LAPORAN TUGAS BESAR 1
IF2123 ALJABAR LINIER DAN GEOMETRI

Disusun Oleh:

Kelompok 34 - Besok Minggu

Muhammad Fauzan Azhim (13522153)
Muhammad Davis Adhipramana (13522157)
Valentino Chryslie Triadi (13522164)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

SEMESTER 1 TAHUN 2022/2023

DAFTAR ISI

Bab I
Deskripsi Masalah

1. Buatlah pustaka (*library* atau *package*) dalam **Bahasa Java** untuk menemukan solusi SPL dengan metode eliminasi Gauss, metode Eliminasi Gauss-Jordan, metode matriks balikan, dan kaidah *Cramer* (kaidah *Cramer* khusus untuk SPL dengan *n* peubah dan *n* persamaan), menghitung determinan matriks dengan reduksi baris dan dengan ekspansi kofaktor, dan menghitung balikan matriks.
2. Gunakan pustaka di atas untuk membuat program penyelesaian berbagai persoalan dalam bentuk SPL, menyelesaikan persoalan interpolasi dan regresi linier, menghitung matriks balikan, menghitung determinan matriks dengan berbagai metode (reduksi baris dan ekspansi kofaktor).

Spesifikasi program adalah sebagai berikut:

1. Program dapat menerima masukan (*input*) baik dari *keyboard* maupun membaca masukan dari *file text*. Untuk SPL, masukan dari *keyboard* adalah *m*, *n*, koefisien a_{ij} , dan b_i. Masukan dari *file* berbentuk matriks *augmented* tanpa tanda kurung, setiap elemen matriks dipisah oleh spasi. Misalnya,

   3 4.5 2.8 10 12

   -3 7 8.3 11 -4

   0.5 -10 -9 12 0

2. Untuk persoalan menghitung determinan dan matriks balikan, masukan dari *keyboard* adalah *n* dan koefisien a_{ij}. Masukan dari *file* berbentuk matriks, setiap elemen matriks dipisah oleh spasi. Misalnya,

   3 4.5 2.8

   -3 7 8.3

   0.5 -10 -9

   Luaran (*output*) disesuaikan dengan persoalan (determinan atau invers) dan penghitungan balikan/invers dilakukan dengan metode matriks balikan dan adjoin.

3. Untuk persoalan interpolasi, masukannya jika dari *keyboard* adalah *n*, $(x_0, y_0), (x_1, y_1)$, …, (x_n, y_n), dan nilai *x* yang akan ditaksir nilai fungsinya. Jika masukannya dari *file*, maka titik-titik dinyatakan pada setiap baris tanpa koma dan tanda kurung. Masukan kemudian dilanjutkan dengan satu buah baris berisi satu buah nilai *x* yang akan ditaksir menggunakan fungsi interpolasi yang telah didefinisikan. Misalnya jika titik-titik datanya adalah (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513) dan akan mencari nilai *y* saat $x = 8.3$, maka di dalam *file text* ditulis sebagai berikut:

8.0 2.0794

9.0 2.1972

9.5 2.2513

8.3

4. Untuk persoalan regresi, masukannya jika dari *keyboard* adalah n (jumlah peubah x), m (jumlah sampel), semua nilai-nilai x_{1i}, x_{2i}, …, x_{ni}, nilai y_i, dan nilai-nilai x_k yang akan ditaksir nilai fungsinya. Jika masukannya dari *file*, maka titik-titik dinyatakan pada setiap baris tanpa koma dan tanda kurung.
5. Untuk persoalan SPL, luaran program adalah solusi SPL. Jika solusinya tunggal, tuliskan nilainya. Jika solusinya tidak ada, tuliskan solusi tidak ada, jika solusinya banyak, maka tuliskan solusinya dalam bentuk parametrik (misalnya $x_4 = -2$, $x_3 = 2s - t$, $x_2 = s$, dan $x_1 = t$).
6. Untuk persoalan polinom interpolasi dan regresi, luarannya adalah persamaan polinom/regresi dan taksiran nilai fungsi pada x yang diberikan. Contoh luaran untuk interpolasi adalah

   $$f(x) = -0.0064x^2 + 0.2266x + 0.6762, \quad f(5) = \ldots$$

   dan untuk regresi adalah

   $$f(x) = -9.5872 + 1.0732x_1, \quad f(x_k) = \ldots$$
7. Untuk persoalan *bicubic spline interpolation*, masukan dari *file text* (.txt) yang berisi matriks berukuran 4 x 4 yang berisi konfigurasi nilai fungsi dan turunan berarah disekitarnya, diikuti dengan nilai a dan b untuk mencari nilai $f(a, b)$.

   Misalnya jika nilai dari $f(0, 0)$, $f(1, 0)$, $f(0, 1)$, $f(1, 1)$, $f_x(0, 0)$, $f_x(1, 0)$, $f_x(0, 1)$, $f_x(1, 1)$, $f_y(0, 0)$, $f_y(1, 0)$, $f_y(0, 1)$, $f_y(1, 1)$, $f_{xy}(0, 0)$, $f_{xy}(1, 0)$, $f_{xy}(0, 1)$, $f_{xy}(1, 1)$ berturut-turut adalah 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 serta nilai a dan b yang dicari berturut-turut adalah 0.5 dan 0.5 maka isi *file text* ditulis sebagai berikut:

   1 2 3 4

   5 6 7 8

   9 10 11 12

   13 14 15 16

   0.5 0.5

   Luaran yang dihasilkan adalah nilai dari $f(0.5, 0.5)$.
8. Luaran program harus dapat ditampilkan **pada layar komputer dan dapat disimpan ke dalam *file***.
9. Bahasa program yang digunakan adalah Java. Anda bebas untuk menggunakan versi java apapun dengan catatan di atas java versi 8 (8/9/11/15/17/19/20).
10. Program **tidak harus** berbasis GUI, cukup *text-based* saja, namun boleh menggunakan GUI (memakai kakas *Eclipse* misalnya).
11. Program dapat dibuat dengan pilihan menu. Urutan menu dan isinya dipersilakan dirancang masing-masing. Misalnya, menu:

    MENU

    1. Sistem Persamaaan Linier

2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic Spline
6. Regresi linier berganda
7. Keluar

Untuk pilihan menu nomor 1 ada sub-menu lagi yaitu pilihan metode:

1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer

Begitu juga untuk pilihan menu nomor 2 dan 3.

BAB II
Teori Singkat

I. Sistem Persamaan Linier

Sistem persamaan linier (SPL) banyak ditemukan di dalam bidang sains dan rekayasa. 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). Terdapat 3 jenis solusi SPL, yaitu tidak ada solusi, banyak solusi (tidak berhingga), atau hanya satu solusi (unik/tunggal).

$$\begin{bmatrix} 0 & \mathbf{2} & 1 & -1 \\ 0 & 0 & \mathbf{3} & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad \begin{bmatrix} 0 & \mathbf{1} & 0 & -\frac{2}{3} \\ 0 & 0 & \mathbf{1} & \frac{1}{3} \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Gambar 1. Eliminasi Gauss dilakukan dengan matriks eselon baris dan eliminasi Gauss-Jordan dengan matriks eselon baris tereduksi.

II. OBE, Eliminasi Gauss, dan Eliminasi Gauss-Jordan

Operasi baris elementer (OBE) merupakan operasi aritmatika (penjumlahan dan perkalian) yang dikenakan pada setiap unsur dalam suatu baris pada sebuah matriks. Operasi baris elementer meliputi :

1. Pertukaran Baris
2. Perkalian suatu baris dengan konstanta tak nol
3. Penjumlahan suatu baris dan baris lainnya yang sudah dikalikan suatu konstanta tak nol.

Tujuan dilakukan operasi baris elementer pada suatu matriks adalah menghasilkan matriks eselon baris yang memenuhi beberapa sifat berikut :

1. Dalam suatu baris dengan elemen tak nol maka elemen tak nol pertama adalah 1 (satu utama).
2. Urutan atau susunan baris yang lebih rendah memuat 1 utama yang lebih ke kanan dibanding baris yang lebih tinggi.
3. Jika ada baris nol (baris yang semua elemennya nol), maka ia diletakkan pada baris paling bawah.
4. Pada kolom yang memuat unsur 1 utama, maka elemen yang lainnya adalah nol.

Jika sifat 1, 2, dan 3 terpenuhi maka akan terbentuk matriks eselon baris dengan proses yang dijalankan adalah **eliminasi gauss**. Jika sifat 1, 2, dan 3 terpenuhi dan sifat ke-4 juga terpenuhi, maka matriks hasil OBE berbentuk matriks eselon baris tereduksi dengan proses yang dijalankan adalah **eliminasi gauss jordan**.

III. Kaidah Cramer

Kaidah Cramer dapat menyelesaikan SPL dengan menggunakan determinan. Jika Ax = b adalah SPL yang terdiri dari n persamaan linier dengan n peubah (variabel) dan det(A) ≠ 0, maka SPL tersebut memiliki solusi yang unik yaitu:

$$x1 = \frac{det(A1)}{det(A)}, x2 = \frac{det(A2)}{det(A)}, x3 = \frac{det(A3)}{det(A)}, \dots, xn = \frac{det(An)}{det(A)}$$

IV. Matriks Kofaktor dan Adjoin

Misalkan A adalah matriks berukuran n x n

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

Didefinisikan:

M_{ij} = minor entri aij dan merupakan determinan submatrix yang elemen-elemennya tidak berada pada baris i dan kolom j

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

Misalkan C adalah matriks kofaktor dari A, maka C dapat digambarkan sebagai berikut:

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

Adjoin dari A adalah transpose matriks C dengan C adalah matriks kofaktor dari A :

$$adj(a) = C^T$$

V. Matriks Balikan

Matriks Balikan dapat dicari dengan menggunakan 2 metode, yaitu metode adjoin dan metode matriks identitas. Matriks balikan hanya bisa didapat jika matriks input merupakan matriks persegi. Metode adjoin menggunakan matriks kofaktor dan adjoin dengan persamaan sebagai berikut:

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

Keterangan:

A^{-1} = *Matriks balikan dari A*

$det(A)$ = *determinan dari matriks A*, $det(A) \neq 0$

$adj(A)$ = *adjoin dari matriks A*

Metode selanjutnya adalah dengan menggunakan matriks identitas sebagai matriks augmented dan melakukan OBE hingga matriks utama berubah menjadi matriks identitas. Matriks augmented hasil OBE merupakan balikan dari matriks A. Untuk lebih jelasnya dapat melihat gambar berikut:

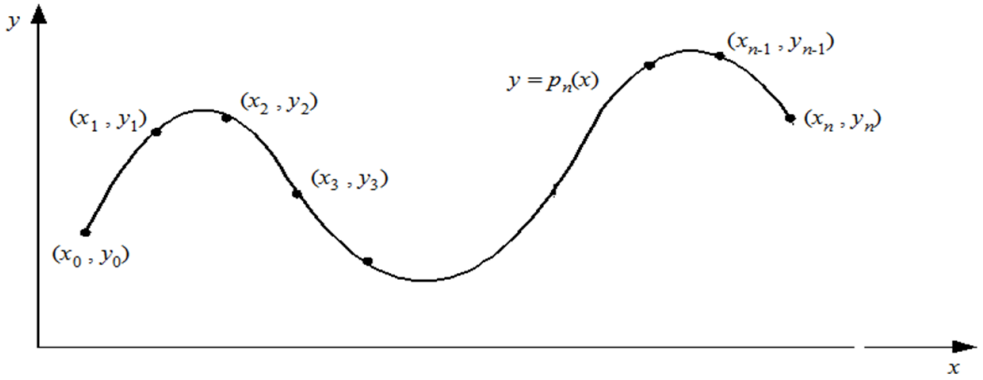
$$[A|I] \overset{G\text{-}J}{\sim} [I|A^{-1}]$$

Keterangan:

$G - J$ = *Operasi Gauss Jordan* (OBE)

VI. Interpolasi Polinom

Persoalan interpolasi polinom adalah sebagai berikut: Diberikan n+1 buah titik berbeda, (x_0, y_0), (x_1, y_1), ..., (x_n, y_n). Tentukan polinom $p_n(x)$ yang menginterpolasi (melewati) semua titik-titik tersebut sedemikian rupa sehingga $y_i = p_n(x_i)$ untuk $i = 0, 1, 2, \ldots, n$.



Gambar 2. Ilustrasi beberapa titik yang diinterpolasi secara polinomial.

Setelah polinom interpolasi $p_n(x)$ ditemukan, $p_n(x)$ dapat digunakan untuk menghitung perkiraan nilai y di sembarang titik di dalam selang $[x_0, x_n]$.

