

LAPORAN
TUGAS BESAR 1 IF2123 ALJABAR LINIER DAN
GEOMETRI
SISTEM PERSAMAAN LINIER, DETERMINAN, DAN
APLIKASINYA

oleh

Fakhri Muhammad Mahendra 13521045

Fatih Nararya Rashadyfa 13521060

Arsa Izdihar Islam 13521101



**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2022**

IF2123

DAFTAR ISI

DAFTAR ISI

BAB 1	1
DESKRIPSI MASALAH	1
1.1 Spesifikasi Tugas	1
BAB II	6
TEORI SINGKAT	6
2.1 Persamaan Linier	6
2.2 Operasi Baris Elementer	6
2.3 Matriks eselon baris dan eselon baris tereduksi	7
2.4 Metode Eliminasi Gauss	7
2.2 Metode Eliminasi Gauss-Jordan	8
2.3 Determinan	8
2.3.1 Determinan dari Matriks 2x2	8
2.3.2 Determinan dari Matriks 2x2	8
2.3.3 Determinan dari Matriks secara umum	9
2.4 Matriks Balikan	9
2.4.1 Metode OBE	9
2.4.2 Metode Matriks Adjoin	10
2.5. Minor	10
2.6 Matriks Kofaktor	11
2.6 Matriks Adjoin	12
2.7 Kaidah Kramer	12
2.8 Interpolasi Polinom	12
2.9 Interpolasi Bicubic	13
2.10 Regresi Linier Berganda	14
BAB III	16
IMPLEMENTASI PUSTAKA DAN PROGRAM DALAM JAVA	16
3.1 Class Matrix	16
3.2 Class Pair	24
3.3 Class Image	24
3.4 Class Errors	25

3.5 Class FromFile	26
3.6 Class FromKeyboard	28
3.7 Class IOLib	30
3.7.1 Class RowError	30
3.7.2 Class SPLSolution	31
3.7.3 Class MLR	33
3.8 Class ToFile	33
3.8.1 Class CreateFileCondition	35
3.9 Class ToKeyboard	36
3.10 Class CoreFunctionality	36
3.10.1 Class solveSPL	36
3.10.2 Class Inverse	37
3.10.2 Class Interpolation	37
3.10.3 Class Determinant	38
3.10.4 Class MLR	38
3.11 Class Interface	39
3.11.1 Class MenuText	39
BAB IV	41
EXPERIMEN	41
4.1 SPL	41
4.1.1 Studi Kasus 1a	41
4.2.2 Studi Kasus 1b	41
4.2.3 Studi Kasus 1.c	42
4.2.4 Studi Kasus 1d Matriks Hilbert dengan $n = 6$	43
4.2.5 Studi Kasus 1d Matriks Hilbert dengan $n = 10$	44
4.2.6 Studi Kasus 2a	45
4.2.7 Studi Kasus 2b	45
4.2.8 Studi Kasus 3a	46
4.2.9 Studi Kasus 3b	47
4.2 Determinan	48
4.2.1 Determinan dengan Kofaktor	48
4.2.2 Determinan dengan Matriks Segitiga Atas	49
4.3 Balikan	50
4.3.1 Balikan dengan OBE	50
4.3.2 Balikan dengan Matriks Adjoin	50

4.4 Interpolasi Polinom	51
4.4.1 Studi Kasus 4a	51
4.4.2 Interpolasi Data Covid	52
4.4.3 Interpolasi fungsi	53
4.5 Interpolasi Bicubic	54
4.6 Regresi Linier Berganda	55
4.7 Perbesaran Gambar	57
4.7.1 Gambar Logo HMIF	57
4.7.2 Gambar Logo ITB	59
4.7.3 Gambar Einstein	60
BAB 5	62
KESIMPULAN, SARAN, DAN REFLEKSI	62
5.1 Kesimpulan	62
5.2 Saran	62
5.3 Refleksi	63
DAFTAR PUSTAKA	64
LAMPIRAN	65

BAB 1

DESKRIPSI MASALAH

1.1 Spesifikasi Tugas

Pada tugas besar 1, mahasiswa diminta membuat *library* aljabar linier dalam Bahasa Java. Program dalam *library* harus dapat menyelesaikan permasalahan berikut.

1. Melakukan eliminasi Gauss.
2. Melakukan eliminasi Gauss-Jordan.
3. Mencari solusi dari SPL dengan eliminasi Gauss, eliminasi Gauss-Jordan, matriks balikan, dan kaidah Cramer.
4. Mencari matriks balikan dengan metode OBE dan metode kofaktor.
5. Mencari determinan dengan metode segitiga atas, dan ekspansi kofaktor.
6. Melakukan interpolasi polinom.
7. Melakukan interpolasi bikubik.
8. Melakukan regresi linier berganda.
9. Sebagai bonus, melakukan perbesaran citra gambar digital.

Spesifikasi program yang dibuat 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

3. Untuk persoalan interpolasi, masukannya jika dari keyboard adalah n, (x_0, y_0) , (x_0, 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. Misalnya jika titik-titik datanya adalah (8.0, 2.0794), (9.0, 2.1972), dan (9.5, 2.2513), maka di dalam file text ditulis sebagai berikut:

8.0	2.0794
9.0	2.1972
9.5	2.2513

4. Untuk persoalan regresi, masukannya jika dari keyboard adalah n (jumlah peubah x), m (jumlah sampel), semua nilai-nilai $x_{1i}, x_{2i}, \dots, x_{ni}$, 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 (output) 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 determinan dan matriks balikan, maka luarannya sesuai dengan persoalan masing-masing

$$\begin{array}{ccccc} -3 & 7 & 8.3 & 11 & -4 \\ 3 & 4.5 & 2.8 & 10 & 12 \\ 0.5 & -10 & -9 & 12 & 0 \\ 0.1 & 0.2 & & & \end{array}$$

7. Untuk persoalan polinom interpolasi dan regresi, luarannya adalah persamaan polinom/regresi dan taksiran nilai fungsi pada x yang diberikan. Contoh luaran untuk interpolasi adalah $f(x) = -0.0064x^2 + 0.2266x + 0.6762$, $f(5) = \dots$ dan untuk regresi adalah $f(x) = -9.5872 + 1.0732x_i$, $f(x_k) = \dots$
8. Untuk persoalan interpolasi bicubic, masukan dari file text (.txt) yang berisi matriks berukuran 4x4 yang berisi nilai $f(i,j)$ dengan i dan j adalah indeks matriks diikuti dengan nilai a dan b untuk mencari nilai $f(a,b)$. misalnya jika 6 nilai dari $f(-1,-1)$, $f(-1,0)$, $f(-1,1)$, $f(-1,2)$, $f(0,-1)$, $f(0,0)$, $f(0,1)$, $f(0,2)$, $f(1,-1)$, $f(1,0)$, $f(1,1)$, $f(1,2)$, $f(2,-1)$, $f(2,0)$, $f(2,1)$, $f(2,2)$ 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 $f(0.5,0.5)$. masukannya adalah matriks 4×4 , diikuti oleh nilai a dan b, maka luarannya adalah nilai $f(a,b)$.

9. Luaran program harus dapat ditampilkan pada layar komputer dan dapat disimpan ke dalam file.
10. Bahasa program yang digunakan adalah Java.
11. Program tidak harus berbasis GUI, cukup text-based saja, namun boleh menggunakan GUI (memakai kakas Eclipse misalnya).
12. Program dapat dibuat dengan pilihan menu. Urutan menu dan isinya dipersilakan dirancang masing-masing. Misalnya, menu:

MENU

1. Sistem Persamaan Linier
2. Determinan
3. Matriks balikan
4. Interpolasi Polinom
5. Interpolasi Bicubic
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

2.1 Persamaan Linier

Persamaan linear dengan n peubah x_1, x_2, \dots, x_n didefinisikan sebagai persamaan yang bisa dibuat ekspresikan dalam bentuk

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n = b$$

dengan a_1, a_2, \dots, a_n dan b adalah suatu konstan, dan tidak semua dari ‘ a ’ bernilai nol.

Sistem persamaan linear didefinisikan sebagai himpunan terbatas dari persamaan linear. Sistem persamaan linear bisa diekspresikan dalam bentuk matriks persegi panjang. Sebagai contoh, sistem persamaan linear dibawah:

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

ekuivalen dengan bentuk matriks berikut:

$$\left[\begin{array}{ccccc} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array} \right]$$

Matriks ini disebut matriks augmentasi dari sistem persamaan linear.

2.2 Operasi Baris Elementer

Pada matriks dapat dilakukan tiga operasi baris elementer (OBE) yaitu:

1. Mengalikan sebuah baris dengan konstanta tidak nol.
2. Menukarkan dua buah baris.
3. Menambahkan sebuah baris dengan kelipatan baris lainnya.

2.3 Matriks eselon baris dan eselon baris tereduksi

Matriks disebut dalam bentuk eselon baris jika memenuhi syarat berikut.

1. Jika sebuah baris memiliki elemen tidak semua nol, maka bilangan tak nol pertama yang ditemukan pada baris adalah 1. Angka ini kita sebut sebagai 1 utama.
2. Untuk semua baris yang elemennya hanya berisi nol, maka baris tersebut dikumpulkan kebagian bawah matriks.
3. Untuk semua dua baris berurutan yang elemennya tidak semua nol, 1 utama dari baris dibawah muncul lebih ke kanan dari 1 utama dari baris diatas.

Sementara matriks berada dalam bentuk eselon baris tereduksi jika memenuhi semua dari persyaratan matriks eselon baris, ditambah syarat keempat:

4. Setiap kolom yang mengandung 1 utama memiliki nilai nol selain pada baris tersebut.

2.4 Metode Eliminasi Gauss

Metode eliminasi Gauss adalah metode yang dipakai untuk menyelesaikan sistem persamaan linier. Metodenya adalah

1. Sistem persamaan linier diubah ke bentuk matriks augmentasi.
2. OBE diterapkan pada matriks augmentasi sampai terbentuk matriks eselon baris.

3. Pecahkan persamaan yang berkorespondensi pada matriks eselon baris dengan teknik substitusi mundur.

2.2 Metode Eliminasi Gauss-Jordan

Metode eliminasi Gauss-Jordan juga merupakan metode untuk menyelesaikan sistem persamaan linier. Metode ini mirip dengan eliminasi Gauss.. Metodenya adalah

1. Sistem persamaan linier diubah ke bentuk matriks augmentasi.
2. Terapkan OBE pada matriks augmentasi sampai terbentuk matriks eselon baris tereduksi.
3. Nilai dari peubah yang dicari didapat dari kolom paling kanan dari matriks augmentasi.

2.3 Determinan

Determinan adalah sebuah nilai skalar yaitu sebuah fungsi dari matriks persegi. Determinan dari suatu matriks A, dinotasikan dengan $\det(A)$, $\det A$, atau $|A|$. Berikut diberikan rumus untuk mencari nilai beberapa ukuran determinan

2.3.1 Determinan dari Matriks 2x2

Determinan dari Matriks 2x2 A bisa didefinisikan sebagai

$$|A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

2.3.2 Determinan dari Matriks 2x2

Determinan dari Matriks 3x3 A bisa didefinisikan sebagai

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = a \begin{vmatrix} e & f \\ h & i \end{vmatrix} - b \begin{vmatrix} d & f \\ g & i \end{vmatrix} + c \begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

2.3.3 Determinan dari Matriks secara umum

Jika A adalah $n \times n$ matrix, maka angka yang didapat dari mengalikan entri dari baris atau kolom manapun dari A dengan pasangan kofaktornya dan menjumlahkan hasilnya merupakan nilai determinan dari A, dan hasil jumlahnya sendiri disebut ekspansi kofaktor dari A, yaitu

$$\det(A) = a_{1j}C_{1j} + a_{2j}C_{2j} + \cdots + a_{nj}C_{nj}$$

dan

$$\det(A) = a_{i1}C_{i1} + a_{i2}C_{i2} + \cdots + a_{in}C_{in}$$

2.4 Matriks Balikan

Jika A adalah satu matriks persegi, dan ada matriks B dengan ukuran yang sama sehingga $AB = BA = I$, maka A disebut *invertible* (atau *nonsingular*) dan B disebut sebagai invers dari A. Jika tidak ada matriks B yang memenuhi kondisi tersebut maka matriks A disebut matriks *singular*.

