

# LAPORAN TUGAS BESAR 1

## IF2123 ALJABAR LINEAR GEOMETRI

Kelompok TBA (Tugas Besar Algeo)



Disusun oleh:

Nadia Mareta Putri Leiden    13520007

Hansel Valentino Tanoto    13520046

Adelline Kania Setiyawan    13520084

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA  
INSTITUT TEKNOLOGI BANDUNG

2021

## DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>2</b>
<b>BAB 1 DESKRIPSI MASALAH .....</b>	<b>3</b>
<b>BAB 2 TEORI SINGKAT .....</b>	<b>4</b>
<b>BAB 3 IMPLEMENTASI PROGRAM.....</b>	<b>10</b>
<b>BAB 4 EKSPERIMEN .....</b>	<b>39</b>
<b>BAB 5 KESIMPULAN DAN SARAN .....</b>	<b>65</b>
<b>DAFTAR REFERENSI .....</b>	<b>66</b>

## BAB 1 DESKRIPSI MASALAH

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

Di dalam Tugas Besar 1 ini, anda diminta membuat satu atau lebih *library* aljabar linier dalam Bahasa Java. *Library* tersebut berisi fungsi-fungsi seperti eliminasi Gauss, eliminasi Gauss-Jordan, menentukan balikan matriks, menghitung determinan, kaidah *Cramer* (kaidah *Cramer* khusus untuk SPL dengan  $n$  peubah dan  $n$  persamaan). Selanjutnya, gunakan *library* tersebut di dalam program Java untuk menyelesaikan berbagai persoalan yang dimodelkan dalam bentuk SPL, menyelesaikan persoalan interpolasi, dan persoalan regresi. Penjelasan tentang interpolasi dan regresi adalah seperti di bawah ini.

Beberapa tulisan cara membuat *library* di Java:

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

## BAB 2 TEORI SINGKAT

### I. Gauss dan Gauss-Jordan

Eliminasi Gauss merupakan sebuah metode untuk menyelesaikan persamaan melalui matriks. Eliminasi Gauss menggunakan prinsip matriks eselon, yang memiliki bentuk sebagai berikut:

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

Metode ini bertumpu pada satu prinsip, yakni bilangan bukan nol harus dimulai dengan angka 1, lalu boleh diikuti oleh angka lainnya dalam satu baris selain angka 1.

Sedangkan metode Eliminasi Gauss-Jordan adalah metode eliminasi yang merupakan kelanjutan dari Eliminasi Gauss. Apabila prinsip Eliminasi Gauss disebut dengan matriks eselon, Eliminasi Gauss-Jordan disebut dengan matriks eselon tereduksi yang memiliki bentuk sebagai berikut:

$$(A|b) = \left[ \begin{array}{ccc|c} 1 & 0 & 3 & 7 \\ 0 & 1 & 5 & 11 \\ 0 & 0 & 0 & 0 \end{array} \right]$$

Angka 1 harus diapit angka nol, sehingga nanti hasil persamaannya langsung berupa variabel persamaan yang ingin dicari oleh pengguna.

### II. Determinan

Determinan adalah nilai yang dapat dihitung dari elemen-elemen suatu matriks persegi yaitu matriks yang memiliki jumlah baris dan kolom yang sama (matriks  $n \times n$ ). Notasi untuk determinan dapat dituliskan sebagai  $\det(A)$  atau  $|A|$ . Misalnya, untuk matriks berukuran  $2 \times 2$ , determinannya dapat dihitung dengan cara:

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

Sedangkan untuk matriks  $3 \times 3$ , determinannya dapat dihitung dengan cara:

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

Terdapat sifat khusus saat menghitung nilai determinan dari matriks segitiga atas maupun matriks segitiga bawah. Matriks segitiga atas (*upper triangular*) adalah matriks yang semua elemen di bawah diagonal utamanya adalah nol, sedangkan matriks segitiga bawah (*lower triangular*) adalah matriks yang semua elemen di atas diagonal utamanya adalah nol. Determinan dari matriks segitiga dapat diperoleh hanya dengan mengalikan semua elemen pada diagonal utamanya. Jadi untuk suatu matriks segitiga  $A$  berukuran  $n \times n$ ,

$$\det(A) = a_{11}a_{22}a_{33} \dots a_{nn}$$

Salah satu metode untuk menghitung determinan adalah metode reduksi baris yaitu dengan menerapkan OBE (Operasi Baris Elementer) sampai didapatkan bentuk matriks segitiga, lalu nilai determinannya dapat dihitung menggunakan rumus determinan untuk matriks segitiga. Beberapa sifat determinan ketika diterapkan OBE adalah sebagai berikut:

1. Mengalikan suatu baris pada matriks  $A$  dengan konstanta  $k$ , maka determinannya menjadi  $k \cdot \det(A)$
2. Menukarkan dua buah baris pada matriks  $A$ , maka determinannya menjadi  $-\det(A)$
3. Menambahkan sebuah baris dengan kelipatan baris yang lain, maka determinannya tetap  $\det(A)$

Jadi, secara umum, menghitung determinan dengan metode reduksi baris dapat diperoleh dengan cara:

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

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

dengan  $p$  menyatakan banyaknya operasi pertukaran baris selama penerapan OBE.

Selain metode reduksi baris, ada metode lain untuk menghitung determinan yaitu metode ekspansi kofaktor. Untuk matriks  $A$  berukuran  $n \times n$ , dapat didefinisikan:

1.  $M_{ij}$  (Minor entri  $a_{ij}$ ) adalah determinan submatriks yang elemen-elemennya tidak berada pada baris  $i$  dan kolom  $j$ , dan
2.  $C_{ij}$  (Kofaktor entri  $a_{ij}$ ) =  $(-1)^{i+j}M_{ij}$

Jadi determinan matriks  $A$  berukuran  $n \times n$  dapat dihitung dengan metode ekspansi kofaktor menggunakan salah satu rumus berikut ini:

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

### III. Matriks Kofaktor dan Adjoin

Matriks kofaktor ( $C$ ) dari suatu matriks  $A$  adalah matriks persegi yang ukurannya sesuai dengan ukuran matriks  $A$  dan setiap elemennya ( $C_{ij}$ ) merupakan kofaktor dari entri  $a_{ij}$  pada matriks  $A$ . Elemen  $C_{ij}$  dapat dihitung menggunakan rumus seperti yang tertera pada subbab II Bab 2. Jadi matriks kofaktor dapat dituliskan sebagai berikut:

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

Adjoin dari matriks  $A$  adalah *transpose* dari matriks kofaktor untuk matriks  $A$ . Matriks adjoin berguna untuk mencari *invers* (matriks balikan) yang penjelasan lebih lanjutnya akan dijelaskan pada subbab berikutnya (subbab 3). Notasi dari adjoin suatu matriks  $A$  dapat dituliskan sebagai  $Adj(A)$ .

### IV. Matriks Balikan

Sebuah matriks bisa diinverskan apabila memiliki  $A^{-1}$  sehingga memiliki ukuran yang sama dengan  $A$ . Sehingga bisa dikatakan, apapun yang terjadi pada  $A$ , maka pada  $A^{-1}$  akan berlaku sebaliknya. Hasil perkalian dari keduanya adalah berupa matriks identitas ( $I$ ) sebagai berikut:

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

Ada 2 cara dalam mencari matriks balikan yaitu dengan matriks kofaktor dan dengan metode reduksi baris. Pada metode matriks kofaktor, matriks balikan dapat dihitung menggunakan rumus:

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

dimana adjoin didapatkan dengan melakukan *transpose* terhadap matriks kofaktor. Sedangkan untuk metode reduksi baris, matriks balikan diperoleh dengan cara membuat matriks *augmented* berukuran  $n \times 2n$  yang tersusun atas matriks  $A$  dan matriks identitas  $I$  lalu diterapkan OBE hingga bagian kiri matriks augmented (matriks  $A$ ) berbentuk matriks identitas dan matriks balikannya adalah matriks pada sisi kanan matriks *augmented*,

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

## V. Kaidah Cramer

Kaidah *Cramer* adalah rumus atau metode yang dapat digunakan untuk menyelesaikan system persamaan linear dengan jumlah variabel sama dengan jumlah persamannya sehingga kaidah *Cramer* hanya berlaku pada sistem persamaan linear yang memiliki solusi tunggal / unik. Untuk mendapatkan solusi dari suatu SPL dengan  $n$  buah persamaan dan  $n$  buah variabel, metode ini membutuhkan nilai determinan dari matriks koefisien dan determinan dari  $n$  buah matriks lain yang masing-masing diperoleh dengan mengganti salah satu kolom matriks koefisien dengan matriks konstanta (ruas kanan pada SPL). Jika  $Ax = b$  merupakan SPL yang terdiri dari  $n$  persamaan dengan  $n$  buah variabel, maka solusi dari SPL tersebut adalah:

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

dengan,

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

## VI. Interpolasi Polinom

Interpolasi polinom adalah salah satu metode untuk memprediksi nilai dari sebuah titik  $p_n(x)$  dengan memperkirakan nilai  $y$  tersebut pada sembarang titik pada selang  $[x_0, \dots, x_n]$ . Bentuk polinom interpolasi derajat  $n$  yang menginterpolasikan titik-titik  $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$  adalah  $p_n(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$ . Dengan menyulihkan semua nilai  $(x_i, y_i)$  untuk  $i = 1, 2, \dots, n$ , akan didapatkan  $n$  buah sistem persamaan linier

$$a_0 + a_1x_0 + a_2x_0^2 + \dots + a_nx_0^n = y_0$$

$$a_0 + a_1x_1 + a_2x_1^2 + \dots + a_nx_1^n = y_1$$

$$\dots \qquad \dots$$

$$a_0 + a_1x_n + a_2x_n^2 + \dots + a_nx_n^n = y_n$$

Solusi sistem persamaan linier ini, yaitu nilai  $a_0, a_1, \dots, a_n$ , diperoleh dengan menggunakan metode eliminasi Gauss. Sebagai contoh, misalkan diberikan tiga buah titik yaitu  $(8.0, 2.0794)$ ,  $(9.0, 2.1972)$ , dan  $(9.5, 2.2513)$ . Tentukan polinom interpolasi kuadrat lalu estimasi nilai fungsi pada  $x = 9.2$ . Polinom kuadrat berbentuk  $p_2(x) = a_0 + a_1x + a_2x^2$ . Dengan menyulihkan ketiga buah titik data ke dalam polinom tersebut, diperoleh sistem persamaan linier 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 Linier Berganda

Regresi Linear merupakan salah satu metode untuk memprediksi nilai selain menggunakan Interpolasi Polinom. Meskipun sudah ada rumus jadi untuk menghitung regresi linear sederhana, terdapat rumus 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  $\beta_i$  dapat digunakan *Normal Estimation Equation for Multiple Linear Regression* sebagai berikut:

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

Sistem persamaan linier tersebut diselesaikan dengan menggunakan metode eliminasi Gauss.

