LAPORAN TUGAS BESAR 1

IF2123 ALJABAR LINEAR GEOMETRI

Kelompok SERINGAI

DISUSUN OLEH:

Ariel Jovananda	13521086
Muhammad Abdul Aziz Ghazali	13521128
Muhammad Rifqi Farhansyah	13521166

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2022

DAFTAR ISI

BAB 1: DESKRIPSI MASALAH

Sistem persamaan linier (SPL) banyak ditemukan di dalam bidang sains dan rekayasa. Anda sudah mempelajari berbagai metode untuk menyelesaikan SPL, termasuk menghitung determinan matriks. Sembarang SPL dapat diselesaikan dengan beberapa metode, yaitu metode eliminasi Gauss, metode eliminasi Gauss-Jordan, metode matriks balikan ($x = A^{-1}b$), dan kaidah *Cramer* (khusus untuk SPL dengan n peubah dan n persamaan). Solusi sebuah SPL mungkin tidak ada, banyak (tidak berhingga), atau hanya satu (unik/tunggal).

Di dalam Tugas Besar 1 ini, anda diminta 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, kaidah Cramer (kaidah Cramer khusus untuk SPL dengan n peubah dan n persamaan). Selanjutnya, gunakan *library* tersebut di dalam program Java untuk menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi. Penjelasan tentang interpolasi dan regresi adalah seperti di bawah ini.

Beberapa tulisan cara membuat library di Java:

1. https://www.programcreek.com/2011/07/build-a-java-library-for-yourself/
2. https://developer.ibm.com/tutorials/j-javalibrary/
3. https://stackoverflow.com/questions/3612567/how-to-create-my-own-java-libraryapi

BAB 2: TEORI SINGKAT

1. Elminasi Gauss/Gauss-Jordan

Masalah berbasis matriks dapat diselesaikan dengan menggunakan teknik eliminasi Gaussian. Prinsip matriks eselon, yang memandu eliminasi Gauss, mengambil bentuk berikut:

$$\begin{bmatrix} 1 & * & * & \dots & * & * \\ 0 & 1 & * & \dots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

Pendekatan ini didasarkan pada gagasan bahwa angka bukan nol harus dimulai dengan angka 1, dan bahwa angka lain dapat muncul setelahnya berturut-turut setelah itu.

Sebagai kelanjutan dari Eliminasi Gauss, metode Eliminasi Gauss-Jordan adalah teknik eliminasi. Eliminasi Gauss-Jordan disebut sebagai matriks eselon tereduksi dan memiliki bentuk sebagai berikut:

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 & * \\ 0 & 1 & 0 & \dots & 0 & * \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 1 & * \end{bmatrix}$$

Untuk memastikan bahwa hasil persamaan selanjutnya berbentuk variabel persamaan yang diinginkan pengguna, angka 1 harus diapit oleh nol.

2. Determinan

Determinan, adalah angka yang dapat dihitung dari komponen matriks persegi, atau matriks dengan jumlah baris dan kolom yang sama. Notasi determinan dapat dinyatakan sebagai det(A) atau |A|. Untuk matriks 2 x 2, misalnya, determinannya dapat ditentukan sebagai berikut:

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$$

Untuk matriks 3 x 3 bisa menggunakan metode berikut:

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{12}a_{21}a_{33} - a_{11}a_{23}a_{32}$$

Saat menentukan nilai determinan segitiga atas dan bawah, ada fitur unik. Semua elemen dalam matriks segitiga atas (juga dikenal sebagai "segitiga atas") adalah nol, sedangkan semua elemen dalam matriks segitiga bawah (juga dikenal sebagai "segitiga bawah") adalah nol. Determinan segitiga segitiga dapat ditemukan hanya dengan mengalikan semua elemen di sepanjang diagonal utama. Oleh karena itu, jika diketahui segitiga A berukuran,

Pendekatan reduksi baris yang menerapkan OBE (Elementary Row Operation) hingga diperoleh matriks segitiga merupakan salah satu cara untuk menentukan determinan. Dari sana, nilai determinan dapat diturunkan menggunakan rumus determinan untuk matriks segitiga. Berikut adalah beberapa sifat OBE

1. Ketika sebuah baris matriks dikalikan dengan konstanta k, determinannya berubah menjadi. det(A)

2. Jika dua baris matriks dialihkan, determinannya berubah menjadi det(A).

3. Setelah menambahkan baris yang merupakan kelipatan dari baris lain, determinannya dipertahankan sebagai det(A).

Jadi, secara umum, determinan dapat dihitung menggunakan metode reduksi baris dengan cara berikut:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \overset{OBE}{\longleftrightarrow} \begin{bmatrix} a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & a'_{22} & \cdots & a'_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & a'_{nn} \end{bmatrix}$$

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

Dengan mendaftar berapa banyak operasi pertukaran baris yang dilakukan saat OBE sedang digunakan. Metode ekspansi kofaktor adalah cara lain untuk menghitung determinan selain pendekatan reduksi baris. Matriks n x n bisa didefinikan sebagai berikut:

1. Submatriks yang komponennya tidak berada pada baris I dan kolom j ditentukan oleh apa yang disebut "entri minor" (Mij (minot entri aij)

2. C_{ij} (Kofaktor entri a_{ij}) $= (-1)^{i+j} M_{ij}$

Oleh karena itu, salah satu rumus berikut dapat digunakan untuk menentukan matriks determinan A berukuran n menggunakan kofaktor teknik ekspansi:

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

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

3. Matriks Kofaktor dan Adjoin

Matriks kofaktor matriks (C) adalah matriks bujur sangkar yang dimensinya sesuai dengan matriks, dan masing-masing komponennya (Cij) berfungsi sebagai kofaktor untuk entri dalam matriks A. Dengan menerapkan rumus yang diberikan pada bagian II Bab 2, elemen dapat dihitung. Berikut ini adalah bagaimana matriks kofaktor dapat dinyatakan:

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

Matriks kofaktor ditransposisikan ke matriks untuk membuat adjoin. Menemukan invers (matriks invers), yang akan dirinci lebih lanjut pada bagian berikut, berguna ketika menggunakan matriks adjoin (bagian 3). Notasi adjoin matriks dapat direpresentasikan sebagai (A).

4. Matriks Balikan

Jika sebuah matriks memiliki A^{-1}, matriks tersebut dapat dibalik dan memiliki ukuran yang sama dengan matriks A. Dengan kata lain, apapun yang terjadi pada A, akan terjadi kebalikan di A^{-1}. Keduanya menghasilkan matriks identitas (I) yang terlihat seperti ini:

$$I = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Matriks kofaktor dan metode reduksi baris adalah dua metode untuk mencari matriks invers. Matriks invers untuk pendekatan matriks kofaktor dapat dihitung dengan menggunakan rumus
berikut:

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

Dimana adjoint diturunkan melalui transposisi matriks kofaktor. Mengenai metode reduksi baris, matriks invers dibuat dengan terlebih dahulu membuat matriks augmented berukuran 2n yang terdiri dari matriks dan matriks identitas. Kemudian, OBE diterapkan sampai sisi kiri matriks yang diperbesar (matriks) berbentuk matriks identitas, dan matriks inversnya adalah matriks di sisi kanan.

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & 0 & 0 & \cdots & 1 \end{bmatrix} \overset{OBE}{\longleftrightarrow} \begin{bmatrix} 1 & 0 & \cdots & 0 & a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & 1 & \cdots & 0 & a'_{21} & a'_{22} & \cdots & a'_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & a'_{n1} & a'_{n2} & \cdots & a'_{nn} \end{bmatrix}$$

5. Kaidah Cramer

Kaidah Cramer hanya berlaku untuk sistem persamaan linier yang memiliki solusi tunggal/unik, dan Kaidah Cramer adalah rumus atau prosedur yang dapat digunakan untuk menyelesaikan persamaan sistem linier dengan jumlah variabel sama dengan jumlah persamaan. Pendekatan ini membutuhkan nilai determinan matriks koefisien dan determinan matriks lain, yang keduanya dihasilkan dengan mensubstitusi matriks konstan untuk salah satu matriks koefisien, untuk menemukan solusi SPL dengan persamaan dan variabel (sisi kanan SPL). Jika Ax= B adalah SPL dengan n persamaan yang mengandung n variabel, maka berikut ini adalah solusi SPL:

$$x_1 = \frac{\det(\mathrm{A_1})}{\det(A)}, x_2 = \frac{\det(\mathrm{A_2})}{\det(A)}, \ldots, x_n = \frac{\det(\mathrm{A_n})}{\det(A)}$$

$$A_1 = \begin{vmatrix} b_1 & a_{12} & \cdots & a_{1n} \\ b_2 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_n & a_{n2} & \cdots & a_{nn} \end{vmatrix}, A_2 = \begin{vmatrix} a_{11} & b_1 & \cdots & a_{1n} \\ a_{21} & b_2 & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & b_n & \cdots & a_{nn} \end{vmatrix}, \ldots, A_n = \begin{vmatrix} a_{11} & a_{12} & \cdots & b_1 \\ a_{21} & a_{22} & \cdots & b_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & b_n \end{vmatrix}$$

6. Interpolasi Polinom

Teknik untuk menaksir nilai suatu titik (x) dengan aproksimasi pada sembarang titik pada interval [xo,...,] adalah interpolasi polinomial. Interpolasi derajat polinomial yang menginterpolasi titik (x,y),(x,y),...,(x,y) adalah $pn(x) = a0 + a1x + a2x^2 + \cdots + anx^n$. Kami memperoleh sistem persamaan linier dengan mensubstitusi semua nilai (xi, yi) untuk =1, 2,..., n.

$$a_0 + a_1x_0 + a_2{x_0}^2 + \ldots + a_n{x_0}^n = y_0$$

$$a_0 + a_1x_1 + a_2{x_1}^2 + \ldots + a_n{x_1}^n = y_1$$

$$\ldots \qquad\qquad \ldots$$

$$a_0 + a_1x_n + a_2{x_n}^2 + \ldots + a_n{x_n}^n = y_n$$

Pendekatan eliminasi Gaussian digunakan untuk mendapatkan nilai 0, 1,..., yang mewakili solusi sistem persamaan linier ini. Sebagai ilustrasi, perhatikan tiga poin berikut: (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513). Perkirakan nilai fungsi pada = 9,2 setelah menentukan polinomial interpolasi kuadrat. polinomial kuadrat 2(x) = 0 + 1 + 2x2. Sistem persamaan linier yang dibuat dengan menambahkan tiga titik data ke polinomial adalah

$$a_0 + 8.0a_1 + 64.00a_2 = 2.0794$$

$$a_0 + 9.0a_1 + 81.00a_2 = 2.1972$$

$$a_0 + 9.5a_1 + 90.25a_2 = 2.2513$$

Hasil dari metode eliminasi Gaussian yang digunakan untuk menyelesaikan sistem persamaan adalah 0 = 0,6762, 1 = 0,2266, dan 2 = 0,0064. 2(x) = 0,6762 + 0,2266x 0,0064x2 adalah polinomial terinterpolasi yang melalui tiga titik. Nilai fungsi pada = 9,2 dapat diperkirakan sebagai berikut dengan menggunakan polinomial ini: 2(9,2) = 0,6762 + 0,2266(9,2) 0,0064(9,2)2 = 2,2192.

7. Interpolasi Bicubic

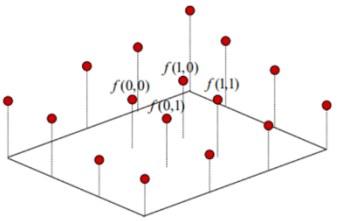
Sebuah pengembangan dari metode interpolasi linier dan kubik yang tercakup dalam kursus metode numerik dalam aljabar geometris, interpolasi bikubik adalah teknik untuk menginterpolasi data 2D yang biasanya digunakan dalam pembesaran gambar. Dengan mensimulasikan persamaan berikut, kita dapat menentukan persamaan interpolasi f(x,y) dengan matriks awal, seperti M:

Normalization: $f(0,0), f(1,0)$
$f(0,1), f(1,1)$

Model: $f(x,y) = \sum_{j=0}^{3}\sum_{i=0}^{3} a_{ij}x^i y^j$

$x = -1, 0, 1, 2$

Solve: a_{ij}

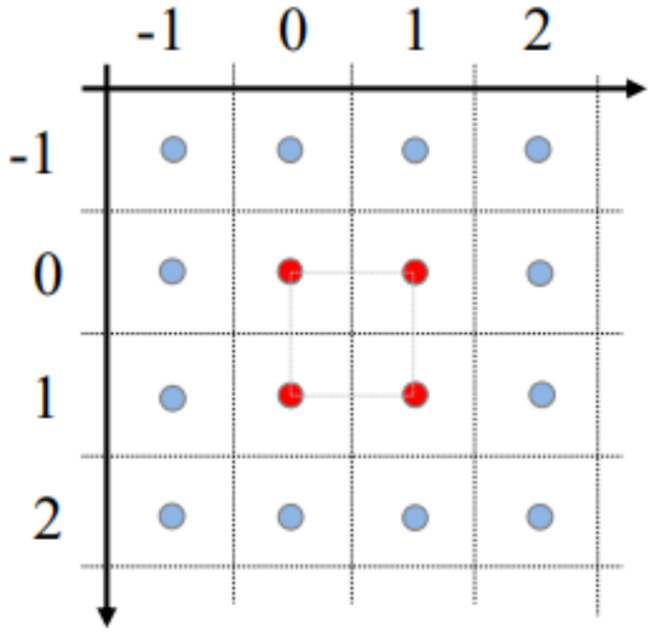


Matriks persamaan dibuat dengan mensubstitusi nilai yang diketahui dalam matriks 4 x 4 ke dalam ekspresi f(x,y):

$$y = Xa$$

$$\begin{bmatrix} f(-1,-1) \\ f(0,-1) \\ f(1,-1) \\ f(2,-1) \\ f(-1,0) \\ f(0,0) \\ f(1,0) \\ f(2,0) \\ f(-1,1) \\ f(0,1) \\ f(1,1) \\ f(2,1) \\ f(-1,2) \\ f(0,2) \\ f(1,2) \\ f(2,2) \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 2 & 4 & 8 & -1 & -2 & -4 & -8 & 1 & 2 & 4 & 8 & -1 & -2 & -4 & -8 \\ 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 4 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 1 & 2 & 4 & 8 & 1 & 2 & 4 & 8 & 1 & 2 & 4 & 8 \\ 1 & -1 & 1 & -1 & 2 & -2 & 2 & -2 & 4 & -4 & 4 & -4 & 8 & -8 & 8 & -8 \\ 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 8 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 4 & 4 & 4 & 4 & 8 & 8 & 8 & 8 \\ 1 & 2 & 4 & 8 & 2 & 4 & 8 & 16 & 4 & 8 & 16 & 32 & 8 & 16 & 32 & 64 \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \\ a_{01} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{02} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{03} \\ a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}$$

Koefisien aij dari persamaan sebelumnya f(x,y) adalah elemen dari matriks X. Misalnya, menurut persamaan xi * yj, elemen pada baris 4, kolom 10, yang merupakan koefisien a12, dapat ditemukan dengan menyelesaikan 21*(-1)2 = 2.
Vektor a dapat diperoleh dari persamaan (dengan menerapkan invers), dan kemudian digunakan sebagai nilai variabel dalam f. (x,y). sedemikian rupa sehingga fungsi interpolasi bikubik berbasis model dibuat. Tugas Anda adalah menyimpulkan persamaan f(x,y) dari matriks input 4 x 4, dan kemudian melakukan interpolasi berdasarkan f(a,b). kotak merah).