Polinom interpolasi derajat n yang menginterpolasi titik-titik (x_0, y_0), (x_1, y_1), ..., (x_n, y_n). adalah berbentuk $p_n(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$. Jika hanya ada dua titik, (x_0, y_0) dan (x_1, y_1), maka polinom yang menginterpolasi kedua titik tersebut adalah $p_1(x) = a_0 + a_1x$ yaitu berupa persamaan garis lurus. Jika tersedia tiga titik, (x_0, y_0), (x_1, y_1), dan (x_2, y_2), maka polinom yang menginterpolasi ketiga titik tersebut adalah $p_2(x) = a_0 + a_1x + a_2x^2$ atau persaman kuadrat dan kurvanya berupa parabola. Jika tersedia empat titik, (x_0, y_0), (x_1, y_1), (x_2, y_2), dan (x_3, y_3), polinom yang menginterpolasi keempat titik tersebut adalah $p_3(x) = a_0 + a_1x + a_2x^2 + a_3x^3$, demikian seterusnya. Dengan cara yang sama kita dapat membuat polinom interpolasi berderajat n untuk n yang lebih tinggi asalkan tersedia $(n+1)$ buah titik data. Dengan menyulihkan (x_i, y_i) ke dalam persamaan polinom $p_n(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ untuk $i = 0, 1, 2, \ldots, n$, akan diperoleh n buah sistem persamaan lanjar dalam $a_0, a_1, a_2, \ldots, a_n$,

$$a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_n x_0^n = y_0$$
$$a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_n x_1^n = y_1$$
$$\ldots \qquad \ldots$$
$$a_0 + a_1 x_n + a_2 x_n^2 + \ldots + a_n x_n^n = y_n$$

Solusi sistem persamaan lanjar ini, yaitu nilai a_0, a_1, …, a_n, diperoleh dengan menggunakan metode eliminasi Gauss yang sudah anda pelajari. Sebagai contoh, misalkan diberikan tiga buah titik yaitu (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513). Tentukan polinom interpolasi kuadratik lalu estimasi nilai fungsi pada x = 9.2. Polinom kuadratik berbentuk $p_2(x) = a_0 + a_1 x + a_2 x^2$. Dengan menyulihkan ketiga buah titik data ke dalam polinom tersebut, diperoleh sistem persamaan lanjar yang terbentuk adalah

$$a_0 + 8.0a_1 + 64.00a_2 = 2.0794$$
$$a_0 + 9.0a_1 + 81.00a_2 = 2.1972$$
$$a_0 + 9.5a_1 + 90.25a_2 = 2.2513$$

Penyelesaian sistem persamaan dengan metode eliminasi Gauss menghasilkan a_0 = 0.6762, a_1 = 0.2266, dan a_2 = -0.0064. Polinom interpolasi yang melalui ketiga buah titik tersebut adalah $p_2(x) = 0.6762 + 0.2266x - 0.0064x^2$. Dengan menggunakan polinom ini, maka nilai fungsi pada x = 9.2 dapat ditaksir sebagai berikut: $p_2(9.2) = 0.6762 + 0.2266(9.2) - 0.0064(9.2)^2 = 2.2192$.

VII. Regresi Linear Berganda

Regresi Linear (akan dipelajari lebih lanjut di Probabilitas dan Statistika) merupakan salah satu metode untuk memprediksi nilai selain menggunakan Interpolasi Polinom. Meskipun sudah ada persamaan jadi untuk menghitung regresi linear sederhana, terdapat persamaan umum dari regresi linear yang bisa digunakan untuk regresi linear berganda, yaitu.

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

Untuk mendapatkan nilai dari setiap β_i dapat digunakan *Normal Estimation Equation for Multiple Linear Regression* sebagai berikut:

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

Sistem persamaan linier tersebut diselesaikan dengan menggunakan metode eliminasi Gauss.

VIII. Bicubic Spline Interpolation

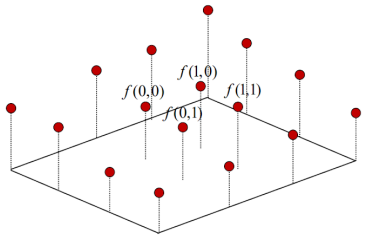
Bicubic spline interpolation adalah metode interpolasi yang digunakan untuk mengaproksimasi fungsi di antara titik-titik data yang diketahui. *Bicubic spline interpolation* melibatkan konsep *spline* dan konstruksi serangkaian polinomial kubik di dalam setiap sel segi empat dari data yang diberikan. Pendekatan ini menciptakan permukaan yang halus dan kontinu, memungkinkan untuk perluasan data secara visual yang lebih akurat daripada metode interpolasi linear.

Dalam pemrosesan menggunakan interpolasi *bicubic spline* digunakan 16 buah titik, 4 titik referensi utama di bagian pusat, dan 12 titik di sekitarnya sebagai aproksimasi turunan dari keempat titik referensi untuk membagun permukaan bikubik. Bentuk pemodelannya adalah sebagai berikut.

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

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

Solve: a_{ij}



Gambar 3. Pemodelan interpolasi *bicubic spline*.

Selain melibatkan model dasar, juga digunakan model turunan berarah dari kedua sumbu, baik terhadap sumbu *x*, sumbu *y,* mapun keduanya. Persamaan polinomial yang digunakan adalah sebagai berikut.

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

$$f_x(x,y) = \sum_{j=0}^{3}\sum_{i=1}^{3} a_{ij} i x^{i-1} y^j$$

$$f_y(x,y) = \sum_{j=1}^{3}\sum_{i=0}^{3} a_{ij} j x^i y^{j-1}$$

$$f_{xy}(x,y) = \sum_{j=0}^{3}\sum_{i=0}^{3} a_{ij} ij x^{i-1} y^{j-1}$$

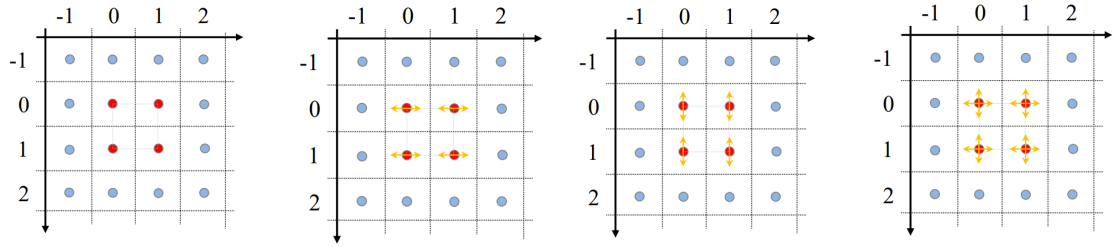
Dengan menggunakan nilai fungsi dan turunan berarah tersebut, dapat terbentuk sebuah matriks solusi *X* yang membentuk persamaan penyelesaian sebagai berikut.

$$y = Xa$$

$$\begin{bmatrix} f(0,0) \\ f(1,0) \\ f(0,1) \\ f(1,1) \\ f_x(0,0) \\ f_x(1,0) \\ f_x(0,1) \\ f_x(1,1) \\ f_y(0,0) \\ f_y(1,0) \\ f_y(0,1) \\ f_y(1,1) \\ f_{xy}(0,0) \\ f_{xy}(1,0) \\ f_{xy}(0,1) \\ f_{xy}(1,1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 2 & 4 & 6 & 0 & 3 & 6 & 9 \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{10} \\ a_{20} \\ a_{30} \\ a_{01} \\ a_{11} \\ a_{21} \\ a_{31} \\ a_{02} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{03} \\ a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}$$

Perlu diketahui bahwa elemen pada matriks X adalah nilai dari setiap komponen koefisien a_{ij} yang diperoleh dari persamaan fungsi maupun persamaan turunan yang telah dijelaskan sebelumnya. Sebagai contoh, elemen matriks X pada baris 8 kolom ke 2 adalah koefisien dari a_{10} pada ekspansi sigma untuk $f_x(1, 1)$ sehingga diperoleh nilai konstanta $1 \times 1^{1-1} \times 1^0 = 1$, sesuai dengan isi matriks X.

Nilai dari vektor a dapat dicari dari persamaan $y = Xa$, lalu vektor a tersebut digunakan sebagai nilai variabel dalam $f(x, y)$, sehingga terbentuk fungsi interpolasi bicubic sesuai model. Tugas Anda pada studi kasus ini adalah membangun persamaan $f(x, y)$ yang akan digunakan untuk melakukan interpolasi berdasarkan nilai $f(a, b)$ dari masukan matriks 4 x 4. Nilai masukan a dan b berada dalam rentang [0, 1]. Nilai yang akan diinterpolasi dan turunan berarah disekitarnya dapat diilustrasikan pada titik berwarna merah pada gambar di bawah.



Gambar 4. Nilai fungsi yang akan di interpolasi pada titik merah, turunan berarah terhadap sumbu x, terhadap sumbu y, dan keduanya (kiri ke kanan).

IX. Image Scaler

Seperti yang telah dijelaskan sebelumnya bahwa interpolasi *bicubic spline* dapat digunakan untuk menciptakan permukaan yang halus pada gambar. Oleh karena itu, selain persamaan dasar $y = Xa$ yang telah dijabarkan, persamaan ini juga dapat menggunakan data sebuah citra untuk menciptakan kualitas gambar yang lebih baik. Misalkan $I(x, y)$ merupakan nilai dari suatu citra gambar pada posisi (x, y), maka dapat digunakan persamaan nilai dan persamaan turunan berarah sebagai berikut.

$$f(x,y) = I(x,y)$$

$$f_x(x,y) = [I(x+1,y) - I(x-1,y)]/2$$

$$f_y(x,y) = [I(x,y+1) - I(x,y-1)]/2$$

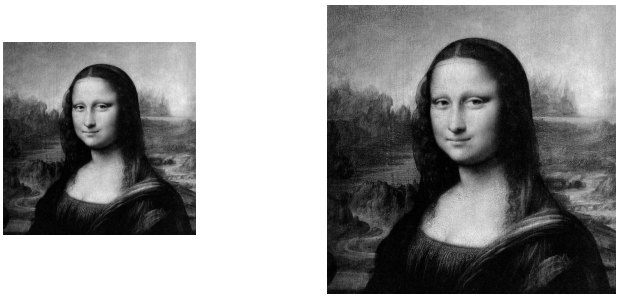
$$f_{xy}(x,y) = [I(x+1,y+1) - I(x-1,y) - I(x,y-1) - I(x,y)]/4$$

Sistem persamaan tersebut dapat dipetakan menjadi sebuah matriks (dalam hal ini matriks *D*) dengan gambaran lengkap seperti yang tertera di bawah.

$$y = DI$$

$$\begin{bmatrix} f(0,0) \\ f(1,0) \\ f(0,1) \\ f(1,1) \\ f_x(0,0) \\ f_x(1,0) \\ f_x(0,1) \\ f_x(1,1) \\ f_y(0,0) \\ f_y(1,0) \\ f_y(0,1) \\ f_y(1,1) \\ f_{xy}(0,0) \\ f_{xy}(1,0) \\ f_{xy}(0,1) \\ f_{xy}(1,1) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I(-1,-1) \\ I(0,-1) \\ I(1,-1) \\ I(2,-1) \\ I(-1,0) \\ I(0,0) \\ I(1,0) \\ I(2,0) \\ I(-1,1) \\ I(0,1) \\ I(1,1) \\ I(2,1) \\ I(-1,2) \\ I(0,2) \\ I(1,2) \\ I(2,2) \end{bmatrix}$$

Dengan menggunakan kedua persamaan nilai *y* yang telah disebutkan dan dibahas sebelumnya, dapatkan nilai *a* yang lebih baik dan akurat dalam pemrosesan citra gambar, kemudian gunakan nilai dan persamaan *f*(*x, y*) yang terbentuk untuk memperbaiki kualitas citra gambar monokrom pasca perbesaran dengan skala tertentu dengan melakukan interpolasi *bicubic spline*. Berikut adalah contohnya.



Gambar 5. Sebuah citra gambar asal (kiri) dan hasil perbesarannya dengan skala 1.5 (kanan).

Bab III
Implementasi Pustaka Dan Program Java

Pada tugas besar kali ini, kami membuat 2 buah package. Dengan package pertama bernama "besokminggu.algeotubes" dan package kedua bernama "besokminggu.fungsialgeo". Pada package pertama berisi 21 class yang akan menampilkan kode kami dalam bentuk GUI menggunakan library javax.swing, java.io, java.awt, dan java.util. Dalam package kedua berisi fungsi-fungsi yang akan digunakan untuk mencari dan mengolah data input serta mengeluarkan output dengan format yang diinginkan.

3.1. besokminggu.algeotubes

Pada Package ini kami menampilkan GUI menggunakan library javax.swing, java.io, java.awt, dan java.util. Terdapat 21 file java yang masing-masing berfungsi untuk menampilkan program dan menerima input serta mengeluarkan output sesuai dengan yang sudah dipilih sebelumnya. Program ini terdiri dari komponen Main Menu, Submenu SPL, Submenu Determinan, Submenu Inverse, Interpolasi Polinom, Regresi Linier berganda, Interpolasi bicubic spline, Menu input from keyboard, dan Menu input From File.