## BAB 3 IMPLEMENTASI PROGRAM

### I. Main Class

*Main Class* adalah *class* program utama dari tugas besar 1 ini. Pada *Main Class*, hanya terdapat satu prosedur yaitu *main*. Pada prosedur ini, pertama-tama pengguna akan diberikan informasi ringkas tentang apa yang bisa dilakukan program ini kemudian akan ditampilkan menu yang bisa pengguna pilih sesuai kebutuhan. Terdapat enam menu, yaitu Sistem Persamaan Linier, Determinan, Matriks Balikasn, Interpolasi Polinom, dan Regresi Linier Berganda. Setelah memilih menu, implementasi algoritma untuk masing-masing menu akan dijelaskan lebih lanjut pada *class* lainnya.

Setiap kali pengguna selesai melakukan perhitungan pada suatu menu tertentu, pengguna akan diberikan opsi untuk kembali melanjutkan perhitungan dengan program atau keluar. Apabila pengguna ingin melanjutkan perhitungan, pengguna akan kembali ke halaman menu dan bisa kembali memilih menu yang diinginkan. Sedangkan, apabila pengguna ingin keluar, pengguna akan langsung keluar dari program.

### II. Gauss\_gauss\_jordan Class

Fungsi yang digunakan sebagai fungsi utama operator adalah sebagai berikut secara terurut: *elimination\_before()*, *gauss()*, dan *gauss\_jordan()*. Prosedur utama yang digunakan pada kelas ini adalah *gauss\_jordan\_main()* yang akan dipanggil pada *Main Class*. Sedangkan, fungsi yang lainnya adalah fungsi yang dibuat untuk mendukung ketiga fungsi di atas dan digunakan untuk memenuhi kondisi-kondisi tertentu.

Seluruh fungsi di bawah ini disimpan di dalam satu *class* yaitu *gauss\_gauss\_jordan.class* dan *file gauss\_gauss\_jordan.java*.

a. `public static void swap(float m[][], int i, int j, int neffrow, int neffcols)`

---

**procedure** `swap(input/output m: array of array of real, input i: integer, input j: integer, input neffrow: integer, input neffcols: integer)`

{ I.S. terdapat matriks yang sedang dilakukan pengecekan baris  
F.S. terjadi penukaran baris }

**KAMUS LOKAL**

temp: real

k: integer

**ALGORITMA**

i traversal [0..neffcols-1]

temp  $\leftarrow$  m[j][k]

m[j][k]  $\leftarrow$  m[i][k]

m[i][k]  $\leftarrow$  temp

b. public static void check\_gauss(float m[][], int neffrow, int neffcols, int swap\_counter[])

---

**procedure** check\_gauss(input/output m: array of array of real, input neffrow: integer, input neffcols: integer, input/output swap\_counter: array of integer)

{ I.S. matriks belum rapi dan terstruktur untuk dilakukan eliminasi gauss

F.S. matriks sudah terurut dan siap untuk diubah ke bentuk matriks

eselon }

**KAMUS LOKAL**

h, i, j, n: integer

**ALGORITMA**

h traversal [0..neffrow-1]

i traversal [1..neffrow-1]

j traversal [0..neffcols-1]

if (m[i][j]  $\neq$  0 and m[i-1][j] = 0) then

n  $\leftarrow$  i - 1

swap(m, i, n , neffrow, neffcols)

swap\_counter[0]  $\leftarrow$  swap\_counter[0] + 1

break

c. public static void gauss(float m[][], int neffrow, int neffcols)

---

**procedure** gauss(input/output m: array of array of real, input neffrow: integer, input neffcols: integer, input/output swap\_counter: array of integer)

{ I.S. sudah dieliminasi, terurut, dan membentuk matriks segitiga bawah

F.S. akan terbentuk matriks eselon }

**KAMUS LOKAL**

i, j, k: integer

divider: real

**ALGORITMA**

```
i traversal [0..neffrow-1]
  j traversal [0..neffcols-1]
    if (m[i][j] ≠ 0) then
      divider ← m[i][j]
      k traversal [j..neffcols]
        m[i][k] ← m[i][k] `div` divider
      break
```

d. public static void gauss\_jordan(float m[][], int neffrow, int neffcols)

---

**procedure** gauss\_jordan (input/output m: array of array of real, input neffrow: integer, input neffcols: integer)

{ I.S. Sudah di lakukan eliminasi gauss

F.S. Terbentuk matriks eselon baris tereduksi }

**KAMUS LOKAL**

index\_row, i, j, k, z: integer  
arr: array of integer  
count: integer  
divider, idx: real

**ALGORITMA**

```
index_row ← -999
count ← 0
i traversal [0..neffcols-1]
  arr[i] ← -9999
i traversal [0..neffrow-2]
  j traversal [0..neffcols-1]
    index_row ← search_row(m, j, neffrow, i)
    if (index ≠ -999) then
      if (check_availability(arr, index_row, j)) then
        arr[j] ← index_row
        divider ← m[i][j] / m[index_row][j]
        k traversal [0..neffcols-1]
          idx ← m[index_row][k]
          m[i][k] ← m[i][k] - divider*idx
```

```

        else
            count  $\leftarrow$  index_row
            while (check_availability (arr, index_row, j) = false and
count < neffrow) do
                index_row  $\leftarrow$  search_row(m, j, neffrow, index_row)
                count  $\leftarrow$  count + 1
        z traversal [0..neffcols-1]
            arr[z]  $\leftarrow$  -9999

```

e. public static boolean check\_availability(int a[], int val, int j)

---

**function** check\_availability (a: array of integer, val: integer, j: integer)  $\rightarrow$  boolean

{ Mengecek apakah baris ini sudah digunakan untuk mengeliminasi atau belum oleh elemen yang lainnya, menghindari hasil yang negatif }

**KAMUS LOKAL**

flag: boolean

k: integer

**ALGORITMA**

```

flag  $\leftarrow$  true
k traversal [0..j-1]
    if (a[k] = val) then
        flag  $\leftarrow$  false
        break
 $\rightarrow$  flag

```

f. public static int search\_row(float m[][], int j, int neffrow, int i)

---

**function** search\_row(m: array of array of real, j: integer, neffrow: integer, i: integer)  $\rightarrow$  integer

{ Mencari row untuk mengeliminasi baris yang dituju }

**KAMUS LOKAL**

index: integer

k: integer

**ALGORITMA**

```

index  $\leftarrow$  -999

```

```

k traversal [i+1..neffrow-1]
  if (m[k][j] = 1) then
    index  $\leftarrow$  k
    break
   $\rightarrow$  index

```

g. public static void elimination\_before(float m[][], int neffrow, int neffcols, int swap\_counter[])

---

**procedure** elimination\_before (input/output m: array of array of real, input neffrow: integer, input neffcols: integer, input swap\_counter: array of integer)

{I.S. matriks sudah di swap terlebih dahulu

F.S. terbentuk matriks dengan karakteristik segitiga bawah}

#### KAMUS LOKAL

```

idx1, idx2: real
i, j, k: integer
count: integer
M: array [0...neffcols-1] of array [0...neffrows-1] of real

```

#### ALGORITMA

```

check_gauss(m, neffrow, neffcols, swap_counter)
if (neffrow > neffcols) then
  i traversal [1..neffrow-1]
    if (i  $\geq$  neffcols) then
      count  $\leftarrow$  neffcols
    else
      count  $\leftarrow$  i
  j traversal [0..i-1]
    if (m[j][j] = 0) then
      if (is_singular(m, j, i, neffcols)) then
        { do nothing }
      else
        check_gauss(m, neffrow, neffcols, swap_counter)
        i  $\leftarrow$  1
        break
    else

```

```

        idx1 ← m[i][j]
        idx2 ← m[j][j]
        k traversal [0..neffcols-1]
            m[i][k] ← m[i][k] - (idx1*m[j][k]) `div` 2
        m[i][j] ← 0
    else
        i traversal [1..neffrow-1]
            j traversal [0..i-1]
                if (m[j][j] = 0) then
                    if (is_singular(m, j, i, neffcols)) then
                        {do nothing}
                    else
                        check_gauss(m, neffrow, neffcols, swap_counter)
                        i ← 1
                        break
                else
                    idx1 ← m[i][j]
                    idx2 ← m[j][j]
                    k traversal [0..neffcols-1]
                        m[i][k] ← m[i][k] - (idx1*m[j][k]) `div` 2
                    m[i][j] ← 0

```

h. public static boolean is\_singular(float m[][], int j, int i, int neffcols)

---

**function** is\_singular(m: array of array of real, j: integer, i: integer, neffcols: integer) → boolean

{ini adalah fungsi yang digunakan di dalam proses eliminasi untuk melihat apakah kedua baris sama2 matriks yang seluruh elemennya nol, menentukan apakah perlu di swap atau tidak}