8. Regresi Lininer Berganda

Pendekatan yang berbeda untuk prediksi nilai dari interpolasi polinomial adalah regresi linier. Meskipun metode perhitungan regresi linier sederhana sudah ada, regresi linier berganda juga dapat dihitung menggunakan rumus umum untuk regresi linier:
$y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \epsilon_i$
Persamaan Estimasi Normal untuk Regresi Linier Berganda dapat diterapkan sebagai berikut untuk menentukan nilai masing-masing:

$$
\begin{aligned}
nb_0 + b_1\sum_{i=1}^{n} x_{1i} \quad &+ b_2\sum_{i=1}^{n} x_{2i} \quad + \cdots + b_k\sum_{i=1}^{n} x_{ki} \quad = \sum_{i=1}^{n} y_i \\
b_0\sum_{i=1}^{n} x_{1i} + b_1\sum_{i=1}^{n} x_{1i}^2 \quad &+ b_2\sum_{i=1}^{n} x_{1i}x_{2i} + \cdots + b_k\sum_{i=1}^{n} x_{1i}x_{ki} = \sum_{i=1}^{n} x_{1i}y_i \\
\vdots \quad &\quad \vdots \quad\quad\quad \vdots \quad\quad\quad \vdots \quad\quad \vdots \\
b_0\sum_{i=1}^{n} x_{ki} + b_1\sum_{i=1}^{n} x_{ki}x_{1i} &+ b_2\sum_{i=1}^{n} x_{ki}x_{2i} + \cdots + b_k\sum_{i=1}^{n} x_{ki}^2 \quad = \sum_{i=1}^{n} x_{ki}y_i
\end{aligned}
$$

BAB 3: IMPLEMENTASI PROGRAM

1. Main Class

Class utama tugas besar 1 yang signifikan ini disebut *Main Class*. Hanya ada satu prosedur, yang disebut main, di *Main Class*. Dalam proses ini, pengguna pertama-tama akan memberikan deskripsi singkat tentang kemampuan aplikasi sebelum melihat menu dari mana mereka dapat memilih opsi yang paling sesuai dengan kebutuhan mereka. Sistem Persamaan Linier, Determinan, Matriks Balikan, Interpolasi Polinomial, Interpolasi Bicubic, dan Regresi Linier Berganda adalah enam opsi menu. Setelah memilih menu, kelas lain akan masuk ke detail lebih lanjut tentang bagaimana setiap menu diimplementasikan.

Pilihan untuk melanjutkan menggunakan program untuk membuat perhitungan atau keluar ditampilkan kepada pengguna setiap kali mereka menyelesaikan perhitungan pada menu tertentu. Jika pengguna ingin melanjutkan perhitungan, mereka dapat kembali ke halaman menu dan memilih menu pilihan mereka sekali lagi. Jika pengguna memutuskan untuk keluar dari program, mereka akan segera melakukannya.

2. Matrix Class

Matrix *Class* berisi konstruktor dan semua fungsi-fungsi yang digunakan untuk kalkulator matriks ini. Seperti gauss(), gaussJordan(), adjoin(), kofaktor(), determinanKofaktor(), determinanOBE(), invers(), dan lain-lain.

Fungsi-fungsi untuk display, membaca input, membuat matriks, menyalin matriks, dan lain-lain juga berada di dalam *class* ini. Contoh-nya adalah readMatrixCLI(), readMatrixFILE(), displayMatrix(), createMatrix(), extendMatrix(), dan lain-lain.

Di *class* ini juga ada fungsi-fungsi operasi di dalam matriks, dan memeriksa matriks. Contoh-contoh dari fungsi ini adalah, isColsZero(), isRowsZero(), isSegitigaAtas(), addMulitplyRow(), swapRow(), divideRow(), dan lain-lain.

Untuk atribut, ada *matriks* yaitu *array of array of real*, *rows* yaitu variabel integer untuk menyimpan jumlah baris, *cols* yaitu variabel integer untuk menyimpan jumlah kolom, dan *epsilon* yaitu variabel bilangan riil (*double*) yang meyimpan konstanta yang sangat kecil untuk mengoreksi nilai negatif nol.

Berikut adalah definisi, input, dan output dari semua fungsi-fungsi yang berada di dalam *class* ini (tidak dalam urutan apapun).

public boolean isMatriksNol()

Fungsi untuk memeriksa apakah semua elemen matriks nol atau tidak, dan mengembalikkan *true* jika nol dan *false* jika tidak.

Procedure isMatriksNol() -> Boolean

{I.S Matriks belum di periksa apakah matriks nol atau tidak}

{F.S Matriks sudah di periksa apakah matriks nol atau tidak, dan mengembalikan *true* jika nol dan *false* jika tidak

public int firstZeroInRow(int i)

Fungsi untuk mengembalikan nilai indeks pertama elemen 0 pada baris.

Procedure firstZeroInRow(i: integer) -> int

{I.S. Matriks sudah terbentuk}

{F.S. Matriks sudah diperiksa, dan mengembalikkan indeks pertama elemen 0 pada baris}

public int firstNonZeroInRow(int i)

Fungsi untuk mengembalikan nilai indeks pertama elemen non 0 pada baris.

Procedure firstNonZeroInRow(i: integer) -> int

{I.S. Matriks sudah terbentuk}

{F.S. Matriks sudah diperiksa, dan mengembalikkan indeks pertama elemen non 0 pada baris}

public void divideRow(int row, Double n)

Fungsi untuk membagi baris dengan suatu bilangan n.

Procedure divideRow(row: integer, n: double)

{I.S. Matriks sudah terbentuk}

{F.S. Suatu barisan matriks berhasil dibagi oleh suatu bilangan n}

public void swapWithZeroRow(int i ,int j)

Fungsi untuk menukarkan suatu baris dengan baris yang berelemen nol.

Procedure swapWithZeroRow(i: integer, j: integer)

{I.S. Nilai i dan j sudah diketahui nol}

{F.S. Suatu baris di matriks berhasil di tukar dengan baris lainnya}

public void addMultiplyRow (int row1, int row2, double n)

Fungsi untuk menambahkan suatu baris awal *row* 1 dengan kelipatan n dari baris *row* 2

Procedure addMultiplyRow(row1: integer, row2: integer, n: double)

{I.S. Matriks sudah terbentuk}

{F.S. Suatu baris matriks berhasil ditambahkan oleh kelipatan n barisan lainnya}

public void gauss()

Fungsi untuk melakukan eliminasi gauss kepada suatu matriks

Procedure gauss()

{I.S. Matriks sudah dalam bentuk *augmented*}

{F.S. Matriks berhasil dieliminasi gauss}

public void gaussJordan()

Fungsi untuk melakukan eliminasi gauss-jordan kepada suatu matriks

Procedure gaussJordan()

{I.S. Matriks sudah dieliminasi gauss}

{F.S. Matriks berhasil dieliminasi gauss-jordan}

public void readMatrixCLI(int rows, int cols)

Fungsi untuk membaca masukan matriks dari input user

Procedure readMatrixCLI(rows: integer, cols:integer)

{I.S. Matriks dengan baris berjumlah *rows* dan kolom berjumlah *cols* kosong dan belum terdefinisi}

{F.S. Matriks sudah terdefinisi dengan nilai-nilai yang di input oleh user.}

public void readMatrixFILE(String FileName)

Fungsi untuk membaca matriks dari file .txt

Procedure readMatrixFILE(FileName: string)

{I.S. Matriks kosong}

{F.S. Matriks sudah terbentuk dan berhasil di baca dari file .txt}

public void displayMatrix()

Fungsi untuk mengeluarkan matriks ke layar user.

Procedure displayMatrix()

{I.S. Matriks sudah terbentuk dan sudah terdefinisi}

{F.S. Matriks di cetak ke layar untuk di lihat user}

public void createMatrix(int rows, int cols)

Fungsi untuk membentuk matriks dengan masukan *rows* dan *cols*.

Procedure createMatrix(rows: integer, cols: integer)

{I.S. Jumlah *rows* dan *cols* sudah terdefinsi}

{F.S. Matriks terbentuk dengan baris berjumlah *rows* dan kolom berjumlah *cols*}

public Matrix copyMatrix()

Fungsi untuk menyalin sebuah matriks ke matriks baru

Procedure copyMatrix() -> Matrix

{I.S. Suatu matriks sudah terbentuk dan terdefinisi}

{F.S. Matriks baru berhasil dibuat dengan elemen-elemen yang sama dengan matriks tersebut}

public Matrix extendMatrix(int nRows, int nCols)

Fungsi untuk memperbesar ukuran matriks

Procedure extendMatrix(nRows: integer, nCols: integer) -> Matrix

{I.S. Suatu matriks sudah terbentuk dan terdefinisi}

{F.S. Matriks tersebut sudah diperbesar dengan elemen-elemen yang lama}

public boolean isRowsZero(int i)

Fungsi untuk memeriksa apakah sebuah baris berisi nol atau tidak

Procedure isRowsZero(i: integer) -> boolean

{I.S. Matriks sudah terdefinisi}

{F.S. Mengembalikkan *true* jika ada nol di sebuah baris yang ada di matriks tersebut dan mengembalikkan *false* jika tidak ada nol}

public boolean isColsZero(int j)

Fungsi untuk memeriksa apakah sebuah kolom berisi nol atau tidak

Procedure isColsZero(i: integer) -> boolean

{I.S. Matriks sudah terdefinisi}

{F.S. Mengembalikkan *true* jika ada nol di sebuah kolom yang ada di matriks tersebut dan mengembalikkan *false* jika tidak ada nol}

public boolean isAtLeastRowZero()

Fungsi untuk memeriksa apakah ada nilai 0 dalam suatu baris

public boolean isAtleastRowZero() -> boolean

{I.S. Matriks sudah terdefinisi}

{F.S. Mengembalikkan *true* jika ada nol di sebuah baris yang ada di matriks tersebut dan mengembalikkan *false* jika tidak ada nol}

public boolean isAtLeastColZero()

---

public boolean isAtleastColZero() -> boolean

{I.S. Matriks sudah terdefinisi}

{F.S. Mengembalikkan *true* jika ada nol di sebuah kolom yang ada di matriks tersebut dan mengembalikkan *false* jika tidak ada nol}

public void corrZero()

Fungsi untuk memperbaiki nilai negatif nol

Procedure corrZero()

{I.S. Matriks sudah terdefinisi}

{F.S. Elememn matriks yang bernilai negatif nol berhasil diubah menjadi nol}

public boolean isSegitigaAtas()

Fungsi untuk memeriksa apakah suatu matriks memiliki sifat segitiga atas

Procedure isSegitigaAtas() -> boolean

{I.S. Matriks sudah terdefinisi}

{{F.S. Mengembalikkan *true* jika matriks bersifat segitiga atas dan mengembalikkan *false* jika tidak}

public void reduceMatrix(double M[][], int i, int j)

Fungsi untuk mengurangkan ukuran suatu matriks

Procedure extendMatrix(M: array of array of real, i: integer, j: integer)

{I.S. Suatu matriks sudah terbentuk dan terdefinisi}

{F.S. Matriks tersebut berhasil dikurangkan ukurannya}

boolean isSquare()

Fungsi untuk memeriksa apakah suatu matriks berukuran n x n

Procedure isSquare() -> boolean
{I.S. baris dan kolom sudah terdefinisi}

{F.S. mengembalikkan *true* jika nilai baris dan kolom sama dan *false* jika nilainya beda}

int jumlahSolusi()

Fungsi untuk menghitung jumlah solusi dari sebuah SPL

Procedure jumlahSolusi() -> integer

{I.S. Matriks sudah terdefinisi}

{F.S. Mengembalikkan satu dari tiga integer, nol apabila tidak ada solusi, satu apabila solusi unik, dan dua apabila solusi tak hingga}

public void swapRow(int row1, int row2)

Fungsi untuk menukar 2 baris tertentu

Procedure swapRow (row1: integer, row2: integer)

{I.S. Matriks sudah terdefinisi}

[F.S. Dua baris tertentu dari matriks tersebut sudah ditukar}

public Matrix multiplyMatrix(Matrix m1, Matrix m2)

Fungsi untuk mengalikan 2 buah matriks

Procedure multiplyMatrix(m1: Matrix, m2: Matrix) -> Matrix

{I.S. Dua buah matriks sudah terdefinisi}

{F.S. Dua matriks tersebut berhasil di kalikan dan fungsi mengembalikannya}

public void OBE(int a)

Fungsi untuk melakukan Operasi Baris Elementer (OBE)

Procedure OBE(a: integer)

{I.S. Matriks sudah terdefinisi dan baris pertamanya bukan baris nol, atau baris pertama merupakan baris dengan nilai non nol terkiri yang ada di matriks}

{F.S. OBE berhasil dilakukan di matriks tersebut}

public int lastZeroInRow(int i)

Fungsi untuk mengembalikkan indeks bilangan nol terakhir dalam suatu baris

Procedure lastZeroInRow(i: integer) -> integer

{I.S. Matriks sudah terdefinisi}

{F.S. Mengembalikkan indeks yang menunjukkan bilangan nol terakhir dalam suatu baris di matriks tersebut]

public boolean isAllRowBelowZero(int i)

Fungsi untuk memeriksa nilai nol di bawah suatu elemen indeks

Procedure isAllRowBelowZero(i: integer) -> boolean

{I.S. Matriks dan nilai i sudah terdefinisi}

{F.S. Mengembalikkan *true* jika seluruh elemen baris di bawah suatu baris adalah nol dan mengembalikkan false jika tidak}

public void swapDown(int i)

Fungsi untuk menukar suatu baris di dalam matriks dengan baris di bawah nya

Procedure swapDown(i: integer)

{I.S. Matriks dan nilai i sudah terdefinisi}

{F.S. Berhasil menukarkan baris matriks dengan baris dibawahnya}

public boolean isRowAugZero(int i)

Fungsi untuk memeriksa apakah suatu baris berisi dengan nilai nol

Procedure isRowAugZero(i: integer) -> boolean

{I.S. Matriks dan nilai i sudah terdefinisi}

{F.S. Mengembalikkan *true* jika semua elemen dalam baris adalah nol dan false jika tidak}

public double inverseOBE()

