4	77.9	29.28
1.03	20.1	76.9	29.48	0.78	96.6	78.7	29.29
0.77	72.2	77.7	29.09	0.82	107.4	86.8	29.03
1.07	24.0	67.7	29.60	0.95	54.9	70.9	29.37

Source: Charles T. Hare, "Light-Duty Diesel Emission Correction Factors for Ambient Conditions," EPA-600/2-77-116. U.S. Environmental Protection Agency.

Gunakan *Normal Estimation Equation for Multiple Linear Regression* untuk mendapatkan regresi linear berganda dari data pada tabel di atas, kemudian estimasi nilai Nitrous Oxide apabila Humidity bernilai 50%, temperatur 76°F, dan tekanan udara sebesar 29.30.

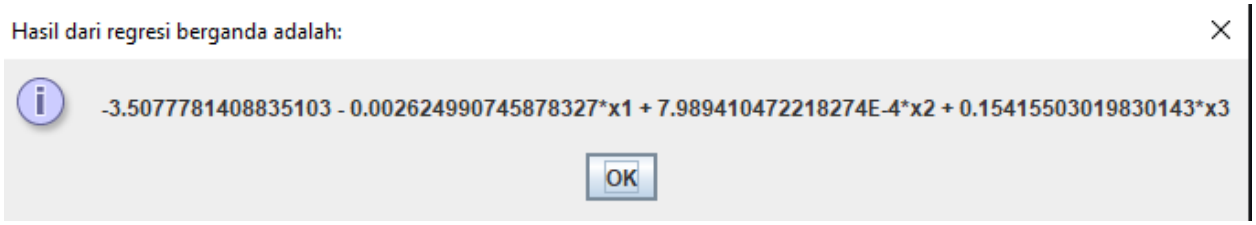
Dari data-data tersebut, apabila diterapkan *Normal Estimation Equation for Multiple Linear Regression*, maka diperoleh sistem persamaan linear sebagai berikut.

$$20b_0 + 863.1b_1 + 1530.4b_2 + 587.84b_3 = 19.42$$

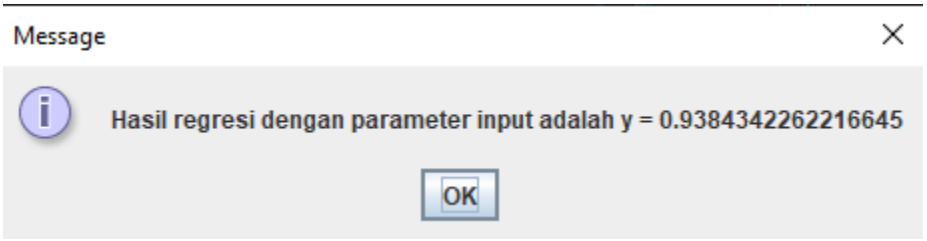
$$863.1b_0 + 54876.89b_1 + 67000.09b_2 + 25283.395b_3 = 779.477$$

$$1530.4b_0 + 67000.09b_1 + 117912.32b_2 + 44976.867b_3 = 1483.437$$

$$587.84b_0 + 25283.395b_1 + 44976.867b_2 + 17278.5086b_3 = 571.1219$$



HASIL PENAKSIRAN :



Analisis tambahan: Hasil dari regresi berganda adalah sebuah persamaan linier dengan beberapa peubah yang dimana apabila kita masukkan parameter x1,x2, … xn, akan didapatkan taksiran nilai y dengan kondisi sesuai parameter. Dalam hal ini, nilai y yang didapat konsisten dengan tabel (nilai tidak jauh dari data yang ada).

BAB 5
Kesimpulan

5.1 Kesimpulan

SPL dapat diselesaikan menggunakan berbagai macam metode, diantaranya metode eliminasi Gauss, Gauss - Jordan, matriks balikan, dan kaidah cramer. SPL ini juga ternyata dapat digunakan untuk menaksir suatu nilai menggunakan interpolasi polinom dan regresi linear berganda.

Untuk tugas besar ini, kami berhasil membuat kalkulator matriks dengan implementasi algoritma yang sudah ada ke dalam bahasa Java dan dapat menyelesaikan berbagai permasalahan menggunakan matriks, contohnya adalah menggunakan interpolasi polinom dan regresi linear berganda untuk memprediksi data covid dan dapat memperkirakan banyak Nitrous Oxide berdasarkan beberapa nilai yang ada.

5.2 Saran

Saran untuk kelompok kami diantaranya :

- Seharusnya pengecekan test case dilakukan lebih baik lagi, agar tidak terjadi lagi penemuan code bug mendekati deadline.
- Kode program diberi komentar yang lebih jelas lagi. Agar kode yang digunakan lebih mudah dimengerti dan semua anggota bisa melakukan *debugging* pada algoritma yang bermasalah
- Seharusnya dilakukan pemecahan masalah terlebih dahulu di awal pengerjaan, agar *workload* masing - masing anggota lebih jelas.

5.3 Refleksi

Refleksi yang kami dapatkan dari tugas ini adalah kami bisa memperbaiki lagi kinerja kami dalam berbagai hal, contohnya pembuatan timeline kerja agar selalu ada progress dalam tugas tiap harinya. Kami juga merasa masih harus memperbaiki pembagian waktu karena pengerjaan tugas masih terlalu berdekatan dengan *deadline*. Hal lain yang kami dapatkan dari tugas ini adalah pengalaman dan pengetahuan membuat program dalam bahasa Java.

DAFTAR PUSTAKA

https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2021-2022/Pengantar-Pemrograman-dengan-Bahasa-Java-2021.pdf

https://www.w3schools.com/java/java_try_catch.asp

https://www.w3schools.com/java/default.asp

https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2021-2022/algeo21-22.htm

https://www.petanikode.com/java-swing-joptionpane/

https://docs.oracle.com/javase/7/docs/api/javax/swing/JOptionPane.html

https://docs.oracle.com/javase/7/docs/api/javax/swing/JPanel.html

https://docs.oracle.com/en/java/javase/13/docs/specs/man/javac.html

https://docs.oracle.com/en/java/javase/15/docs/api/java.base/java/util/Scanner.html