Jika A adalah suatu matriks yang *invertible* maka matriks balikannya biasa dinotasikan sebagai A^{-1} .

2.4.1 Metode OBE

Matriks balikan dapat dicari dengan beberapa cara, salah satunya adalah dengan menggunakan determinan-adjoint, dan menggunakan OBE. Metode OBE dapat dilakukan dengan

1. Augmentasi matriks persegi yang ingin dicari balikannya dengan matriks identitas yang besarnya.
2. Lakukan OBE pada matriks hasil augmentasi tersebut sampai ke bentuk matriks eselon baris tereduksi.
3. Jika setelah OBE, matriks augmentasi bagian kiri memiliki baris yang semua nilainya nol, maka matriks tersebut tidak mempunyai balikan.

Namun jika sebaliknya terjadi, maka bagian kanan matriks augmentasi yang awalnya merupakan matriks identitas, adalah matriks balikan yang dicari.2.5 Matriks.

2.4.2 Metode Matriks Adjoin

Cara lain untuk menemukan balikan dari suatu matriks A adalah dengan memanfaat sifat matriks adjoin, yaitu

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

2.5. Minor

Jika A adalah suatu matriks persegi, maka entri minor dari a_{ij} yang dinotasikan sebagai M_{ij} didefinisikan sebagai determinan dari submatrix yang tersisa ketika baris ke-i dan kolom ke-j dihapus dari A. Jika A adalah suatu matriks persegi

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1j} \\ a_{21} & a_{22} & \cdots & a_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} \end{bmatrix}$$

maka matriks minor dari A adalah

$$\text{minor}(A) = \begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1j} \\ M_{21} & M_{22} & \cdots & M_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ M_{i1} & M_{i2} & \cdots & M_{ij} \end{bmatrix}$$

2.6 Matriks Kofaktor

Jika A adalah suatu matriks persegi, maka entri kofaktor a_{ij} yang dinotasikan sebagai C_{ij} didefinisikan sebagai hasil dari $(-1)^{i+j} M_{ij}$. Jika A adalah suatu matriks persegi

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1j} \\ a_{21} & a_{22} & \cdots & a_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} \end{bmatrix}$$

maka matriks kofaktor dari A adalah

$$\text{cofactor}(A) = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1j} \\ C_{21} & C_{22} & \cdots & C_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ C_{i1} & C_{i2} & \cdots & C_{ij} \end{bmatrix}$$

2.6 Matriks Adjoin

Matriks adjoin didefinisikan sebagai matriks transpos dari matriks kofaktor.

Dilanjut dengan definisi matriks A yang sebelumnya didapat $\text{adjoin}(A)$ adalah

$$\text{adjoin}(A) = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1j} \\ C_{21} & C_{22} & \cdots & C_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ C_{i1} & C_{i2} & \cdots & C_{ij} \end{bmatrix}^T = \begin{bmatrix} C_{11} & C_{21} & \cdots & C_{i1} \\ C_{12} & C_{22} & \cdots & C_{i2} \\ \vdots & \vdots & \ddots & \vdots \\ C_{1j} & C_{2j} & \cdots & C_{ij} \end{bmatrix}$$

2.7 Kaidah Kramer

Jika $Ax = b$ adalah sistem n persamaan linier dengan n peubah sehingga $\det(A) \neq 0$, maka sistem tersebut mempunyai solusi unik. Solusi tersebut adalah

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

dimana A_j adalah matriks yang didapat dengan menggantikan kolumn ke-j dari A dengan entri dari matriks

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

2.8 Interpolasi Polinom

Interpolasi polinomial adalah interpolasi dari sekumpulan data menggunakan polinomial dengan derajat terendah yang melewati titik-titik dari data tersebut. Sebagai contoh, jika diketahui $n+1$ buah titik data, misal pasangan (x_i, y_i) dengan

$i = 0, 1, \dots, n$. Maka terdapat polinom derajat n yang bisa menginterpolasi titik-titik data tersebut. Andai polinom tersebut adalah

$$P_n(x) = a_0 + a_1x + \dots + a_nx^n$$

dengan a_0, a_1, \dots, a_n adalah nilai yang dicari. Dari polinom dan titik-titik tersebut dapat dibuat $n + 1$ buah persamaan linier dengan $n + 1$ peubah yang tidak diketahui nilainya.

$$\begin{aligned} a_0 + a_1x_0 + a_2x_0^2 + \dots + a_nx_0^n &= y_0 \\ a_0 + a_1x_1 + a_2x_1^2 + \dots + a_nx_1^n &= y_1 \\ &\dots \\ a_0 + a_1x_n + a_2x_n^2 + \dots + a_nx_n^n &= y_n \end{aligned}$$

Untuk mendapat nilai a_0, a_1, \dots, a_n dapat digunakan metode eliminasi Gauss atau metode penyelesaian SPL lainnya.

2.9 Interpolasi Bicubic

Interpolasi bicubic merupakan teknik interpolasi data dua dimensi menggunakan pendekatan polinomial kubik. Karena polinomial kubik dipakai untuk dua dimensi, maka model berbentuk

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

Mirip dengan interpolasi polinomial, dalam pemodelan ini, a_{ij} merupakan nilai yang tidak diketahui nilainya. Karena itu, diperlukan 16 titik data untuk menyelesaikan sistem persamaan tersebut.

Jika $x,y = -1, 0, 1, 2$. kita dapat mensubstitusi nilai-nilai yang diketahui tersebut ke dalam model, sehingga menghasilkan matriks sistem persamaan sebagai berikut

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

Vektor koefisien a dapat dicari dengan menggunakan matriks balikan. Jika persamaan tersebut dimodelkan dalam bentuk $y = Xa$, maka kita dapat mencari nilai dari matriks balikan X yaitu X^{-1} . Lalu dengan melakukan operasi kali matriks pada kedua ruas persamaan, didapat $X^{-1}y = a$. Karena nilai dari matriks X dan $f(x,y)$ sudah diketahui, maka vektor koefisien a dapat diketahui nilainya. Dari koefisien=koefisien yang sudah didapat, maka model berhasil didapatkan, dan bisa dipakai untuk mencari interpolasi dari data tersebut.

2.10 Regresi Linier Berganda

Model regresi digunakan untuk mendeskripsikan hubungan antar variabel dengan garis linear dari data yang didapat. Regresi linier berganda merupakan

kasus khusus dimana terdapat dua peubah independen dan satu peubah terikat.

Maka model regresi linier berganda bisa menjelaskan bagaimana variabel terikat Y bergantung secara linier dengan dua peubah independen. Rumus umum regresi linear yang bisa digunakan untuk regresi linear berganda adalah

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

Agar nilai dari tiap-tiap β_i dapat diketahui, bisa digunakan *Normal Estimation*

Equation for Multiple Linear Regression sebagai berikut

$$\begin{aligned} n\beta_0 + \beta_1 \sum_{i=1}^n x_{1i} + \beta_2 \sum_{i=1}^n x_{2i} + \dots + \beta_k \sum_{i=1}^n x_{ki} &= \sum_{i=1}^n y_i \\ \beta_0 \sum_{i=1}^n x_{1i} + \beta_1 \sum_{i=1}^n x_{1i}^2 + \beta_2 \sum_{i=1}^n x_{1i}x_{2i} + \dots + \beta_k \sum_{i=1}^n x_{1i}x_{ki} &= \sum_{i=1}^n x_{1i}y_i \\ \vdots &\quad \vdots & \vdots & \vdots \\ \beta_0 \sum_{i=1}^n x_{ki} + \beta_1 \sum_{i=1}^n x_{ki}x_{1i} + \beta_2 \sum_{i=1}^n x_{ki}x_{2i} + \dots + \beta_k \sum_{i=1}^n x_{ki}^2 &= \sum_{i=1}^n x_{ki}y_i \end{aligned}$$

Bentuk ini merupakan sebuah sistem persamaan linier dan dapat diselesaikan dengan berbagai metode pilihan, seperti eliminasi Gauss.

BAB III

IMPLEMENTASI PUSTAKA DAN PROGRAM DALAM JAVA

3.1 Class Matrix

PROPERTIES
<pre>private double[][] contents // tempat menyimpan matriks</pre> <pre>Static final Matrix inverseBicubicCoefficientMatrix // tempat menyimpan singleton hasil inverse koefisien untuk perhitungan bicubic</pre>
CONSTRUCTOR
<pre>public Matrix(int nRow, int nCol) // Matriks konstruktor. Menginisiasi matriks dengan ukuran baris x kolom. // Prekondisi: nRow, nCol >= 1</pre>
CLASS METHODS
<pre>public int getNRow() // mengembalikan jumlah baris pada matriks</pre>
<pre>public int getNCol() // mengembalikan jumlah kolom pada matriks</pre>
<pre>public double[] getRow(int rowIdx) // mengembalikan baris indeks ke-rowIdx</pre>
<pre>public void setRow(int rowIdx, double[] row) // Mengubah baris indeks ke-rowIdx dengan baris baru // Prekondisi: baris baru memiliki jumlah kolom sama dengan matriks. // I.S. matriks terdefinisi // F.S. baris indeks ke-rowIdx diubah dengan baris baru</pre>
<pre>public double[] getCol(int colIdx) // mengembalikan kolom indeks ke-colIdx</pre>
<pre>public double[][] getContents() // mengembalikan konten matriks</pre>

```

public void setContents(double[][] contents)
// Mengubah keseluruhan matriks
// Prekondisi: contents merupakan array dengan
panjang >= 1 yang seluruh elemennya merupakan array
dengan panjang sama dan >= 1
// I.S matriks terdefinisi
// F.S. matriks berubah sesuai dengan contents

public double getElmt(int rowIdx, int colIdx)
// mengembalikan elemen matriks pada indeks (rowIdx,
colIdx)

public void setElmt(int rowIdx, int colIdx, double val)
// Mengubah elemen matriks pada indeks (rowIdx,
colIdx) menjadi val
// I.S: matriks terdefinisi, rowIdx dan colIdx
merupakan indeks yang valid
// F.S: elemen matriks pada indeks (rowIdx, colIdx)
menjadi val

public boolean isSPL()
// Mengecek apakah matriks merupakan matriks SPL yang
valid (minimal baris sejumlah kolom - 1)
// mengembalikan true jika matriks merupakan matriks
SPL yang valid, false jika tidak

public boolean isSquare()
// Mengembalikan true jika matriks merupakan matriks
persegi, false jika tidak

public boolean isPoints()
// Mengembalikan true jika matriks merupakan kumpulan
point (satu baris terdiri atas dua kolom x dan y)

public boolean isRowEmpty(int rowIdx)
// Mengembalikan true jika semua elemen pada baris
ke-rowIdx bernilai 0, false jika tidak

public void swapRow(int idx1, int idx2)
// Menukar baris ke-idx1 dengan baris ke-idx2
// I.S. matriks terdefinisi, idx1 dan idx2 merupakan
indeks yang valid
// F.S. baris ke-idx1 menjadi baris ke-idx2 dan
sebaliknya

public void transpose()