Fungsi untuk mencari balikan dari suatu matriks menggunakan OBE

Procedure inverseOBE() -> real

{I.S. Matriks sudah terdefinisi dan matriks adalah persegi}

{F.S. Hasil dari matriks balikan berhasil didapatkan dan dikembalikan}

public void inverseSPL()

Fungsi untuk mencari solusi SPL dengan matriks balikan

Procedure inverserSPL()

{I.S. Matriks sudah terdefinisi dan sudah dibalik}

{F.S. Solusi SPL dari matriks tersebut berhasil didapatkan}

public boolean isInverseUrut()

Fungsi untuk memerika apakah suatu matriks balikan berurutan

Procedure isInverseUrut()

{I.S. Matriks sudah terdefinisi}

{F.S. Mengembalikkan *true* jika sudah berurutan dan *false* jika tidak}

public boolean OBEdet(int a, boolean evenSwap)

Fungsi untuk mencari determinan matrix menggunakan OBE

Procedure OBEdet(a: integer, evenSwap: boolean) -> boolean

{I.S. Matriks sudah terdefinisi dan baris pertama matriks bukan baris nol, atau baris pertama merupakan baris dengan nilai non nol terkiri yang ada di matriks}

{F.S. evenSwap dikembalikan}

public double determinanOBE()

Fungsi untuk mencari determinan matrix menggunakan OBE

Procedure determinanOBE() -> real

{I.S. Matriks sudah terdefinisi dan berbentuk persegi}

{F.S. Determinan berhasil dihitung dan dikembalikan}

void kofaktor()

Fungsi untuk melakukan operasi kofaktor

Procedure kofaktor()

{I.S. Matriks sudah terdefinisi}

{F.S. Operasi kofaktor berhasil dilakukan di matriks tersebut}

public double determinanKofaktor()

Fungsi untuk mencari nilai determinan dengan kofaktor

Procedure determinanKofaktor() -> real

{I.S. Matriks sudah terdefinisi dan sudah dilakukan operasi kofaktor pada matriks tersebut}

{F.S. Determinan berhasil dihitung dan determinan dikembalikkan}

void transpose()

Fungsi untuk melakukan transpose kepada matriks

Procedure transpose()

{I.S. Matriks sudah terdefinisi}

{F.S. Semua elemen matriks [i][j] berhasil ditukar dengan elemen matriks [j][i]}

void adjoin()

Fungsi untuk mencari matrix adjoin

Procedure adjoin()

{I.S. Matriks sudah terdefinisi}

{F.S. Matriks adjoin berhasil ditemukan}

void copyMatrixInverse(double M[][])

Fungsi untuk menyalin matrix invers

Procedure copyMatrixInverse(M: array of array of real)

{I.S. Suatu matriks sudah terbentuk dan terdefinisi}

{F.S. Matriks baru berhasil dibuat dengan elemen-elemen yang sama dengan matriks tersebut}

void invers()

Fungsi untuk melakukan invers pada suatu matriks

Procedure invers()

{I.S. Matriks sudah terdefinisi}

{F.S. Matriks berhasil di invers}

String Cramer(double m[][])

Fungsi untuk melakukan kaidah cramer pada matriks

Procedure Cramer(m: array of array of real)

{I.S. Matriks sudah terdefinisi}

{F.S. Kaidah Cramer berhasil dilakukan di matriks tersebut dan mengembalikkan output-nya)

String multiplyInvers(double M[][])

Fungsi untuk mengalikan matriks A invers dengan matriks B

Procedure multiplyInvers(M: array of array of real) -> string

{I.S. Matriks sudah terdefinsi)

(F.S. Perkalian antara kedua matriks berhasil dilakukan dan mengembalikkan string}

BAB 4: EKSPERIMEN

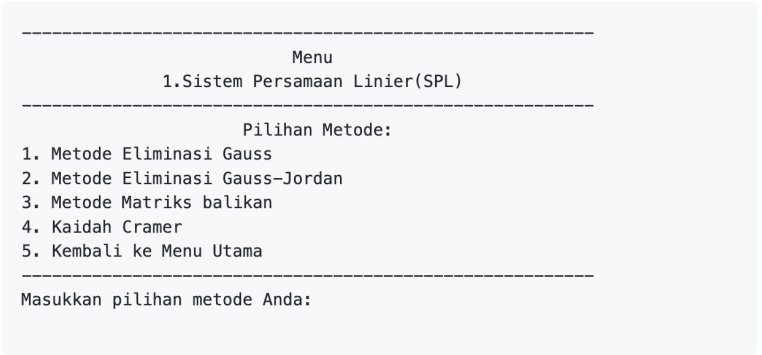
1. Menu Utama

Setelah program dijalankan, CLI akan menampilkan menu utama sebagai berikut:

Masukkan SubMenu yang hendak dijalankan berupa sebuah masukan angka, antara 1-6 (inklusif).

2. Sistem Persamaan Linier

Apabila pengguna memasukkan angka 1 (Sistem Persamaan Linier), maka akan muncul pilihan metode yang dapat digunakan sesuai kebutuhan pengguna. Masing-masing pilihan berkaitan dengan metode penyelesaian Sistem Persamaan Linier(SPL).



Pilihan sumber

Pada tiap pilihan metode yang telah ditentukan oleh pengguna, akan memunculkan sebuah menu lanjutan berupa pilihan sumber input yang dapat ditentukan sendiri oleh pengguna. pengguna dapat memilih untuk langsung memasukkan data olahan via CLI/keyboard atau melalui file.txt

```
------------------------------------------------------------
                          Metode
                1. Metode Eliminasi Gauss
------------------------------------------------------------
                     Pilihan Sumber:
------------------------------------------------------------
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode
------------------------------------------------------------
Masukkan pilihan sumber Anda:
```

Metode Eliminasi Gauss

Mula-mula pengguna akan diminta untuk memasukkan matriks augmented dari Sistem Persamaan Linear yang hendak diselesaikan. Misal, sebuah sistem persamaan linier terdiri dari tiga persamaan di bawah ini:

$x + y + z = 0;$
$2x + 3y + z = 1;$
$3x + y + 2z = 1;$

Maka matriks augmented-nya adalah
1 1 1 0
2 3 1 1
3 1 2 1

Untuk masukan via CLI, pengguna akan diminta untuk memasukkan nilai tiap elemen baris dan kolom pada matriks augmented, program yang akan muncul sebagai berikut (gambar berada di halaman selanjutnya):

```
------------------------------------------------------------
                          Sumber
                  1. Masukan dari CLI
------------------------------------------------------------
Masukkan jumlah baris:
3
Masukkan jumlah kolom:
4
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
1
Masukkan nilai untuk baris ke-1 kolom ke-2:
1
Masukkan nilai untuk baris ke-1 kolom ke-3:
1
Masukkan nilai untuk baris ke-1 kolom ke-4:
0
Masukkan nilai untuk baris ke-2 kolom ke-1:
2
Masukkan nilai untuk baris ke-2 kolom ke-2:
3
Masukkan nilai untuk baris ke-2 kolom ke-3:
1
Masukkan nilai untuk baris ke-2 kolom ke-4:
1
Masukkan nilai untuk baris ke-3 kolom ke-1:
3
Masukkan nilai untuk baris ke-3 kolom ke-2:
1
Masukkan nilai untuk baris ke-3 kolom ke-3:
2
Masukkan nilai untuk baris ke-3 kolom ke-4:
1
Matriks berhasil dibaca.
------------------------------------------------------------
```

Sementara untuk masukan via file.txt, pengguna akan diminta untuk memasukkan nama file (*.txt) yang telah disimpan dalam folder test/input.

```
------------------------------------------------------------
                          Sumber
                2. Masukan dari file .txt
------------------------------------------------------------
Masukkan nama file (.txt) dalam folder test:
```

Misalkan nama file-nya adalah inputSPLGauss.txt, ketika nama file telah benar, maka akan muncul tampilan sebagai berikut:

```
------------------------------------------------------------
                          Sumber
                2. Masukan dari file .txt
------------------------------------------------------------
Masukkan nama file (.txt) dalam folder test:
inputSPLGauss.txt
Matriks (file:inputSPLGauss.txt) berhasil dibaca.
------------------------------------------------------------
```

Program akan menjalankan eliminasi Gauss pada matriks augmented tersebut. Proses eliminasi akan berjalan seperti di bawah ini:

Matriks yang dibaca:
1.000000 1.000000 1.000000 0.000000
2.000000 3.000000 1.000000 1.000000
3.000000 1.000000 2.000000 1.000000

Kurangi baris ke-2 dengan 2.000000 kali baris ke-1
1.000000 1.000000 1.000000 0.000000
0.000000 1.000000 -1.000000 1.000000
3.000000 1.000000 2.000000 1.000000

Kurangi baris ke-3 dengan 3.000000 kali baris ke-1
1.000000 1.000000 1.000000 0.000000
0.000000 1.000000 -1.000000 1.000000
0.000000 -2.000000 -1.000000 1.000000

Kurangi baris ke-3 dengan -2.000000 kali baris ke-2
1.000000 1.000000 1.000000 0.000000
0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 -3.000000 3.000000

Bagi setiap elemen di baris ke-1 dengan 1.00
1.000000 1.000000 1.000000 0.000000
0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 -3.000000 3.000000

Bagi setiap elemen di baris ke-2 dengan 1.00
1.000000 1.000000 1.000000 0.000000
0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 -3.000000 3.000000

Bagi setiap elemen di baris ke-3 dengan -3.00
1.000000 1.000000 1.000000 0.000000
0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 1.000000 -1.000000

Di akhir proses eliminasi Gauss, akan dihasilkan sebuah matriks eselon baris. Lalu program akan melakukan substitusi balik dan menghasilkan solusi seperti berikut:

```
------------------------------------------------------------
Dengan metode Eliminasi Gauss, diperoleh solusi SPL:
X1=1.00 X2=0.00 X3=-1.00
------------------------------------------------------------
```

Metode Eliminasi Gauss-Jordan

Pada metode eliminasi Gauss-Jordan, langkah-langkah yang dilakukan sama dengan metode eliminasi gauss. Namun, pada akhir proses eliminasi akan dihasilkan matriks eselon baris tereduksi. Berikut ini adalah simulasi proses penyelesaian Sistem Persamaan Linier dengan metode Gauss-Jordan:

Matriks yang dibaca:
1.000000 1.000000 2.000000 4.000000
2.000000 -1.000000 1.000000 2.000000
1.000000 2.000000 3.000000 7.000000

Kurangi baris ke-2 dengan 2.000000 kali baris ke-1
1.000000 1.000000 2.000000 4.000000
0.000000 -3.000000 -3.000000 -6.000000
1.000000 2.000000 3.000000 7.000000

Kurangi baris ke-3 dengan 1.000000 kali baris ke-1
1.000000 1.000000 2.000000 4.000000
0.000000 -3.000000 -3.000000 -6.000000
0.000000 1.000000 1.000000 3.000000

Kurangi baris ke-3 dengan -0.333333 kali baris ke-2
1.000000 1.000000 2.000000 4.000000
0.000000 -3.000000 -3.000000 -6.000000
0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-1 dengan 1.00
1.000000 1.000000 2.000000 4.000000
0.000000 -3.000000 -3.000000 -6.000000
0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-2 dengan -3.00
1.000000 1.000000 2.000000 4.000000
0.000000 1.000000 1.000000 2.000000
0.000000 0.000000 0.000000 1.000000

Kurangi baris ke-1 dengan 1.000000 kali baris ke-2
1.000000 0.000000 1.000000 2.000000
0.000000 1.000000 1.000000 2.000000
0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-1 dengan 1.00
1.000000 0.000000 1.000000 2.000000
0.000000 1.000000 1.000000 2.000000
0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-2 dengan 1.00

1.000000 0.000000 1.000000 2.000000
0.000000 1.000000 1.000000 2.000000
0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-1 dengan 1.00
1.000000 0.000000 1.000000 2.000000
0.000000 1.000000 1.000000 2.000000
0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-2 dengan 1.00
1.000000 0.000000 1.000000 2.000000
0.000000 1.000000 1.000000 2.000000
0.000000 0.000000 0.000000 1.000000

Di akhir proses eliminasi Gauss-Jordan, akan dihasilkan sebuah matriks eselon baris tereduksi dan sebuah solusi SPL, sebagai berikut:

```
------------------------------------------------------------
Dengan metode Eliminasi Gauss-Jordan, diperoleh solusi SPL:
SPL tidak memiliki solusi
------------------------------------------------------------
```

Apabila sebuah sistem persamaan linier memiliki solusi tak hingga/banyak, maka program akan menuliskan solusinya, sebagai berikut:

```
------------------------------------------------------------
Dengan metode Eliminasi Gauss-Jordan, diperoleh solusi SPL:
X1=-2.00t+3.00s+7.00 X2=1.00t-4.00s+5.00 X3=t X4=s
------------------------------------------------------------
```

Metode Matriks Balikan

Dengan metode ini, matriks akan disajikan dalam model Ax=B. Misal, sebuah sistem persamaan linier terdiri dari empat persamaan di bawah ini:

a + b - c - d = 1;
2a + 5b- 7c- 5d = 2;
2a - b + c + 3d = 4;
5a + 2b - 4c + 2d = 6;

Untuk masukan via CLI, pengguna akan diminta untuk memasukkan nilai tiap elemen baris dan kolom pada matriks A terlebih dahulu, kemudian mengisi matriks B, program yang akan muncul sebagai berikut:

Sumber

1. Masukan dari CLI

```
---------------------------------------------------------
Masukkan jumlah baris:
4
Masukkan jumlah kolom:
4
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
1
Masukkan nilai untuk baris ke-1 kolom ke-2:
1
Masukkan nilai untuk baris ke-1 kolom ke-3:
-1
Masukkan nilai untuk baris ke-1 kolom ke-4:
-1
Masukkan nilai untuk baris ke-2 kolom ke-1:
2
Masukkan nilai untuk baris ke-2 kolom ke-2:
5
Masukkan nilai untuk baris ke-2 kolom ke-3:
-7
Masukkan nilai untuk baris ke-2 kolom ke-4:
-5
Masukkan nilai untuk baris ke-3 kolom ke-1:
2
Masukkan nilai untuk baris ke-3 kolom ke-2:
-1
Masukkan nilai untuk baris ke-3 kolom ke-3:
1
Masukkan nilai untuk baris ke-3 kolom ke-4:
3
Masukkan nilai untuk baris ke-4 kolom ke-1:
5
Masukkan nilai untuk baris ke-4 kolom ke-2:
2
Masukkan nilai untuk baris ke-4 kolom ke-3:
-4
Masukkan nilai untuk baris ke-4 kolom ke-4:
2
Matriks berhasil dibaca.
---------------------------------------------------------
```

Namun, karena pada studi kasus yang dipilih, nilai determinan Matriks A bernilai 0, maka pengguna tidak akan diminta memasukkan nilai elemen matriks B (Matriks A tidak memiliki Matriks Balikan). Program akan mengeluarkan tampilan sebagai berikut:

```
------------------------------------------------------------
Matriks yang dibaca:
1.000000 1.000000 -1.000000 -1.000000
2.000000 5.000000 -7.000000 -5.000000
2.000000 -1.000000 1.000000 3.000000
5.000000 2.000000 -4.000000 2.000000
------------------------------------------------------------
Oleh karena determinan matriks = 0, maka
Metode Matriks Balikan tidak dapat diterapkan
------------------------------------------------------------
```

Apabila masukan via file (*.txt), maka penulisan matriks A dan B harus dilakukan secara augmented seperti yang dilakukan pada metode Gauss dan Gauss-Jordan.