#### KAMUS LOKAL

flag: boolean

k: integer

#### ALGORITMA

```

flag ← true
k traversal [0..neffcols-1]
  if (m[i][k] ≠ m[j][k]) then
    if (m[i][k] = 0 && m[j][k] ≠ 0) then
      flag ← true
      break
    else if (m[i][k] ≠ 0 && m[j][k] = 0) then
      flag ← false
      break
→ flag

```

i. public static boolean swap\_singular(float m[][], int neffrow, int neffcols, int [] swap\_counter)

**procedure** swap\_confirm (input/output m: array of array of real, input neffrow: integer, input neffcols: integer, input swap\_counter: array [0] of integer)

{I.S. Sudah melewati fase eliminasi segitiga bawah untuk dianalisis apakah ada yang harus di eliminasi kembali

F.S. Fungsi berhasil dieliminasi}

#### KAMUS LOKAL

idx, idx2, i, k: integer

divider: float

#### ALGORITMA

```

i...traversal [0...neffrow-2]
  k...traversal [0...neffcols-1]
    if (m[i][k] != 0) then
      if (m[i+1][k] != 0) then
        idx2 <- k
        idx <- i

```



```

        divider <- m[idx+1][idx2] / m[idx][idx2]

        z...traversal [idx2...neffcols-1]

            m[idx+1][z] <- m[idx][z]*divider

        m[idx+1][idx2] <- 0

        i <- 0

        elimination_before(m_ neffrow, neffcols, swap_counter)

        break

    else

        break

```

### III. Determinan Class

*Determinan Class* adalah kelas yang berisi fungsi-fungsi dan prosedur untuk menghitung nilai determinan suatu matriks dengan 2 metode yaitu metode reduksi baris dan metode ekspansi kofaktor. Pada kelas ini, terdapat 7 *method* yang bersifat *public*. *Method-method* tersebut adalah:

#### a. public static void determinan()

*Method* ini berbentuk prosedur dan bersifat *public* sehingga bisa digunakan pada kelas-kelas lainnya, terutama pada *Main Class*. *Method* ini merupakan *method* utama dari kelas ini yang akan dipanggil di *Main Class* pada bagian menu determinan. *Method* ini akan membaca masukan matriks dari pengguna berupa *input keyboard* atau *file* dengan memanggil *method* public static float[][] readInput(boolean isSquare) dari *ReadDisplayArray Class* dengan parameter *input*-nya bernilai *true* karena nilai determinan hanya terdefinisi apabila matriksnya berbentuk persegi ( $n \times n$ ). Setelah itu akan dilakukan validasi juga untuk masukan dari *file* apakah berbentuk matriks persegi atau tidak. Kemudian, akan ditampilkan submenu berupa pilihan metode yang ingin digunakan untuk menghitung determinan yaitu metode reduksi baris dan metode ekspansi kofaktor. Apabila submenu yang dipilih adalah metode reduksi baris, maka akan dipanggil *method* public static float detKofaktor(float[][] matriks), sedangkan apabila submenu yang dipilih adalah metode ekspansi kofaktor, maka akan dipanggil *method* public static float detKofaktor(float[][] matriks). Pada akhir *method* ini, program akan memberi opsi jenis *output* apa yang diinginkan (*keyboard* atau *file*).

---

**procedure** determinan()

**KAMUS LOKAL**

matriks: array of array of real

resultString: string

row: integer

choiceMenu: integer

chooseInput: integer

**ALGORITMA**

matriks ← ReadDisplayArray.readInput(true)

resultString ← ""

row traversal [0..matriks.length-1]

resultString ← resultString + Arrays.toString(matriks[row]) + "\n"

if (matriks.length ≠ matriks[0].length) then

resultString ← resultString + "Matriks di atas tidak memiliki  
determinan karena bukan matriks persegi"

output(resultString)

else

output("nSUB-MENU DETERMINAN")

output("1. Metode Reduksi Baris\n2. Metode Ekspansi Kofaktor\n")

choiceMenu ← Utils.chooseOptionValidation(1, 2)

if (choiceMenu = 1) then

resultString ← resultString + "\nHasil determinan untuk matriks di  
atas dengan metode reduksi baris adalah " + Determinan.detReduksiBaris(matriks)

else {(choiceMenu = 2)}

resultString ← resultString + "\nHasil determinan dari matriks di  
atas dengan metode ekspansi kofaktor adalah " + Determinan.detKofaktor(matriks)

output("Jenis output apa yang ingin diberikan: ")

output("1. Keyboard\n2. File\n")

chooseInput ← Utils.chooseOptionValidation(1, 2)

if (chooseInput = 2) then

```

        Random randomNum ← new Random()

        FileWriter DetFile ← new FileWriter("../test/Determinan" +
randomNum.nextInt(100) + ".txt")

        DetFile.write(resultString)

        DetFile.close()

    else {(chooseInput = 2)}

        output(resultString)

```

b. `public static float[][] minor(float[][] matriks, int i, int j)`

*Method* ini berbentuk fungsi dan bersifat *public* sehingga dapat digunakan pada kelas-kelas yang lain. *Method* ini berfungsi untuk menghasilkan minor ( $M_{ij}$ ) dari suatu matriks yaitu dengan cara menghilangkan / menghapus baris  $i$  dan kolom  $j$  pada matriks yang di-*input* melalui iterasi (*traversal*) pada setiap elemen matriks. Parameter dari fungsi ini berupa matriks yang ingin dicari minornya serta *integer*  $i$  dan  $j$  yang berasosiasi dengan  $M_{ij}$ . *Method* ini akan mengembalikan nilai bertipe `float[][]`.

---

**function** `minor(matriks: array of array of real, i: integer, j: integer) → array of array of real`

#### KAMUS LOKAL

size: integer  
result: array of array of real  
a, b, c, d: integer

#### ALGORITMA

```

size ← matriks.length - 1
result ← array[0..size-1] of array[0..size-1] of real
c ← 0
a traversal [0..matriks.length-1]
    if (a = i) then      {Skip baris i}
        continue
    d ← 0
    b traversal [0..matriks.length-1]

```

```

        if (b = j) then           {Skip kolom j}
            continue
        result[c][d] ← matriks[a][b]
        d ← d + 1
    c ← c + 1
→ result

```

c. `public static float nilaiKofaktor(float[][] matriks, int i, int j)`

*Method* ini berbentuk fungsi juga dan bersifat *public* sehingga dapat diakses atau digunakan pada kelas-kelas lainnya. Parameter fungsi ini berupa matriks yang ingin dicari nilai kofaktornya, serta *integer*  $i$  dan  $j$  yang berasosiasi dengan  $M_{ij}$ . Fungsi ini berfungsi untuk menghitung nilai kofaktor  $M_{ij}$  dengan memanggil *method* `public static float detKofaktor(float[][] matriks)` yang parameter matriksnya diisi dengan memanggil *method* `public static float[][] minor(float[][] matriks, int i, int j)`. *Method* ini akan mengembalikan sebuah nilai bertipe *float*.

---

**function** nilaiKofaktor(matriks: array of array of real, i: integer, j: integer) → real

KAMUS LOKAL

ALGORITMA

→ `detKofaktor(minor(matriks, i, j)) * Math.pow(-1, i + j)`

d. `public static float[][] matriksKofaktor(float[][] matriks)`

*Method* ini juga berbentuk fungsi dan bersifat *public* sehingga dapat digunakan pada kelas-kelas lainnya. *Method* ini berfungsi untuk menghasilkan matriks kofaktor yaitu matriks yang elemennya adalah kofaktor-kofaktor dari entri matriks yang di-input. *Method* ini akan memanggil *method* `public static float nilaiKofaktor(float[][] matriks, int i, int j)` untuk mengisi elemen-elemen dari matriks kofaktor. *Method* ini akan mengembalikan nilai bertipe `float[][]`.

---

**function** matriksKofaktor(matriks: array of array of real) → array of array of real

#### KAMUS LOKAL

size: integer

result: array of array of real

#### ALGORITMA

size  $\leftarrow$  matriks.length

result  $\leftarrow$  array[0..size-1] of array[0..size-1] of real

a traversal [0..size-1]

    b traversal [0..size-1]

        result[a][b]  $\leftarrow$  nilaiKofaktor(matriks, a, b)

$\rightarrow$  result

e. public static float[][] adjoin(float[][] matriks)

*Method* ini juga berbentuk fungsi dan bersifat *public* sehingga dapat digunakan pada kelas-kelas lainnya, terutama *Invers Class*. *Method* ini berfungsi untuk menghasilkan matriks adjoin yaitu matriks *transpose* dari matriks kofaktor. *Method* ini akan memanggil *method* public static float[][] matriksKofaktor(float[][] matriks) yang digunakan sebagai parameter *input* fungsi ini. *Method* ini akan mengembalikan nilai bertipe float[][].

---

**function** adjoin(matriks: array of array of real)  $\rightarrow$  array of array of real

#### KAMUS LOKAL

size: integer

result: array of array of real

kofaktor: array of array of real

#### ALGORITMA

size  $\leftarrow$  matriks.length

result  $\leftarrow$  array[0..size-1] of array[0..size-1] of real

kofaktor  $\leftarrow$  matriksKofaktor(matriks)

a traversal [0..size-1]

    b traversal [0..size-1]

```

        result[a][b] ← kofaktor[b][a]
    → result

```

f. `public static float detKofaktor(float[][] matriks)`

*Method* ini juga berbentuk fungsi dan bersifat *public* sehingga dapat digunakan pada kelas-kelas lainnya, terutama *Invers Class* dan *Cramer Class*. *Method* ini digunakan untuk menghitung determinan matriks dengan metode ekspansi kofaktor. Fungsi ini bekerja secara rekursif dengan terus menerus menghitung determinan dari minornya hingga matriks tidak memiliki minor lagi. Jadi dalam penerapannya, fungsi ini akan memanggil *method* `public static float[][] minor(float[][] matriks, int i, int j)`. Fungsi ini akan mengembalikan sebuah nilai bertipe data *float*.