```

```

// Mentranspose matriks
// I.S. matriks terdefinisi
// F.S. matriks menjadi transpose dari matriks
// sebelumnya (a[i][j] menjadi a[j][i])

public void fillZero()
// Mengisi seluruh elemen pada matriks dengan nol
// I.S. matriks terdefinisi
// F.S. seluruh elemen pada matriks menjadi nol

public void multiplyRowScalar(int rowIdx, double scalar)
// Mengalikan baris ke-rowIdx dengan skalar
// I.S. matriks terdefinisi, rowIdx merupakan indeks
// yang valid
// F.S. baris ke-rowIdx menjadi baris ke-rowIdx
// dikali skalar

public void multiplyScalar(double scalar)
// Mengalikan matriks dengan skalar
// I.S. matriks terdefinisi
// F.S. semua elemen pada matriks dikalikan dengan
// skalar

public Matrix getCopyMatrixByColumn(int startColIdx,
int endColIdx)
// Prekondisi: startColIdx dan endColIdx merupakan
// indeks kolom yang valid dan endColIdx > startColIdx
// Mengembalikan matriks baru yang merupakan salinan
// matriks dari kolom startColIdx sampai kolom endColIdx

public Matrix getNonZeroRowIdx(int startRowIndex, int
endRowIndex, int colIdx)
// Prekondisi: startRowIndex dan endRowIndex merupakan
// indeks baris yang valid dan endRowIndex > startRowIndex.
// colIdx merupakan indeks kolom yang valid
// mengembalikan idx baris ditemukan pertama kali
// yang tidak nol dalam satu kolom. Jika tidak
// ditemukan, akan mengembalikan (-1)

public void makeColumnZero(int leadingOneRowIndex, int
startRowIndex, int endRowIndex, int mainColIdx)
// I.S. nilai dari matriks pada index startRowIndex dan
// rowIdx sudah bernilai 1
// F.S. satu kolom dari matriks dari startRowIndex
// hingga endRowIndex memiliki nilai nol

```

```

public Pair<Matrix, Double> getEchelonForm(int startColIdx, int endColIdx)
// Prekondisi: startColIdx dan endColIdx merupakan indeks kolom yang valid dan endColIdx > startColIdx
// Mengembalikan pair berisi bentuk matriks eselon dari startColIdx hingga endColIdx dan total perkalian yang dilakukan

public Matrix getReducedForm(int startColIdx, int endColIdx)
// Prekondisi: startColIdx dan endColIdx merupakan indeks kolom yang valid dan endColIdx > startColIdx
// Mengembalikan matriks dengan kolom yang terambil sudah dalam bentuk baris eselon tereduksi

public Matrix getSolG() throws NoSolutionException
// Prekondisi: jumlah kolom matriks >= 2
// Matriks dengan matriks format solusi yang dapat ditampilkan, termasuk dengan variabel parametrik
// throws NoSolutionException ketika matriks SPL tidak memiliki solusi

public Matrix getSolGJ() throws NoSolutionException
// Prekondisi: jumlah kolom matriks >= 2
// Mengembalikan matriks dengan matriks format solusi yang dapat ditampilkan, termasuk dengan variabel parametrik
// throws NoSolutionException ketika matriks SPL tidak memiliki solusi

public Matrix getSolInverse() throws NoInverseException
// Prekondisi: jumlah kolom matriks >= 2
// Mengembalikan Matriks solusi SPL dengan metode pengalian dengan inverse (kolom 1)
// throws NoSolutionException ketika matriks variabel pada SPL tidak memiliki inverse

public Matrix getAugmentedMatrixByIdentity()
// Prekondisi: matriks merupakan maktriks square
// mengembalikan matriks yang telah di augmentasi dengan matriks identitas

public Matrix getInverseOBE() throws NoInverseException
// mengembalikan invers matriks dengan metode OBE
// throws NoInverseException jka matriks tidak

```

```

memiliki inverse

public Matrix getInverseAdjoin() throws
NoInverseException
// mengembalikan invers matriks dengan metode adjoin
// throws NoInverseException jika matriks tidak
memiliki inverse

public double getDeterminantTriangle() throws
InvalidMatrixSquareException
// Mengembalikan nilai determinan matriks dengan
metode segitiga atas
// throws InvalidMatrixSquareException ketika matriks
masukan bukanlah matrix persegi

public Matrix getMinor(int rowIdx, int colIdx)
// Prekondisi: (rowIdx, colIdx) merupakan indeks yang
valid
// Mengembalikan matriks dengan baris ke rowIdx dan
kolumn ke colIdx dihilangkan

public void toValidSPL()
// Mengubah matriks menjadi matriks SPL yang valid
dengan membuat baris baru dengan elemen 0 hingga
jumlah baris sama dengan jumlah kolumn - 1
// I.S. matriks terdefinisi
// F.S. matriks merupakan matriks SPL yang valid

public double getDeterminantCofactor() throws
InvalidMatrixSquareException
// Mengembalikan determinan matriks dengan metode
kofaktor
// throws InvalidMatrixSquareException jika bukan
matriks persegi

public Matrix getSolCramer() throws
Errors.NoSolutionException
// Mencari solusi SPL dengan metode cramer
// Mengembalikan matriks solusi SPL unik (kolumn
berjumlah 1)
// throws NoSolutionException jika SPL tidak memiliki
solusi yang unik

public Matrix getPolynomialFunction() throws
NoSolutionException
// Fungsi untuk mendapatkan solusi dari interpolasi
polinomial

```

```

// Prekondisi: matriks berupa matriks point yaitu
dengan kolom berjumlah 2
// Mengembalikan matriks solusi interpolasi
polinomial derajat n yaitu matriks (n+1) x
// throws NoSolutionException jika SPL dari polinom
tidak memiliki solusi yang unik

public double getValuePolinomial(double x)
// Fungsi untuk mendapatkan hasil estimasi nilai x
pada interpolasi polinomial
// Prekondisi: matriks berupa hasil return dari
getPolynomialFunction
// Mengembalikan hasil estimasi

public double getValueBicubicSpecific(double b,
double a)
// Fungsi untuk mendapatkan solusi dari interpolasi
polinomial
// Prekondisi: matriks berupa matriks point yaitu
dengan kolom berjumlah 2
// Mengembalikan nilai f(a,b) yang telah di
interpolasi

public Matrix getBicubicFunction(int rowStart, int
colStart)
// Prekondisi: rowStart + 3 < this.getNRow() dan
colStart + 3 < this.getNCol()
// Mengembalikan matriks yang berisi koefisien2 dari
interpolasi bicubic

public double getValueBicubic(double a, double b, int
startRow, int startCol)
// Fungsi untuk mendapatkan nilai interpolasi bicubic
dari matriks solusi bicubic
// Prekondisi: matriks merupakan matriks hasil dari
this.getBicubicFunction
// Mengembalikan nilai f(a,b) yang telah di
interpolasi

public double getValueBilinear(int rowStart, int
colStart, double a, double b)
// Fungsi untuk mendapatkan solusi dari interpolasi
bilinear
// Mengembalikan nilai f(a,b) yang telah
diinterpolasi

public Matrix getNTimesSizeMatrix(int scalingFactor)

```

```

// Mengembalikan Matriks yang sudah diperbesar
scalingFactor kali dengan interpolasi bicubic dan
interpolasi bilinier pada edges nya (digunakan untuk
memperbesar gambar)

public int getMostDigit(int digitAfterComma)
// Menghasilkan panjang dari angka paling panjang
dalam matriks setelah diformat dengan digitAfterComma
angka di belakang koma

public int getMostDigit()
// Mengembalikan panjang dari angka paling panjang
dalam matriks setelah diformat dengan 2 angka di
belakang koma

public double getAverageByColumnAndRow(int
startRowIdx, int endRowIdx, int startColIdx, int
endColIdx)
// Mencari rata-rata dari sebuah bagian matriks.
Index inklusif.
// Prekondisi: 0 <= rowStart <= rowEnd < getNRow() &&
0 <= colStart <= colEnd < getNCol()
// Mengembalikan rata-rata dari nilai sel pada bagian
yang diberikan

public double getAverageByColumn(int startColIdx, int
endColIdx)
// Mencari rata-rata dari sebagian matriks dari kolom
sekian sampai kolom sekian. Baris yang dihitung
adalah seluruh baris. Index inklusif.
// Prekondisi: 0 <= colStart <= colEnd < getNCol()
// Mengembalikan rata-rata dari nilai sel pada bagian
yang diberikan

public double getAverageOfColumn(int colIdx)
// Mencari rata-rata dari salah satu kolom pada
matrix.
// Prekondisi: colIdx merupakan index kolom yang
valid
// Mengembalikan rata-rata dari nilai sel pada kolom
yang diberikan

public double getAverageByRow(int startRowIdx, int
endRowIdx)
// Mencari rata-rata dari sebagian matriks dari baris
sekian sampai baris sekian. Kolom yang dihitung
adalah seluruh kolom. Index inklusif.

```

```

// Prekondisi: 0 <= startRowIndex <= endRowIndex <
getNRow()
// Mengembalikan rata-rata dari nilai sel pada bagian
yang diberikan

public double getAverageOfRow(int rowIdx)
// Mencari rata-rata dari salah satu baris pada
matrix.
// Prekondisi: rowIdx merupakan index baris yang
valid
// Mengembalikan rata-rata dari nilai sel pada baris
yang diberikan

STATIC METHODS

public static boolean isNColumnSame(Matrix matriks1,
Matrix matriks2)
// Mengembalikan apakah kedua matriks memiliki jumlah
kolom yang sama.

public static boolean isNRowSame(Matrix matriks1,
Matrix matriks2)
// Mengembalikan apakah kedua matriks memiliki jumlah
baris yang sama.

public static boolean isDimensionSame(Matrix
matriks1, Matrix matriks2)
// Mengembalikan apakah kedua matriks memiliki jumlah
kolom yang sama.

public static boolean isMultipliable(Matrix matriks1,
Matrix matriks2)
// Mengembalikan apakah matriks pertama punya jumlah
kolom sebanyak matriks kedua.

public static Matrix multiply(Matrix matriks1, Matrix
matriks2) throws InvalidMatrixSizeException
// Mengalikan 2 buah matriks
// Mengembalikan matriks baru yang merupakan hasil
perkalian kedua matriks yang
// throws InvalidMatrixSizeException ketika kedua
matriks tidak multipliable

public static Matrix add(Matrix matriks1, Matrix
matriks2) throws InvalidMatrixSizeException
// Menjumlahkan 2 buah matriks
// Mengembalikan matriks baru yang merupakan hasil

```

```

penjumlahan dua buah matriks
// throws InvalidMatrixSizeException ketika kedua
matriks tidak memiliki dimensi yang sama

public static Matrix subtract(Matrix matriks1, Matrix
matriks2) throws InvalidMatrixSizeException
// Mengurangkan 2 buah matriks. Matriks pertama
dikurangi matriks kedua.
// Mengembalikan matriks baru yang merupakan hasil
pengurangan matriks pertama oleh matriks kedua
// throws InvalidMatrixSizeException ketika kedua
matriks tidak memiliki dimensi yang sama

private static Matrix getInverseBicubicCoefficient()
// Mengembalikan matriks yang merupakan hasil inverse
dari koefisien bicubic

```

3.2 Class Pair

PROPERTIES
<pre> public F first // nilai pertama pair dengan tipe generic public S second // nilai kedua pair dengan tipe generic </pre>
CONSTRUCTOR
<pre> public Pair(F first, S second) // Membuat pair dengan first sebagai elemen pertama dan second sebagai elemen kedua. </pre>

3.3 Class Image

PROPERTIES
<pre> private Matrix rMat, gMat, bMat, aMat // Matriks untuk menyimpan nilai r,g,b,a gambar </pre>
CONSTRUCTOR
<pre> public Image(File file) throws IOException // Membuat image dari file </pre>

```

// throws IOException jika file bukan merupakan
gambar

public Image(Matrix rMat, Matrix gMat, Matrix bMat,
Matrix aMat)
// Membuat image dari matrix r,g,b,a
// Prekondisi: semua matrix memiliki ukuran yang sama

CLASS METHODS

public int getPixelSize()
// Mengembalikan banyak total pixel pada gambar

private void setMatrixFromImage(BufferedImage img)
// I.S: image bisa terdefinisi atau tidak
// F.S: r,g,b,a image terisi dari bufferedimage yang
diberikan

public BufferedImage getBufferedImage()
// Mengembalikan bufferedimage berdasarkan matriks
r,g,b,a image

private static int onColorRange(int color)
// Mengembalikan nilai color jika berada di range
0-255, 0 jika kurang dari 0, 255 jika lebih dari 255

public void scale(int scalingFactor)
// I.S. image terdefinisi
// F.S. image diperbesar sebanyak scalingFactor kali