Metode Cramer

Dengan metode ini, matriks akan disajikan dalam model Ax=B. Misal, sebuah sistem persamaan linier terdiri dari empat persamaan di bawah ini:

w + x - y - z = 1;
2w + 5x - 7y - 5z = -2;
2w - x + y + 3z = 4;
5w + 2x - 4y + 2z = 6;

Untuk masukan via CLI, pengguna akan diminta untuk memasukkan nilai tiap elemen baris dan kolom pada matriks A terlebih dahulu, kemudian mengisi matriks B, program yang akan muncul sebagai berikut:

```
------------------------------------------------------------
Matriks yang dibaca:
1.000000 -1.000000 0.000000 0.000000 1.000000
1.000000 1.000000 0.000000 -3.000000 0.000000
2.000000 -1.000000 0.000000 1.000000 -1.000000
-1.000000 2.000000 0.000000 -2.000000 -1.000000
------------------------------------------------------------
Oleh karena matriks bukan persegi, maka
Metode Kaidah Cramer tidak dapat diterapkan
------------------------------------------------------------
```

Apabila masukan via file (*.txt), maka penulisan matriks A dan B harus dilakukan secara augmented seperti yang dilakukan pada metode Gauss dan Gauss-Jordan.

3. Determinan Matriks

Apabila pengguna memasukkan angka 2 (Determinan Matriks), maka akan muncul pilihan metode yang dapat digunakan sesuai kebutuhan pengguna. Masing-masing pilihan berkaitan dengan metode penyelesaian nilai determinan sebuah matriks.

```
------------------------------------------------------------
                          Menu
                       2.Determinan
------------------------------------------------------------
                     Pilihan Metode:
1. Metode Eliminasi Gauss
2. Metode Ekspansi Kofaktor
3. Kembali ke Menu Utama
------------------------------------------------------------
Masukkan pilihan metode Anda:
```

Metode Eliminasi Gauss

Metode perhitungan determinan ini sebenarnya menggunakan operasi baris elementer (metode Gauss, tetapi diagonal utama matriks tidak perlu bernilai 1). Matriks akan dioperasikan sehingga berbentuk segitiga atas (seluruh elemen matriks di bawah diagonal utama bernilai 0). Perhitungan determinan akan dilakukan dengan mengalikan seluruh elemen diagonal utama matriks hasil OBE.

Pada metode Eliminasi Gauss, untuk masukan via CLI, pengguna akan diminta untuk memasukkan nilai tiap elemen baris dan kolom pada matriks augmented (matriks harus berupa matriks persegi, ukuran baris = ukuran kolom), program yang akan muncul sebagai berikut:

```
------------------------------------------------------
                    Sumber
                1. Masukan dari CLI
------------------------------------------------------
Masukkan ukuran matriks persegi (NxN) N:
3
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
4
Masukkan nilai untuk baris ke-1 kolom ke-2:
2
Masukkan nilai untuk baris ke-1 kolom ke-3:
8
Masukkan nilai untuk baris ke-2 kolom ke-1:
2
Masukkan nilai untuk baris ke-2 kolom ke-2:
1
Masukkan nilai untuk baris ke-2 kolom ke-3:
5
Masukkan nilai untuk baris ke-3 kolom ke-1:
3
Masukkan nilai untuk baris ke-3 kolom ke-2:
2
Masukkan nilai untuk baris ke-3 kolom ke-3:
4
```

Matriks berhasil dibaca.

Di akhir proses eliminasi Gauss, akan dihasilkan nilai determinan matriks, sebagai berikut:

```
------------------------------------------------------------
Matriks yang dibaca:
4.000000 2.000000 8.000000
2.000000 1.000000 5.000000
3.000000 2.000000 4.000000
------------------------------------------------------------
Dengan metode Eliminasi Gauss,
diperoleh nilai determinan:
-2.0
------------------------------------------------------------
```

Apabila masukan via file (*.txt), maka penulisan matriks A harus dilakukan secara augmented (dengan syarat, matriks A merupakan matriks persegi, ukuran baris = ukuran kolom) seperti yang dilakukan pada metode-metode sebelumnya.

Metode Kofaktor

Metode perhitungan determinan ini menggunakan minor dan kofaktor pada matriks masukan.Perhitungan determinan akan menggunakan elemen pada baris pertama yang masing-masing akan dikalikan dengan kofaktor-nya. Masing-masing kofaktor akan dihitung dengan mencari minornya yang kemudian determinannya akan dihitung pula menggunakan fungsi determinan metode kofaktor (rekursi).

Pada metode Kofaktor, untuk masukan via CLI, pengguna akan diminta untuk memasukkan nilai tiap elemen baris dan kolom pada matriks augmented (matriks harus berupa matriks persegi, ukuran baris = ukuran kolom), program yang akan muncul sebagai berikut:

Sumber
1. Masukan dari CLI

Masukkan ukuran matriks persegi (NxN) N:
3
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
2
Masukkan nilai untuk baris ke-1 kolom ke-2:
0
Masukkan nilai untuk baris ke-1 kolom ke-3:
1
Masukkan nilai untuk baris ke-2 kolom ke-1:
3

Masukkan nilai untuk baris ke-2 kolom ke-2:
1
Masukkan nilai untuk baris ke-2 kolom ke-3:
2
Masukkan nilai untuk baris ke-3 kolom ke-1:
4
Masukkan nilai untuk baris ke-3 kolom ke-2:
2
Masukkan nilai untuk baris ke-3 kolom ke-3:
3
Matriks berhasil dibaca.

Di akhir proses metode kofaktor, akan dihasilkan nilai determinan matriks, sebagai berikut:

```
------------------------------------------------------------
Matriks yang dibaca:
2.000000 0.000000 1.000000
3.000000 1.000000 2.000000
4.000000 2.000000 3.000000
------------------------------------------------------------
Dengan metode Ekspansi Kofaktor,
diperoleh nilai determinan:
0.0
------------------------------------------------------------
```

Apabila masukan via file (*.txt), maka penulisan matriks A harus dilakukan secara augmented (dengan syarat, matriks A merupakan matriks persegi, ukuran baris = ukuran kolom) seperti yang dilakukan pada metode-metode sebelumnya.

4. Matriks Balikan

Apabila pengguna memasukkan angka 3 (Matriks Balikan), maka akan muncul pilihan metode yang dapat digunakan sesuai kebutuhan pengguna. Masing-masing pilihan berkaitan dengan metode penyelesaian sebuah matriks balikan dari masukan pengguna.

```
------------------------------------------------------------
                          Menu
                  3. Matriks Balikan
------------------------------------------------------------
                     Pilihan Metode:
1. Metode Eliminasi Gauss
2. Metode Ekspansi Kofaktor
3. Kembali ke Menu Utama
------------------------------------------------------------
Masukkan pilihan metode Anda:
```

Metode Eliminasi Gauss

Metode perhitungan determinan ini sebenarnya menggunakan operasi baris elementer (metode Gauss, tetapi diagonal utama matriks tidak perlu bernilai 1). Matriks akan dioperasikan beriringan dengan matriks identitas (matriks dengan elemen diagonal utama bernilai 1, dan elemen lainnya bernilai 0) sehingga akan terbentuk matriks baru yang merupakan matriks balikan dari matriks awal.

[A|I] -> [I|A^-1]

Misalkan untuk studi kasus matriks:
4 2 -1
0 2 3
-1 1 5

Pada metode Eliminasi Gauss, untuk masukan via CLI, pengguna akan diminta untuk memasukkan nilai tiap elemen baris dan kolom pada matriks augmented (matriks harus berupa matriks persegi, ukuran baris = ukuran kolom), program yang akan muncul sebagai berikut:

```
---------------------------------------------------------
                    Sumber
                1. Masukan dari CLI
---------------------------------------------------------
Masukkan ukuran matriks persegi (NxN) N:
3
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
4
Masukkan nilai untuk baris ke-1 kolom ke-2:
2
Masukkan nilai untuk baris ke-1 kolom ke-3:
-1
Masukkan nilai untuk baris ke-2 kolom ke-1:
0
Masukkan nilai untuk baris ke-2 kolom ke-2:
2
Masukkan nilai untuk baris ke-2 kolom ke-3:
-3
Masukkan nilai untuk baris ke-3 kolom ke-1:
-1
Masukkan nilai untuk baris ke-3 kolom ke-2:
1
Masukkan nilai untuk baris ke-3 kolom ke-3:
5
Matriks berhasil dibaca.
---------------------------------------------------------
```

Berikut ini adalah simulasi proses penyelesaian matriks balikan dengan metode Gauss:

```
---------------------------------------------------------
Matriks yang dibaca:
4.000000 2.000000 -1.000000
0.000000 2.000000 -3.000000
-1.000000 1.000000 5.000000

4.000000 2.000000 -1.000000 1.000000 0.000000 0.000000
0.000000 2.000000 -3.000000 0.000000 1.000000 0.000000
-1.000000 1.000000 5.000000 0.000000 0.000000 1.000000

Kurangi baris ke-3 dengan -0.250000 kali baris ke-1
4.000000 2.000000 -1.000000 1.000000 0.000000 0.000000
0.000000 2.000000 -3.000000 0.000000 1.000000 0.000000
0.000000 1.500000 4.750000 0.250000 0.000000 1.000000

Kurangi baris ke-3 dengan 0.750000 kali baris ke-2
4.000000 2.000000 -1.000000 1.000000 0.000000 0.000000
0.000000 2.000000 -3.000000 0.000000 1.000000 0.000000
0.000000 0.000000 7.000000 0.250000 -0.750000 1.000000

Bagi setiap elemen di baris ke-1 dengan 4.00
1.000000 0.500000 -0.250000 0.250000 0.000000 0.000000
0.000000 2.000000 -3.000000 0.000000 1.000000 0.000000
0.000000 0.000000 7.000000 0.250000 -0.750000 1.000000

Bagi setiap elemen di baris ke-2 dengan 2.00
1.000000 0.500000 -0.250000 0.250000 0.000000 0.000000
0.000000 1.000000 -1.500000 0.000000 0.500000 0.000000
0.000000 0.000000 7.000000 0.250000 -0.750000 1.000000

Bagi setiap elemen di baris ke-3 dengan 7.00
1.000000 0.500000 -0.250000 0.250000 0.000000 0.000000
0.000000 1.000000 -1.500000 0.000000 0.500000 0.000000
0.000000 0.000000 1.000000 0.035714 -0.107143 0.142857

Kurangi baris ke-2 dengan -1.500000 kali baris ke-3
1.000000 0.500000 -0.250000 0.250000 0.000000 0.000000
0.000000 1.000000 0.000000 0.053571 0.339286 0.214286
0.000000 0.000000 1.000000 0.035714 -0.107143 0.142857

Kurangi baris ke-1 dengan -0.250000 kali baris ke-3
1.000000 0.500000 0.000000 0.258929 -0.026786 0.035714
0.000000 1.000000 0.000000 0.053571 0.339286 0.214286
0.000000 0.000000 1.000000 0.035714 -0.107143 0.142857
```

Kurangi baris ke-1 dengan 0.500000 kali baris ke-2
1.000000 0.000000 0.000000 0.232143 -0.196429 -0.071429
0.000000 1.000000 0.000000 0.053571 0.339286 0.214286
0.000000 0.000000 1.000000 0.035714 -0.107143 0.142857

Di akhir proses eliminasi Gauss, akan dihasilkan matriks balikan sebagai berikut:

```
------------------------------------------------------------
Dengan metode Eliminasi Gauss,
diperoleh matriks balikan:
0.232143 -0.196429 -0.071429
0.053571 0.339286 0.214286
0.035714 -0.107143 0.142857
------------------------------------------------------------
```

Apabila masukan via file (*.txt), maka penulisan matriks A harus dilakukan secara augmented (dengan syarat, matriks A merupakan matriks persegi, ukuran baris = ukuran kolom) seperti yang dilakukan pada metode-metode sebelumnya.

Metode Ekspansi Kofaktor

Metode perhitungan determinan ini memanfaatkan nilai determinan suatu matriks dan matriks adjoin (matriks transpose dari sebuah matriks kofaktor). Metode ini berdasarkan pada rumus:

[A^-1] = 1/det(A) * [adj(A)]

Misalkan untuk studi kasus matriks:
3 3 0 5
2 2 0 -2
4 1 -3 0
2 10 3 2

Pada metode ekspansi kofaktor, untuk masukan via CLI, pengguna akan diminta untuk memasukkan nilai tiap elemen baris dan kolom pada matriks augmented (matriks harus berupa matriks persegi, ukuran baris = ukuran kolom), program yang akan muncul sebagai berikut:

Sumber
1. Masukan dari CLI

Masukkan ukuran matriks persegi (NxN) N:
4
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
3

Masukkan nilai untuk baris ke-1 kolom ke-2:
3
Masukkan nilai untuk baris ke-1 kolom ke-3:
0
Masukkan nilai untuk baris ke-1 kolom ke-4:
5
Masukkan nilai untuk baris ke-2 kolom ke-1:
2
Masukkan nilai untuk baris ke-2 kolom ke-2:
2
Masukkan nilai untuk baris ke-2 kolom ke-3:
0
Masukkan nilai untuk baris ke-2 kolom ke-4:
-2
Masukkan nilai untuk baris ke-3 kolom ke-1:
4
Masukkan nilai untuk baris ke-3 kolom ke-2:
1
Masukkan nilai untuk baris ke-3 kolom ke-3:
-3
Masukkan nilai untuk baris ke-3 kolom ke-4:
0
Masukkan nilai untuk baris ke-4 kolom ke-1:
2
Masukkan nilai untuk baris ke-4 kolom ke-2:
10
Masukkan nilai untuk baris ke-4 kolom ke-3:
3
Masukkan nilai untuk baris ke-4 kolom ke-4:
2
Matriks berhasil dibaca.

Di akhir proses ekspansi kofaktor, akan dihasilkan matriks balikan sebagai berikut:

```
------------------------------------------------------------
Matriks yang dibaca:
3.000000 3.000000 0.000000 5.000000
2.000000 2.000000 0.000000 -2.000000
4.000000 1.000000 -3.000000 0.000000
2.000000 10.000000 3.000000 2.000000
------------------------------------------------------------
Dengan metode Ekspansi Kofaktor,
diperoleh matriks balikannya:
0.325000 0.612500 -0.200000 -0.200000
-0.200000 -0.300000 0.200000 0.200000
0.366667 0.716667 -0.533333 -0.200000
0.125000 -0.187500 0.000000 0.000000
------------------------------------------------------------
```

Apabila masukan via file (*.txt), maka penulisan matriks A harus dilakukan secara augmented (dengan syarat, matriks A merupakan matriks persegi, ukuran baris = ukuran kolom) seperti yang dilakukan pada metode-metode sebelumnya.