---

**function** `detKofaktor(matriks: array of array of real) → real`

#### KAMUS LOKAL

```

size: integer
result: integer
j: integer

```

#### ALGORITMA

```

size ← matriks[0].length
result ← 0
if (size = 1) then
    → matriks[0][0]
else if (size = 2) then
    → (matriks[0][0] * matriks[1][1]) - (matriks[1][0] * matriks[0][1])
else
    j ← 0
    i traversal [0..size-1]
        result ← result + matriks[i][j] * detKofaktor(minor(matriks, i,
j)) * Math.pow(-1, i + j)
    → result

```

g. `public static float detReduksiBaris(float[][] matriks)`

*Method* ini juga berbentuk fungsi dan bersifat *public* sehingga dapat digunakan pada kelas-kelas lainnya, terutama *Invers Class* dan *Cramer Class*. *Method* ini digunakan untuk menghitung determinan matriks dengan metode reduksi baris. Jadi dalam penerapannya, fungsi ini akan memanggil *method* `public static void elimination_before(float m[][], int neffrow, int neffcols, int swap_counter[])` untuk melakukan OBE. Fungsi ini akan mengembalikan sebuah nilai bertipe *float*.

---

**function** `detReduksiBaris(matriks: array of array of real) → real`

#### KAMUS LOKAL

size: integer

swap: array of integer

det: real

#### ALGORITMA

swap ← array[1] of integer

size ← `matriks.length`

if (size = 1) then

→ `matriks[0][0]`

else

det ← 1

`gauss_gauss_jordan.elimination_before(matriks, size, size, swap)`

i ← 0

i traversal [0..size-1]

det ← `det * matriks[i][i]`

→ `det * Math.pow(-1, swap[0])`

## IV. *Invers Class*

Penjelasan:

1. Terdapat 1 prosedur utama yaitu `InversMain()` yang akan digunakan pada *Main Class*. Fungsi utama pada kelas ini adalah `invers_mat_reduc()`, `invers_mat_kofaktor()`,

dan SPLInvers() yang parameternya berupa array dua dimensi, fungsi yang lainnya adalah fungsi pendukung yakni multiply\_matrix()

2. Disimpan dalam *class* sendiri *invers.class* dan *file* *invers.java*

Beberapa fungsi yang terdapat pada kelas ini adalah sebagai berikut:

a. `public static float[][] invers_mat_kofaktor(float a[][])`

---

**function** `invers_mat_kofaktor` (a: array of array of real) → array of array of real

{ Fungsi utama untuk melakukan invers }

**KAMUS LOKAL**

determinan: real

size: integer

adjoint: array of array of real

array: array [0..size-1] of array [0..size-1] of real

**ALGORITMA**

size ← a.length

determinan ← Determinan.detKofaktor(a)

array ← multiply\_matrix(adjoint, 1/determinan)

→ array

b. `public static float[][] multiply_matrix_kofaktor(float a[][], float var)`

---

**function** `multiply_matrix_kofaktor` (a: array of array of real, var: real) → array of array of real

{ Mengalikan matrix dengan suatu konstanta }

**KAMUS LOKAL**

size: integer

array: array [0..size-1] of array [0..size-1] of real

i, j: integer

**ALGORITMA**

size ← a.length

i traversal [0..size-1]

    j traversal [0..size-1]

        array[i][j] ← a[i][j] \* var

→ array



c. `public static float[][] multiply_matrix_reduc(float a[][], float var)`

**function** multiply\_matrix\_reduc (a: array of array of real, var: real) → array of array of real

{ Mengalikan matrix dengan suatu konstanta untuk reduksi }

#### KAMUS LOKAL

size: integer

array: array [0..size-1] of array [0..size-1] of real

i, j: integer

#### ALGORITMA

size ← a.length

i traversal [0..size-1]

    j traversal [0..size-1]

        array[i][j] ← a[i][j] \* var

→ array

d. `public static float[][] multiply_matrix_reduc(float a[][], float var)`

**function** invers\_mat\_reduc (a: array of array of real) → array of array of real

{ Fungsi utama untuk melakukan invers menggunakan reduksi }

#### KAMUS LOKAL

determinan: real

size: integer

adjoint: array of array of real

array: array [0..size-1] of array [0..size-1] of real

#### ALGORITMA

size ← a.length

determinan ← Determinan.detKofaktor(a)

array ← multiply\_matrix(adjoint, 1/determinan)

→ array

## V. *Cramer Class*

*Cramer Class* adalah kelas yang berisi fungsi untuk menyelesaikan SPL (Sistem Persamaan Linier) dengan  $n$  persamaan dan  $n$  variabel menggunakan Kaidah *Cramer*. Pada kelas ini,

terdapat 3 *method* yang terdiri dari 2 *method public* dan 1 *method private*. Kedua *method* tersebut adalah:

a. `public static String CramerMain(float[][] matriks, int rows, int cols, String resultString)`

*Method* ini berbentuk fungsi dan bersifat *public* sehingga dapat dipanggil oleh kelas lain, terutama *Main Class*. *Method* ini merupakan *method* utama pada kelas ini yang akan dipanggil pada *Main Class*. Nilai (*value*) yang dikembalikan fungsi ini bertipe *string*.

b. `private static float[][] insertConst(float[][] matriks, int j, float[] constant)`

*Method* ini berbentuk fungsi dan bersifat *private* sehingga tidak akan bisa diakses atau dipanggil oleh kelas lain. *Method* ini berfungsi untuk membentuk matriks yang salah satu kolomnya sudah diganti dengan matriks konstanta ( $n \times 1$ ) yaitu matriks yang elemennya adalah konstanta / bagian ruas kanan pada SPL. Nilai (*value*) yang dikembalikan fungsi ini bertipe `float[][]`.

---

**function** `insertConst(matriks: array of array of real, j: integer, constant: array of real) → array of array of real`

#### KAMUS LOKAL

rows, cols: integer

result: array of array of real

#### ALGORITMA

rows ← matriks.length

cols ← matriks[0].length

result ← array[0..rows-1] of array[0..cols-1] of real

l traversal [0..rows-1]

    k traversal [0..cols-1]

        result[l][k] ← matriks[l][k]

i traversal [0..rows-1]

    result[i][j] ← constant[i]

→ result

c. `public static float[] cramerSol(float[][] mAugmented)`

*Method* ini berbentuk fungsi dan bersifat *public* sehingga dapat diakses atau dipanggil pada kelas-kelas yang lain karena *method* ini merupakan *method* utama dari kelas ini. *Method* ini berfungsi untuk menghitung solusi dari suatu SPL. *Method* ini akan memanggil *method* `public static float detKofaktor(float[][] matriks)` dan *method* `private static float[][] insertConst(float[][] matriks, int j, float[] constant)` untuk menghitung nilai determinan sesuai dengan rumus yang sudah dipaparkan pada subbab V Bab 2. Nilai yang dikembalikan fungsi ini memiliki tipe data `float[]` yang menampung solusi dari semua variabel pada SPL.

---

**function** `cramerSol(mAugmented: array of array of real) → array of real`

**KAMUS LOKAL**

rows, cols: integer  
mKcoef: array of array of real  
temp: array of array of real  
detKcoef: real  
constant: array of real  
solution: array of real

**ALGORITMA**

```
rows ← mAugmented.length  
cols ← mAugmented[0].length  
mKcoef ← array[0..rows-1] of array[0..cols-2] of real  
i traversal [0..rows-1]  
    j traversal [0..cols-2]  
        mKcoef[i][j] ← mAugmented[i][j]  
detKcoef ← Determinan.detKofaktor(mKcoef)  
constant ← array[0..rows-1] of real  
i traversal [0..rows-1]  
    constant[i] ← mAugmented[i][cols-1]  
solution ← array[0..rows-1] of real  
j traversal [0..cols-1]
```

```

        temp ← mKoeff
        solution[j] ← (Determinan.detKofaktor(insertConst(temp, j, constant)))
/ detKoeff
    → solution

```

## VI. ReadDisplayArray Class

Fungsi utama dari *ReadDisplayArray Class* adalah untuk menangani pembacaan *input* dari pengguna berupa *input keyboard* atau *file*, baik dalam bentuk matriks maupun titik untuk polinom dan regresi, serta menangani *output* yang akan diberikan, *output keyboard* atau *file*, sesuai dengan keinginan pengguna. Terdapat 12 *method* pada *class* ini, namun hanya 6 *method* yang bersifat *public*. *Method-method public* tersebut adalah:

### a. public static float[][] readInput(boolean isSquare)

*Method* berbentuk fungsi ini bersifat *public* dan dapat digunakan di kelas-kelas lainnya untuk membaca input dari pengguna, terkhusus untuk pembacaan matriks. *Method* ini memiliki parameter bertipe *boolean*, yaitu *isSquare*. Parameter ini untuk menentukan apakah input matriks yang nanti akan dibaca merupakan matriks persegi atau bukan.

*Method* ini, membaca *input* berupa *input keyboard* dan *file*. Apabila inputnya *file*, maka akan dipanggil *method* khusus yang membaca *file* dan mengembalikan sebuah matriks, yaitu *method* `private static float[][] readFiletoMatrix()`. Apabila *input* (berupa *input keyboard*) matriks berupa matriks persegi, maka *method* yang akan dipanggil oleh *method* ini adalah *method* `private static float[][] readSquareMatrix()` pada poin g, sedangkan, jika matriks bukan persegi, *method* ini akan memanggil *method* `private static float[][] readMatrix()`. *Method* ini, pada akhirnya, akan mengembalikan sebuah matriks bertipe `float[][]`.

---

**function** readInput(isSquare: boolean) → array of array of real

#### KAMUS LOKAL

chooseInput: integer

matrix: array of array of real

#### ALGORITMA

output("Jenis input apa yang ingin diberikan:")

output("1. Keyboard\n2. File\n")

chooseInput ← Utils.chooseOptionValidation(1,2)

```

if chooseInput = 2 then
    matrix ← readFiletoMatrix()
else
    if isSquare then
        matrix ← readSquareMatrix()
    else
        matrix ← readMatrix()

→ matrix

```

b. `public static float[][] readInputPoint()`

*Method* ini juga berbentuk fungsi dan bersifat *public* sehingga dapat digunakan pada kelas-kelas lainnya untuk membaca *input* dari pengguna. *Method* fungsi ini dikhususkan untuk membaca *n* buah poin/titik. *Method* ini tidak memiliki parameter apapun.