```

3.4 Class Errors

```

static public class NoSolutionException extends
Exception
// Error SPL tidak memiliki solusi

static public class NoInverseException extends
Exception
// Error matriks tidak memiliki invers

static public class InvalidMatrixSquareException
extends Exception
// Error matrix tidak persegi

static public class InvalidMatrixSizeException
extends Exception

```

```
// Error ukuran matriks tidak valid
```

3.5 Class FromFile

CLASS METHODS

```
static private String[] readFile()
// Menggunakan fungsi getFile untuk mendapat nama
file, kemudian mengembalikan array berisi string
baris tiap file

static private RowError isFileContentValid(String[]
rowStrings)
// Mengecek apakah isi file adalah barisan matriks
yang valid.
// Valid berarti semua karakter adalah angka (spasi
dikecualikan), tidak ada kelebihan spasi antar kolom,
dan jumlah kolom antar baris konsisten

static private Matrix rowStringsToMatrix(String[]
rowStrings)
// Menerima array berisi string yang merupakan isi
file, kemudian mengkonversinya menjadi matriks
// Prekondisi : Array masukan adalah barisan SPL yang
valid

static private Matrix Matrix()
// Menggunakan readFile, rowStringToMatrix, dan
isFileContentValid untuk meminta file berisi matrix
dari pengguna lalu mengkonversinya menjadi matrix
// Mencetak pesan error jika file masukan setelah
dicek isFileContentValid memiliki error

static private Matrix MatrixSquare()
// Menggunakan readFile, rowStringToMatrix, dan
isFileContentValid untuk meminta file berisi matrix
dari pengguna lalu mengkonversinya menjadi matrix
// Mencetak pesan error jika file masukan setelah
dicek isFileContentValid memiliki error

static private Matrix MatrixSquare()
// Sama seperti fungsi Matrix di class ini, namun
matriks dalam file harus berbentuk persegi.
// Jika tidak persegi maka diberikan pesan error dan
diulangi.
```

```

static public Matrix pointsInput(String message)
// Meminta file berisi titik-titik, lalu
mengkonversinya menjadi sebuah matriks
// Memberikan pesan error dan mengulangi input jika
isi file bukan titik-titik

static public Matrix matrixInput(String message,
boolean isSquare)
// Mencetak parameter message yang diberikan
// Meminta file berisi matriks dari pengguna lalu
mengembalikan matriks dalam file tersebut.

static public Matrix SPL()
// Menggunakan matrixInput untuk meminta file berisi
SPL dari pengguna lalu mengembalikan SPL dalam file
tersebut
// Memberikan pesan yang sesuai untuk meminta SPL ke
matrixInput

static public Matrix matrixToDetermine()
// Menggunakan matrixInput untuk meminta file berisi
matriks yang ingin dicari determinannya dari pengguna
lalu mengembalikan matriks dalam file tersebut
// Memberikan pesan yang sesuai untuk meminta matriks
ke matrixInput dan matriks dalam file haruslah
persegi

static public Matrix matrixToInterpolate()
// Menggunakan matrixInput untuk meminta file berisi
matriks yang ingin dicari interpolasinya dari
pengguna lalu mengembalikan matriks dalam file
tersebut
// Memberikan pesan yang sesuai untuk meminta matriks
ke matrixInput

static public Matrix matrixToInvert()
// Menggunakan matrixInput untuk meminta file berisi
matriks yang ingin dicari inversnya dari pengguna
lalu mengembalikan matriks dalam file tersebut
// Memberikan pesan yang sesuai untuk meminta matriks
ke matrixInput

static public Matrix[] MLR()
// Menggunakan matrixInput untuk meminta file berisi
sampel data yang ingin dicari regresi linear
bergandanya
// Setelah mendapat matrix berisi sampel data dari

```

```

matrixInput kemudian meminta file berisi data yang
ingin diprediksi
// Mengembalikan array berisi matriks sampel data dan
matriks data yang ingin diprediksi

static public Image readImage()
// Meminta file gambar dengan fungsi getFile lalu
mengkonversinya menjadi instance dari class Image,
instance kemudian dikembalikan

```

3.6 Class FromKeyboard

CLASS METHODS

```

static public int readInt(String numberName, int
lowerBound, int upperBound)
// Mencetak prompt message dengan menggunakan apa
yang diminta (numberName)
// Membaca bilangan bulat masukan pengguna lalu
mengembalikan masukannya.
// Valid adalah berada di antara lowerBound dan
upperBound inklusif
// Jika tidak valid maka mengulang input sampai valid

static public int readIntWithMinimum(String
numberName, int lowerBound)
// Mencetak prompt message dengan menggunakan apa
yang diminta (numberName)
// Membaca bilangan bulat masukan pengguna lalu
mengembalikan masukannya.
// Valid adalah berada di atas atau sama dengan
lowerBound

static public int readIntWithMaximum(String
numberName, int upperBound)
// Mencetak prompt message dengan menggunakan apa
yang diminta (numberName)
// Membaca bilangan bulat masukan pengguna lalu
mengembalikan masukannya.
// Valid adalah berada di bawah atau sama dengan
upperBound

static public double readDouble(String numberName)
// Mencetak prompt message dengan menggunakan apa
yang diminta (numberName)

```

```

// Membaca bilangan real masukan pengguna lalu dari
// pengguna lalu mengembalikan masukannya

static public String readString(String prompt)
// Mencetak parameter prompt
// Membaca line string masukan pengguna lalu
mengembalikan masukannya.

static private double[] readRow(int nCol)
// Membaca masukan baris matriks dari pengguna lalu
mengkonversinya menjadi sebuah array berisi double
// Jika tidak valid (jumlah kolom tidak sesuai, isi
kolom bukan angka, ada spasi yang berlebih) maka
mengulangi input sampai valid

static private Matrix Matrix(int nRow, int nCol,
String message)
// Membaca matriks dengan baris sebanyak nRow dan
kolom sebanyak nCol dari pengguna, baris per baris.
// Mengembalikan matriks yang telah dibaca.

static public Matrix MatrixSquare()
// Membaca matriks persegi dengan jumlah baris dan
kolom sebanyak masukan dari pengguna
// Mengembalikan matriks yang telah dibaca.

static public Matrix Points(String prompt)
// Mencetak prompt
// Meminta jumlah titik yang ingin dimasukkan
// Membaca matriks dengan kolom sepanjang 2 dan baris
sebanyak jumlah titik yang dimasukkan

static public Matrix SPL()
// Membaca sistem persamaan linear dengan jumlah
variabel dan jumlah persamaan diberikan oleh pengguna
// Mengembalikan matriks augmented dari SPL yang
telah dibaca.

static public Matrix Matrix()
// Membaca matriks dengan jumlah baris dan kolom
sebanyak masukan dari pengguna
// Mengembalikan matriks yang telah dibaca.

static public Matrix[] MLR()
// Membaca sampel data untuk regresi linier berganda
dengan jumlah variabel dan jumlah sampel data
diberikan oleh pengguna

```

```

// Setelah membaca sampel data, kemudian membaca data
// yang ingin diprediksi
// Mengembalikan array berisi matriks sampel data dan
// matriks data yang ingin diprediksi

```

3.7 Class IOLib

CLASS METHODS
static public RowError checkRow(String rowString, int nCol) // Mencetak prompt message dengan menggunakan apa yang diminta (numberName) // Membaca bilangan bulat masukan pengguna lalu mengembalikan masukannya. // Valid adalah berada di atas atau sama dengan lowerBound
static public boolean chooseToReadFromFile() // Menanyakan pengguna apakah ingin membaca masukan dari file // Mengembalikan boolean apakah pengguna ingin membaca dari file
static public boolean chooseToWriteToFile() // Menanyakan pengguna apakah ingin menulis masukan ke file // Mengembalikan boolean apakah pengguna ingin menulis ke file
static public String addWhiteSpace(String string, int amount) // Mengembalikan parameter string yang sudah ditambah whitespace ke bagian depannya sejumlah parameter amount

3.7.1 Class RowError

PROPERTIES
private boolean containWhiteSpace; // apakah baris memiliki kelebihan baris private boolean columnDiscrepancy;

```

// apakah baris jumlah kolomnya tidak sesuai
private boolean containNaN;
// apakah kolom pada baris memiliki elemen yang bukan
angka
private boolean empty;
// apakah baris kosong

```

CONSTRUCTOR

```

public RowError()
// Membuat instance dengan tidak ada error

```

CLASS METHODS

```

public boolean anyError()
// apakah ada error apapun pada instance

```

```

public void setError(String errorType, boolean newValue)
// Mengubah property pada instance
// errorType melambangkan properti mana yang diubah,
newValue adalah nilai baru properti yang diubah

```

```

public boolean isWhitespace()
// mengembalikan nilai properti containWhitespace

```

```

public boolean isNaN()
// mengembalikan properti containNaN

```

```

public boolean isEmpty()
// mengembalikan nilai properti empty

```

```

public void reset()
// membuat nilai semua properti menjadi false

```

3.7.2 Class SPLSolution

CLASS METHODS

```

static private String parameterVariableMaker(int index)
// Membentuk variabel yang unik untuk tiap indeksnya
dengan penggabungan karakter yang diperbolehkan di
dalam fungsi

```

```

static private boolean isNthVariableParametric(Matrix

```

```

solution, int i)
// Mengubah property pada instance
// errorType melambangkan properti mana yang diubah,
newValue adalah nilai baru properti yang diubah

static private boolean
isNthVariableContainParametric(Matrix solution, int
i)
// Mengecek apakah suatu variabel dalam matriks
solusi nilainya bergantung pada variabel parametrik

static public String[] createSolutionTexts(Matrix
solution, String variableSymbol, int
digitsAfterComma)
// Mengubah matriks solusi menjadi array berisi
string solusi variabel yang bisa dibaca manusia

static public String[] createSolutionTexts(Matrix
solution, String variableSymbol)
// Mengubah matriks solusi menjadi array berisi
string solusi variabel yang bisa dibaca manusia
dengan angka diformat ke 2 angka di belakang koma

static public void print(Matrix solution, String
variableSymbol)
// Mencetak matriks solusi yang telah diubah menjadi
persamaan dan variabel ke layar

static public void print(Matrix solution)
// Mencetak matriks solusi yang telah diubah menjadi
persamaan dan variabel ke layar

static public void printUnique(Matrix solution,
String variableSymbol)
// Mencetak matriks solusi unik menjadi persamaan dan
variabel ke layar dengan

static public void printUnique(Matrix solution)
// Mencetak matriks solusi unik menjadi persamaan dan
variabel ke layar

static public void print(Matrix solution, String
variableSymbol)
// membuat nilai semua properti menjadi false

```

3.7.3 Class MLR

CLASS METHODS
<pre>static public String createEquationText(List<Double> betaList) // Membuat persamaan fungsi hasil regresi linear dengan list berisi koefisien fungsi // Mengembalikan string hasil persamaannya</pre>
<pre>static public String createResultText(double[] predictedDataRow) // Mengembalikan String hasil prediksi data regresi yang dibuat dari predictedDataRow // predictedDataRow berisi nilai variabel data yang ingin diprediksi kecuali anggota terakhirnya yang adalah hasil prediksi dengan data tersebut</pre>
<pre>static public String[] createArrayOfResultText(Matrix predictedData) // Menerima matriks berisi hasil prediksi lalu mengonversinya menjadi array of string yang tiap anggotanya adalah persamaan hasil regresi // Mengembalikan hasil konversi</pre>