5. Interpolasi Polinom

Apabila pengguna memasukkan angka 4 (Interpolasi Polinom), maka akan muncul pilihan metode yang dapat digunakan sesuai kebutuhan pengguna. Masing-masing pilihan berkaitan dengan metode penyelesaian interpolasi polinom dari masukan pengguna.

Misalkan untuk studi kasus:

x	0.1	0.3	0.5	0.7	0.9	1.1	1.3
y	0.003	0.067	0.148	0.248	0.370	0.518	0.697

Pada menu interpolasi polinom, untuk masukan via CLI, pengguna akan diminta untuk memasukkan banyak nilai sampel, serta ordinat dan , program yang akan muncul sebagai berikut:

```
------------------------------------------------------------
                          Sumber
                    1. Masukan dari CLI
------------------------------------------------------------
Masukkan banyaknya titik sampel (X,Y) = 7
------------------------------------------------------------
Masukkan X1 = 0.1
Masukkan Y1 = 0.003
------------------------------------------------------------
Masukkan X2 = 0.3
Masukkan Y2 = 0.067
------------------------------------------------------------
Masukkan X3 = 0.5
Masukkan Y3 = 0.148
------------------------------------------------------------
Masukkan X4 = 0.7
Masukkan Y4 = 0.248
------------------------------------------------------------
Masukkan X5 = 0.9
Masukkan Y5 = 0.370
------------------------------------------------------------
Masukkan X6 = 1.1
Masukkan Y6 = 0.518
------------------------------------------------------------
Masukkan X7 = 1.3
Masukkan Y7 = 0.697
------------------------------------------------------------
```

Berikut ini adalah simulasi proses penyelesaian interpolasi polinom:

Matriks koefisien dari persamaan derajat 7 :
1.000000 0.100000 0.010000 0.001000 0.000100 0.000010 0.000001 0.003000
1.000000 0.300000 0.090000 0.027000 0.008100 0.002430 0.000729 0.067000
1.000000 0.500000 0.250000 0.125000 0.062500 0.031250 0.015625 0.148000
1.000000 0.700000 0.490000 0.343000 0.240100 0.168070 0.117649 0.248000
1.000000 0.900000 0.810000 0.729000 0.656100 0.590490 0.531441 0.370000
1.000000 1.100000 1.210000 1.331000 1.464100 1.610510 1.771561 0.518000
1.000000 1.300000 1.690000 2.197000 2.856100 3.712930 4.826809 0.697000

Kurangi baris ke-2 dengan 1.000000 kali baris ke-1
1.000000 0.100000 0.010000 0.001000 0.000100 0.000010 0.000001 0.003000
0.000000 0.200000 0.080000 0.026000 0.008000 0.002420 0.000728 0.064000
1.000000 0.500000 0.250000 0.125000 0.062500 0.031250 0.015625 0.148000
1.000000 0.700000 0.490000 0.343000 0.240100 0.168070 0.117649 0.248000
1.000000 0.900000 0.810000 0.729000 0.656100 0.590490 0.531441 0.370000
1.000000 1.100000 1.210000 1.331000 1.464100 1.610510 1.771561 0.518000
1.000000 1.300000 1.690000 2.197000 2.856100 3.712930 4.826809 0.697000

(...perhitungan tidak ditampilkan seluruhnya untuk mempersingkat...)

Kurangi baris ke-1 dengan 0.010000 kali baris ke-3
1.000000 0.100000 0.000000 0.000000 0.000000 0.000000 0.000000 0.001023
0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.240000
0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.197396
0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.026042
0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000

Kurangi baris ke-1 dengan 0.100000 kali baris ke-2
1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -0.022977
0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.240000
0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.197396
0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.026042
0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000

Selanjutnya, akan dihasilkan sebuah solusi dari interpolasi polinom, sebagai berikut:

Solusi SPL f(x) = 0.000000x^6 + 0.000000x^5 + 0.026042x^4 + 0.000000x^3 +
0.197396x^2 + 0.240000x + -0.022977

Pengguna kemudian akan diminta untuk memasukkan nilai X (ordinat) yang hendak ditaksir nilainya, program selanjutnya akan menampilkan solusi dari nilai taksiran yang diinginkan pengguna:

```
------------------------------------------------------------
Masukan nilai x yang akan ditaksir : 0.2
Hasil interpolasi f(0.200000) = 0.032961
------------------------------------------------------------
```

6. Interpolasi Bicubic

Apabila pengguna memasukkan angka 5 (Interpolasi Bicubic), maka akan muncul pilihan metode yang dapat digunakan sesuai kebutuhan pengguna. Masing-masing pilihan berkaitan dengan metode penyelesaian interpolasi Bicubic dari masukan pengguna.

Misalkan untuk studi kasus:

(x,y)	(-1,-1)	(-1,0)	(-1,1)	(-1,2)	(0,-1)	(0,0)	(0,1)	(0,2)	(1,-1)	(1,0)	(1,1)
f(x,y)	4	6	8	12	4	9	10	2	1	6	13

(1,2)	(2,-1)	(2,0)	(2,1)	(2,2)
4	7	18	9	4

Pada menu interpolasi bicubic, untuk masukan via CLI, pengguna akan diminta untuk memasukkan banyak nilai sampel, serta ordinat dan absis, program yang akan muncul sebagai berikut:

Sumber
1. Masukan dari CLI

Masukkan nilai f(-1,-1) = 4
Masukkan nilai f(-1,0) = 6
Masukkan nilai f(-1,1) = 8
Masukkan nilai f(-1,2) = 12
Masukkan nilai f(0,-1) = 4
Masukkan nilai f(0,0) = 9
Masukkan nilai f(0,1) = 10
Masukkan nilai f(0,2) = 2
Masukkan nilai f(1,-1) = 1
Masukkan nilai f(1,0) = 6
Masukkan nilai f(1,1) = 13
Masukkan nilai f(1,2) = 4
Masukkan nilai f(2,-1) = 7
Masukkan nilai f(2,0) = 18

Masukkan nilai f(2,1) = 9
Masukkan nilai f(2,2) = 4

Selanjutnya, pengguna akan diminta untuk menentukan nilai a dan b yang hendak ditentukan hasil f(a,b)-nya dengan interpolasi bicubic:

Misalkan

a = 0.1;
b = 0.3;

Berikut ini adalah simulasi proses penyelesaian interpolasi bicubic:

```
------------------------------------------------------------
Kurangi baris ke-2 dengan 1.000000 kali baris ke-1
1.000000 -1.000000 1.000000 -1.000000 -1.000000 1.000000 -1.000000 1.000000 1.000000 -1.000000 1.000000 -1.000000 -1.000000 1.000000 -1.000000 1.000000 4.000000
0.000000 1.000000 -1.000000 1.000000 0.000000 -1.000000 1.000000 -1.000000 0.000000 1.000000 -1.000000 1.000000 0.000000 -1.000000 1.000000 -1.000000 2.000000
1.000000 1.000000 1.000000 1.000000 -1.000000 -1.000000 -1.000000 -1.000000 1.000000 1.000000 1.000000 1.000000 -1.000000 -1.000000 -1.000000 -1.000000 8.000000
1.000000 2.000000 4.000000 8.000000 -1.000000 -2.000000 -4.000000 -8.000000 1.000000 2.000000 4.000000 8.000000 -1.000000 -2.000000 -4.000000 -8.000000 12.000000
1.000000 -1.000000 1.000000 -1.000000 0.000000 -0.000000 0.000000 -0.000000 0.000000 -0.000000 0.000000 -0.000000 0.000000 -0.000000 0.000000 -0.000000 4.000000
1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 9.000000
1.000000 1.000000 1.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 10.000000
1.000000 2.000000 4.000000 8.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 2.000000
1.000000 -1.000000 1.000000 -1.000000 1.000000 -1.000000 1.000000 -1.000000 1.000000 -1.000000 1.000000 -1.000000 1.000000 -1.000000 1.000000 -1.000000 1.000000
1.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 6.000000
1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 13.000000
1.000000 2.000000 4.000000 8.000000 1.000000 2.000000 4.000000 8.000000 1.000000 2.000000 4.000000 8.000000 1.000000 2.000000 4.000000 8.000000 4.000000
1.000000 -1.000000 1.000000 -1.000000 2.000000 -2.000000 2.000000 -2.000000 4.000000 -4.000000 4.000000 -4.000000 8.000000 -8.000000 8.000000 -8.000000 7.000000
1.000000 0.000000 0.000000 0.000000 2.000000 0.000000 0.000000 0.000000 4.000000 0.000000 0.000000 0.000000 8.000000 0.000000 0.000000 0.000000 18.000000
1.000000 1.000000 1.000000 1.000000 2.000000 2.000000 2.000000 2.000000 4.000000 4.000000 4.000000 4.000000 8.000000 8.000000 8.000000 8.000000 9.000000
1.000000 2.000000 4.000000 8.000000 2.000000 4.000000 8.000000 16.000000 4.000000 8.000000 16.000000 32.000000 8.000000 16.000000 32.000000 64.000000 4.000000

(...perhitungan tidak ditampilkan seluruhnya untuk mempersingkat...)

Kurangi baris ke-1 dengan -1.000000 kali baris ke-2
1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 9.000000
0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 3.833333
0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -2.000000
0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -0.833333
0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -3.500000
0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 7.027778
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 3.666667
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -3.361111
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -3.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.500000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 2.500000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 -0.500000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 3.500000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 -3.361111
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 -3.166667
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 1.694444
------------------------------------------------------------
```

Di akhir proses interpolasi bicubic, program akan menampilkan nilai f(a,b) hasil dari masukan nilai a dan b pengguna:

```
------------------------------------------------------------
Maka f(0.100000,0.300000) = 8.363646
------------------------------------------------------------
```

7. Regresi Linier Berganda

Apabila pengguna memasukkan angka 6 (Regresi Linier Berganda), maka akan muncul pilihan metode yang dapat digunakan sesuai kebutuhan pengguna. Masing-masing pilihan berkaitan dengan metode penyelesaian regresi linier berganda dari masukan pengguna.

Misalkan untuk studi kasus:

n (banyak sampel) = 3;
x (banyak variabel x) = 1;

X1	3	1	5
y	2	4	6

Pada menu regresi linier berganda, untuk masukan via CLI, pengguna akan diminta untuk memasukkan banyak nilai sampel, serta banyak variabel X yang akan digunakan , program yang akan muncul sebagai berikut:

```
------------------------------------------------------------
                          Sumber
                  1. Masukan dari CLI
------------------------------------------------------------
Masukkan banyak sampel n : 3
Masukkan banyak varibel x : 1
------------------------------------------------------------
Masukkan nilai X1 untuk sampel 1 : 3
Masukkan nilai y untuk sampel 1 : 2
------------------------------------------------------------
Masukkan nilai X1 untuk sampel 2 : 1
Masukkan nilai y untuk sampel 2 : 4
------------------------------------------------------------
Masukkan nilai X1 untuk sampel 3 : 5
Masukkan nilai y untuk sampel 3 : 6
------------------------------------------------------------
```

Selanjutnya, pengguna akan diminta untuk menentukan nilai-nilai x yang hendak ditaksir dengan regresi linier berganda:

Misalkan:

X1 = 3;

```
------------------------------------------------------------
Masukkan nilai-nilai X yang akan ditaksir...
Nilai X1 : 3
------------------------------------------------------------
```

Berikut ini adalah simulasi proses penyelesaian regresi linier berganda:

```
------------------------------------------------------------
3.000000 9.000000 1.000000 0.000000
9.000000 35.000000 0.000000 1.000000

Kurangi baris ke-2 dengan 3.000000 kali baris ke-1
3.000000 9.000000 1.000000 0.000000
0.000000 8.000000 -3.000000 1.000000

Bagi setiap elemen di baris ke-1 dengan 3.00
1.000000 3.000000 0.333333 0.000000
0.000000 8.000000 -3.000000 1.000000

Bagi setiap elemen di baris ke-2 dengan 8.00
1.000000 3.000000 0.333333 0.000000
0.000000 1.000000 -0.375000 0.125000

Kurangi baris ke-1 dengan 3.000000 kali baris ke-2
1.000000 0.000000 1.458333 -0.375000
0.000000 1.000000 -0.375000 0.125000
------------------------------------------------------------
```

Di akhir proses regresi linier berganda, program akan menampilkan persamaan yang didapat, serta hasil taksiran nilai variabel masukan pengguna:

```
------------------------------------------------------------
Maka didapatkan persamaan..
f(x) = 2.500000 + 1.142857X1
y = 2.500000 x 1.000000 + 1.142857 x 3.000000 = 5.928571
------------------------------------------------------------
```

8. Penutup

Apabila pengguna memasukkan angka 7 (Penutup), maka program akan tertutup secara otomatis.

```
------------------------------------------------------------
                      TUBES ALGEO-01
  SISTEM PERSAMAAN LINIER, DETERMINAN, DAN APLIKASINYA
------------------------------------------------------------
                        Menu Utama
------------------------------------------------------------
1. Sistem Persamaan Linier(SPL)
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic
6. Regresi Linier berganda
7. Keluar
------------------------------------------------------------
Masukkan pilihan menu Anda:
7
```

9. Menyimpan File

Pada tiap pilihan metode, ketika hasil telah dicetak ke layar, pengguna dapat memilih untuk menyimpan file di local directory-nya atau tidak, dengan mengetikan 'y' untuk menyimpan file dalam bentuk (*.txt) serta 'n' untuk tidak menyimpan file dan mengembalikan program ke tampilan menu utama. Apabila pengguna mengetikan 'y', tampilan program sebagai berikut:

```
------------------------------------------------------------
Hasil ingin disimpan? (y/n):
y
------------------------------------------------------------
Nama output file (*.txt):
coba.txt
------------------------------------------------------------
File coba.txt berhasil disimpan.
------------------------------------------------------------
```

Pengguna akan diminta menentukan nama file yang hendak disimpan dengan extension .txt. File (*.txt) yang disimpan nantinya akan berada dalam folder test\output , sementara itu ketika pengguna mengetikan 'n', tampilan program sebagai berikut:

```
------------------------------------------------------------
Hasil ingin disimpan? (y/n):
n
------------------------------------------------------------
------------------------------------------------------------
            Operasi Sistem Persamaan Linier
                       SELESAI
                 Kembali ke Menu Utama
------------------------------------------------------------
------------------------------------------------------------
                     TUBES ALGEO-01
  SISTEM PERSAMAAN LINIER, DETERMINAN, DAN APLIKASINYA
------------------------------------------------------------
                       Menu Utama
------------------------------------------------------------
1. Sistem Persamaan Linier(SPL)
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic
6. Regresi Linier berganda
7. Keluar
------------------------------------------------------------
Masukkan pilihan menu Anda:
```

Pengguna akan langsung dikembalikan ke menu utama secara otomatis.