*Method* ini dapat membaca *input* berupa *input keyboard* dan *file*. Apabila inputnya *file*, maka akan dipanggil *method* khusus yang membaca *file* dan mengembalikan sebuah matriks yang berisi poin/titik, yaitu *method* `private static float[][] readFiletoMatrix()`. Apabila *input* berupa *keyboard*, akan dipanggil *method* `private static float[][] readPointMatrix()`. *Method* ini, pada akhirnya, akan mengembalikan sebuah matriks bertipe `float[][]`.

---

**function** `readInputPoint()` → array of array of real

**KAMUS LOKAL**

chooseInput: integer  
matrix: array of array of real

**ALGORITMA**

```

output("Jenis input apa yang ingin diberikan:")
output("1. Keyboard\n2. File\n")
chooseInput ← Utils.chooseOptionValidation(1,2)
if chooseInput = 2 then
    matrix ← readFiletoMatrix()
else
    matrix ← readPointMatrix()

→ matrix

```

c. `private static float[][] readRegressionPoints()`

*Method* fungsi ini dikhususkan untuk membaca  $n$  buah poin/titik sebanyak  $k$  data. *Method* ini tidak memiliki parameter apapun. *Method* ini hanya dapat membaca *input* dari *keyboard*. *Method* ini, pada akhirnya, akan mengembalikan sebuah matriks bertipe `float[][]`.

---

**function** `readRegressionPoints()` → array of array of real

**KAMUS LOKAL**

$n, k, i, j$ : integer

matrix: array of array of real

**ALGORITMA**

output("Masukkan jumlah titik x (n)\n n = ")

input(n)

output("Masukkan jumlah data (k)\n k = ")

input(k)

output("Masukkan input dengan format x1 x2 .. xn y sebanyak k data ")

i traversal [0..k+1]

    j traversal [0..n+2]

output("")

input(matrix[i][j])

→ matrix

d. `public static float[][] readRointMatrix()`

*Method* fungsi ini dikhususkan untuk membaca  $n$  buah poin/titik. *Method* ini tidak memiliki parameter apapun. *Method* ini hanya dapat membaca *input* dari *keyboard*. *Method* ini, pada akhirnya, akan mengembalikan sebuah matriks bertipe `float[][]`.

---

**function** `readPointMatrix()` → array of array of real

**KAMUS LOKAL**

$n, k, i, j$ : integer

matrix: array of array of real

**ALGORITMA**

output("Masukkan jumlah titik x (n)\n n = ")

input(n)

output("Masukkan input dengan format x y")

i traversal [0..n+1]

    j traversal [0..2]

```

        output("")
        input(matrix[i][j])
    → matrix

```

e. `public static void displayOutput(float[][] matrix)`

*Method displayOutput berfungsi untuk menampilkan output kepada pengguna, baik melalui text pada layar command maupun file. Apabila output yang ditampilkan berupa matriks, maka akan dipanggil method private static void displayMatrix(float[][] matrix), sedangkan, apabila ingin menyimpannya dalam sebuah file, akan dipanggil method private static void createFileFromMatrix(float[][] matrix). Method yang berfungsi untuk membaca nilai-nilai float pada sebuah file txt kemudian mengubahnya menjadi format matriks float.*

---

**procedure** displayOutput(input matrix: array of array of real)

**KAMUS LOKAL**

chooseOutput: integer

**ALGORITMA**

```

output("Jenis input apa yang ingin diberikan:")
output("1. Keyboard\n2.File\n")
chooseOutput ← Utils.chooseOptionValidation(1,2)
if chooseOutput = 1 then
    displayMatrix(matrix)
else
    createFileFromMatrix(matrix)
    output("File matriks sudah berhasil tersimpan")

```

f. `public static void displayOutputSPL(float[][] matrix)`

*Method displayOutputSPL berfungsi untuk menampilkan output berupa persamaan SPL beserta taksiran nilainya kepada pengguna, baik melalui text pada layar command maupun file.*

---

**procedure** displayOutputSPL(input resultString: string)

**KAMUS LOKAL**

chooseOutput: integer

SPLFile: File

#### ALGORITMA

```
output("Jenis input apa yang ingin diberikan:")  
output("1. Keyboard\n2.File\n")  
chooseOutput ← Utils.chooseOptionValidation(1,2)  
if chooseOutput = 1 then  
    output(resultString)  
else  
    SPLFile.write(resultString)  
    SPLFile.close()
```

## VII. Regresi Linier Class

Regresi Linier *Class* berfungsi untuk menyelesaikan permasalahan interpolasi polinom dan regresi linier. Terdapat dua *method* public pada *class* ini, yaitu:

### a. public static void interpolasiPolinom()

*Method* ini berfungsi untuk menentukan nilai taksiran  $x_k$  dengan menggunakan interpolasi polinom. Langkah-langkah algortimanya adalah sebagai berikut:

1. Membaca input berupa  $n$  poin/titik dari *input keyboard* ataupun *file* dan menyimpan *input* dalam matriks
2. Matriks akan diubah dari bentuk poin/titik menjadi bentuk persamaan SPL sesuai dengan aturan polinom kemudian menyimpan matriks baru tersebut pada variabel *matriks\_final*
3. Menyelesaikan persamaan SPL interpolasi polinom dengan metode Gauss Jordan yang terdapat pada *Gauss\_gauss\_jordan Class*
4. Menaksir nilai  $x_k$  dengan mensubstitusikan pada persamaan interpolasi polinom yang terbentuk
5. Menampilkan persamaan interpolasi polinom dan taksiran nilai  $x_k$ , baik lewat *output* pada layar atau dalam bentuk *file*.

---

procedure interpolasiPolinom()

KAMUS LOKAL



```

matriks, matrix_final, tempMatrix: array of array of real
swap_counter: array[1] of integer
x, result: real
rows, cols, i, j, p, k, m, n: integer

```

#### ALGORITMA

```

output("Jenis input apa yang ingin diberikan: \n1. Keyboard\n2.File\n")
chooseInput ← Utils.chooseOptionValidation(1,2)
if chooseInput = 1 then
    matriks ← ReadDisplayArray.readPointMatrix()
    output("Nilai x yang akan ditaksir = ")
    input(x)
else
    tempMatrix ← ReadDisplayArray.readFiletoMatrix()
    m ← 0
    i traversal [0..tempMatrix.length-1]
        n ← 0
        j traversal [0..tempMatrix[0].length]
            matriks[i][j] <- tempMatrix[m][n]
            n ← n + 1
        m ← m + 1
    x ← tempMatrix[tempMatrix.length - 1][0]

rows ← matriks.length
cols ← rows + 1
k ← 0
i traversal [0..rows]
    p ← 0

```

```

j traversal [0..cols]
    if j = rows then
        matrix_final[i][j] ← matriks[k][1]
    else
        matrix_final[i][j] ← Utils.power(matriks[k][0], p)
    p ← p + 1
k ← k + 1
swap_counter ← [1]
rows ← matrix_final.length
cols ← matriks_final[0].length

gauss_gauss_jordan.elimination_before(matriks_final, rows, cols, swap_counter)

gauss_gauss_jordan.gauss(matriks_final, rows, cols)
gauss_gauss_jordan.gauss_jordan(matriks_final, rows, cols)

SPL ← SPLInterpolasiRegresi(matrix_final, true)
result ← 0
i traversal [0..rows]
    result ← result + (matrix_final[i][cols-1] * Utils.power(x, i))

resultString ← SPL + "\nHasil taksiran nilai untuk x = " + x + " adalah "
+ result

ReadDisplayArray.displayOutputSPL(resultString)

```

b. `public static void regresiliniier()`

*Method* ini berfungsi untuk menentukan nilai taksiran  $x_{k1}, x_{k2}, \dots, x_{ki}$  dengan menggunakan metode regresi linier. Langkah-langkah algortimanya adalah sebagai berikut:

1. Membaca *input* berbentuk  $n$  titik  $x$  dan nilai  $y$  sebanyak  $k$  data baik dalam bentuk input *keyboard* maupun *file* dan menyimpan *input* dalam matriks
2. Matriks akan diubah dari bentuk  $n$  titik  $x$  dan nilai  $y$  sebanyak  $k$  data menjadi bentuk persamaan SPL sesuai dengan aturan regresi linier kemudian menyimpan matriks baru tersebut pada variabel *matriks\_final*
3. Menyelesaikan persamaan SPL regresi linier dengan metode Gauss Jordan yang terdapat pada *Gauss\_gauss\_jordan Class*
4. Menaksir nilai  $x_{k1}, x_{k2}, \dots, x_{ki}$  dengan mensubstitusikan pada persamaan regresi linier yang terbentuk
5. Menampilkan persamaan regresi linier dan taksiran nilai  $x_{k1}, x_{k2}, \dots, x_{ki}$ , baik lewat *output* pada layar atau dalam bentuk *file*.