3.8 Class ToFile

CLASS METHODS
<pre>private static String getValidFilename() // Meminta masukan nama file yang ingin dibuat dari pengguna // Jika ketika membuat file dengan nama tersebut terdapat error, berikan pesan error yang sesuai dan minta nama file lagi sampai tidak ada error // Kembalikan nama file yang telah berhasil dibuat</pre>
<pre>public static void writeToFile(String[] rowStrings) // Membuat file baru dengan nama file didapatkan dengan getValidFilename // Menulis array rowStrings ke file dengan tiap anggotanya dalam satu baris file</pre>
<pre>public static void SPL(Matrix solution) // Mengonversi matriks solusi ke bentuk string</pre>

```

persamaan
// Menuliskannya hasil konversi ke dalam file dengan
fungsi writeToFile

public static void determinant(double determinant)
// Menuliskan pesan determinan matriks dengan
parameter determinant yang diberikan menggunakan
fungsi writeToFile

public static String[] matrixToString(Matrix
outputtedMatrix, int digitAfterComma)
// Mengubah matriks yang diberikan menjadi array
berisi string tiap baris matriks
// String adalah baris matriks dengan tiap kolom
dipisahkan spasi
// Anggota dari matriks diformat digit di belakang
komanya sejumlah parameter digitAfterComma

public static String[] matrixToString(Matrix
outputtedMatrix)
// Mengubah matriks yang diberikan menjadi array
berisi string tiap baris matriks
// String adalah baris matriks dengan tiap kolom
dipisahkan spasi
// Anggota dari matriks diformat menjadi 2 digit di
belakang koma

public static void inverse(Matrix invertedMatrix)
// Menuliskan matriks yang diberikan ke file dengan
fungsi writeToFile

public static void MLR(List<Double> betaList, Matrix
predictedData)
// Menerima list berisi koefisien persamaan linier
dan matriks berisi data yang ingin diprediksi dan
prediksinya.
// Menuliskan fungsi regresi yang didapatkan dari
betaList dan hasil prediksi data ke file dengan
writetofile

public static void exportImageFile(Image img)
// Menuliskan instance Image yang diterima lalu
menuliskannya ke format png

```

3.8.1 Class CreateFileCondition

PROPERTIES
<pre>private boolean createFileSuccessful; // apakah file berhasil dibuat private boolean fileAlreadyExists; // apakah file sudah ada</pre>
CONSTRUCTOR
<pre>public CreateFileCondition() // Membuat instance dengan dua properti yang ada bernilai false</pre>
CLASS METHODS
<pre>public void setCreateSuccessful(boolean newValue) // Mengubah properti createFileSuccessful menjadi bernilai newValue</pre>
<pre>public void setCreateSuccessful() // Mengubah properti createFileSuccessful menjadi bernilai True</pre>
<pre>public void setFileExists(boolean newValue) // Mengubah properti fileAlreadyExists menjadi bernilai newValue</pre>
<pre>public void setFileExists() // Mengubah properti fileAlreadyExists menjadi bernilai True</pre>
<pre>public boolean otherError() // Mengembalikan apakah ada error lain selain file sudah ada // True ketika gagal membuat file tapi file belum ada</pre>
<pre>public boolean isSuccessful() // Mengembalikan nilai properti createFileSuccessful</pre>
<pre>public boolean isExists() // Mengembalikan nilai properti fileAlreadyExists</pre>

3.9 Class ToKeyboard

CLASS METHODS
<pre>static public void printMatrix(Matrix outputtedMatrix, int digitAfterComma) // Menuliskan matriks ke CLI baris per baris dengan tiap elemen diformat dua digit di belakang koma</pre>
<pre>static public void printNumber(double number, String message) // Mencetak parameter number ke CLI dalam satu baris tersendiri</pre>
<pre>static public void printMessage(String message) // Mencetak parameter message ke CLI dalam satu baris tersendiri</pre>
<pre>public static void clearConsole() // Membersihkan / mereset isi CLI</pre>
<pre>static public void printMLR(List<Double> betaList, Matrix predictedData) // Menerima list berisi koefisien persamaan linier dan matriks berisi data yang ingin diprediksi dan prediksinya. // Menuliskan fungsi regresi yang didapatkan dari betaList dan hasil prediksi data ke CLI</pre>

3.10 Class CoreFunctionality

CLASS METHODS

3.10.1 Class solveSPL

CLASS METHODS
<pre>static public void gauss() // Prosedur yang menerima masukan SPL dari pengguna lalu mencetak solusinya. Solusi didapatkan dari eliminasi Gauss.</pre>

```

static public void gaussJordan()
// Prosedur yang menerima masukan SPL dari pengguna
lalu mencetak solusinya. Solusi didapatkan dari
eliminasi Gauss-Jordan.

static public void inverseMatrix()
// Prosedur yang menerima masukan SPL dari pengguna
lalu mencetak solusinya. Solusi didapatkan dari
metode matriks balikan.

static public void inverse()
// Prosedur yang menerima masukan SPL dari pengguna
lalu mencetak solusinya. Solusi didapatkan dari
matriks balikan.

static public void cramer()
// Prosedur yang menerima masukan SPL dari pengguna
lalu mencetak solusinya. Solusi didapatkan dari
matriks balikan.

```

3.10.2 Class Inverse

CLASS METHODS
static public void adjoin() // Prosedur yang menerima masukan matriks dari pengguna lalu mencetak inverse-nya. Jika matriks singular maka error ditangkap dan dikeluarkan ke layar bahwa matriks singular. static public void obe() // Prosedur yang menerima masukan matriks dari pengguna lalu mencetak inverse-nya. Jika matriks singular maka error ditangkap dan dikeluarkan ke layar bahwa matriks singular.

3.10.2 Class Interpolation

CLASS METHODS
static public void polinomial() // Prosedur untuk menerima input titik-titik yang ingin dilakukan interpolasi polinomial lalu mencetak

hasil interpolasinya
<pre>static public void bicubic() // Prosedur untuk menerima input matriks 4x4 yang ingin dilakukan interpolasi bicubic lalu menampilkan hasilnya</pre>
<pre>static public void imageScaling() // Prosedur untuk menerima input gambar lalu melakukan scaling sesuai scale faktor permintaan pengguna</pre>

3.10.3 Class Determinant

CLASS METHODS
<pre>static public void ekspansiKofaktor() // Prosedur yang menerima masukan matriks dari pengguna lalu mencetak determinannya. // Determinan dihitung dengan ekspansi kofaktor.</pre>
<pre>static public void segitigaAtas() // Prosedur yang menerima masukan matriks dari pengguna lalu mencetak determinannya. // Determinan dihitung dengan mengubah ke bentuk eselon tereduksi.</pre>

3.10.4 Class MLR

CLASS METHODS
<pre>static private List<Double> betaProducer(Matrix data) // Menerima matriks sampel data lalu mengembalikan list beta dari fungsi regresi linier bergandanya</pre>
<pre>static private Matrix resultGenerator(List<Double> betaList, Matrix dataToPredict) // Menghasilkan prediksi data yang diberikan dengan menggunakan betaList</pre>
<pre>static public void compute() // Meminta sampel data dan data yang ingin diprediksi // Mencetak fungsi regresi linier</pre>

3.11 Class Interface

CLASS METHODS
<pre>static public void printMenu(String[] menu) // Mencetak menu yang diberikan beserta nomornya.</pre>
<pre>static public void menuSPL() // Prosedur untuk menampilkan submenu SPL // Menerima input pilihan submenu pengguna dan menjalankan subprogram yang bersesuaian</pre>
<pre>static public void menuInverse() // Prosedur untuk menampilkan submenu penghitungan inverse // Menerima input pilihan submenu pengguna dan menjalankan subprogram yang bersesuaian</pre>
<pre>static public void menuDeterminant() // Prosedur untuk menampilkan submenu penghitungan determinan // Menerima input pilihan submenu pengguna dan menjalankan subprogram yang bersesuaian</pre>
<pre>static public void mainEventLoop() // Menerima list berisi koefisien persamaan linier dan matriks berisi data yang ingin diprediksi dan prediksinya. // Menuliskan fungsi regresi yang didapatkan dari betaList dan hasil prediksi data ke CLI</pre>

3.11.1 Class MenuText

PROPERTIES
<pre>private static final String[] main // Pilihan menu utama private static final String[] SPL // Pilihan submenu SPL private static final String[] inverse // Pilihan submenu inverse private static final String[] determinant // Pilihan submenu determinan</pre>

BAB IV

EXPERIMEN

4.1 SPL

4.1.1 Studi Kasus 1a

Input
$A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 5 & -7 & -5 \\ 2 & -1 & 1 & 3 \\ 5 & 2 & -4 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 6 \end{bmatrix}$
<pre>test > 1a.txt 1 1 1 -1 -1 1 2 2 5 -7 -5 -2 3 2 -1 1 3 4 4 5 2 -4 2 6</pre>
Output
<pre>1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Kaidah Cramer 5. Kembali ke main menu Masukkan pilihan sub menu : 1 Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 1 Masukkan nama file berisi matriks augmented yang merepresentasikan SPL : ./test/1a.txt SPL tidak punya solusi Tekan enter untuk kembali ke menu utama █</pre>
Keterangan
Ax = b tidak memiliki solusi, diselesaikan dengan Gauss.

4.2.2 Studi Kasus 1b

Input

	<pre>test > 1b.txt</pre> $A = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -3 & 0 \\ 2 & -1 & 0 & 1 & -1 \\ -1 & 2 & 0 & -2 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 6 \\ 5 \\ -1 \end{bmatrix}$	$\begin{array}{ccccccccc} 1 & 1 & -1 & 0 & 0 & 1 & 3 \\ 2 & 1 & 1 & 0 & -3 & 0 & 6 \\ 3 & 2 & -1 & 0 & 1 & -1 & 5 \\ 4 & -1 & 2 & 0 & -2 & -1 & -1 \end{array}$
Output		
<pre> 1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Kaidah Cramer 5. Kembali ke main menu Masukkan pilihan sub menu : 2 Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 1 Masukkan nama file berisi matriks augmented yang merepresentasikan SPL : ./test/1b.txt Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI. Masukkan pilihan anda : 2 Solusi matriks adalah seperti di bawah ini. Perlu diingat bahwa variabel (contoh) ab bukanlah a * b melainkan variabel unik tersendiri x1 = 3+r x2 = 2r x3 = s x4 = -1+r x5 = r Tekan enter untuk kembali ke menu utama █ </pre>		
Keterangan		
Terdapat 2 variabel bebas yaitu x3 dan x5, diselesaikan dengan Gauss-Jordan.		

4.2.3 Studi Kasus 1.c

	<p>Input</p> $A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ -1 \\ 1 \end{bmatrix}$	<pre>test > 1c.txt</pre> $\begin{array}{ccccccccc} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 2 \\ 2 & 0 & 0 & 0 & 1 & 1 & 0 & -1 \\ 3 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 4 & & & & & & & \end{array}$
Output		

```

1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer
5. Kembali ke main menu

Masukkan pilihan sub menu : 3
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks augmented yang merepresentasikan SPL :
../test/lc.txt
SPL tidak dapat diselesaikan dengan metode ini karena tidak mempunyai inverse ! Berikut solusinya dengan eliminasi Gauss-Jordan :
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 2
Solusi matriks adalah seperti di bawah ini. Perlu diingat bahwa variabel (contoh) ab bukanlah a * b melainkan variabel unik tersendiri
x1 = t
x2 = 1-1r
x3 = s
x4 = -2-1r
x5 = 1+r
x6 = r
Tekan enter untuk kembali ke menu utama

```

Keterangan

Mencoba diselesaikan dengan matriks balikan tetapi matriks tidak mempunyai balikan sehingga program mencoba menggunakan Gauss-Jordan.