3.2. besokminggu.fungsialgeo

3.2.1 Determinan.java

Pada class ini dikumpulkan fungsi-fungsi yang berhubungan dengan mencari determinan.

Fungsi/Prosedur	Deskripsi
getDeterminanOutput (double[][] matriks, String Function)	Mendapatkan output dalam string sesuai dengan cara yang diinginkan.
determinangauss(double[][] matrix)	Mencari determinan menggunakan cara eliminasi gauss dan mengalikan diagonal utamanya.
determinan_kofaktor(double[][] m)	Mencari determinan menggunakan cara ekspansi kofaktor.
DeterminanNxN(double[][] matriks)	Mencari determinan menggunakan cara sarrus, khusus untuk 2x2 atau 3x3.

3.2.2 Inverse.java

Pada class ini dikumpulkan fungsi-fungsi yang berhubungan dengan mencari invers.

Fungsi/Prosedur	Deskripsi
getInverseOutput(double[][] matriks, String Function)	Mendapatkan output dalam string sesuai dengan cara yang diinginkan.
MatriksIdentity_Maker(double[][] matrix)	Membuat matriks identitas sebesar panjang matriks yang diinput.
Add_MatrixIdentity(double[][] matrix)	Membuat array 2D yang menggabungkan matriks input dengan matriks identitas yang sudah dibuat sebelumnya
zeroCounter(double[][] matrix, int row)	Menghitung jumlah kemunculan angka 0 pada sisi kiri matrix[row].
swapping_Operation(double[][] matrix)	Mencari row yang dapat di swap untuk mendapatkan leading 1 dan menjalankan fungsi swapping row.
matrix_Swapping(double[][] matrix, int row_will_be_changed, int row_who_changed)	Menukar row dari matrix dengan row lain di matriks yang sama.
Multipy_Operation(double[][] matrix, int row_will_be_changed, int column_will_be_changed)	Menjadikan suatu elemen matriks pada row dan column tertentu menjadi 1 agar menjadi leading 1
Reduce_Operation(double[][] matrix, int row_will_be_changed,int column_will_be_changed, int row_who_changed)	Mengurangi matrix pada row tertentu sehingga menjadi 0 dan membentuk pola matriks eliminasi gauss.
Gauss_Operation(double[][] matrix)	Melakukan operasi gauss jordan sehingga matriks saat ini menjadi matriks identitas.
get_Inverse_Matriks_fromIdentity(double[][] matrix)	Fungsi utama untuk menjalankan fungsi matriks invers menggunakan cara matriks balikan.
getInverseFromAdjoin(double[][] matriks)	Fungsi utama untuk mendapatkan matriks dari adjoin
getKofaktor(double[][]matriks)	Fungsi mendapatkan Kofaktor dari matriks
getKofaktorValue(double[][] matriks, int x, int y)	Fungsi mendapatkan determinan menggunakan teknik kofaktor
getTranspose(double[][] matriks)	Fungsi mendapatkan matriks yang telah di

	transpose.

3.2.3 SPL.java

Pada class ini dikumpulkan fungsi-fungsi yang berhubungan dengan mencari Sistem persamaan linear.

Fungsi/Prosedur	Deskripsi
SPLCramer(double[][] matrix)	Mendapatkan output SPL menggunakan teknik Cramer
SPL_From_Inverse(double[][] matrix_input)	Mendapatkan output SPL menggunakan teknik inverse
perkalian_Matriks(double[][] matriks1, double[][] matriks2)	Mereturn suatu matriks yang merupakan perkalian antara matriks1 dan matriks2
String_Converter(double[][] matrix)	Mengembalikan String sebagai output untuk solusi matriks_balikan dan spl solusi tunggal
Multipy_Operation(double[][] matrix, int row_will_be_changed, int column_will_be_changed)	Menjadikan suatu elemen matriks pada row dan column tertentu menjadi 1 agar menjadi leading 1
Reduce_Operation(double[][] matrix, int row_will_be_changed,int column_will_be_changed, int row_who_changed)	Mengurangi matrix pada row tertentu sehingga menjadi 0 dan membentuk pola matriks eliminasi gauss.
zeroCounter(double[][] matrix, int row)	mengembalikan jumlah 0 yang ditemukan pada row yang dipilih.
swapping_Operation(double[][] matrix)	Mencari row yang dapat di swap untuk mendapatkan leading 1 dan menjalankan fungsi swapping row.
matrix_Swapping(double[][] matrix, int row_will_be_changed, int row_who_changed)	Menukar row dari matrix dengan row lain di matriks yang sama.
Gauss_Elimination(double[][] matriks)	Menjadikan suatu matriks menjadi matriks eselon
class ParametricVariable{double NonParamValue, double[] ParamValue,	Membuat datatype baru untuk SPL yang memiliki hasil berupa parametric. data type

String[] valueRepresentation}	ini mengandung value yang non parameter, representasi parameternya, dan value untuk parameternya
ParametricVariable(double NonParamValue, double[] ParamValue, String[] valueRepresentation, int zeroSolution)	Constructor untuk datatype ParametricVariable
printVariable(int x_counter, int Params_counter) note: ada didalam class ParametricVariable	Mengembalikan String sebagai output untuk SPL yang bersifat parametric.
zero_col_counter(double[][] matriks)	Menghitung jumlah kolom yang keseluruhan barisnya 0
zero_rows_counter(double[][] matriks)	Mengembalikan jumlah baris yang keseluruhan kolomnya 0
parametric_Solution(double[][] matrix, int zero_rows, int zero_cols)	Fungsi yang digunakan untuk calculasi matriks yang memiliki solusi parametrik serta mengembalikan output untuk seluruh variable SPL.
Check_Exception(double[][] matriks)	mengembalikan state jika matriks tersebut termasuk matriks yang tidak memiliki solusi, solusi parametric atau single-solution
Gauss_Output(double[][] matriks)	Mengembalikan output yang akan ditampilkan di GUI.
SPL_From_Gauss(double[][] matriks)	Main function yang akan mengembalikan hasil dari variable SP
SPLGaussJordanFromMatrix(double[][] matrix)	Sebagai main function (function yang akan dipanggil) dan mengeluarkan string berupa hasil perhitungan SPL Gauss Jordan
indexOf(int val, int[] arr)	Mengembalikan index (int) dari value yang dicari (val)
getTotal0(double[][] matrix)	Mencari jumlah semua 0 sebelum 1 utama pada setiap baris dan mengembalikan array yang berisi jumlah semua 0 sebelum 1 utama per barisnya
getJumping0(int[] sum_0)	Mencari i pada Xi dimana Xi merupakan solusi dengan nilai real (Xi = Real) dan

	mengembalikan array yang berisi 1 untuk index i yang tidak memenuhi Xi = Real dan berisi 0 untuk index i yang memenuhi Xi = Real
getExtreme(boolean isMax, int[] arr)	Mencari dan mengembalikan nilai ekstrim (nilai maksimum atau nilai minimum)
SwapLess0(double[][] matrix)	Menukarkan baris dengan jumlah 0 yang lebih sedikit keatas baris dengan jumlah 0 yang lebih banyak
OBE(double[][] matrix)	Melakukan OBE hingga didapat matriks eselon baris tereduksi
SPLSolution(double[][] matrix)	Sebagai prosedur utama (dipanggil untuk melakukan pengecekan jenis solusi) pengembalian solusi perhitungan SPL Gauss Jordan
simplifyMatriks(double[][] matrix, int row)	Menghilangkan baris yang berisi 0 untuk setiap kolomnya
getSingleSolution(double[][] matrix,int row)	Mencari solusi tunggal SPL Gauss Jordan
getParametricSolution(double[][] matrix)	Mencari solusi parametrik SPL Gauss Jordan

3.2.4 RegresiLinierBerganda.java

Fungsi/Prosedur	Deskripsi
RLB(double[][] matrix)	Sebagai fungsi utama (fungsi yang dapat dipanggil) yang akan memanggil fungsi NEE dan mengembalikan string sebagai output yang berisi persamaan regresi linier berganda dan hasil taksirannya
NEE(double[][] matrix)	Melakukan normal estimation equation kepada matrix inputan dan mengembalikan matrix hasil normal estimation equation

3.2.5 InterpolasiPolinom.java

Fungsi/Prosedur	Deskripsi
getPolinomOutput(double[][] matrix, double x)	Mengeluarkan output fungsi interpolasi polinom dan nilainya.
SPLSolutionMatrix(double[][] m)	Mendapatkan hasil solusi SPL menggunakan gauss.
pointToFungsiInterpolasi(double[][] matrix_point)	Mengubah point-point yang sudah di input menjadi matrix

3.2.6 InterpolasiBicubicSpline.java

Fungsi/Prosedur	Deskripsi
matrixOutput(double[][] matriks)	mengembalikan String output untuk matriks saat ini
setMatrix(double[][] input)	MainFunction untuk mendapatkan matriks D inverted dan matriks koofisien vektor a.
constant_inverted()	Akan mengembalikan Inverse dari Matriks X
konstanta(int i)	Mengisi row matriks D satu array.
valueTurunanBerarah(int x, int y, int i)	Mendapatkan nilai untuk matriks D.
GetAValue(int i, int j)	mengembalikan value untuk matriks A, pada saat i,j
getBicubicOutput(double x, double y)	Mengeluarkan output untuk Bicubic.
OutputMX()	Mengeluarkan matriks X.

3.2.7 Implementasi_BicubicSpline.java

Fungsi/Prosedur	Deskripsi
displayImage(Image image)	akan menampilkan Image yang sudah di scale pada GUI
input_image(String path)	Mengembalikan image yang di input berdasarkan path yang diberikan.
ImageScaler(String input_Path, double scale)	akan memperbesar ukuran gambar yang sudah di input dengan skala yang diinput juga
matriksI(BufferedImage image, int current_x, int current_y)	akan mengembalikan suatu matriks 16 x 1 yang berisi value dari I(current_x,current_y)
state(int n, int[] x, int[] y)	akan mengubah x[0] dan y[0] menjadi 1 atau 0 sesuai state dari n
matriksD()	akan mengembalikan matriks dengan nilai constant D
GetAValue(double[][] matriksA, int i, int j)	akan mengembalikan nilai dari matriksA pada index yang ditentukan
getBicubicOutput(double[][] matriksA, double x, double y)	akan mengembalikan hasil penjumlahan untuk matriksA dengan rumus tertentu
BicubicInterpolation(Buffered image, double x, double y)	MainFunction yang akan mengembalikan nilai dari a berdasarkan a = X^-1 D I
saveImage(String outputPath)	download the image and save it to desire path

3.2.8 TextToMatrix.java

Fungsi/Prosedur	Deskripsi
readMatrixFromFile(String path)	Sebagai fungsi utama (fungsi yang dapat dipanggil) yang akan melakukan pembacaan file dan memasukannya kedalam variable bertipe string
getMatrixFromText(String text)	Melakukan parsing karakter per karakter dan mengubahnya ke bentuk matrix.

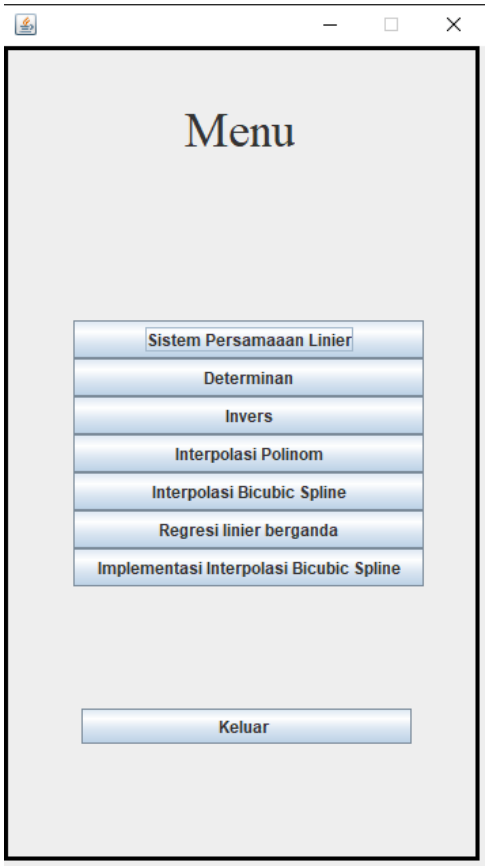
3.3. Error Handler

Error Code	Deskripsi
Wrong file input (error code = 1)	Inverse spl not square, Determinan not square, Det NxN not under 3
Wrong keyboard input for matriks size (error code = 2)	example: 0 x 0 matriks
Wrong keyboard input for determinan, not square (error code = 3)	matrix input is not square
Path must be inputted again. (error code = 4)	return val != 0
Matriks input file false (error code = 5)	col(x) != col(x+1) for every col.