---

procedure regresiLinier()

#### KAMUS LOKAL

main\_matriks, main\_matrix, matrix2, matrix\_final: array of array of real  
 $n, k, i, j, \text{row\_matrix2}, \text{col\_matrix2}, b$ : integer  
 $xs, \text{array\_sum\_xy}, \text{array\_sum}$ : array of real  
 swap: array of integer  
 SPL: string  
 result: real

#### ALGORITMA

output("Jenis input apa yang ingin diberikan: \n1. Keyboard\n2.File\n")  
 chooseInput  $\leftarrow$  Utils.chooseOptionValidation(1,2)  
if chooseInput = 1 then  
     main\_matriks  $\leftarrow$  ReadDisplayArray.readRegressionPoints()  
      $n \leftarrow \text{main\_matriks}[0].\text{length} - 1$   
      $k \leftarrow \text{main\_matrix}.\text{length}$   
else  
     main\_matriks  $\leftarrow$  ReadDisplayArray.readFiletoMatrix()

```

n ← main_matriks[0].length - 1
k ← main_matrix.length - 1

i traversal [0..k+1]
  j traversal [0..n+2]
    main_matrix[i][j] ← main_matriks[i][j]

if chooseInput = 1 then
  output("Nilai x yang akan ditaksir (masukkan sebanyak k data)")
  input(xs)
else
  i traversal [0..n+1]
    xs[i] ← main_matriks[main_matriks.length - 1][i]

array_sum_xy ← [n+1]
j traversal [0..n+2)
  array_sum_xy[j] ← 0
  i traversal [0..k+1]
    array_sum_xy ← array_sum_xy + main_matrix[i][j]

row_matrix2 ← k
col_matrix2 ← n * (n + 1)
matrix2 ← [row_matrix2][col_matrix2]
i traversal [0..row_matrix2+1]
  j traversal [0..col_matrix2+1]
    if j < n then
      matrix2[i][j] ← main_matrix[i][n] * main_matrix[i][j]

```

```

        else
            matrix2[i][j] ← main_matrix[i][(j/n)-1] * main_matrix[i][j%n]

array_sum ← [col_matrix2]
j traversal [0..col_matrix2+1]
    array_sum[j] ← 0
    i traversal [0..row_matrix2+1]
        array_sum[j] ← array_sum[j] + matrix2[i][j]

matrix_final ← [n+1][n+2]
matrix_final[0][0] ← k
matrix_final[0][n+1] ← array_sum_xy[n]
j traversal [1..n+2]
    matrix_final[0][j] ← array_sum_xy[j-1]
i traversal [1..n+2]
    matrix_final[i][0] ← array_sum_xy[i-1]
    matrix_final[i][n+1] ← array_sum[i-1]
b ← n
i traversal [1..n+2]
    j traversal [1..n+2]
        matrix_final[i][j] ← array_sum[b]
        b ← b + 1

gauss_gauss_jordan.elimination_before(matriks_final, rows, cols, swap_counter)

gauss_gauss_jordan.gauss(matriks_final, rows, cols)

gauss_gauss_jordan.gauss_jordan(matriks_final, rows, cols)

SPL = SPLInterpolasiRegresi(matrix_final, false)

```

```

result ← 0

i traversal [0..rows]

    result ← result + (matrix_final[i][n+1] * xs[i-1])

resultString ← SPL + "\nNilai taksiran untuk x " + xs + " adalah " + result

ReadDisplayArray.displayOutputSPL(resultString)

```

## VIII. Utils Class

Utils *Class* berisi *method-method* yang membantu dalam penyelesaian masalah pada program secara umum. Terdapat dua *method* public pada *class* ini yang dapat digunakan pada semua *class* di program ini, yaitu:

- a. `public static int chooseOptionValidation(int numStart, int numEnd)`  
*Method* yang berfungsi untuk memvalidasi *input* opsi dari pengguna agar selalu berada pada *range* [numStart...numEnd] sesuai dengan parameter yang diberikan.
- b. `public static void displayMenu()`  
*Method* untuk menampilkan menu pada *Main class*.
- c. `public static boolean goToMenu()`  
*Method* untuk menanyakan pengguna apakah masih ingin berada pada program (kembali ke menu) atau sudah selesai dan ingin keluar dari program. *Method* ini digunakan pada *Main class*.
- d. `public static float power(float x, int p)`  
Fungsi untuk memangkatkan bilangan *float* *x* sebanyak *p*.

## BAB 4 EKSPERIMEN

### A. Tampilan Awal dan Akhir

```
*****WELCOME*****

MENU
1. Sistem Persamaan Linier
2. Determinan
3. Matriks Balikan
4. Interpolasi Polinom
5. Regresi Linier Berganda
6. Keluar

Masukkan nomor opsi yang diinginkan = █

Apakah masih ingin melanjutkan perhitungan (Y/N)? N
Terima kasih! Sampai bertemu lagi..
```

### B. Sistem Persamaan Linier

#### *Eksperimen 1 - Eliminasi Gauss (Keyboard) Matrix 11 x 11*

```
Masukkan jumlah baris (m) dan kolom (n) matrix
m = 11
n = 11

8 7 2 3 4 5 6 8 7 1 2
9 7 6 5 3 2 1 3 3 2 1
9 0 2 1 4 2 3 4 5 2 1
6 5 3 2 1 4 5 6 2 3 4
2 3 4 1 2 3 5 6 8 7 2
9 8 7 6 2 1 3 4 5 2 1
3 4 2 1 2 5 4 3 2 1 2
7 6 5 4 3 2 1 4 5 2 3
4 3 2 2 1 2 3 4 5 6 4
1 3 4 5 8 9 2 9 3 5 6
8 7 6 2 1 5 4 3 2 1 2

Matrix setelah Eliminasi Gauss:

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 0.875, 0.25, 0.375, 0.5, 0.625, 0.75, 1.0, 0.875, 0.125, 0.25]
[0.0, 1.0, -4.285714, -1.8571428, 1.7142857, 4.142857, 6.571429, 6.857143, 5.571429, -1.0, 1.4285715]
[0.0, 0.0, 1.0, -1.6666666, -3.6666663, 2.9999998, 4.9999995, 3.9999998, -4.333333, 4.6666665, 6.6666665]
[0.0, 0.0, 0.0, 1.0, 1.8076922, -1.730769, -2.826923, -2.3076923, 2.1730766, -1.9230767, -3.4230766]
[0.0, 0.0, 0.0, 0.0, 1.0, 0.18260898, -0.44782674, 0.017391803, 1.3608694, 0.2782611, -0.73913026]
[0.0, 0.0, 0.0, 0.0, 0.0, 1.0, -2.9087243, -4.46825, -5.781737, -3.0396786, -0.39682743]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.20213269, 0.46809205, 0.5851095, -1.5212727]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.3104057, 0.9767772, 1.9441763]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0296034, 0.73378956]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.6595356]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]
```

## *Eksperimen 2 - Eliminasi Gauss-Jordan (Keyboard) Matrix 11 x 11*

Matrix setelah Eliminasi Gauss-Jordan:

Jenis output apa yang ingin diberikan:

1. Keyboard
2. File

Masukkan nomor opsi yang diinginkan = 1

```
[1.0, 0.0, 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, 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, 0.0, 0.0, 1.0, 0.0, 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, 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, 0.0, 0.0, 1.0, 0.0, 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, 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, 0.0, 0.0, 1.0, 0.0]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]
```

```
[8.0, 7.0, 2.0, 3.0, 4.0, 5.0, 6.0, 8.0, 7.0, 1.0, 2.0]
[9.0, 7.0, 6.0, 5.0, 3.0, 2.0, 1.0, 3.0, 3.0, 2.0, 1.0]
[9.0, 0.0, 2.0, 1.0, 4.0, 2.0, 3.0, 4.0, 5.0, 2.0, 1.0]
[6.0, 5.0, 3.0, 2.0, 1.0, 4.0, 5.0, 6.0, 2.0, 3.0, 4.0]
[2.0, 3.0, 4.0, 1.0, 2.0, 3.0, 5.0, 6.0, 8.0, 7.0, 2.0]
[9.0, 8.0, 7.0, 6.0, 2.0, 1.0, 3.0, 4.0, 5.0, 2.0, 1.0]
[3.0, 4.0, 2.0, 1.0, 2.0, 5.0, 4.0, 3.0, 2.0, 1.0, 2.0]
[7.0, 6.0, 5.0, 4.0, 3.0, 2.0, 1.0, 4.0, 5.0, 2.0, 3.0]
[4.0, 3.0, 2.0, 2.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 4.0]
[1.0, 3.0, 4.0, 5.0, 8.0, 9.0, 2.0, 9.0, 3.0, 5.0, 6.0]
[8.0, 7.0, 6.0, 2.0, 1.0, 5.0, 4.0, 3.0, 2.0, 1.0, 2.0]
```

Solusi SPL:

SPL tidak memiliki solusi

## *Eksperimen 3 - SPL dengan kaidah Cramer Matrix 6 x 6*

Masukkan jumlah baris (m) dan kolom (n) matrix

m = 6

n = 6

8 7 6 2 3 1

8 9 6 7 2 1

9 1 6 3 4 2

5 6 3 2 1 1

4 3 2 2 1 0

7 8 5 6 4 3

```
[8.0, 7.0, 6.0, 2.0, 3.0, 1.0]
[8.0, 9.0, 6.0, 7.0, 2.0, 1.0]
[9.0, 1.0, 6.0, 3.0, 4.0, 2.0]
[5.0, 6.0, 3.0, 2.0, 1.0, 1.0]
[4.0, 3.0, 2.0, 2.0, 1.0, 0.0]
[7.0, 8.0, 5.0, 6.0, 4.0, 3.0]
```

Solusi SPL:

Solusi tidak dapat dihitung dengan metode ini karena jumlah persamaan != jumlah variabel atau determinannya bernilai 0



#### *Eksperimen 4 – SPL dengan metode matriks balikan*

```
Masukkan nomor opsi yang diinginkan = 1

***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah baris (m) dan kolom (n) matrix
m = 4
n = 5
1 2 9 7 6
2 -1 5 3 1
2 9 6 5 3
-2 -3 5 1 8

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 3

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 2.0, 9.0, 7.0, 6.0]
[2.0, -1.0, 5.0, 3.0, 1.0]
[2.0, 9.0, 6.0, 5.0, 3.0]
[-2.0, -3.0, 5.0, 1.0, 8.0]
Solusi SPL:
x0=-1.625 x1=0.1704545 x2=1.1363637 x3=-0.42045462
Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

### C. Determinan

#### *Eksperimen 1 – Input dan Output Keyboard – Metode Reduksi Baris*