4.2.4 Studi Kasus 1d Matriks Hilbert dengan n = 6

Input
$H = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & 1 & \frac{1}{4} & \cdots & \frac{1}{n+1} \\ \frac{1}{3} & \frac{1}{4} & 1 & \cdots & \frac{1}{n+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \cdots & \frac{1}{2n+1} \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$
<pre>test > 1d1.txt 1 1 0.5 0.333 0.25 0.2 0.167 1 2 0.5 0.333 0.25 0.2 0.167 0.142857 0 3 0.333 0.25 0.2 0.167 0.142857 0.125 0 4 0.25 0.2 0.167 0.142857 0.125 0.11111 0 5 0.2 0.167 0.142857 0.125 0.11111 0.1 0 6 0.167 0.142857 0.125 0.11111 0.1 0.90901 0 7 0.142857 0.125 0.11111 0.1 0.90901 0.08333333 0 8</pre>
Output

```

1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer
5. Kembali ke main menu

Masukkan pilihan sub menu : 1
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks augmented yang merepresentasikan SPL :
../test/1d1.txt
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 2
Solusi matriks adalah seperti di bawah ini. Perlu diingat bahwa variabel (contoh) ab bukanlah a * b melainkan variabel unik tersendiri
x1 = 3.4877-0.0973s-0.1321r
x2 = 21.2292+0.9977s+1.1964r
x3 = -101.7484-1.8695s-1.9077r
x4 = 83.1195+0.0839s+0.0084r
x5 = s
x6 = r
Tekan enter untuk kembali ke menu utama []

```

Keterangan

Terdapat dua variabel bebas, x_5 dan x_6 .

4.2.5 Studi Kasus 1d Matriks Hilbert dengan n = 10

Input

```

test > 1d2.txt
1 0.5 0.3333333333333333 0.25 0.2 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 1
2 0.5 0.3333333333333333 0.25 0.2 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0
3 0.3333333333333333 0.25 0.2 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0
4 0.3333333333333333 0.25 0.2 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0
5 0.1 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0 0.07692387692307693 0.07112857142857142 0.0625 0
6 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0 0.07692387692307693 0.07112857142857142 0.0625 0
7 0.1666666666666666 0.14285714285714285 0.125 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0 0.07692387692307693 0.07112857142857142 0.0625 0
8 0.125 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0 0.07692387692307693 0.07112857142857142 0.0625 0.058823529411764765 0
9 0.1111111111111111 0.1 0.0969999999999999 0.0969999999999999 0.0833333333333333 0 0.07692387692307693 0.07112857142857142 0.0625 0.058823529411764765 0
10 0.1 0.0969999999999999 0.0833333333333333 0 0.07692387692307693 0.07112857142857142 0.0625 0.058823529411764765 0.05555555555555555555 0.05263157894736842 0
11

```

Output

```

1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer
5. Kembali ke main menu

Masukkan pilihan sub menu : 1
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks augmented yang merepresentasikan SPL :
../test/1d2.txt
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 2
Solusi matriks adalah seperti di bawah ini. Perlu diingat bahwa variabel (contoh) ab bukanlah a * b melainkan variabel unik tersendiri
x1 = 99.9977
x2 = -4.949.7976
x3 = 79.195.6718
x4 = -600.560.5288
x5 = 2.522.331.2543
x6 = -6.305.780.0628
x7 = 9.668.745.4654
x8 = -8.750.772.9823
x9 = 4.375.365.2785
x10 = -923.684.2945
Tekan enter untuk kembali ke menu utama []

```

Keterangan

Dengan Gauss.

4.2.6 Studi Kasus 2a

Input
$\begin{bmatrix} 1 & -1 & 2 & -1 & -1 \\ 2 & 1 & -2 & -2 & -2 \\ -1 & 2 & -4 & 1 & 1 \\ 3 & 0 & 0 & -3 & -3 \end{bmatrix}$ <p> 1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Kaidah Cramer 5. Kembali ke main menu </p> <pre> Masukkan pilihan sub menu : 2 Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 2 Masukkan jumlah persamaan : 4 Masukkan jumlah variabel : 4 Masukkan SPL baris per baris dengan koefisien variabel dipisahkan spasi dan elemen terakhir sebagai b 1 -1 2 -1 -1 2 1 -2 -2 -2 -1 2 -4 1 1 3 0 0 -3 -3 </pre>
Output
<p>Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.</p> <p>Masukkan pilihan anda : 2</p> <p>Solusi matriks adalah seperti di bawah ini. Perlu diingat bahwa variabel (contoh) ab bukanlah a * b melainkan variabel unik tersendiri</p> <pre> x1 = -1+r x2 = 2s x3 = s x4 = r Tekan enter untuk kembali ke menu utama [] </pre>
Keterangan
Masukan dari <i>keyboard</i> , dengan Gauss-Jordan.

4.2.7 Studi Kasus 2b

Input																																										
$\begin{bmatrix} 2 & 0 & 8 & 0 & 8 \\ 0 & 1 & 0 & 4 & 6 \\ -4 & 0 & 6 & 0 & 6 \\ 0 & -2 & 0 & 3 & -1 \\ 2 & 0 & -4 & 0 & -4 \\ 0 & 1 & 0 & -2 & 0 \end{bmatrix}$ <pre> test > 2b.txt </pre> <table style="margin-left: 200px;"> <tr><td>1</td><td>2</td><td>0</td><td>8</td><td>0</td><td>8</td></tr> <tr><td>2</td><td>0</td><td>1</td><td>0</td><td>4</td><td>6</td></tr> <tr><td>3</td><td>-4</td><td>0</td><td>6</td><td>0</td><td>6</td></tr> <tr><td>4</td><td>0</td><td>-2</td><td>0</td><td>3</td><td>-1</td></tr> <tr><td>5</td><td>2</td><td>0</td><td>-4</td><td>0</td><td>-4</td></tr> <tr><td>6</td><td>0</td><td>1</td><td>0</td><td>-2</td><td>0</td></tr> <tr><td>7</td><td></td><td></td><td></td><td></td><td></td></tr> </table>	1	2	0	8	0	8	2	0	1	0	4	6	3	-4	0	6	0	6	4	0	-2	0	3	-1	5	2	0	-4	0	-4	6	0	1	0	-2	0	7					
1	2	0	8	0	8																																					
2	0	1	0	4	6																																					
3	-4	0	6	0	6																																					
4	0	-2	0	3	-1																																					
5	2	0	-4	0	-4																																					
6	0	1	0	-2	0																																					
7																																										

Output
<pre>test > 2bres.txt 1 x1 = 0 2 x2 = 2 3 x3 = 1 4 x4 = 1 5</pre>
Keterangan
Masukan dari file, keluaran ke file.

4.2.8 Studi Kasus 3a

Input
<pre>test > 3a.txt 1 8 1 3 2 0 2 2 9 -1 -2 1 3 1 3 2 -1 2 4 1 0 6 4 3</pre>
Output
<pre>1. Metode eliminasi Gauss 2. Metode eliminasi Gauss-Jordan 3. Metode matriks balikan 4. Kaidah Cramer 5. Kembali ke main menu Masukkan pilihan sub menu : 4 Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 1 Masukkan nama file berisi matriks augmented yang merepresentasikan SPL : ./test/3a.txt Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI. Masukkan pilihan anda : 2 x1 = -0.224324 x2 = 0.182432 x3 = 0.709459 x4 = -0.258108 Tekan enter untuk kembali ke menu utama ■</pre>

```

1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer
5. Kembali ke main menu

Masukkan pilihan sub menu : 3
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks augmented yang merepresentasikan SPL :
./test/3a.txt
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 2
x1 = -0.224324
x2 = 0.182432
x3 = 0.709459
x4 = -0.258108
Tekan enter untuk kembali ke menu utama []

```

Keterangan

Diselesaikan dengan balikan dan Cramer.

4.2.9 Studi Kasus 3b

Input

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

```

test > 3b.txt
1 0 0 0 0 0 1 1 1 13
2 0 0 0 1 1 1 0 0 0 15
3 1 1 1 0 0 0 0 0 0 8
4 0 0.75 0.04289 0 0.04289 0.75 0.04289 0.75 0.61396 14.79
5 0 0.25 0.91421 0.25 0.91421 0.25 0.91421 0.25 0 14.31
6 0.61396 0.75 0.04289 0.75 0.04289 0 0.04289 0 0 3.81
7 0 0 1 0 0 1 0 0 1 18.00
8 0 1 0 0 1 0 0 1 0 12.00
9 1 0 0 1 0 0 1 0 0 6.00
10 0.04289 0.75 0.61396 0 0.04289 0.75 0 0 0.04289 10.51
11 0.91421 0.25 0 0.25 0.91421 0.25 0 0.25 0.91421 16.13
12 0.04289 0 0 0.75 0.04289 0 0.61396 0.75 0.04289 7.04
13

```

Output

```

1. Metode eliminasi Gauss
2. Metode eliminasi Gauss-Jordan
3. Metode matriks balikan
4. Kaidah Cramer
5. Kembali ke main menu

Masukkan pilihan sub menu : 3
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks augmented yang merepresentasikan SPL :
./test/3b.txt
SPL tidak dapat diselesaikan dengan metode ini karena tidak mempunyai inverse ! Berikut solusinya dengan eliminasi Gauss-Jordan :
Matriks tidak punya solusi.
Tekan enter untuk kembali ke menu utama ■

```

Keterangan

Tidak ada solusi.

4.2 Determinan

4.2.1 Determinan dengan Kofaktor

Input

```

test > ➜ det1.txt
      1   6 4 4 5 9
      2   1 4 3 0 1
      3   2 3 1 1 4
      4   6 3 1 0 6
      5   3 7 2 1 7

```

Output

```

1. Metode ekspansi kofaktor
2. Metode segitiga atas
3. Keluar

Masukkan pilihan sub menu : 1
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks persegi yang ingin dicari determinannya :
./test/det1.txt
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 2
Determinan matriksnya adalah : -2.000000

Tekan enter untuk kembali ke menu utama ■

```

Keterangan

Dengan menggunakan kofaktor didapatkan determinan senilai -2.

4.2.2 Determinan dengan Matriks Segitiga Atas

Input
<pre>test > det2.txt 1 7.00786 2.00410 7.44904 0.31207 4.37409 0.11767 4.82695 5.28113 4.37863 4.09600 2 8.67422 4.60029 0.37151 0.88141 8.79151 2.05468 6.65723 3.42437 2.33242 6.85086 3 8.02497 8.65655 0.18030 6.14105 5.26827 3.54167 2.37399 4.70419 1.13786 0.14897 4 1.19984 5.81018 1.27704 6.57136 7.49057 0.79670 2.98123 8.19650 3.54704 5.59107 5 3.22447 3.54918 7.70066 6.68732 5.05549 5.36151 3.50212 1.64942 2.61616 4.07774 6 7.87993 7.81803 3.26927 3.07017 2.78297 0.23020 2.01491 1.95593 3.64098 2.17825 7 2.70669 4.18255 8.38155 6.01990 3.52401 1.91225 2.95685 1.16567 0.85257 4.75922 8 5.93561 5.44610 6.29657 4.02732 2.14718 6.55184 6.27186 7.62294 0.72638 2.08148 9 7.48717 5.33803 4.38929 4.64185 6.82677 3.34714 4.89424 7.60451 6.15153 4.17306 10 8.88529 7.79975 0.60454 2.40295 4.74086 5.65437 4.90990 6.63977 2.26460 7.18044 11</pre>
Output
<pre>1. Metode ekspansi kofaktor 2. Metode segitiga atas 3. Keluar Masukkan pilihan sub menu : 2 Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 1 Masukkan nama file berisi matriks persegi yang ingin dicari determinannya : ./test/det2.txt Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI. Masukkan pilihan anda : 2 Determinan matriksnya adalah : -15766935.528107 Tekan enter untuk kembali ke menu utama █</pre>
Keterangan
Matriks 10x10.