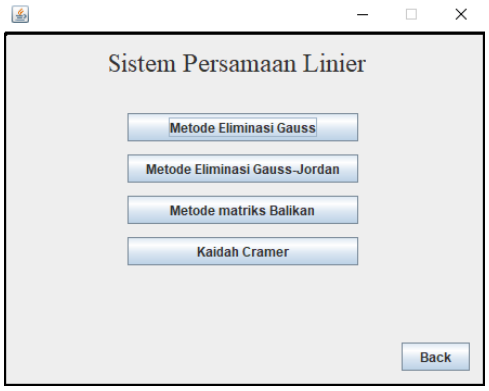
Bab IV
Eksperimen

1. GUI

a. Main Menu



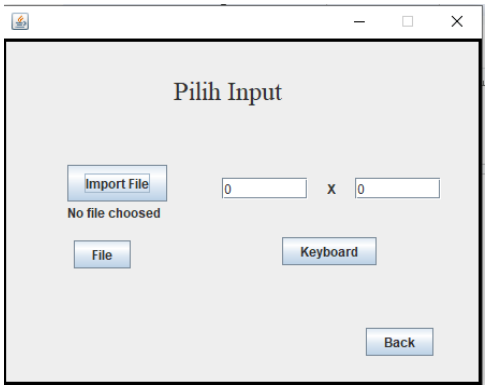
b. SPL

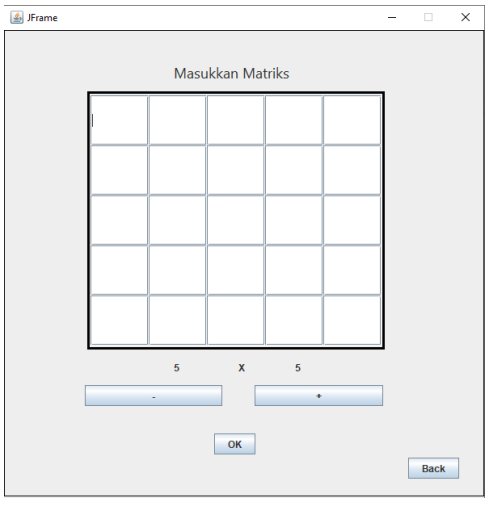


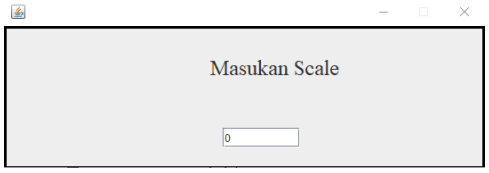
c. Determinan

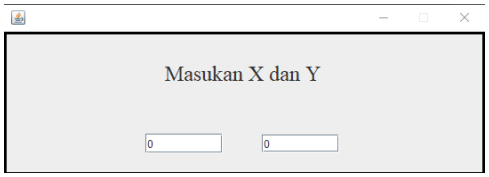
d. Invers

e. Input

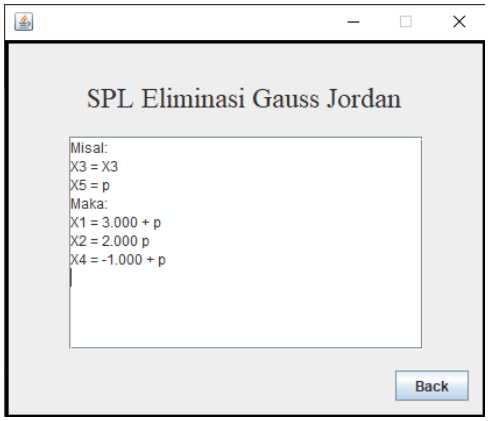








f. Output



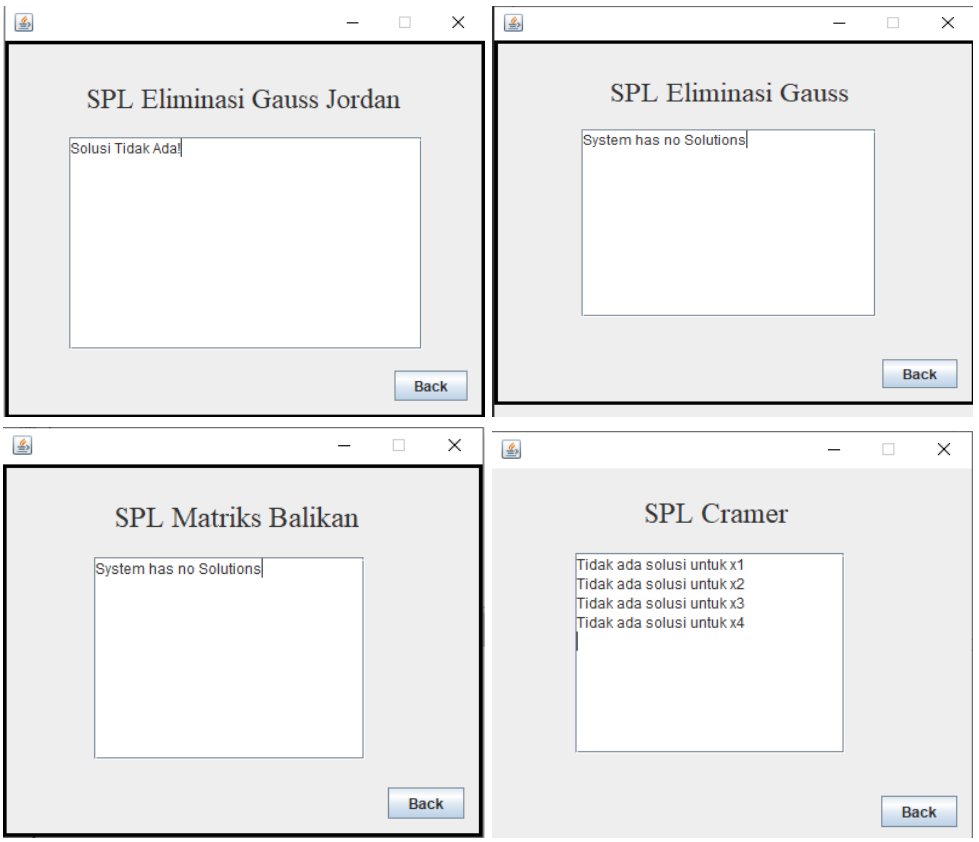
2. Test Case Studi Kasus

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

a.

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

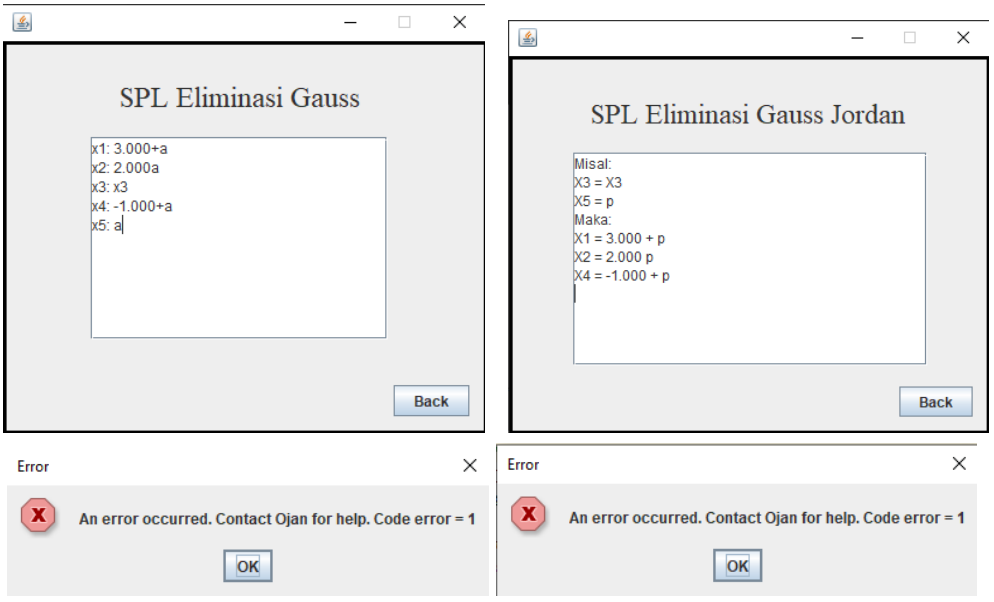
Hasil:



b.

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

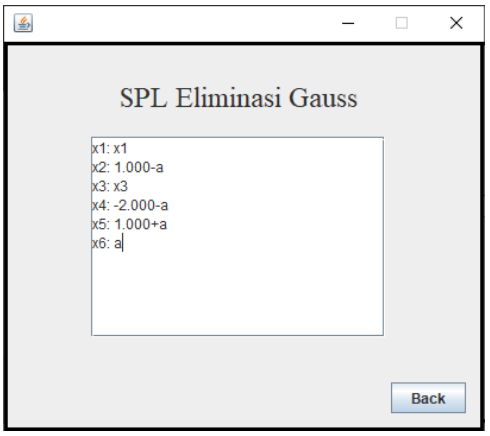
Hasil:

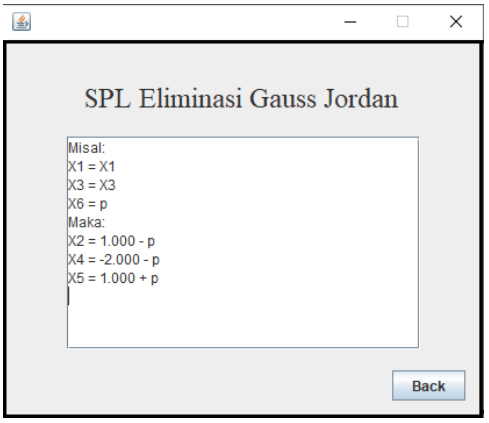


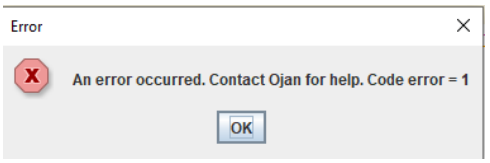
c.

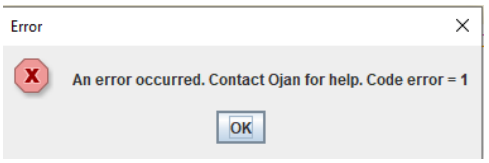
$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$$

Hasil:









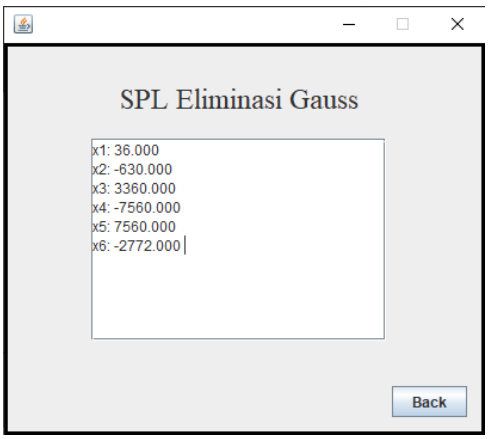
d.

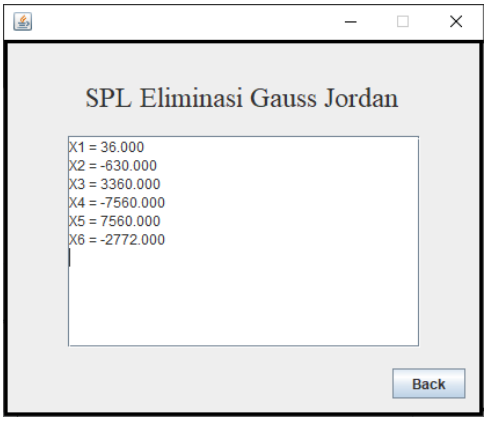
$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{n+2} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2}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

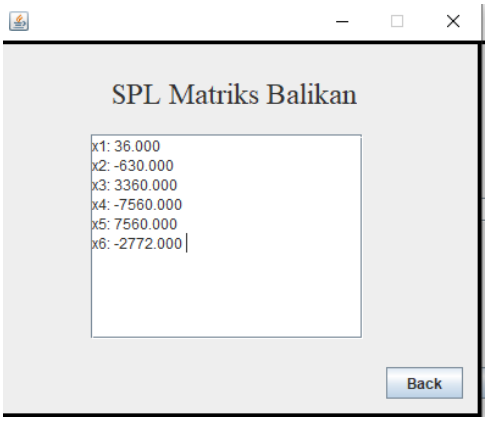
H adalah matriks *Hilbert*. Cobakan untuk $n = 6$ dan $n = 10$.