```
***** DETERMINAN *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah baris dan kolom (n) matrix
n = 4
1 2 3 4
2 8 6 4
3 7 1 8
3 7 5 6

SUB-MENU DETERMINAN
1. Metode Reduksi Baris
2. Metode Ekspansi Kofaktor

Masukkan nomor opsi yang diinginkan = 1

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 2.0, 3.0, 4.0]
[2.0, 8.0, 6.0, 4.0]
[3.0, 7.0, 1.0, 8.0]
[3.0, 7.0, 5.0, 6.0]

Hasil determinan untuk matriks di atas dengan metode reduksi baris adalah 112.0
Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

## ***Eksperimen 2 – Input File dan Output Keyboard – Metode Ekspansi Kofaktor***

```
***** DETERMINAN *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/determinan.txt

SUB-MENU DETERMINAN
1. Metode Reduksi Baris
2. Metode Ekspansi Kofaktor

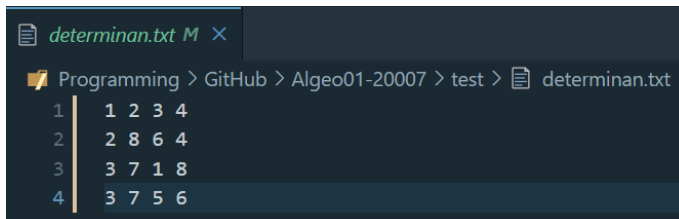
Masukkan nomor opsi yang diinginkan = 2

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 2.0, 3.0, 4.0]
[2.0, 8.0, 6.0, 4.0]
[3.0, 7.0, 1.0, 8.0]
[3.0, 7.0, 5.0, 6.0]

Hasil determinan dari matriks di atas dengan metode ekspansi kofaktor adalah 112.0
Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

*Input file:* determinan.txt



```
determinan.txt M X
Programming > GitHub > Algeo01-20007 > test > determinan.txt
1 | 1 2 3 4
2 | 2 8 6 4
3 | 3 7 1 8
4 | 3 7 5 6
```

## ***Eksperimen 3 – Input Keyboard dan Output File – Metode Reduksi Baris***

```
***** DETERMINAN *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah baris dan kolom (n) matrix
n = 4
1 1 -1 -1
2 5 -7 -5
2 -1 1 3
5 2 -4 2

SUB-MENU DETERMINAN
1. Metode Reduksi Baris
2. Metode Ekspansi Kofaktor

Masukkan nomor opsi yang diinginkan = 1

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2
```

Output file: Determinan75.txt

```
Determinan75.txt U X
Programming > GitHub > Algeo01-20007 > test > Determinan75.txt
1 [1.0, 1.0, -1.0, -1.0]
2 [2.0, 5.0, -7.0, -5.0]
3 [2.0, -1.0, 1.0, 3.0]
4 [5.0, 2.0, -4.0, 2.0]
5
6 Hasil determinan untuk matriks di atas dengan metode reduksi baris adalah -0.0
```

#### Eksperimen 4 – Input bukan matriks persegi

```
***** DETERMINAN *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/determinan.txt
[1.0, 2.0, 3.0, 4.0]
[2.0, 8.0, 6.0, 4.0]
[3.0, 7.0, 1.0, 8.0]
Matriks di atas tidak memiliki determinan karena bukan matriks persegi
Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

Input file: determinan.txt

```
determinan.txt M X
Programming > GitHub > Algeo01-20007 > test > determinan.txt
1 1 2 3 4
2 2 8 6 4
3 3 7 1 8
```

## D. Matriks Balikan

#### Eksperimen 1 – Input dan Output Keyboard – Metode Kofaktor

```
***** MATRIKS BALIKAN *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah baris dan kolom (n) matrix
n = 3
1 2 3
4 9 2
0 7 6

SUB-MENU MATRIKS BALIKAN
1. Metode Kofaktor
2. Metode Reduksi Baris
Masukkan nomor opsi yang diinginkan = 1

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1
```

```
[1.0, 2.0, 3.0]
[4.0, 9.0, 2.0]
[0.0, 7.0, 6.0]

Matriks Balikan:
[0.5263158, 0.118421055, -0.3026316]
[-0.31578946, 0.078947365, 0.13157895]
[0.36842105, -0.09210526, 0.013157895]

Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

## ***Eksperimen 2 – Input Keyboard dan Output File – Metode Reduksi Baris***

```
***** MATRIKS BALIKAN *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah baris dan kolom (n) matrix
n = 4
1 2 4 1
7 5 3 1
2 9 6 4
-2 0 3 7

SUB-MENU MATRIKS BALIKAN
1. Metode Kofaktor
2. Metode Reduksi Baris
Masukkan nomor opsi yang diinginkan = 2

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

### ***Output file: Invers93.txt***

```
Programming > GitHub > Algeo01-20007 > test > Invers93.txt
1 [1.0, 2.0, 4.0, 1.0]
2 [7.0, 5.0, 3.0, 1.0]
3 [2.0, 9.0, 6.0, 4.0]
4 [-2.0, 0.0, 3.0, 7.0]
5
6 Matriks Balikan:
7 [-0.01323043, 0.18191841, -0.09812569, 0.03197354]
8 [-0.17861082, -0.044101436, 0.1753032, -0.06835722]
9 [0.38478503, -0.04079383, -0.062844545, -0.01323043]
10 [-0.16868798, 0.06945976, -0.0011025359, 0.15766263]
11
```

## ***Eksperimen 3 – Determinan = 0***

```

***** MATRIKS BALIKAN *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah baris dan kolom (n) matrix
n = 3
1 2 4
2 9 8
0 0 0
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 2.0, 4.0]
[2.0, 9.0, 8.0]
[0.0, 0.0, 0.0]

Matriks Balikan:
Matriks tidak memiliki balikan karena determinannya bernilai 0

```

## E. Interpolasi Polinom

### *Eksperimen 1*

```

***** INTERPOLASI POLINOM *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah titik (n)
n = 5
Masukkan input dengan format x y
1 2
2 12
3 36
4 80
5 150
Nilai x yang akan ditaksir = 3.28
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

```

Output *file* yang diberikan:

```

≡ SPL28.txt
1 + 1.0x^2 + 1.0x^3
2 Hasil taksiran nilai untuk x = 3.28 adalah 46.04595

```

### *Eksperimen 2*

```

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-20007/test/interpolasipolinomtes.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

23.6 - 40.6x + 22.600002x^2 - 3.6000001x^3
Hasil taksiran nilai untuk x = 4.0 adalah -7.5999603
Apakah masih ingin melanjutkan perhitungan (Y/N)? 

```

File data yang diberikan:

```

interpolasipolinomtes.txt M X
test > interpolasipolinomtes.txt
1 1 2
2 3.5 4
3 2 4
4 3 8
5 4

```

## F. Regresi Linier Berganda

### Eksperimen 1

```

***** REGRESI LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-20007/test/regresiliniertes.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

3.9187276 + 2.4911659x1 - 0.4664309x2
Nilai taksiran untuk x [4.0, 5.0] adalah 11.551237

```

File data yang diberikan (sumber: <https://www.youtube.com/watch?v=4oSfPMec0BY>):

```

regresiliniertes.txt M X
test > regresiliniertes.txt
1 10 7 23
2 2 3 7
3 4 2 15
4 6 4 17
5 8 6 23
6 7 5 22
7 4 3 10
8 6 3 14
9 7 4 20
10 6 3 19
11 4 5

```

## Eksperimen 2

```
***** REGRESI LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-20007/test/regresiliniertes2.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

28.321392 + 0.5195998x1 + 0.051929697x2
Nilai taksiran untuk x [7.0, 125.78] adalah 38.49031
Apakah masih ingin melanjutkan perhitungan (Y/N)? y
```

File data yang diberikan (sumber: <https://www.youtube.com/watch?v=aQexxrQBS00>):

```
regresiliniertes2.txt M X
test > regresiliniertes2.txt
3 7 100 34
4 12 122 39
5 9 129 40
6 10 128 42
7 7 98 38
8 8 103 42
9 11 130 40
10 8 95 36
11 10 115 41
12 8 105 38
13 7 125.78
```

## Eksperimen 3

```
***** REGRESI LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah titik x (n)
n = 2
Masukkan jumlah data (k)
k = 7
Masukkan input dengan format x1 x2 .. xn y sebanyak k data
4 3 6
7 6 8
6 5 6
14 16 15
10 12 20
7 9 11
8 9 10
Nilai x yang akan ditaksir (masukkan x sejumlah n) = 5 14
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2
```

Sumber data: <https://www.youtube.com/watch?v=mtm4WMkK27I>

Output *file* yang diberikan:

```

≡ SPL55.txt U X
≡ SPL55.txt
1 5.445012 - 1.4607239x1 + 1.9947577x2
2 Nilai taksiran untuk x [5.0, 14.0] adalah 26.068

```

## G. Studi Kasus

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

a.

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

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

Solusi:

```

***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/studikassus1a.txt

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 2

Matrix setelah Eliminasi Gauss-Jordan:
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 0.0, 0.0, 0.6666666, 1.6666669]
[0.0, 1.0, 0.0, -2.6666665, 0.33333325]
[0.0, 0.0, 1.0, -1.0, 0.0]
[0.0, 0.0, 0.0, 0.0, 1.0]
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 1.0, -1.0, -1.0, 1.0]
[2.0, 5.0, -7.0, -5.0, -2.0]
[2.0, -1.0, 1.0, 3.0, 4.0]
[5.0, 2.0, -4.0, 2.0, 6.0]
Solusi SPL:
SPL tidak memiliki solusi

Apakah masih ingin melanjutkan perhitungan (Y/N)? █

```



b.