4.3 Balikan

4.3.1 Balikan dengan OBE

Input
<pre>test > inverse1.txt 1 5 0 8 3 2 3 2 9 7 9 3 4 4 3 7 3 1 2 1 5 4 9 7 2 3 4 9 5 5 6 0 4 0 0 6 7 5 6 1 1 4 7</pre>

Output

- ```
1. Operasi Baris Elementer
2. Matrix adjoin
3. Kembali ke main menu
```

```
Masukkan pilihan sub menu : 1
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks persegi yang ingin dicari inverse-nya :
./test/inversel.txt
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 2
-0.1491 0.2597 0.5101 -0.2659 -0.0918 -0.1872
-0.0515 -0.0734 -0.2787 0.1227 0.0949 0.1842
0.1018 -0.0744 -0.1839 0.0286 0.0091 0.1637
0.2636 -0.2146 -0.2195 0.1483 0.2224 -0.0423
-0.1438 0.4266 0.176 -0.032 -0.1594 -0.4668
0.1426 -0.3042 -0.2575 0.24 0.0127 0.2291
Tekan enter untuk kembali ke menu utama □
```

### 4.3.2 Balikan dengan Matriks Adjoin

### Input

- ```
1. Operasi Baris Elementer
2. Matrix adjoin
3. Kembali ke main menu
```

```
Masukkan pilihan sub menu : 2
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 2
Masukkan jumlah baris dan kolom : 8
Masukkan matriks baris per baris dengan tiap elemen dipisahkan spasi
39 17 48 26 21 26 48 50
22 47 45 25 10 3 44 20
21 42 42 41 27 33 34 25
43 5 50 7 24 25 14 22
6 11 29 3 42 50 8 41
46 43 41 18 34 29 17 29
15 6 45 44 25 25 44 11
27 26 34 40 25 18 19 49
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 1
Masukkan nama file yang ingin diisi dengan invers dari matriks yang telah dimasukkan :
./test/inverse2res.txt
```

Output

```
test > inverse2res.txt
1  0.0178 -0.0218 -0.0006 -0.0031 -0.0191 0.0272 0.0001 -0.0077
2  -0.0055 0.0098 0.0154 -0.005 0.0005 0.0045 -0.0142 -0.0038
3  -0.0248 0.0245 0.0117 0.0431 0.0068 -0.0398 -0.0104 0.0102
4  -0.0068 -0.0156 0.0198 0.0018 -0.0139 -0.0075 0.0051 0.0174
5  -0.0213 0.0172 -0.0879 -0.04 0.0218 0.0599 0.0546 0.0115
6  0.0198 -0.0309 0.0696 0.0152 -0.0052 -0.023 -0.0284 -0.0255
7  0.0253 0.0035 -0.0222 -0.0299 0.0023 0.0208 0.0189 -0.021
8  0.0097 0.0038 -0.0042 -0.0006 0.0068 -0.0138 -0.0142 0.0171
```

Keterangan

Masukan dari *keyboard* dan keluaran ke file.

4.4 Interpolasi Polinom

4.4.1 Studi Kasus 4a

Input

```
test > 4a.txt
1  0.4 0.043
2  0.7 0.005
3  0.11 0.058
4  0.14 0.072
5  0.17 0.1
6  0.2 0.13
7  0.23 0.147
```

Output

```

Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
Masukkan nama file berisi matriks dengan 2 kolom yang ingin dicari interpolasi polinomialnya :
./test/4a.txt
Hasil interpolasi polinomial:
p6(x) = -4212.4345x^6 + 7192.3992x^5 + -4346.3140x^4 + 1220.8549x^3 + -163.9157x^2 + 10.2764x + -0.1846
Masukkan point untuk diestimasi : 0.2
Nilai dari p6(0.2000) ≈ 0.1300

Apakah ingin melanjutkan estimasi (y/n)? y
Masukkan point untuk diestimasi : 0.55
Nilai dari p6(0.5500) ≈ 2.1376

Apakah ingin melanjutkan estimasi (y/n)? y
Masukkan point untuk diestimasi : 0.85
Nilai dari p6(0.8500) ≈ -66.2696

Apakah ingin melanjutkan estimasi (y/n)? y
Masukkan point untuk diestimasi : 1.28
Nilai dari p6(1.2800) ≈ -3485.1449

Apakah ingin melanjutkan estimasi (y/n)? ■

```

Keterangan

Titik yang diestimasi dimasukkan satu per satu

4.4.2 Interpolasi Data Covid

Input
<pre> test > ≡ 4b.txt 1 6.567 12624 2 7 21807 3 7.258 38391 4 7.451 54517 5 7.548 51952 6 7.839 28228 7 8.161 35764 8 8.484 20813 9 8.709 12408 10 9 10534 </pre>
Output

```

p9(x) = -140994.5598x^9 + 9372906.0628x^8 + -275476226.8079x^7 + 4695835438.3742x^6 + -51132198648.8689x^5 + 368553169428.9829x^4 + -1756821693899.8840x^3 + 5334238927153.8950x^2 + -9347057986137.9340x + 7187117988941.8590
Masukkan point untuk diestimasi : 7.52
Nilai dari p9(7.5200) ≈ 53374.4316

Apakah ingin melanjutkan estimasi (y/n)? y
Masukkan point untuk diestimasi : 8.32
Nilai dari p9(8.3200) ≈ 36438.3945

Apakah ingin melanjutkan estimasi (y/n)? y
Masukkan point untuk diestimasi : 9.17
Nilai dari p9(9.1700) ≈ -694222.5860

Apakah ingin melanjutkan estimasi (y/n)? y
Masukkan point untuk diestimasi : 7.55
Nilai dari p9(7.5500) ≈ 51830.9865

```

Keterangan

Hasil interpolasi polinomial memang memiliki nilai negatif di $x > 9$.

4.4.3 Interpolasi fungsi

Input
<pre> test > 4c1.txt 1 0 0 2 0.4 0.418884230141 3 0.8 0.50715796853 4 1.2 0.560924674815 5 1.6 0.583685661287 6 2 0.576651529752 7 </pre>
Output
<pre> Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 1 Masukkan nama file berisi matriks dengan 2 kolom yang ingin dicari interpolasi polinomialnya : ./test/4c1.txt Hasil interpolasi polinomial: p5(x) = 0.2363x^5 + -1.4213x^4 + 3.2371x^3 + -3.5527x^2 + 2.0353x + -0.0000 Masukkan point untuk diestimasi : 0.5 Nilai dari p5(0.5000) ≈ 0.4527 Apakah ingin melanjutkan estimasi (y/n)? y Masukkan point untuk diestimasi : 0.9 Nilai dari p5(0.9000) ≈ 0.5209 Apakah ingin melanjutkan estimasi (y/n)? ■ </pre>
Keterangan

4.5 Interpolasi Bicubic

Input / Output
<pre>Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 2 Masukkan matriks 4 kali 4 baris per baris dengan tiap elemen dipisahkan spasi 153 59 210 96 125 161 72 81 98 101 42 12 21 51 0 16 Masukkan nilai a dan b dalam satu baris dan dipisahkan spasi, dimana f(a,b) adalah nilai yang ingin dicari 0 0 Didapat f(0.0000,0.0000) = 161.0000 Tekan enter untuk kembali ke menu utama █</pre>
<pre>Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 2 Masukkan matriks 4 kali 4 baris per baris dengan tiap elemen dipisahkan spasi 153 59 210 96 125 161 72 81 98 101 42 12 21 51 0 16 Masukkan nilai a dan b dalam satu baris dan dipisahkan spasi, dimana f(a,b) adalah nilai yang ingin dicari 0.5 0.5 Didapat f(0.5000,0.5000) = 97.7266 Tekan enter untuk kembali ke menu utama █</pre>
<pre>Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 2 Masukkan matriks 4 kali 4 baris per baris dengan tiap elemen dipisahkan spasi 153 59 210 96 125 161 72 81 98 101 42 12 21 51 0 16 Masukkan nilai a dan b dalam satu baris dan dipisahkan spasi, dimana f(a,b) adalah nilai yang ingin dicari 0.25 0.75 Didapat f(0.2500,0.7500) = 105.5148 Tekan enter untuk kembali ke menu utama █</pre>
<pre>Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard. Masukkan pilihan anda : 2 Masukkan matriks 4 kali 4 baris per baris dengan tiap elemen dipisahkan spasi 153 59 210 96 125 161 72 81 98 101 42 12 21 51 0 16 Masukkan nilai a dan b dalam satu baris dan dipisahkan spasi, dimana f(a,b) adalah nilai yang ingin dicari 0.1 0.9 Didapat f(0.1000,0.9000) = 104.2291 Tekan enter untuk kembali ke menu utama █</pre>
Keterangan
Untuk fungsi program interpolasi bicubic, jika masukan dari file maka keluaran juga dari file, begitupun untuk masukan dari CLI.

4.6 Regresi Linier Berganda

Input

```
test > mlr.txt
```

```
1 72.4 76.3 29.18 0.9
2 41.6 70.3 29.35 0.91
3 34.3 77.1 29.24 0.96
4 35.1 68.0 29.27 0.89
5 10.7 79.0 29.78 1.00
6 12.9 67.4 29.39 1.1
7 8.3 66.8 29.69 1.15
8 20.1 76.9 29.48 1.03
9 72.2 77.7 29.09 0.77
10 24.0 67.7 29.60 1.07
11 23.2 76.8 29.38 1.07
12 47.4 86.6 29.35 0.94
13 31.5 76.9 29.63 1.1
14 10.6 86.3 29.56 1.1
15 11.2 86.0 29.48 1.1
16 73.3 76.3 29.4 0.91
17 75.4 77.9 29.28 0.87
18 96.6 78.7 29.29 0.78
19 107.4 86.8 29.03 0.82
20 54.9 70.9 29.37 0.95
21
```

```
test > mlr-var.txt
```

```
1 50 76 29.3
2
```

Output

```
Tekan 1 untuk menerima input dari file atau 2 untuk menerima input dari keyboard.
Masukkan pilihan anda : 1
File berisi data yang sudah ada dan data yang ingin diprediksi nilai y-nya adalah dipisah dan dibedakan.
Masukkan nama file berisi data riil yang sudah ada :
./test/mlr.txt
Masukkan nama file berisi data variabel x yang ingin diprediksi :
./test/mlr-var.txt
Tekan 1 untuk mengeluarkan hasil ke file atau 2 untuk mengeluarkan hasil ke CLI.
Masukkan pilihan anda : 2
Hasil dibulatkan ke sepuluh tempat di belakang koma.
y = -0.0000612834 -0.0031897397 * x1 + 0.0006704501 * x2 + 0.0359761722 * x3
f(50.0, 76.0, 29.3) = -0.0064531419
Tekan enter untuk kembali ke menu utama █
```