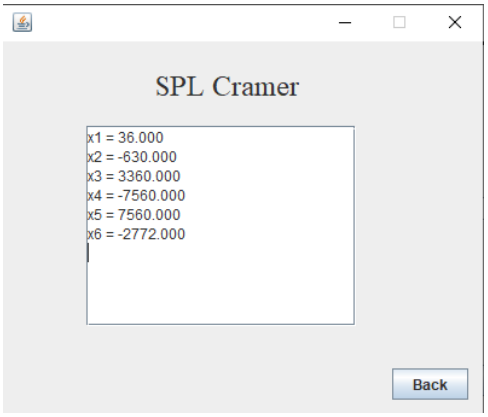
Untuk n = 6, maka

Hasil:



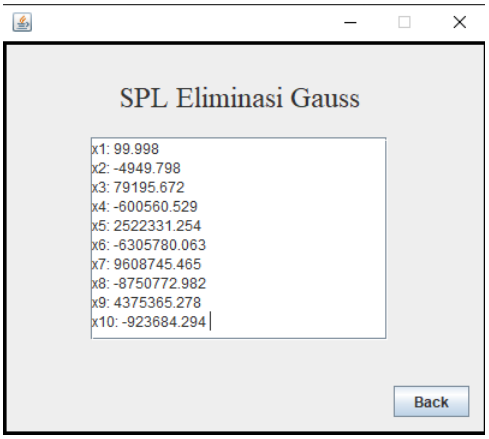


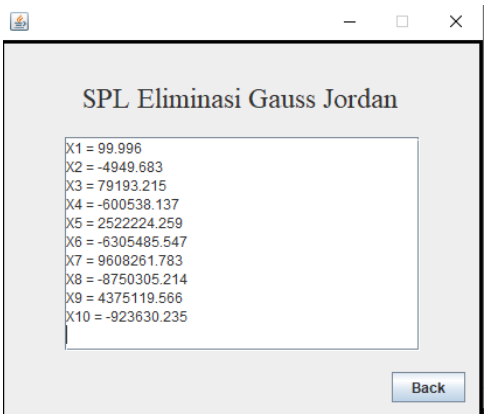


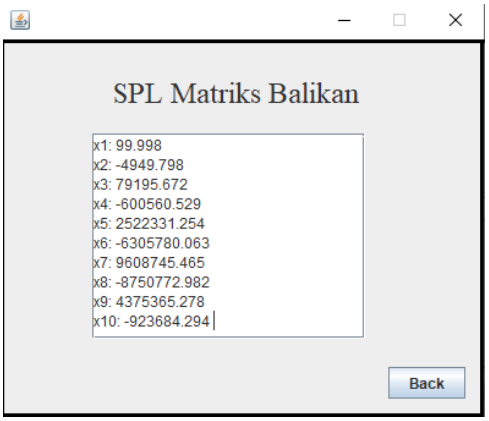


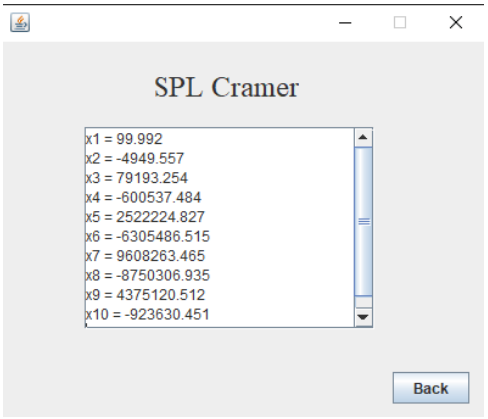
Untuk n = 10, maka

Hasil:







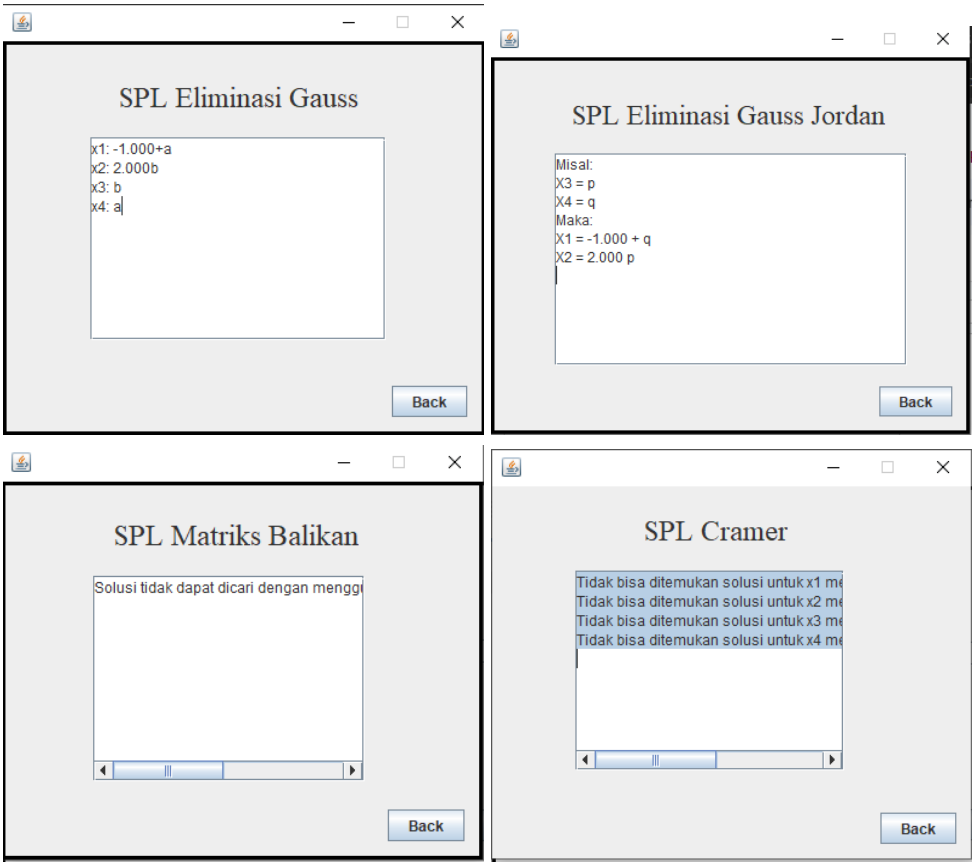


2. **SPL berbentuk matriks *augmented***

a.

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

Hasil:



Solusi SPL Matriks Balikan:

Solusi tidak dapat dicari dengan menggunakan metode ini karena Determinannya 0

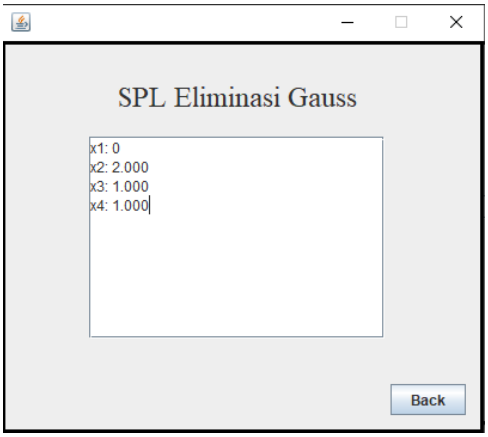
Solusi SPL Cramer:

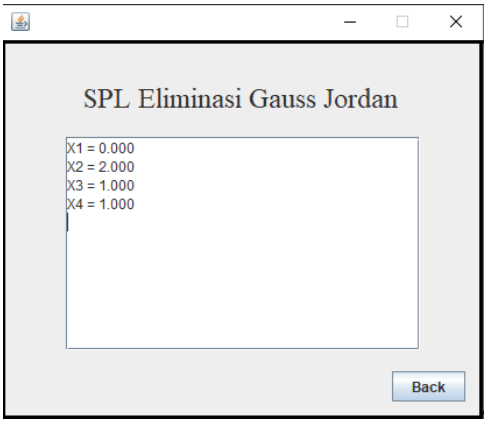
Tidak bisa ditemukan solusi untuk x1 menggunakan cramer karena det = 0
Tidak bisa ditemukan solusi untuk x2 menggunakan cramer karena det = 0
Tidak bisa ditemukan solusi untuk x3 menggunakan cramer karena det = 0
Tidak bisa ditemukan solusi untuk x4 menggunakan cramer karena det = 0

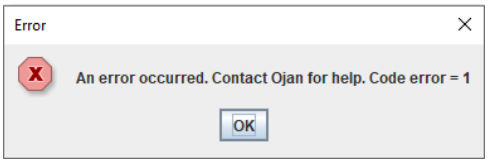
b.

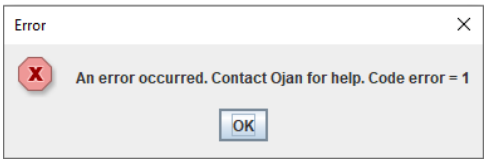
$$\begin{bmatrix} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{bmatrix}.$$

Hasil:





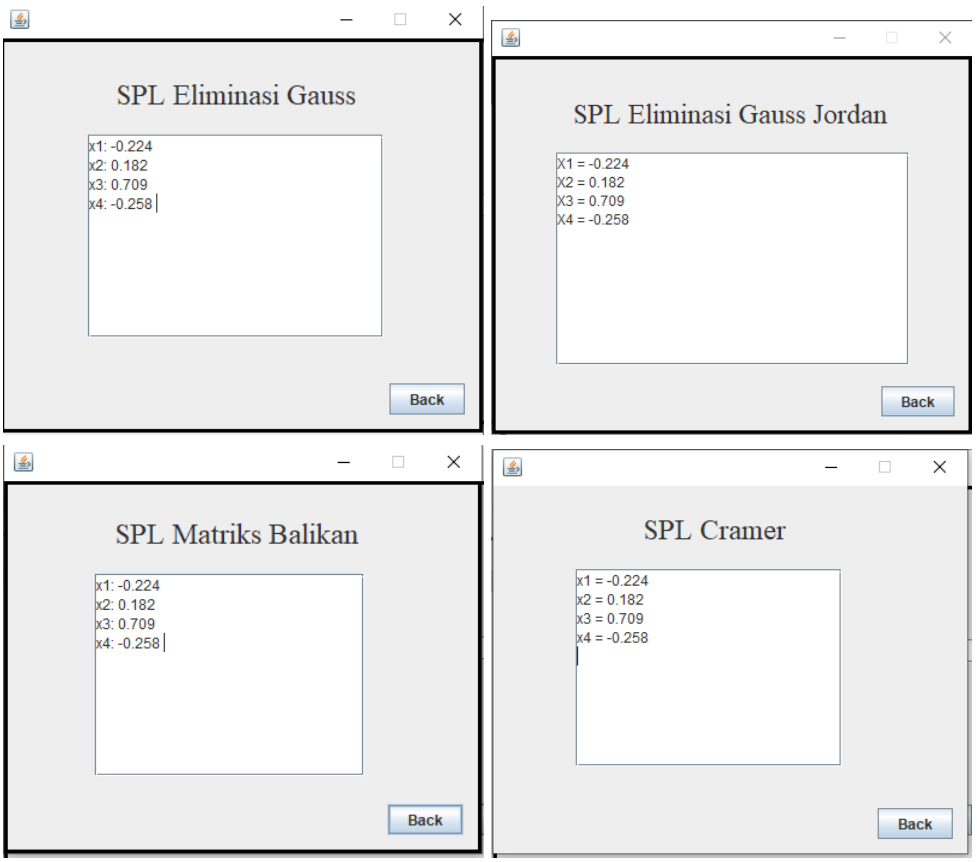




3. SPL berbentuk

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

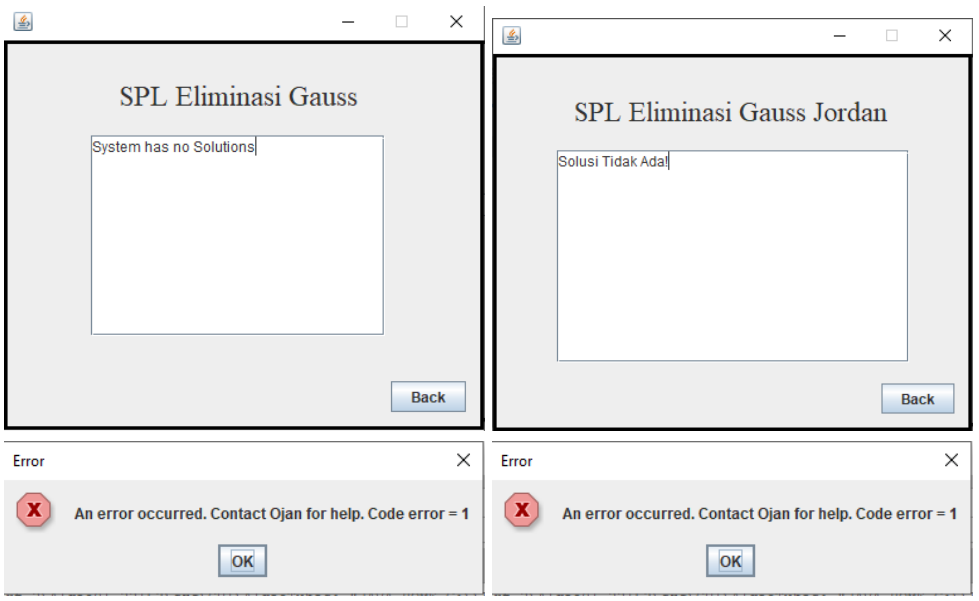
Hasil:



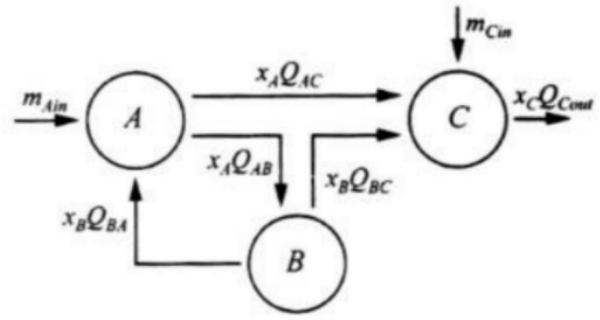
b.