10. Studi Kasus

1. Temukan solusi SPL $Ax = b$,berikut:

   a.

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

$$\text{Matriks Augmented} = \begin{bmatrix} 1 & 1 & -1 & -1 & 1 \\ 2 & 5 & -7 & -5 & -2 \\ 2 & -1 & 1 & 3 & 4 \\ 5 & 2 & -4 & 2 & 6 \end{bmatrix}$$

   Solusi:

```
----------------------------------------------------------
                        Menu
                1.Sistem Persamaan Linier(SPL)
----------------------------------------------------------
                    Pilihan Metode:
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks balikan
4. Kaidah Cramer
5. Kembali ke Menu Utama
----------------------------------------------------------
Masukkan pilihan metode Anda:
2
----------------------------------------------------------
                        Metode
            2. Metode Eliminasi Gauss-Jordan
----------------------------------------------------------
                    Pilihan Sumber:
----------------------------------------------------------
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode
----------------------------------------------------------
Masukkan pilihan sumber Anda:
1
----------------------------------------------------------
                        Sumber
                    1. Masukan dari CLI
----------------------------------------------------------
Masukkan jumlah baris:
4
Masukkan jumlah kolom:
5
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
1
Masukkan nilai untuk baris ke-1 kolom ke-2:
1
Masukkan nilai untuk baris ke-1 kolom ke-3:
-1
Masukkan nilai untuk baris ke-1 kolom ke-4:
-1
Masukkan nilai untuk baris ke-1 kolom ke-5:
1
Masukkan nilai untuk baris ke-2 kolom ke-1:
2
Masukkan nilai untuk baris ke-2 kolom ke-2:
5
Masukkan nilai untuk baris ke-2 kolom ke-3:
```

-7
Masukkan nilai untuk baris ke-2 kolom ke-4:
-5
Masukkan nilai untuk baris ke-2 kolom ke-5:
-2
Masukkan nilai untuk baris ke-3 kolom ke-1:
2
Masukkan nilai untuk baris ke-3 kolom ke-2:
-1
Masukkan nilai untuk baris ke-3 kolom ke-3:
1
Masukkan nilai untuk baris ke-3 kolom ke-4:
3
Masukkan nilai untuk baris ke-3 kolom ke-5:
4
Masukkan nilai untuk baris ke-4 kolom ke-1:
5
Masukkan nilai untuk baris ke-4 kolom ke-2:
2
Masukkan nilai untuk baris ke-4 kolom ke-3:
-4
Masukkan nilai untuk baris ke-4 kolom ke-4:
2
Masukkan nilai untuk baris ke-4 kolom ke-5:
6
Matriks berhasil dibaca.

Matriks yang dibaca:
1.000000 1.000000 -1.000000 -1.000000 1.000000
2.000000 5.000000 -7.000000 -5.000000 -2.000000
2.000000 -1.000000 1.000000 3.000000 4.000000
5.000000 2.000000 -4.000000 2.000000 6.000000

Kurangi baris ke-2 dengan 2.000000 kali baris ke-1
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 3.000000 -5.000000 -3.000000 -4.000000
2.000000 -1.000000 1.000000 3.000000 4.000000
5.000000 2.000000 -4.000000 2.000000 6.000000

Kurangi baris ke-3 dengan 2.000000 kali baris ke-1
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 3.000000 -5.000000 -3.000000 -4.000000
0.000000 -3.000000 3.000000 5.000000 2.000000
5.000000 2.000000 -4.000000 2.000000 6.000000

Kurangi baris ke-4 dengan 5.000000 kali baris ke-1
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 3.000000 -5.000000 -3.000000 -4.000000
0.000000 -3.000000 3.000000 5.000000 2.000000

0.000000 -3.000000 1.000000 7.000000 1.000000

Kurangi baris ke-3 dengan -1.000000 kali baris ke-2
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 3.000000 -5.000000 -3.000000 -4.000000
0.000000 0.000000 -2.000000 2.000000 -2.000000
0.000000 -3.000000 1.000000 7.000000 1.000000

Kurangi baris ke-4 dengan -1.000000 kali baris ke-2
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 3.000000 -5.000000 -3.000000 -4.000000
0.000000 0.000000 -2.000000 2.000000 -2.000000
0.000000 0.000000 -4.000000 4.000000 -3.000000

Kurangi baris ke-4 dengan 2.000000 kali baris ke-3
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 3.000000 -5.000000 -3.000000 -4.000000
0.000000 0.000000 -2.000000 2.000000 -2.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-1 dengan 1.00
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 3.000000 -5.000000 -3.000000 -4.000000
0.000000 0.000000 -2.000000 2.000000 -2.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-2 dengan 3.00
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 1.000000 -1.666667 -1.000000 -1.333333
0.000000 0.000000 -2.000000 2.000000 -2.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-3 dengan -2.00
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 1.000000 -1.666667 -1.000000 -1.333333
0.000000 0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Kurangi baris ke-2 dengan -1.666667 kali baris ke-3
1.000000 1.000000 -1.000000 -1.000000 1.000000
0.000000 1.000000 0.000000 -2.666667 0.333333
0.000000 0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Kurangi baris ke-1 dengan -1.000000 kali baris ke-3
1.000000 1.000000 0.000000 -2.000000 2.000000
0.000000 1.000000 0.000000 -2.666667 0.333333
0.000000 0.000000 1.000000 -1.000000 1.000000

0.000000 0.000000 0.000000 0.000000 1.000000

Kurangi baris ke-1 dengan 1.000000 kali baris ke-2
1.000000 0.000000 0.000000 0.666667 1.666667
0.000000 1.000000 0.000000 -2.666667 0.333333
0.000000 0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Dengan metode Eliminasi Gauss-Jordan, diperoleh solusi SPL:
Bagi setiap elemen di baris ke-1 dengan 1.00
1.000000 0.000000 0.000000 0.666667 1.666667
0.000000 1.000000 0.000000 -2.666667 0.333333
0.000000 0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-2 dengan 1.00
1.000000 0.000000 0.000000 0.666667 1.666667
0.000000 1.000000 0.000000 -2.666667 0.333333
0.000000 0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 0.000000 0.000000 1.000000

Bagi setiap elemen di baris ke-3 dengan 1.00
1.000000 0.000000 0.000000 0.666667 1.666667
0.000000 1.000000 0.000000 -2.666667 0.333333
0.000000 0.000000 1.000000 -1.000000 1.000000
0.000000 0.000000 0.000000 0.000000 1.000000

SPL tidak Memiliki Solusi

SPL tidak memiliki solusi karena baris terakhir tidak ada 1 utama, dan baris terakhir, kolom terakhir ada suatu nilai.

*Penafian, karena output kami juga mencantumkan langkah dari eliminasi Gauss/Gauss-Jordan satu per satu. Demi keterbacaan yang lebih mudah untuk laporan kami, kami tidak akan menunjukkan bagian menu, dan langkah-langkahnya lagi. Untuk tes-tes kasus selanjutnya kami akan hanya menunjukkan input, keadaan pertama matriks, hasil akhir matriks, dan hasil SPL-nya.

b.

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

Solusi:

Metode
2. Metode Eliminasi Gauss-Jordan

Pilihan Sumber:
---
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode

Masukkan pilihan sumber Anda:
1

Sumber
1. Masukan dari CLI

Masukkan jumlah baris:
4
Masukkan jumlah kolom:
6
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
1
Masukkan nilai untuk baris ke-1 kolom ke-2:
-1
Masukkan nilai untuk baris ke-1 kolom ke-3:
0
Masukkan nilai untuk baris ke-1 kolom ke-4:
0
Masukkan nilai untuk baris ke-1 kolom ke-5:
1
Masukkan nilai untuk baris ke-1 kolom ke-6:
3
Masukkan nilai untuk baris ke-2 kolom ke-1:
1
Masukkan nilai untuk baris ke-2 kolom ke-2:
1
Masukkan nilai untuk baris ke-2 kolom ke-3:

```
0
Masukkan nilai untuk baris ke-2 kolom ke-4:
-3
Masukkan nilai untuk baris ke-2 kolom ke-5:
0
Masukkan nilai untuk baris ke-2 kolom ke-6:
6
Masukkan nilai untuk baris ke-3 kolom ke-1:
2
Masukkan nilai untuk baris ke-3 kolom ke-2:
-1
Masukkan nilai untuk baris ke-3 kolom ke-3:
0
Masukkan nilai untuk baris ke-3 kolom ke-4:
1
Masukkan nilai untuk baris ke-3 kolom ke-5:
-1
Masukkan nilai untuk baris ke-3 kolom ke-6:
5
Masukkan nilai untuk baris ke-4 kolom ke-1:
-1
Masukkan nilai untuk baris ke-4 kolom ke-2:
2
Masukkan nilai untuk baris ke-4 kolom ke-3:
0
Masukkan nilai untuk baris ke-4 kolom ke-4:
-2
Masukkan nilai untuk baris ke-4 kolom ke-5:
-1
Masukkan nilai untuk baris ke-4 kolom ke-6:
-1
Matriks berhasil dibaca.
----------------------------------------------------------
Matriks yang dibaca:
1.000000 -1.000000 0.000000 0.000000 1.000000 3.000000
1.000000 1.000000 0.000000 -3.000000 0.000000 6.000000
2.000000 -1.000000 0.000000 1.000000 -1.000000 5.000000
-1.000000 2.000000 0.000000 -2.000000 -1.000000 -1.000000

Bagi setiap elemen di baris ke-3 dengan 1.00
1.000000 0.000000 0.000000 0.000000 -1.000000 3.000000
0.000000 1.000000 0.000000 0.000000 -2.000000 0.000000
0.000000 0.000000 0.000000 1.000000 -1.000000 -1.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000

X1 = a + 3.00
X2 = 2.00a
X3 = b
X4 = a - 1.00
```

X5 = a

Solusi merupakan parametrik karena SPL merupakan solusi banyak, dan karena baris terakhir tidak ada 1 utama dan tidak ada nilai di baris terakhir, kolom terakhir dari hasil gauss-jordan

c.

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

Solusi:

```
------------------------------------------------------
                      Menu
          1.Sistem Persamaan Linier(SPL)
------------------------------------------------------
                   Pilihan Metode:
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks balikan
4. Kaidah Cramer
5. Kembali ke Menu Utama
------------------------------------------------------
Masukkan pilihan metode Anda:
2
------------------------------------------------------
                      Metode
          2. Metode Eliminasi Gauss-Jordan
------------------------------------------------------
                   Pilihan Sumber:
------------------------------------------------------
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode
------------------------------------------------------
Masukkan pilihan sumber Anda:
1
------------------------------------------------------
                      Sumber
                1. Masukan dari CLI
------------------------------------------------------
Masukkan jumlah baris:
3
Masukkan jumlah kolom:
```

```
7
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
0
Masukkan nilai untuk baris ke-1 kolom ke-2:
1
Masukkan nilai untuk baris ke-1 kolom ke-3:
0
Masukkan nilai untuk baris ke-1 kolom ke-4:
0
Masukkan nilai untuk baris ke-1 kolom ke-5:
1
Masukkan nilai untuk baris ke-1 kolom ke-6:
0
Masukkan nilai untuk baris ke-1 kolom ke-7:
2
Masukkan nilai untuk baris ke-2 kolom ke-1:
0
Masukkan nilai untuk baris ke-2 kolom ke-2:
0
Masukkan nilai untuk baris ke-2 kolom ke-3:
0
Masukkan nilai untuk baris ke-2 kolom ke-4:
1
Masukkan nilai untuk baris ke-2 kolom ke-5:
1
Masukkan nilai untuk baris ke-2 kolom ke-6:
0
Masukkan nilai untuk baris ke-2 kolom ke-7:
-1
Masukkan nilai untuk baris ke-3 kolom ke-1:
0
Masukkan nilai untuk baris ke-3 kolom ke-2:
1
Masukkan nilai untuk baris ke-3 kolom ke-3:
0
Masukkan nilai untuk baris ke-3 kolom ke-4:
0
Masukkan nilai untuk baris ke-3 kolom ke-5:
0
Masukkan nilai untuk baris ke-3 kolom ke-6:
1
Masukkan nilai untuk baris ke-3 kolom ke-7:
1
Matriks berhasil dibaca.
--------------------------------------------------------
Matriks yang dibaca:
0.000000 1.000000 0.000000 0.000000 1.000000 0.000000 2.000000
0.000000 0.000000 0.000000 1.000000 1.000000 0.000000 -1.000000
```

0.000000 1.000000 0.000000 0.000000 0.000000 1.000000 1.000000

Bagi setiap elemen di baris ke-3 dengan 1.00
0.000000 1.000000 0.000000 0.000000 0.000000 1.000000 1.000000
0.000000 0.000000 0.000000 1.000000 0.000000 1.000000 -2.000000
0.000000 0.000000 0.000000 0.000000 1.000000 -1.000000 1.000000
X1 = b
X2 = -a + 1.00
X3 = c
X4 = -a - 2.00
X5 = a + 1.00
X6 = a

Solusi merupakan parametrik karena SPL merupakan solusi banyak, dan karena baris terakhir tidak ada 1 utama dan tidak ada nilai di baris terakhir, kolom terakhir dari hasil gauss-jordan

d.

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

H adalah matriks Hilbert. Cobakan untuk n = 6 dan n = 10.