Keterangan

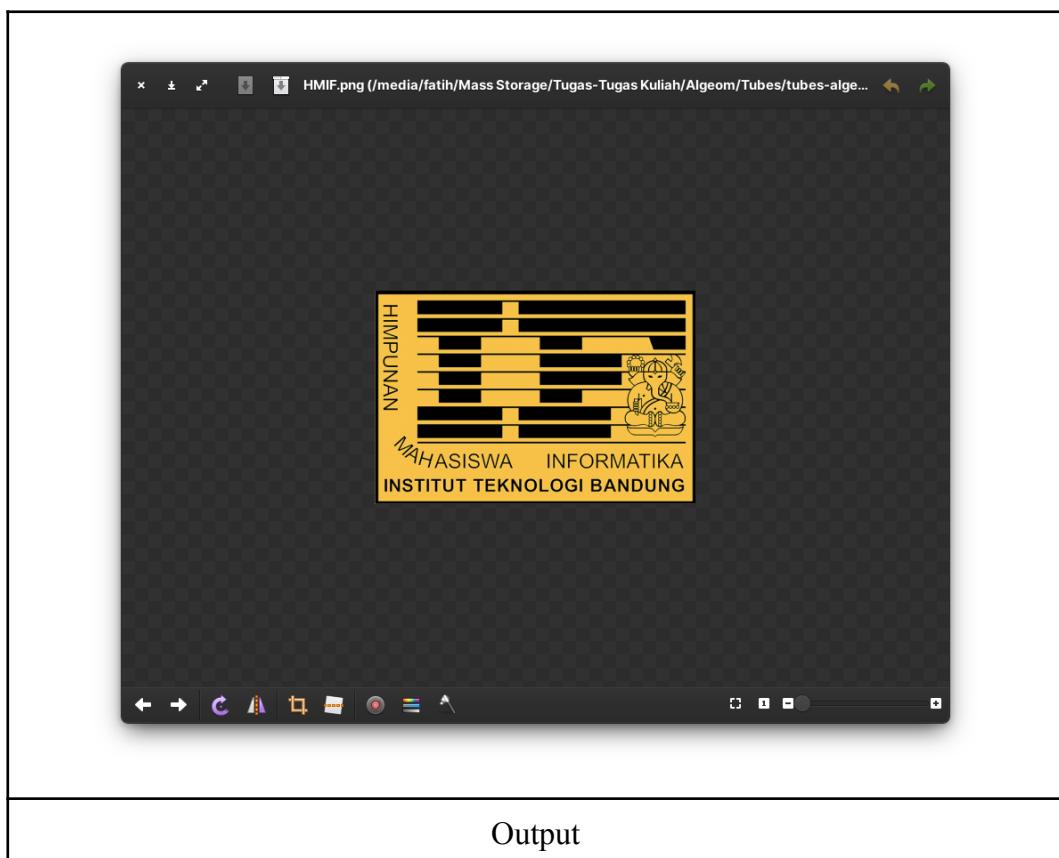
Sampel data terdapat pada file `mlr.txt` dengan kolom paling kanan adalah `y`

Titik data yang ingin diprediksi terdapat pada file `mlr-var.txt` dan (*obviously*) tanpa kolom `y`.
Fungsi yang didapatkan dari regresi linier berganda langsung dikeluarkan ke CLI
Hasil prediksi data dikeluarkan untuk tiap baris data yang ingin diprediksi yang dimasukkan

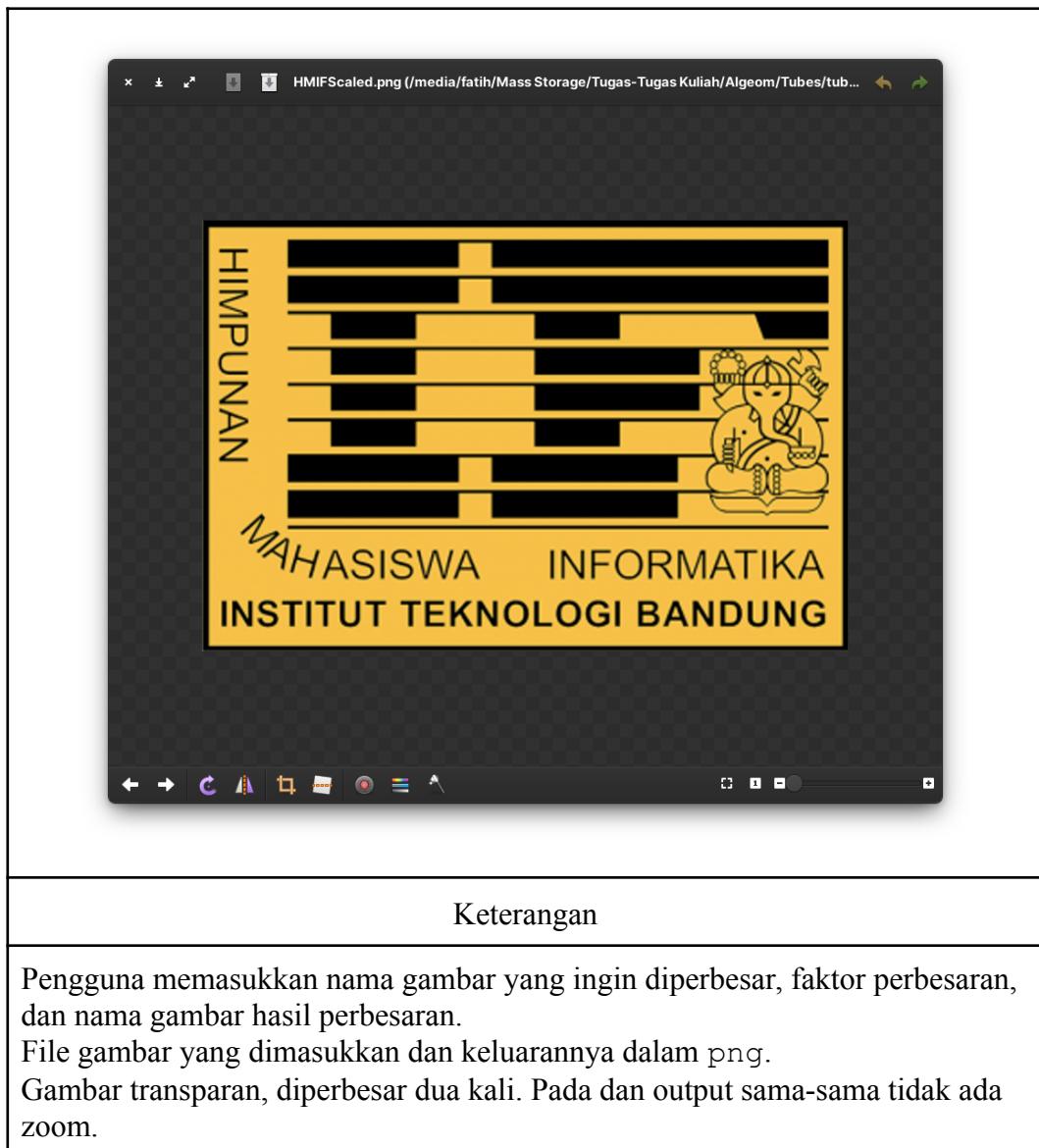
4.7 Perbesaran Gambar

4.7.1 Gambar Logo HMIF

Input
<pre>Masukkan nama file gambar yang ingin diperbesar: ./test/HMIF.png Masukkan kelipatan perbesaran gambar : 2 Memperbesar gambar... (akan memakan waktu beberapa saat) 25% done 50% done 75% done 100% done Masukkan nama file gambar yang ingin diexport : ./test/HMIFScaled.png Gambar berhasil diexport ke file ./test/HMIFScaled.png Tekan enter untuk kembali ke menu utama █</pre>



Output



Keterangan

Pengguna memasukkan nama gambar yang ingin diperbesar, faktor perbesaran, dan nama gambar hasil perbesaran.
File gambar yang dimasukkan dan keluarannya dalam png.
Gambar transparan, diperbesar dua kali. Pada dan output sama-sama tidak ada zoom.

4.7.2 Gambar Logo ITB

Input

```
Masukkan nama file gambar yang ingin diperbesar:  
./test/itb.png  
Masukkan kelipatan perbesaran gambar : 3  
Memperbesar gambar... (akan memakan waktu beberapa saat)  
25% done  
50% done  
75% done  
100% done  
Masukkan nama file gambar yang ingin diexport :  
./test/itbScaled.png  
Gambar berhasil diexport ke file ./test/itbScaled.png  
Tekan enter untuk kembali ke menu utama █
```



Output



Keterangan

Gambar transparan, diperbesar tiga kali. Pada input zoom 61% sedangkan pada output 20%.

4.7.3 Gambar Einstein

Input
<pre>Masukkan nama file gambar yang ingin diperbesar: ./test/einstein.png Masukkan kelipatan perbesaran gambar : 5 Memperbesar gambar... (akan memakan waktu beberapa saat) 25% done 50% done 75% done 100% done Masukkan nama file gambar yang ingin diexport : ./test/einsteinScaled.png Gambar berhasil diexport ke file ./test/einsteinScaled.png Tekan enter untuk kembali ke menu utama █</pre>
Output
Keterangan

Diperbesar lima kali. Pada input zoom 345% sedangkan pada output 82%.

BAB 5

KESIMPULAN, SARAN, DAN REFLEKSI

5.1 Kesimpulan

1. Bagi orang yang masih pemula akan Java, bahasa ini akan terasa *unnecessarily verbose*. Tetapi setelah menghabiskan waktu dengan bahasa ini, Java menjadi intuitif dan mudah dipakai karena sifatnya yang *strongly-typed* dan aspek lain dari bahasa ini seperti semua error yang mungkin harus selalu di-*catch* sehingga program menjadi lebih stabil dan tidak rawan kesalahan.
2. Hal-hal yang krusial dalam berbagai bidang ilmu seperti interpolasi dan regresi linier berganda sampai hal remeh yang kita temui di kehidupan sehari-hari seperti *upscaleing* gambar, dikerjakan dengan menggunakan matriks karena faktanya memang lebih mudah jika menggunakan matriks. Hal ini menunjukkan pentingnya materi dari mata kuliah Algeom.

5.2 Saran

1. Program yang ditulis, meskipun seratus persen fungsional, masih dapat dirapikan baik dari penyederhanaan logika dari fungsi-fungsi di dalamnya maupun penulisan dokumentasi serta *comments* yang lebih deskriptif.
2. Program ini hanya memiliki antarmuka dalam bentuk CLI. Mempertimbangkan *tedious*-nya menuliskan matriks (hal yang sering dilakukan dalam program ini) melalui CLI ataupun menuliskannya ke file txt, penambahan antarmuka grafis akan menghasilkan peningkatan yang berarti dalam aspek *user-friendliness*.

5.3 Refleksi

1. *Testing* pada saat pengembangan perangkat lunak sangatlah krusial untuk memastikan bahwa tidak ada bug, utamanya bug yang bersifat *subtle* sehingga sulit di-*debug*, di dalam program.
2. Tools dokumentasi seperti [Javadoc](#) yang digunakan di proyek ini dapat mempercepat pengembangan aplikasi, terutama pada lingkungan tim (pengembang bukan satu orang saja).
3. Semakin banyak fungsi yang ditulis dan abstraksi yang dibuat, maka semakin banyak pula dokumentasi yang perlu ditulis.
4. *Multi-threading* dapat mempercepat proses tertentu secara signifikan. Hal ini ditunjukkan pada perbedaan kecepatan fitur *scaling* gambar sebelum dan sesudah *multi-threading* diimplementasikan.

DAFTAR PUSTAKA

EXAMPLE: Three-Independent Variables Regression Example. (n.d). EconStats.

Diakses pada 24 September 2022, dari

https://econstats.com/RegressionOut_3_1.htm

Anton, H., Rorres, C. (2014). Elementary Linear Algebra: Applications Version.

Wiley. ISBN: 9781118434413 1118434412 9781118474228 1118474228

D. B. Rowe. (2018). *BiLinear, Bicubic, and In Between Spline Interpolation*.

Marquette University. Diakses pada 3 Oktober 2022, dari

https://www.mssc.mu.edu/~daniel/pubs/RoweTalkMSCS_BiCubic.pdf

Munir, Rinaldi. (2022). *IF2123 Aljabar Geometri - Semester I Tahun 2022/2023*.

Institut Teknologi Bandung. Diakses pada 3 Oktober 2022, dari

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2022-2023/algeo22-23.htm>

Zach. (2020). *Multiple Linear Regression by Hand (Step-by-Step)*. Statology.

Diakses pada 24 September 2022, dari

<https://www.statology.org/multiple-linear-regression-by-hand/>

LAMPIRAN

[Tautan repositori Github](#)