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

Hasil:



4. Lihatlah sistem reaktor pada gambar berikut.



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

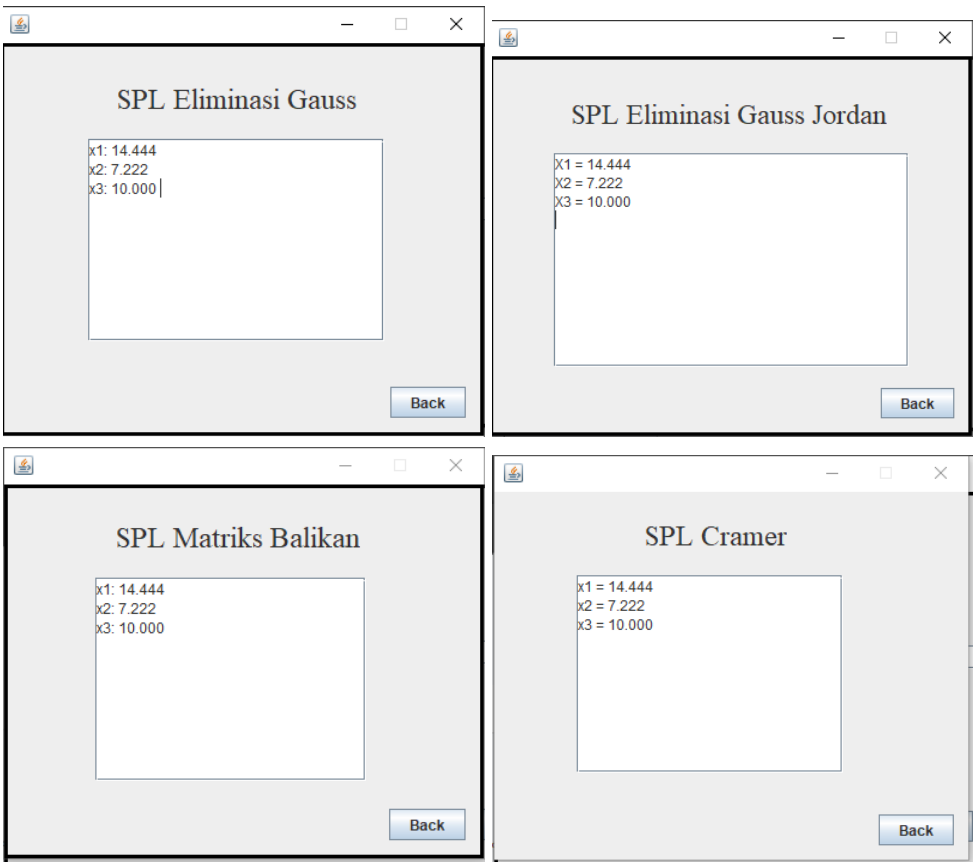
$$\text{A:} \quad m_{A_{in}} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0$$

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

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

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

dengan mengasumsikan $X_A = X_1$, $X_B = X_2$, dan $X_C = X_3$ maka akan didapat



5. Studi Kasus Interpolasi

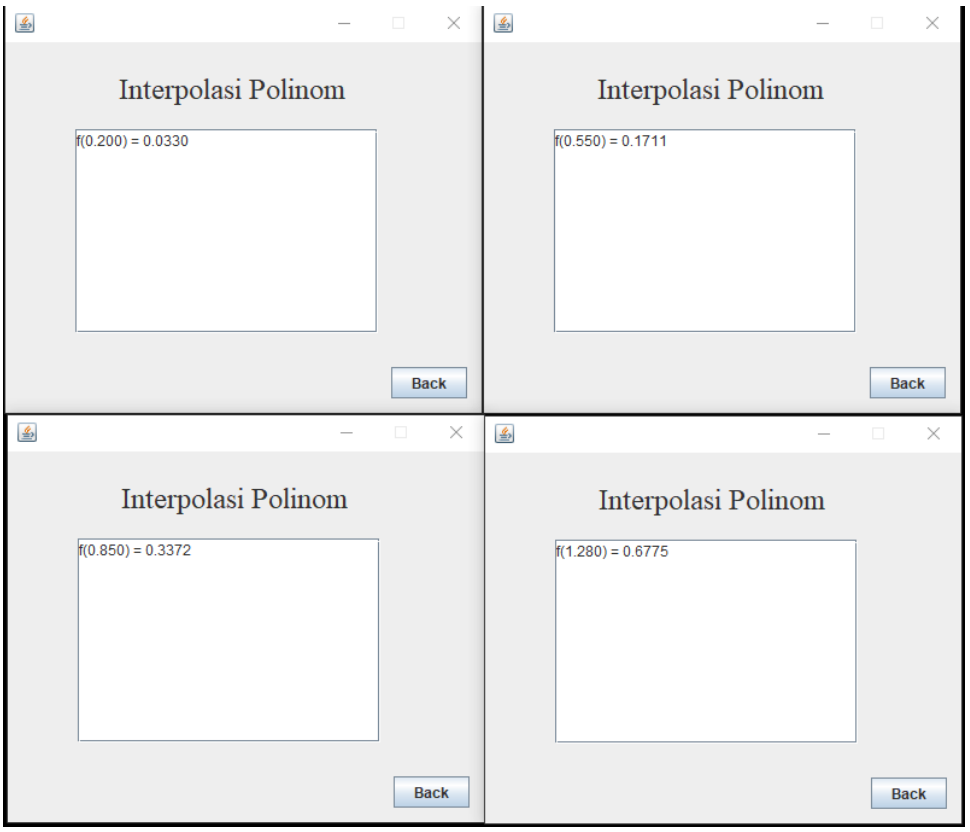
a. Gunakan tabel di bawah ini untuk mencari polinom interpolasi dari pasangan titik-titik yang terdapat dalam tabel. Program menerima masukan nilai x yang akan dicari nilai fungsi $f(x)$.

x	0.1	0.3	0.5	0.7	0.9	1.1	1.3
$f(x)$	0.003	0.067	0.148	0.248	0.370	0.518	0.697

Lakukan pengujian pada nilai-nilai berikut:

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

Hasil:



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:

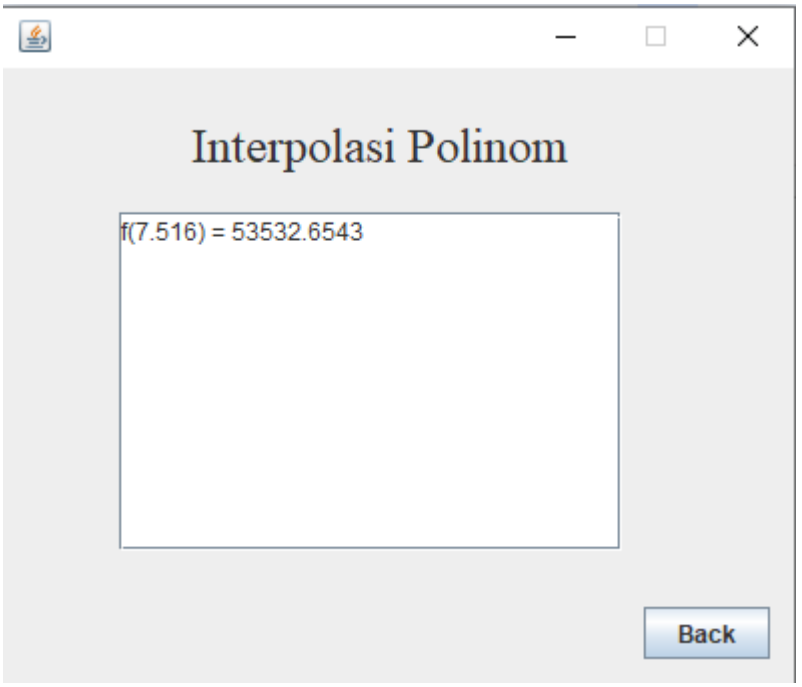
Tanggal (desimal) = bulan + (tanggal / jumlah hari pada bulan tersebut)

Sebagai contoh, untuk tanggal 17/06/2022 (dibaca: 17 Juni 2022) diperoleh tanggal(desimal) sebagai berikut:

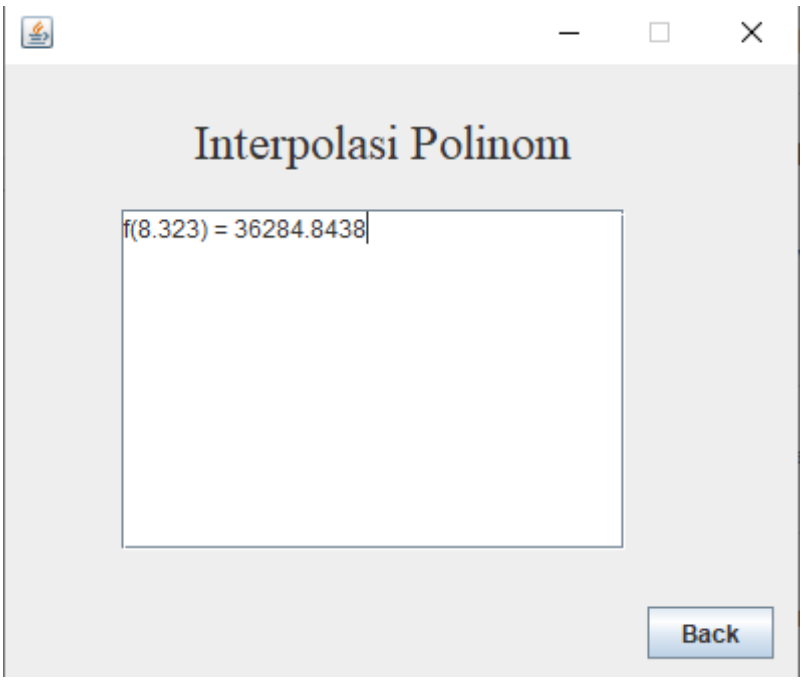
Tanggal (desimal) = 6 + (17/30) = 6,567

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

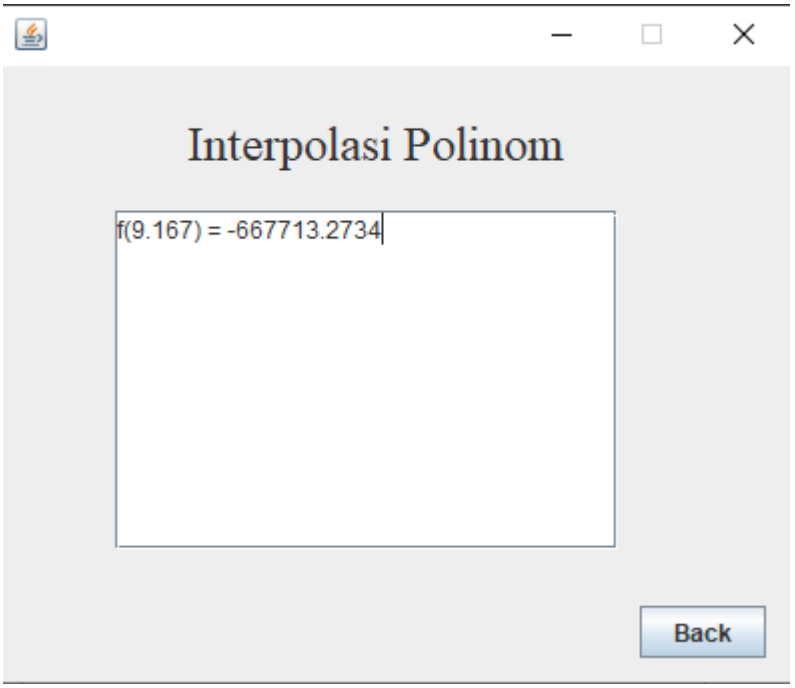
a. 16/07/2022



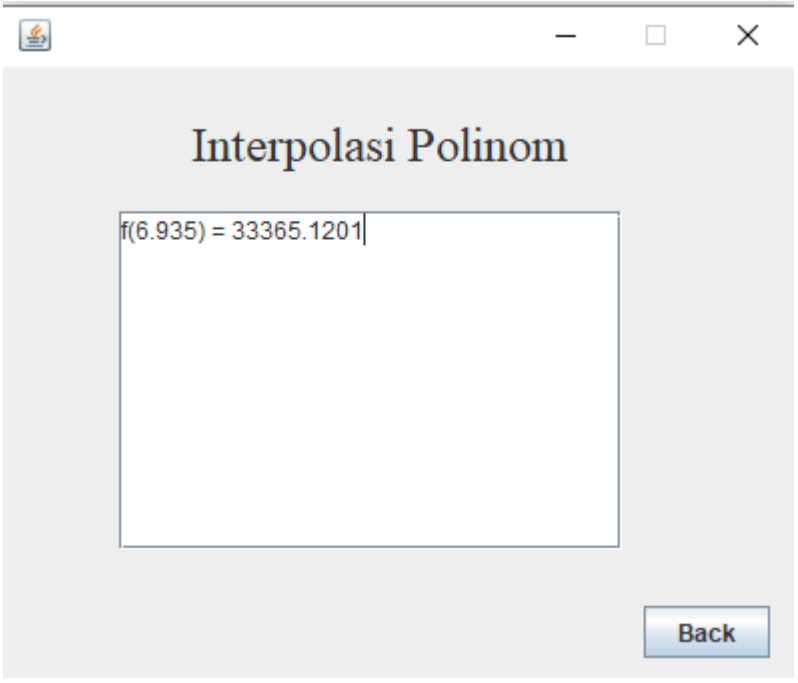
b. 10/08/2022



c. 05/09/2022



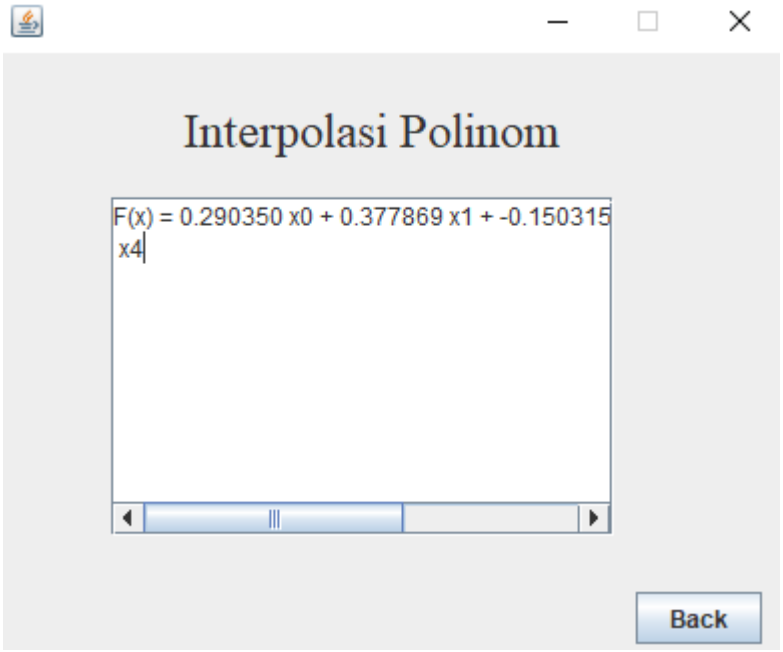
d. Masukan user lainnya berupa **tanggal (desimal) yang sudah diolah** dengan asumsi prediksi selalu dilakukan untuk tahun 2022. **(Masukan = 29/6/2022 atau 6,935 (dalam decimal))**



c. Sederhanakan fungsi $f(x)$ yang memenuhi kondisi

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

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



F(x) = 0.290350 x0 + 0.377869 x1 + -0.150315 x2 + 0.023867 x3 + -0.003695 x4

6. Studi Kasus Regresi Linear Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

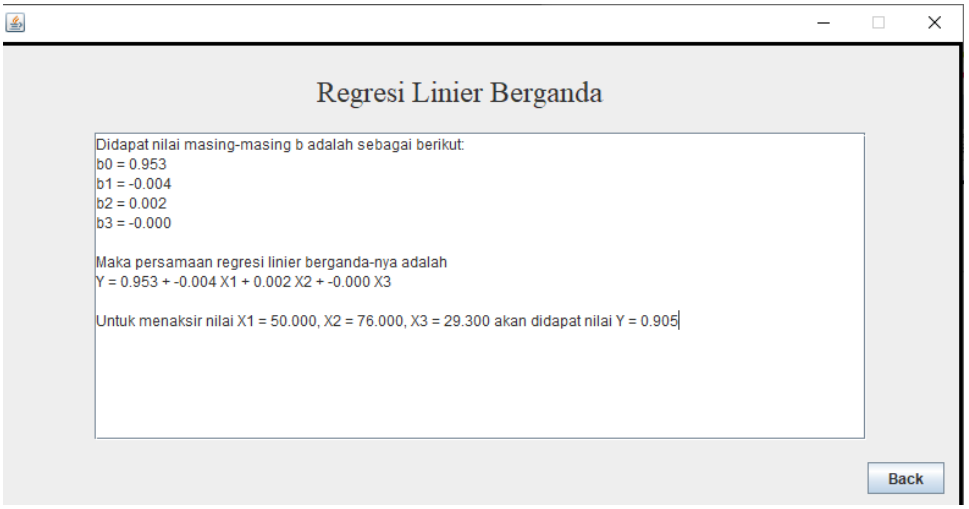
Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3	Nitrous Oxide, y	Humidity, x_1	Temp., x_2	Pressure, x_3
0.90	72.4	76.3	29.18	1.07	23.2	76.8	29.38
0.91	41.6	70.3	29.35	0.94	47.4	86.6	29.35
0.96	34.3	77.1	29.24	1.10	31.5	76.9	29.63
0.89	35.1	68.0	29.27	1.10	10.6	86.3	29.56
1.00	10.7	79.0	29.78	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.

$$
\begin{aligned}
20b_0 &+ 863.1b_1 &+ 1530.4b_2 &+ 587.84b_3 &= 19.42\\
863.1b_0 &+ 54876.89b_1 &+ 67000.09b_2 &+ 25283.395b_3 &= 779.477\\
1530.4b_0 &+ 67000.09b_1 &+ 117912.32b_2 &+ 44976.867b_3 &= 1483.437\\
587.84b_0 &+ 25283.395b_1 &+ 44976.867b_2 &+ 17278.5086b_3 &= 571.1219
\end{aligned}
$$



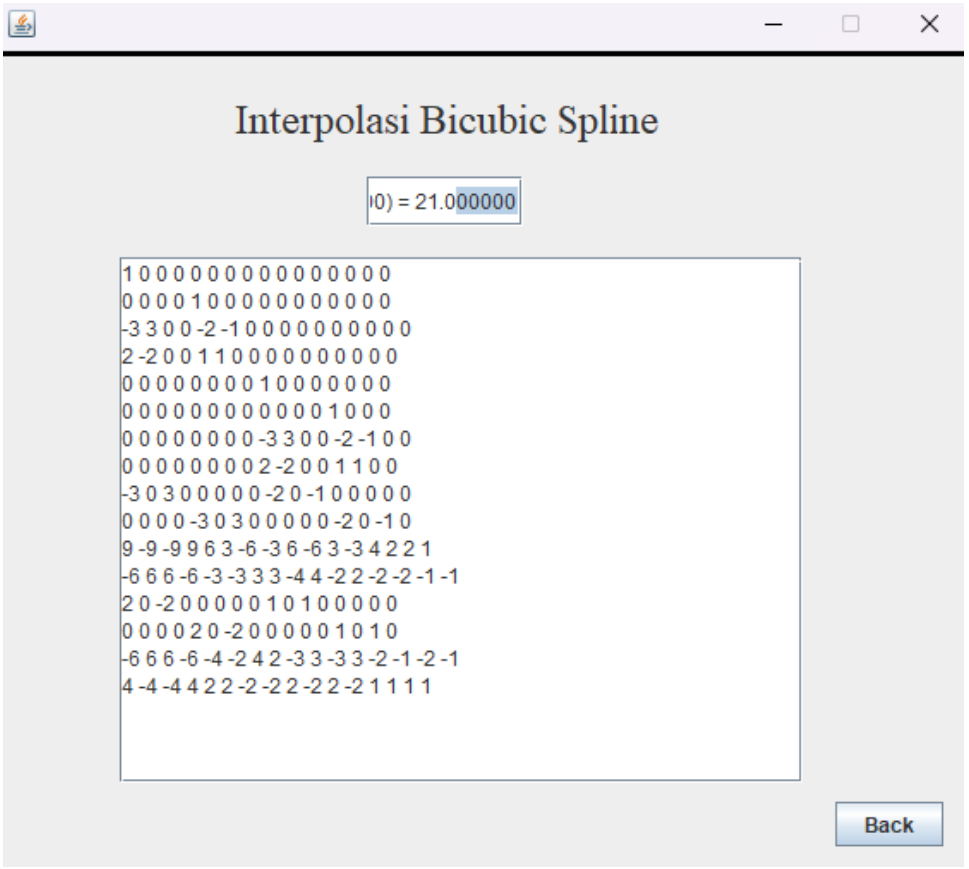
7. Studi Kasus Interpolasi *Bicubic Spline*

Diberikan matriks input dengan bentuk sebagai berikut. Format matriks masukan bukan mewakili nilai matriks, tetapi mengikuti format masukan pada bagian "Spesifikasi Tugas" nomor 7.

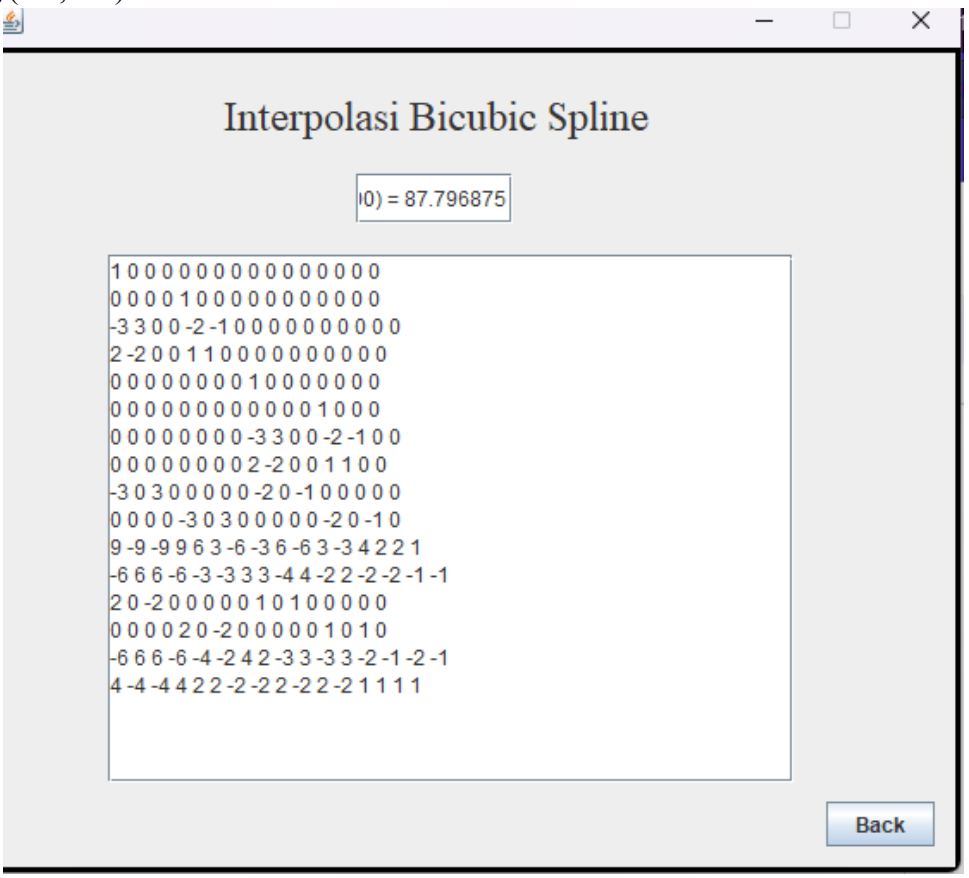
$$
\begin{pmatrix}
21 & 98 & 125 & 153\\
51 & 101 & 161 & 59\\
0 & 42 & 72 & 210\\
16 & 12 & 81 & 96
\end{pmatrix}
$$

Tentukan nilai:

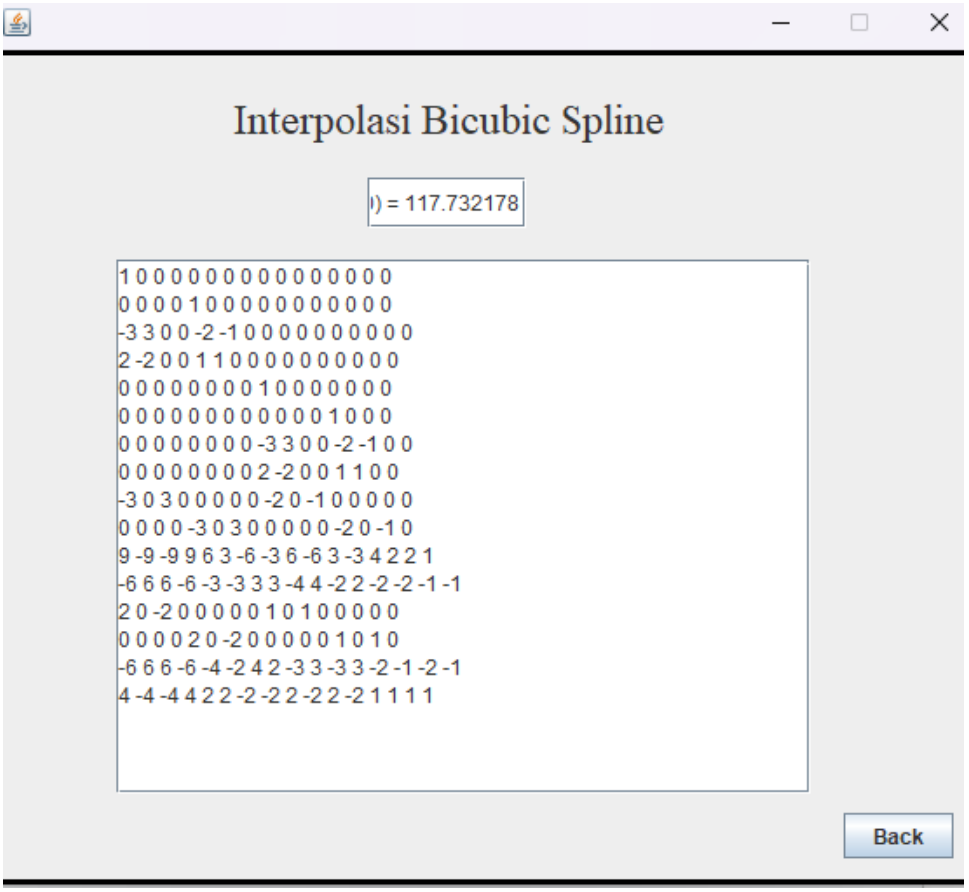
$f(0, 0)$ =



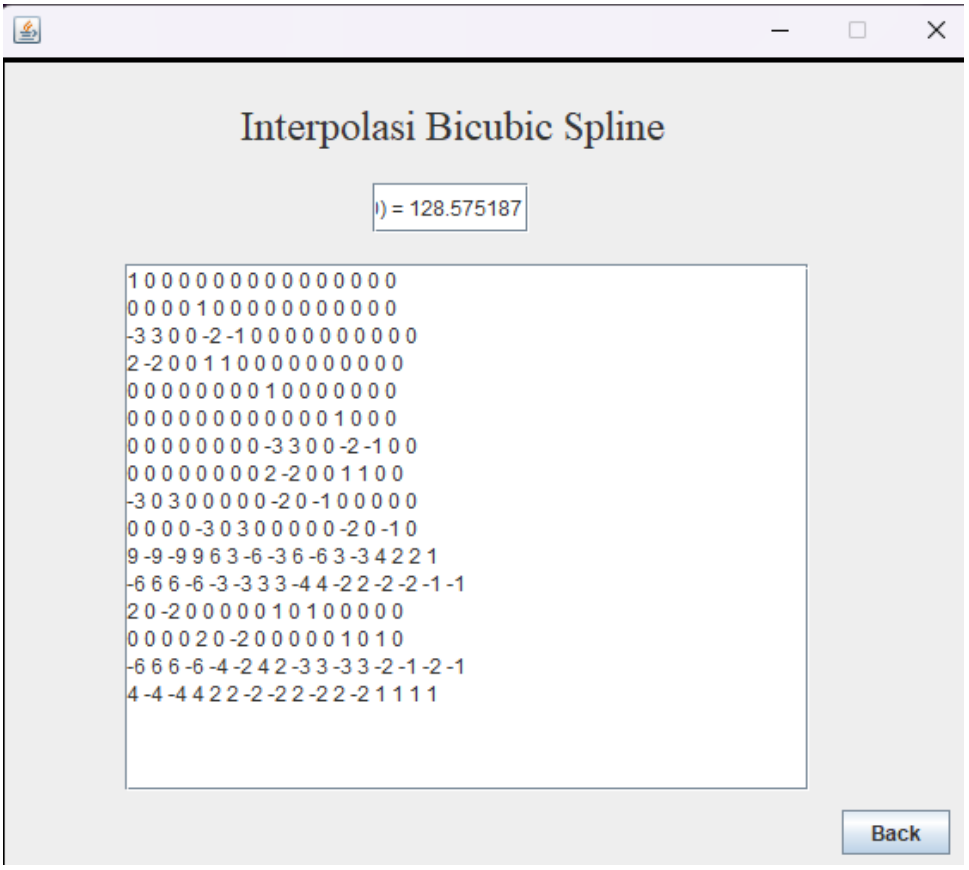
$f(0.5, 0.5)$ =



$f(0.25, 0.75)$ =


Interpolasi Bicubic Spline
) = 117.732178
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
-3 3 0 0 -2 -1 0 0 0 0 0 0 0 0 0 0
2 -2 0 0 1 1 0 0 0 0 0 0 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 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 -3 3 0 0 -2 -1 0 0
0 0 0 0 0 0 0 0 2 -2 0 0 1 1 0 0
-3 0 3 0 0 0 0 0 -2 0 -1 0 0 0 0 0
0 0 0 0 -3 0 3 0 0 0 0 0 -2 0 -1 0
9 -9 -9 9 6 3 -6 -3 6 -6 3 -3 4 2 2 1
-6 6 6 -6 -3 -3 3 3 -4 4 -2 2 -2 -2 -1 -1
2 0 -2 0 0 0 0 0 1 0 1 0 0 0 0 0
0 0 0 0 2 0 -2 0 0 0 0 0 1 0 1 0
-6 6 6 -6 -4 -2 4 2 -3 3 -3 3 -2 -1 -2 -1
4 -4 -4 4 2 2 -2 -2 2 2 -2 2 1 1 1 1
Back

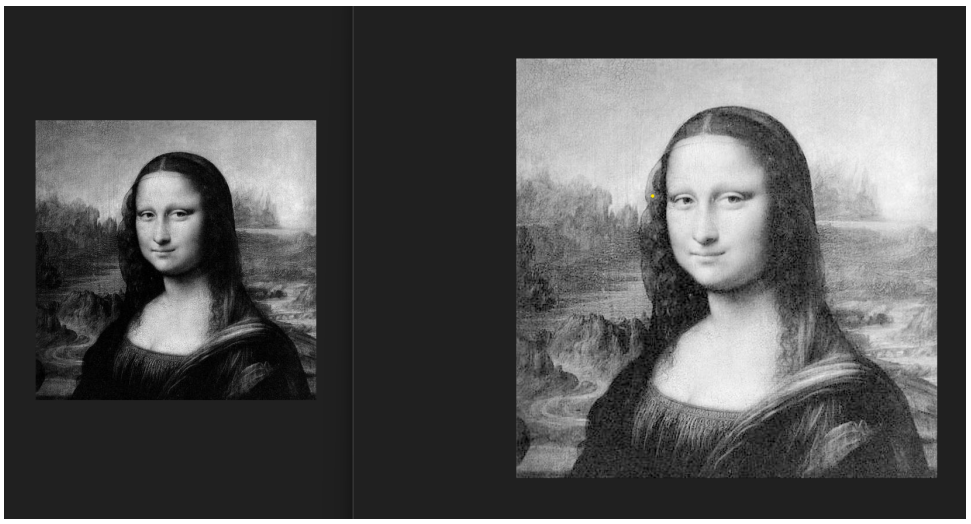
$f(0.1, 0.9)$ =


Interpolasi Bicubic Spline
) = 128.575187
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
-3 3 0 0 -2 -1 0 0 0 0 0 0 0 0 0 0
2 -2 0 0 1 1 0 0 0 0 0 0 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 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 -3 3 0 0 -2 -1 0 0
0 0 0 0 0 0 0 0 2 -2 0 0 1 1 0 0
-3 0 3 0 0 0 0 0 -2 0 -1 0 0 0 0 0
0 0 0 0 -3 0 3 0 0 0 0 0 -2 0 -1 0
9 -9 -9 9 6 3 -6 -3 6 -6 3 -3 4 2 2 1
-6 6 6 -6 -3 -3 3 3 -4 4 -2 2 -2 -2 -1 -1
2 0 -2 0 0 0 0 0 1 0 1 0 0 0 0 0
0 0 0 0 2 0 -2 0 0 0 0 0 1 0 1 0
-6 6 6 -6 -4 -2 4 2 -3 3 -3 3 -2 -1 -2 -1
4 -4 -4 4 2 2 -2 -2 2 2 -2 2 1 1 1 1
Back

8. Studi Kasus untuk Implementasi Interpolasi Bicubic Spline

Diinput suatu image. image ini memiliki format antara jpg ataupun png serta input berapa berapa scale yang ingin digunakan
contoh image:

dari image tersebut kemudian dilakukan suatu image scaling dengan menggunakan metode bicubic spline interpolation. pada contoh ini digunakan scale 1.5 dan hasilnya sebagai berikut. kiri merupakan gambar sebelum di resize, dan yang kanan merupakan gambar setelah dilakukan re-scaling



Bab V
Kesimpulan

Kesimpulan

Kami telah berhasil membuat pustaka dalam Bahasa Java. Pustaka tersebut dapat menemukan solusi SPL dengan metode *eliminasi Gauss,* metode *Eliminasi Gauss-Jordan.* metode *matriks balikan,* dan *kaidah Cramer.* Pustaka tersebut juga dapat menghitung determinan matriks dengan *Reduksi baris* dan dengan ekspansi kofaktor. Pustaka tersebut juga dapat menghitung matriks balikan dengan metode *matriks adjoin* dan *matriks identitas.*

Pustaka tersebut juga dapat menghitung *interpolasi polinom, Interpolasi Bicubic Spline, Regresi Linier Berganda, Implementasi Interpolasi Bicubic Spline.*

Saran

Masih kurangnya presisi angka perhitungan untuk kasus angka lebih besar dari pada 64 Bit, sebaiknya digunakan Bigdecimal. Walau penggunaan Bigdecimal sangat membutuhkan spesifikasi perangkat yang harus bagus, dikarenakan penggunaan Bigdecimal sangat berat.

Refleksi

Seringnya terjadi konflik saat push repository, memang seharusnya untuk melakukan penambahan fitur dilakukan di branch masing-masing. Jika memang dibutuhkan fungsi punya orang lain. Bisa ditambahkan di local terlebih dahulu dan tidak di commit jika ingin di push.

Bab VI
Referensi

memahami-metode-eliminasi-gauss-dan-pembahasan-soal (Teori Singkat Eliminasi-Gauss)
Microsoft PowerPoint - Interpolasi Polinom (itb.ac.id) (Teori interpolasi polinom)
https://www.paulinternet.nl/?page=bicubic (Teori kode Bicubic Interpolation)
Algeo-03-Sistem-Persamaan-Linier-2023.pdf (Teori Sistem Persamaan Linier)
Algeo-04-Tiga-Kemungkinan-Solusi-SPL-2023.pdf (Teori Solusi Sistem Persamaan Linier)
Algeo-05-Sistem-Persamaan-Linier-2023.pdf (Teori Sistem Persamaan Linier)
Algeo-08-Determinan-bagian1-2023.pdf (Teori Determinan Reduksi Baris)
Algeo-09-Determinan-bagian2-2023.pdf (Teori Determinan Ekspansi Kofaktor)

Bab VII
Pembagian Tugas

No	Tugas	Penanggung Jawab	Progress	Prioritas	Pengecek
1	MENU (GUI?)	Ojan	100	High	Valen
2	SPL		100		
2.1	Metode Eliminasi Gauss	Davis	100	High	Ojan
2.2	Metode Eliminasi Gauss-Jordan	Valen	100	Medium	Davis
2.3	Metode matriks Balikan	Davis	100	Low	Ojan
2.4	Kaidah Cramer	Ojan	100	Low	Valen
3	Determinan		100		
3.1	Determinan 2x2 and 3x3 (NxN isril kah? no)	Valen	100	High	Davis
3.2	Determinan dengan Kofaktor	Davis	100	High	Ojan
3.3	Determinan Matriks Segitiga	Ojan	100	High	Valen
4	Invers		100		
4.1	Matriks inverse dengan Adjoin	Valen	100	Medium	Davis
4.2	Metode matriks Balikan	Davis	100	Medium	Ojan
5	Interpolasi Polinom	Ojan	100	Medium	Valen
6	Interpolasi Bicubic Spline	Ojan	100	Medium	Valen

7	Regresi linier berganda	Valen	100	Medium	Davis
8	Input		100		
8.1	Input from Keyboard	Ojan	100	High	Valen
8.2	Input from File	Valen	100	High	Davis
9	Bonus		80		
9.1	interpolasi bicubic spline implementation	Davis	80	Low	Ojan