Solusi:

n = 6

Metode
2. Metode Eliminasi Gauss-Jordan

Pilihan Sumber:
---
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode

Masukkan pilihan sumber Anda:
1

Sumber
1. Masukan dari CLI

Masukkan jumlah baris:
6
Masukkan jumlah kolom:
7
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
1
Masukkan nilai untuk baris ke-1 kolom ke-2:
0.5
Masukkan nilai untuk baris ke-1 kolom ke-3:
0.333333
Masukkan nilai untuk baris ke-1 kolom ke-4:
0.25
Masukkan nilai untuk baris ke-1 kolom ke-5:
0.2
Masukkan nilai untuk baris ke-1 kolom ke-6:
0.166667
Masukkan nilai untuk baris ke-1 kolom ke-7:
1
Masukkan nilai untuk baris ke-2 kolom ke-1:
0.5
Masukkan nilai untuk baris ke-2 kolom ke-2:
0.333333
Masukkan nilai untuk baris ke-2 kolom ke-3:
0.25
Masukkan nilai untuk baris ke-2 kolom ke-4:
0.2
Masukkan nilai untuk baris ke-2 kolom ke-5:
0.166667
Masukkan nilai untuk baris ke-2 kolom ke-6:
0.142857
Masukkan nilai untuk baris ke-2 kolom ke-7:
0
Masukkan nilai untuk baris ke-3 kolom ke-1:
0.333333
Masukkan nilai untuk baris ke-3 kolom ke-2:
0.25
Masukkan nilai untuk baris ke-3 kolom ke-3:
0.2
Masukkan nilai untuk baris ke-3 kolom ke-4:
0.166667
Masukkan nilai untuk baris ke-3 kolom ke-5:
0.142857
Masukkan nilai untuk baris ke-3 kolom ke-6:
0.125
Masukkan nilai untuk baris ke-3 kolom ke-7:

```
0
Masukkan nilai untuk baris ke-4 kolom ke-1:
0.25
Masukkan nilai untuk baris ke-4 kolom ke-2:
0.2
Masukkan nilai untuk baris ke-4 kolom ke-3:
0.166667
Masukkan nilai untuk baris ke-4 kolom ke-4:
0.142857
Masukkan nilai untuk baris ke-4 kolom ke-5:
0.125
Masukkan nilai untuk baris ke-4 kolom ke-6:
0.111111
Masukkan nilai untuk baris ke-4 kolom ke-7:
0
Masukkan nilai untuk baris ke-5 kolom ke-1:
0.2
Masukkan nilai untuk baris ke-5 kolom ke-2:
0.166667
Masukkan nilai untuk baris ke-5 kolom ke-3:
0.142857
Masukkan nilai untuk baris ke-5 kolom ke-4:
0.125
Masukkan nilai untuk baris ke-5 kolom ke-5:
0.111111
Masukkan nilai untuk baris ke-5 kolom ke-6:
0.1
Masukkan nilai untuk baris ke-5 kolom ke-7:
0
Masukkan nilai untuk baris ke-6 kolom ke-1:
0.166667
Masukkan nilai untuk baris ke-6 kolom ke-2:
0.142857
Masukkan nilai untuk baris ke-6 kolom ke-3:
0.125
Masukkan nilai untuk baris ke-6 kolom ke-4:
0.111111
Masukkan nilai untuk baris ke-6 kolom ke-5:
0.1
Masukkan nilai untuk baris ke-6 kolom ke-6:
0.090909
Masukkan nilai untuk baris ke-6 kolom ke-7:
0
Matriks berhasil dibaca.
---------------------------------------------------------
Matriks yang dibaca:
1.000000 0.500000 0.333333 0.250000 0.200000 0.166667 1.000000
0.500000 0.333333 0.250000 0.200000 0.166667 0.142857 0.000000
0.333333 0.250000 0.200000 0.166667 0.142857 0.125000 0.000000
```

0.250000 0.200000 0.166667 0.142857 0.125000 0.111111 0.000000
0.200000 0.166667 0.142857 0.125000 0.111111 0.100000 0.000000
0.166667 0.142857 0.125000 0.111111 0.100000 0.090909 0.000000

Bagi setiap elemen di baris ke-6 dengan 1.00
1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 11.540412
0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 46.616694
0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 -1130.944969
0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 3969.618023
0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 -5045.250153
0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 2158.663954

X1 = 11.54
X2 = 46.62
X3 = -1130.94
X4 = 3969.62
X5 = -5045.25
X6 = 2158.66

Solusi unik karena baris terakhir dari matriks gauss-jordan ada 1 utama dan baris terakhir, kolom terakhir ada suatu nilai.

n = 10

Karena output sangat besar, kami tidak bisa menunjukkan input dan keadaan awal matriks kami hanya bisa menunjukkan hasil akhir matriks, dan hasil SPL-nya

Bagi setiap elemen di baris ke-10 dengan 1.00
1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 19.528495
0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -158.768669
0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 8.945483
0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 2038.975402
0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 -5200.448159
0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 5793.244374
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 -4729.923790
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 3324.637824
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 -640.251596
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 -460.608742

X1 = 19.53
X2 = -158.77
X3 = 8.95
X4 = 2038.98
X5 = -5200.45
X6 = 5793.24
X7 = -4729.92
X8 = 3324.64
X9 = -640.25

X10 = -460.61

Solusi unik karena baris terakhir dari matriks gauss-jordan ada 1 utama dan baris terakhir, kolom terakhir ada suatu nilai.

2. SPL berbentuk matriks *augmented*
   a.

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

Solusi:

```
                    Metode
          2. Metode Eliminasi Gauss-Jordan
-----------------------------------------------------
                  Pilihan Sumber:
-----------------------------------------------------
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode
-----------------------------------------------------
Masukkan pilihan sumber Anda:
1
-----------------------------------------------------
                    Sumber
                1. Masukan dari CLI
-----------------------------------------------------
Masukkan jumlah baris:
4
Masukkan jumlah kolom:
5
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
1
Masukkan nilai untuk baris ke-1 kolom ke-2:
-1
Masukkan nilai untuk baris ke-1 kolom ke-3:
2
Masukkan nilai untuk baris ke-1 kolom ke-4:
-1
Masukkan nilai untuk baris ke-1 kolom ke-5:
-1
```

Masukkan nilai untuk baris ke-2 kolom ke-1:
2
Masukkan nilai untuk baris ke-2 kolom ke-2:
1
Masukkan nilai untuk baris ke-2 kolom ke-3:
-2
Masukkan nilai untuk baris ke-2 kolom ke-4:
-2
Masukkan nilai untuk baris ke-2 kolom ke-5:
-2
Masukkan nilai untuk baris ke-3 kolom ke-1:
-1
Masukkan nilai untuk baris ke-3 kolom ke-2:
2
Masukkan nilai untuk baris ke-3 kolom ke-3:
-4
Masukkan nilai untuk baris ke-3 kolom ke-4:
1
Masukkan nilai untuk baris ke-3 kolom ke-5:
1
Masukkan nilai untuk baris ke-4 kolom ke-1:
3
Masukkan nilai untuk baris ke-4 kolom ke-2:
0
Masukkan nilai untuk baris ke-4 kolom ke-3:
0
Masukkan nilai untuk baris ke-4 kolom ke-4:
-3
Masukkan nilai untuk baris ke-4 kolom ke-5:
-3
Matriks berhasil dibaca.

Matriks yang dibaca:
1.000000 -1.000000 2.000000 -1.000000 -1.000000
2.000000 1.000000 -2.000000 -2.000000 -2.000000
-1.000000 2.000000 -4.000000 1.000000 1.000000
3.000000 0.000000 0.000000 -3.000000 -3.000000

Bagi setiap elemen di baris ke-2 dengan 1.00
1.000000 0.000000 0.000000 -1.000000 -1.000000
0.000000 1.000000 -2.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000

X1 = b - 1.00
X2 = 2.00a
X3 = a
X4 = b

Solusi merupakan parametrik karena SPL merupakan solusi banyak, dan karena baris terakhir tidak ada 1 utama dan tidak ada nilai di baris terakhir, kolom terakhir dari hasil gauss-jordan

b.

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

Solusi:

```
---------------------------------------------------------
                        Menu
                1.Sistem Persamaan Linier(SPL)
---------------------------------------------------------
                    Pilihan Metode:
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks balikan
4. Kaidah Cramer
5. Kembali ke Menu Utama
---------------------------------------------------------
Masukkan pilihan metode Anda:
2
---------------------------------------------------------
                        Metode
                2. Metode Eliminasi Gauss-Jordan
---------------------------------------------------------
                    Pilihan Sumber:
---------------------------------------------------------
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode
---------------------------------------------------------
Masukkan pilihan sumber Anda:
1
---------------------------------------------------------
                        Sumber
                    1. Masukan dari CLI
---------------------------------------------------------
Masukkan jumlah baris:
6
Masukkan jumlah kolom:
5
```

Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
2
Masukkan nilai untuk baris ke-1 kolom ke-2:
0
Masukkan nilai untuk baris ke-1 kolom ke-3:
8
Masukkan nilai untuk baris ke-1 kolom ke-4:
0
Masukkan nilai untuk baris ke-1 kolom ke-5:
8
Masukkan nilai untuk baris ke-2 kolom ke-1:
0
Masukkan nilai untuk baris ke-2 kolom ke-2:
1
Masukkan nilai untuk baris ke-2 kolom ke-3:
0
Masukkan nilai untuk baris ke-2 kolom ke-4:
4
Masukkan nilai untuk baris ke-2 kolom ke-5:
6
Masukkan nilai untuk baris ke-3 kolom ke-1:
-4
Masukkan nilai untuk baris ke-3 kolom ke-2:
0
Masukkan nilai untuk baris ke-3 kolom ke-3:
6
Masukkan nilai untuk baris ke-3 kolom ke-4:
0
Masukkan nilai untuk baris ke-3 kolom ke-5:
6
Masukkan nilai untuk baris ke-4 kolom ke-1:
0
Masukkan nilai untuk baris ke-4 kolom ke-2:
-2
Masukkan nilai untuk baris ke-4 kolom ke-3:
0
Masukkan nilai untuk baris ke-4 kolom ke-4:
3
Masukkan nilai untuk baris ke-4 kolom ke-5:
-1
Masukkan nilai untuk baris ke-5 kolom ke-1:
2
Masukkan nilai untuk baris ke-5 kolom ke-2:
0
Masukkan nilai untuk baris ke-5 kolom ke-3:
-4
Masukkan nilai untuk baris ke-5 kolom ke-4:
0

Masukkan nilai untuk baris ke-5 kolom ke-5:
-4
Masukkan nilai untuk baris ke-6 kolom ke-1:
0
Masukkan nilai untuk baris ke-6 kolom ke-2:
1
Masukkan nilai untuk baris ke-6 kolom ke-3:
0
Masukkan nilai untuk baris ke-6 kolom ke-4:
-2
Masukkan nilai untuk baris ke-6 kolom ke-5:
0
Matriks berhasil dibaca.

Matriks yang dibaca:
2.000000 0.000000 8.000000 0.000000 8.000000
0.000000 1.000000 0.000000 4.000000 6.000000
-4.000000 0.000000 6.000000 0.000000 6.000000
0.000000 -2.000000 0.000000 3.000000 -1.000000
2.000000 0.000000 -4.000000 0.000000 -4.000000
0.000000 1.000000 0.000000 -2.000000 0.000000

Bagi setiap elemen di baris ke-4 dengan 1.00
1.000000 0.000000 0.000000 0.000000 0.000000
0.000000 1.000000 0.000000 0.000000 2.000000
0.000000 0.000000 1.000000 0.000000 1.000000
0.000000 0.000000 0.000000 1.000000 1.000000
0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000

X1 = 0
X2 = 2.00
X3 = 1.00
X4 = 1.00

Solusi unik karena baris terakhir dari matriks gauss-jordan ada 1 utama dan baris terakhir, kolom terakhir ada suatu nilai.

3. SPL berbentuk

a. $8x_1 + x_2 + 3x_3 + 2x_4 = 0$
$2x_1 + 9x_2 - x_3 - 2x_4 = 1$
$x_1 + 3x_2 + 2x_3 - x_4 = 2$
$x_1 + 6x_3 + 4x_4 = 3$

Solusi:

```
---------------------------------------------------------
                    Metode
          2. Metode Eliminasi Gauss-Jordan
---------------------------------------------------------
                  Pilihan Sumber:
---------------------------------------------------------
1. Masukan dari CLI
2. Masukan dari file .txt
3. Kembali ke Menu Pilihan Metode
---------------------------------------------------------
Masukkan pilihan sumber Anda:
1
---------------------------------------------------------
                    Sumber
                1. Masukan dari CLI
---------------------------------------------------------
Masukkan jumlah baris:
4
Masukkan jumlah kolom:
5
Masukkan matriks:
Masukkan nilai untuk baris ke-1 kolom ke-1:
8
Masukkan nilai untuk baris ke-1 kolom ke-2:
1
Masukkan nilai untuk baris ke-1 kolom ke-3:
3
Masukkan nilai untuk baris ke-1 kolom ke-4:
2
Masukkan nilai untuk baris ke-1 kolom ke-5:
0
Masukkan nilai untuk baris ke-2 kolom ke-1:
2
Masukkan nilai untuk baris ke-2 kolom ke-2:
9
Masukkan nilai untuk baris ke-2 kolom ke-3:
-1
Masukkan nilai untuk baris ke-2 kolom ke-4:
-2
```

Masukkan nilai untuk baris ke-2 kolom ke-5:
1
Masukkan nilai untuk baris ke-3 kolom ke-1:
1
Masukkan nilai untuk baris ke-3 kolom ke-2:
3
Masukkan nilai untuk baris ke-3 kolom ke-3:
2
Masukkan nilai untuk baris ke-3 kolom ke-4:
-1
Masukkan nilai untuk baris ke-3 kolom ke-5:
2
Masukkan nilai untuk baris ke-4 kolom ke-1:
1
Masukkan nilai untuk baris ke-4 kolom ke-2:
0
Masukkan nilai untuk baris ke-4 kolom ke-3:
6
Masukkan nilai untuk baris ke-4 kolom ke-4:
4
Masukkan nilai untuk baris ke-4 kolom ke-5:
3
Matriks berhasil dibaca.

Matriks yang dibaca:
8.000000 1.000000 3.000000 2.000000 0.000000
2.000000 9.000000 -1.000000 -2.000000 1.000000
1.000000 3.000000 2.000000 -1.000000 2.000000
1.000000 0.000000 6.000000 4.000000 3.000000

Bagi setiap elemen di baris ke-4 dengan 1.00
1.000000 0.000000 0.000000 0.000000 -0.224324
0.000000 1.000000 0.000000 0.000000 0.182432
0.000000 0.000000 1.000000 0.000000 0.709459
0.000000 0.000000 0.000000 1.000000 -0.258108

X1 = -0.22
X2 = 0.18
X3 = 0.71
X4 = -0.26

Solusi unik karena baris terakhir dari matriks gauss-jordan ada 1 utama dan baris terakhir, kolom terakhir ada suatu nilai.

b.

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

.

Solusi:

Karena output sangat besar, kami tidak bisa menunjukkan bagian input dan keadaan awal matriks, kami hanya bisa menunjukkan hasil akhir matriks dan hasil SPL-nya.

Kurangi baris ke-2 dengan 1.000000 kali baris ke-3
0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 34.307340
1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -17.414087
0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 25.763279
0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -0.349192
0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 -4.739062
0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 -14.568278
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 0.000000 -10.893253
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 -3.024217
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 1.000000 0.000000 26.917471
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 34.099229
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 -30.689154
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000

Dengan metode Eliminasi Gauss-Jordan, diperoleh solusi SPL:
SPL tidak Memiliki Solusi

SPL tidak memiliki solusi karena baris terakhir tidak ada 1 utama, dan baris terakhir, kolom terakhir ada suatu nilai.

4. **Studi Kasus Interpolasi**

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	0.4	0.7	0.11	0.14	0.17	0.2	0.23
$f(x)$	0.043	0.005	0. 058	0.072	0.1	0.13	0.147

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) = ?

Solusi:

Hasil solusi persamaan interpolasi polinom dari matriks kalkulator kami adalah

Solusi SPL f(x) = -4212.434532x^6 + 7102.399163x^5 + -4346.313951x^4 + 1220.854891x^3 + -163.915663x^2 + 10.276384x + -0.184559

Sehingga didapatkan

f(0.2) = 0.13
f(0.55) = 2.13
f(0.85) = -66.26
f(1.28) = -3485.14

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 2022 hingga 31 Agustus 2022:

Tanggal	Tanggal (desimal)	Jumlah Kasus Baru
17/06/2022	6,567	12.624
30/06/2022	7	21.807
08/07/2022	7,258	38.391
14/07/2022	7,451	54.517
17/07/2022	7,548	51.952
26/07/2022	7,839	28.228
05/08/2022	8,161	35.764
15/08/2022	8,484	20.813
22/08/2022	8,709	12.408
31/08/2022	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 17/06/2022 (dibaca: 17 Juni 2022) diperoleh tanggal(desimal) sebagai berikut:

$$\text{Tanggal(desimal)} = 6 + (17/30) = 6{,}567$$

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

a. 16/07/2022
b. 10/08/2022
c. 05/09/2022
d. beserta masukan user lainnya berupa **tanggal (desimal) yang sudah diolah** dengan asumsi prediksi selalu dilakukan untuk tahun 2022.

Solusi:

Dari kalkulator matriks kami di dapatkan y dengan persamaan sebagai berikut:

Solusi SPL f(x) = -141018.352722x^9 + 9374584.419448x^8 + -275528781.137900x^7 + 4696794214.320051x^6 + -51143429292.710880x^5 + 368640760971.040800x^4 + -1757276563260.703100x^3 + 5335755560850.102000x^2 + -9350004058993.094000x + 7189658216471.242000

Kami sudah memeriksa solusi SPL tersebut positif dengan tanggal desimal yang ada di data.

a. 16/07/2022 = 7 + 16/31 = 7.516

   Hasil interpolasi f(7.516000) = 53537.728516

   Hasil bisa kami bulatkan menjadi 53538

   Sehingga perkiraan jumlah covid pada tanggal 16/07/2022 adalah 53538.

b. 10/08/2022 = 8 + 10/31 = 8.323

   Hasil interpolasi f(8.323000) = 36295.011719

   Hasil bisa kami bulatkan menjadi 36295

   Sehingga perkiraan jumlah covid pada tanggal 10/08/2022 adalah 36295.

c. 05/09/2022 = 9 + 5/30 = 9.167

   Karena menggunakan solusi SPL yang diatas mengeluarkan hasil yang negatif

   Hasil interpolasi f(9.167000) = -667726.140625

   Kami menggunakan data dengan jangka dari 26/07/2022 - 31/08/2022

   Dari data dengan jangka tersebut di peroleh solsui SPL sebagai berikut

   Solusi SPL f(x) = -78334.512562x^4 + 2743087.382383x^3 + -35955721.950864x^2 + 209065131.967029x + -454920140.520252

   Dengan hasil interpolasi sebagai berikut

   Hasil interpolasi f(9.167000) = 12640.008962

   Hasil bisa kami bulatkan menjadi 12640

   Sehingga perkiraan jumlah covid pada tanggal 05/09/2022 adalah 12640.

d. 20/08/2022 = 8 + 20/31 = 8.645

   Tanggal desimal 8.645 tidak menghasilkan hasil yang negatif menggunakan solusi SPL a dan b, maka itu kami tetap menggunakan solusi SPL tersebut.

Hasil interpolasi f(8.645000) = 8244.109375

Hasil bisa kami bulatkan menjadi 8244

Sehingga perkiraan jumlah covid pada tanggal 20/08/2022 adalah 8244.

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 $h = (2 - 0)/5 = 0.4$.

Solusi:

Dengan n = 5, diperoleh titik-titik sebagai berikut:

(0.4, 0.41888), (0.8, 0.50715), (1.2, 0.56092), (1.6, 0.58368), (2, 0.57665)

```
---------------------------------------------------------
                      Menu
                4. Interpolasi Polinom
---------------------------------------------------------
                   Pilihan Sumber:
---------------------------------------------------------
1. Masukan dari CLI
2. Masukan dari file .txt
---------------------------------------------------------
Masukkan pilihan sumber Anda:
1
---------------------------------------------------------
                      Sumber
                1. Masukan dari CLI
---------------------------------------------------------
Masukkan banyaknya titik sampel (X,Y) = 5
---------------------------------------------------------
Masukkan X1 = 0.4
Masukkan Y1 = 0.41888
---------------------------------------------------------
Masukkan X2 = 0.8
Masukkan Y2 = 0.50715
---------------------------------------------------------
Masukkan X3 = 1.2
Masukkan Y3 = 0.56092
```

```
---------------------------------------------------------
Masukkan X4 = 1.6
Masukkan Y4 = 0.58368
---------------------------------------------------------
Masukkan X5 = 2
Masukkan Y5 = 0.57665
---------------------------------------------------------

Matriks koefisien dari persamaan derajat 5 :
1.000000 0.400000 0.160000 0.064000 0.025600 0.418880
1.000000 0.800000 0.640000 0.512000 0.409600 0.507150
1.000000 1.200000 1.440000 1.728000 2.073600 0.560920
1.000000 1.600000 2.560000 4.096000 6.553600 0.583680
1.000000 2.000000 4.000000 8.000000 16.000000 0.576650

Kurangi baris ke-1 dengan 0.400000 kali baris ke-2
1.000000 0.000000 0.000000 0.000000 0.000000 0.290350
0.000000 1.000000 0.000000 0.000000 0.000000 0.377869
0.000000 0.000000 1.000000 0.000000 0.000000 -0.150315
0.000000 0.000000 0.000000 1.000000 0.000000 0.023867
0.000000 0.000000 0.000000 0.000000 1.000000 -0.003695

---------------------------------------------------------
Solusi SPL f(x) = -0.003695x^4 + 0.023867x^3 + -0.150315x^2 + 0.377869x + 0.290350
---------------------------------------------------------

 y = 0.290350 + 0.377869x + -0.150315x^2 + 0.023867x^3 - 0.003695x^4
```

4. **Studi Kasus Interpolasi Bicubic**

Diberikan matriks input:

$$\begin{bmatrix} 153 & 59 & 210 & 96 \\ 125 & 161 & 72 & 81 \\ 98 & 101 & 42 & 12 \\ 21 & 51 & 0 & 16 \end{bmatrix}$$

Tentukan nilai:

f(0,0) = ?

f(0.5, 0.5) = ?

f(0.25, 0.75) = ?

f(0.1, 0.9) = ?

Solusi:

Dengan kalkulator matriks kami di dapatkan nilai-nilai sebagai berikut:

f(0.000000,0.000000) = 161.000000

f(0.500000,0.500000) = 97.726562

f(0.250000,0.750000) = 105.514771

f(0.100000,0.900000) = 104.229119

5. 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.

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

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

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

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

Solusi:

```
----------------------------------------------------
                    Menu
            5. Regresi Linier Berganda
----------------------------------------------------
                  Pilihan Sumber:
----------------------------------------------------
1. Masukan dari CLI
```

2. Masukan dari file .txt

Masukkan pilihan sumber Anda:
1

Sumber
1. Masukan dari CLI

Masukkan banyak sampel n : 20
Masukkan banyak varibel x : 3

Masukkan nilai X1 untuk sampel 1 : 72.4
Masukkan nilai X2 untuk sampel 1 : 76.3
Masukkan nilai X3 untuk sampel 1 : 29.18
Masukkan nilai y untuk sampel 1 : 0.9

Masukkan nilai X1 untuk sampel 2 : 41.6
Masukkan nilai X2 untuk sampel 2 : 70.3
Masukkan nilai X3 untuk sampel 2 : 29.35
Masukkan nilai y untuk sampel 2 : 0.91

Masukkan nilai X1 untuk sampel 3 : 34.3
Masukkan nilai X2 untuk sampel 3 : 77.1
Masukkan nilai X3 untuk sampel 3 : 29.24
Masukkan nilai y untuk sampel 3 : 0.96

Masukkan nilai X1 untuk sampel 4 : 35.1
Masukkan nilai X2 untuk sampel 4 : 68.0
Masukkan nilai X3 untuk sampel 4 : 29.27
Masukkan nilai y untuk sampel 4 : 0.89

Masukkan nilai X1 untuk sampel 5 : 10.7
Masukkan nilai X2 untuk sampel 5 : 79.0
Masukkan nilai X3 untuk sampel 5 : 29.78
Masukkan nilai y untuk sampel 5 : 1.00

Masukkan nilai X1 untuk sampel 6 : 12.9
Masukkan nilai X2 untuk sampel 6 : 67.4
Masukkan nilai X3 untuk sampel 6 : 29.39
Masukkan nilai y untuk sampel 6 : 1.10

Masukkan nilai X1 untuk sampel 7 : 8.3
Masukkan nilai X2 untuk sampel 7 : 66.8
Masukkan nilai X3 untuk sampel 7 : 29.69
Masukkan nilai y untuk sampel 7 : 1.15

Masukkan nilai X1 untuk sampel 8 : 20.1
Masukkan nilai X2 untuk sampel 8 : 76.9
Masukkan nilai X3 untuk sampel 8 : 29.48

Masukkan nilai y untuk sampel 8 : 1.03

Masukkan nilai X1 untuk sampel 9 : 72.2
Masukkan nilai X2 untuk sampel 9 : 77.7
Masukkan nilai X3 untuk sampel 9 : 29.09
Masukkan nilai y untuk sampel 9 : 0.77

Masukkan nilai X1 untuk sampel 10 : 24.0
Masukkan nilai X2 untuk sampel 10 : 67.7
Masukkan nilai X3 untuk sampel 10 : 29.60
Masukkan nilai y untuk sampel 10 : 1.07

Masukkan nilai X1 untuk sampel 11 : 23.2
Masukkan nilai X2 untuk sampel 11 : 76.8
Masukkan nilai X3 untuk sampel 11 : 29.38
Masukkan nilai y untuk sampel 11 : 1.07

Masukkan nilai X1 untuk sampel 12 : 47.4
Masukkan nilai X2 untuk sampel 12 : 86.6
Masukkan nilai X3 untuk sampel 12 : 29.35
Masukkan nilai y untuk sampel 12 : 0.94

Masukkan nilai X1 untuk sampel 13 : 31.5
Masukkan nilai X2 untuk sampel 13 : 76.9
Masukkan nilai X3 untuk sampel 13 : 29.63
Masukkan nilai y untuk sampel 13 : 1.10

Masukkan nilai X1 untuk sampel 14 : 10.6
Masukkan nilai X2 untuk sampel 14 : 86.3
Masukkan nilai X3 untuk sampel 14 : 29.56
Masukkan nilai y untuk sampel 14 : 1.10

Masukkan nilai X1 untuk sampel 15 : 11.2
Masukkan nilai X2 untuk sampel 15 : 86.0
Masukkan nilai X3 untuk sampel 15 : 29.48
Masukkan nilai y untuk sampel 15 : 1.10

Masukkan nilai X1 untuk sampel 16 : 73.3
Masukkan nilai X2 untuk sampel 16 : 76.3
Masukkan nilai X3 untuk sampel 16 : 29.40
Masukkan nilai y untuk sampel 16 : 0.91

Masukkan nilai X1 untuk sampel 17 : 75.4
Masukkan nilai X2 untuk sampel 17 : 77.9
Masukkan nilai X3 untuk sampel 17 : 29.28
Masukkan nilai y untuk sampel 17 : 0.87

Masukkan nilai X1 untuk sampel 18 : 96.6
Masukkan nilai X2 untuk sampel 18 : 78.7

Masukkan nilai X3 untuk sampel 18 : 29.29
Masukkan nilai y untuk sampel 18 : 0.78

Masukkan nilai X1 untuk sampel 19 : 107.4
Masukkan nilai X2 untuk sampel 19 : 86.8
Masukkan nilai X3 untuk sampel 19 : 29.03
Masukkan nilai y untuk sampel 19 : 0.82

Masukkan nilai X1 untuk sampel 20 : 54.9
Masukkan nilai X2 untuk sampel 20 : 70.9
Masukkan nilai X3 untuk sampel 20 : 29.37
Masukkan nilai y untuk sampel 20 : 0.95

Masukkan nilai-nilai X yang akan ditaksir...

Nilai X1 : 50.0
Nilai X2 : 76.0
Nilai X3 : 29.3

Kurangi baris ke-1 dengan 43.155000 kali baris ke-2
1.000000 0.000000 0.000000 0.000000 -3.507778
0.000000 1.000000 0.000000 0.000000 -0.002625
0.000000 0.000000 1.000000 0.000000 0.000799
0.000000 0.000000 0.000000 1.000000 0.154155

Maka didapatkan persamaan..

f(x) = -3.507778 + (-0.002625*X1) + (0.000799*X2) + (0.154155*X3)

y = -3.507778 + (-0.002625) x (50.000000) + (0.000799) x (76.000000) + (0.154155) x (29.300000) = 0.938434

Perkiraan nilai nitrous oxide adalah 0,9384268, yang ditentukan dari persamaan linier dengan mensubstitusi nilai 1 = 50 (kelembaban 50%), 2 = 76 (suhu 76°F), dan 3 = 29,3 (tekanan udara 29,30).

BAB 5: KESIMPULAN DAN SARAN

Dari pengerjaan tugas besar pertama mata kuliah Aljabar Linier dan Geometri (IF2123) ini, kelompok kami berhasil membuat suatu program yang mampu melakukan perhitungan permasalahan linier, matriks, matriks balikan, interpolasi polinom, interpolasi bicubic, dan regresi linier berganda. Dengan menggunakan metode eliminasi Gauss/Gauss-Jordan, matriks invers, dan kaidah Cramer.

Namun dari semua program ada kekurangannya masing-masing. Contohnya adalah program kami ada keterbatasan di solusi paramterik yaitu hanya sampai 26 variabel maksimal karena di spesifikasi hanya disuru pakai alfabet, sedangkan alfabet hanya berisi 26, namun apabila simbol bisa ditambah variabel sekaligus juga bisa bertambah. Program kami juga kurang *modular* dalam arti, ada beberapa fungsi yang hanya dipanggil sekali, dan ada beberapa kode yang bisa dijadikan fungsi karena sering dipanggil, alhasil baris program lebih banyak. Program juga belum bisa menyelesaikan semua tes kasus interpolasi polinom, seperti pada studi tes kasus 4b bagian d.

Tugas ini memberi pelajaran pentingnya kerja sama dan komunikasi secara tim. Karena setiap program mempunyai kesulitannya masing-masing, dan tidak mungkin untuk kedepannya setiap program yang memiliki tingkat kesulitan yang tinggi dibikin sendiri-sendiri. Penting untuk bisa bekerja sama, berkoordinasi, dan menjalin komunikasi yang baik dengan programmer-programmer lain. Semoga pengalaman yang didapatkan dari tugas besar pertama ini bisa berguna untuk masa depan.

DAFTAR PUSTAKA

https://www.geeksforgeeks.org/java/

LINK GITHUB

Link *github* untuk repository kami:

https://github.com/semifinal-com/Algeo01-21086