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

$$\text{Matriks Augmented} = \begin{bmatrix} 1 & -1 & 0 & 0 & 1 & 3 \\ 2 & 1 & 0 & -3 & 0 & 6 \\ 2 & -1 & 0 & 1 & -1 & 5 \\ -1 & 2 & 0 & -2 & -1 & -1 \end{bmatrix}$$

Solusi:

```
***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/studikasukas1b.txt

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 2

Matrix setelah Eliminasi Gauss-Jordan:

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, 0.0, 0.0, 0.0, -0.5, 1.5]
[0.0, 1.0, 0.0, 0.0, -1.5, -1.5]
[0.0, 0.0, 0.0, 1.0, -1.0, -1.0]
[0.0, 0.0, 0.0, 0.0, -0.25, 0.75]
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[1.0, -1.0, 0.0, 0.0, 1.0, 3.0]
[2.0, 1.0, 0.0, -3.0, 0.0, 6.0]
[2.0, -1.0, 0.0, 1.0, -1.0, 5.0]
[-1.0, 2.0, 0.0, -2.0, -1.0, -1.0]
Solusi SPL:
x0 = 1.5
x1 = -1.5
x2 = a
x3 = -1.0
x4 = 0.75

Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

c.

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

$$\text{Matriks Augmented} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 1 & 1 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Solusi:

```

***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/studikusus1c.txt

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 2

Matrix setelah Eliminasi Gauss-Jordan:

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[0.0, 1.0, 0.0, 0.0, -1.0, 2.0, 0.0]
[0.0, 0.0, 0.0, 1.0, 0.0, 1.0, -2.0]
[0.0, 0.0, 0.0, 0.0, 1.0, -1.0, 1.0]
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 2.0]
[0.0, 0.0, 0.0, 1.0, 1.0, 0.0, -1.0]
[0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0]
Solusi SPL:
x0 = a
x1 = 1.0 + (-1.0c)
x2 = b
x3 = -2.0 + (-1.0c)
x4 = 1.0 + (1.0c)
x5 = c

Apakah masih ingin melanjutkan perhitungan (Y/N)? █

```

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

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

▪ Untuk  $n = 6$ ,

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} \\ \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\text{Matriks Augmented} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & 1 \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & 0 \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & 0 \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & 0 \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & 0 \\ \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & 0 \end{bmatrix}$$

Solusi:

```
***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/studikasus1d1.txt

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 2

Matrix setelah Eliminasi Gauss-Jordan:

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1
[1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 31.83985]
[0.0, 1.0, 0.0, 0.0, 0.0, 0.0, -468.32205]
[0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 2140.103]
[0.0, 0.0, 0.0, 1.0, 0.0, 0.0, -4216.6035]
[0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 3761.731]
[0.0, 0.0, 0.0, 0.0, 0.0, 1.0, -1249.3833]
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1
[1.0, 0.5, 0.33333, 0.25, 0.2, 0.16667, 1.0]
[0.5, 0.33333, 0.25, 0.2, 0.16667, 0.14286, 0.0]
[0.33333, 0.25, 0.2, 0.16667, 0.14286, 0.125, 0.0]
[0.25, 0.2, 0.16667, 0.14286, 0.125, 0.11111, 0.0]
[0.2, 0.16667, 0.14286, 0.125, 0.11111, 0.1, 0.0]
[0.16667, 0.14286, 0.125, 0.11111, 0.1, 0.09091, 0.0]
Solusi SPL:
x0 = 31.83985
x1 = -468.32205
x2 = 2140.103
x3 = -4216.6035
x4 = 3761.731
x5 = -1249.3833

Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

- Untuk  $n = 10$ ,

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} \\ \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} \\ \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} \\ \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} \\ \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\text{Matriks Augmented} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & 1 \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & 0 \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & 0 \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & 0 \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & 0 \\ \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & 0 \\ \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & 0 \\ \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & 0 \\ \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & 0 \\ \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & 0 \end{bmatrix}$$

Solusi:

```
***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/studikasukus1d2.txt

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 2

Matrix setelah Eliminasi Gauss-Jordan:

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1
```

```

[1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 35.658066]
[0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -524.9935]
[0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 2062.781]
[0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -2608.2913]
[0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 286.54498]
[0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 546.06976]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 369.77875]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, -47.03357]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 949.333]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, -1084.2583]
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1
[1.0, 0.5, 0.33333, 0.25, 0.2, 0.16667, 0.14286, 0.125, 0.11111, 0.1, 1.0]
[0.5, 0.33333, 0.25, 0.2, 0.16667, 0.14286, 0.125, 0.11111, 0.1, 0.09091, 0.0]
[0.33333, 0.25, 0.2, 0.16667, 0.14286, 0.125, 0.11111, 0.1, 0.09091, 0.08333, 0.0]
[0.25, 0.2, 0.16667, 0.14286, 0.125, 0.11111, 0.1, 0.09091, 0.08333, 0.07692, 0.0]
[0.2, 0.16667, 0.14286, 0.125, 0.11111, 0.1, 0.09091, 0.08333, 0.07692, 0.07143, 0.0]
[0.16667, 0.14286, 0.125, 0.11111, 0.1, 0.09091, 0.08333, 0.07692, 0.07143, 0.06667, 0.0]
[0.14286, 0.125, 0.11111, 0.1, 0.09091, 0.08333, 0.07692, 0.07143, 0.06667, 0.0625, 0.0]
[0.125, 0.11111, 0.1, 0.09091, 0.08333, 0.07692, 0.07143, 0.06667, 0.0625, 0.05882, 0.0]
[0.11111, 0.1, 0.09091, 0.08333, 0.07692, 0.07143, 0.06667, 0.0625, 0.05882, 0.05556, 0.0]
[0.1, 0.09091, 0.08333, 0.07692, 0.07143, 0.06667, 0.0625, 0.05882, 0.05556, 0.05263, 0.0]
Solusi SPL:
x0 = 35.658066
x1 = -524.9935
x2 = 2062.781
x3 = -2608.2913
x4 = 286.54498
x5 = 546.06976
x6 = 369.77875
x7 = -47.03357
x8 = 949.333
x9 = -1084.2583
Apakah masih ingin melanjutkan perhitungan (Y/N)?

```

## 2. SPL berbentuk matriks *augmented*

a.

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

Solusi:

```

[1.0, -1.0, 2.0, -1.0, -1.0]
[2.0, 1.0, -2.0, -2.0, -2.0]
[-1.0, 2.0, -4.0, 1.0, 1.0]
[3.0, 0.0, 0.0, -3.0, -3.0]
Solusi SPL:
x0 = -1.0 + (1.0b) + (-0.0a)
x1 = 0.0 + (-0.0b) + (2.0a)
x2 = a
x3 = b

```

b.

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

Solusi:

```
[2.0, 0.0, 8.0, 0.0, 8.0]
[0.0, 1.0, 0.0, 4.0, 6.0]
[-4.0, 0.0, 6.0, 6.0, 6.0]
[0.0, -2.0, 0.0, 3.0, -1.0]
[2.0, 0.0, -4.0, 0.0, -4.0]
[0.0, 1.0, 1.0, -2.0, 0.0]
Solusi SPL:
SPL tidak memiliki solusi
```

### 3. SPL berbentuk

a.  $8x_1 + x_2 + 3x_3 + 2x_4 = 0$

$$2x_1 + 9x_2 - x_3 - 2x_4 = 1$$

$$x_1 + 3x_2 + 2x_3 - x_4 = 2$$

$$x_1 + 6x_3 + 4x_4 = 3$$

$$\text{Matriks Augmented} = \begin{bmatrix} 8 & 1 & 3 & 2 & 0 \\ 2 & 9 & -1 & -2 & 1 \\ 1 & 3 & 2 & -1 & 2 \\ 1 & 0 & 6 & 4 & 3 \end{bmatrix}$$

Solusi:

```
***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/studikasuk3a.txt

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 4

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1
```

```
[8.0, 1.0, 3.0, 2.0, 0.0]
[2.0, 9.0, -1.0, -2.0, 1.0]
[1.0, 3.0, 2.0, -1.0, 2.0]
[1.0, 0.0, 6.0, 4.0, 3.0]
Solusi SPL:
x0=-0.22432433 x1=0.18243243 x2=0.7094595 x3=-0.2581081
Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

b.

$$x_7 + x_8 + x_9 = 13.00$$

$$x_4 + x_5 + x_6 = 15.00$$

$$x_1 + x_2 + x_3 = 8.00$$

$$0.04289(x_3 + x_5 + x_7) + 0.75(x_6 + x_8) + 0.61396x_9 = 14.79$$

$$0.91421(x_3 + x_5 + x_7) + 0.25(x_2 + x_4 + x_6 + x_8) = 14.31$$

$$0.04289(x_3 + x_5 + x_7) + 0.75(x_2 + x_4) + 0.61396x_1 = 14.79$$

$$x_3 + x_6 + x_9 = 18.00$$

$$x_2 + x_5 + x_8 = 12.00$$

$$x_1 + x_4 + x_7 = 6.00$$

$$0.04289(x_1 + x_5 + x_9) + 0.75(x_2 + x_6) + 0.61396x_3 = 10.5$$

$$0.91421(x_1 + x_5 + x_9) + 0.25(x_2 + x_4 + x_6 + x_8) = 16.13$$

$$0.04289(x_1 + x_5 + x_9) + 0.75(x_4 + x_8) + 0.61396x_7 = 7.04$$

Matriks Augmented =

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 13.00 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 15.00 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 8.00 \\ 0 & 0 & 0.04289 & 0 & 0.04289 & 0.75 & 0.04289 & 0.75 & 0.61396 & 14.79 \\ 0 & 0.25 & 0.91421 & 0.25 & 0.91421 & 0.25 & 0.91421 & 0.25 & 0 & 14.31 \\ 0.61396 & 0.75 & 0.04289 & 0.75 & 0.04289 & 0 & 0.04289 & 0 & 0 & 14.79 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 18.00 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 12.00 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 6.00 \\ 0.04289 & 0.75 & 0.61396 & 0 & 0.04289 & 0.75 & 0 & 0 & 0.04289 & 10.5 \\ 0.91421 & 0.25 & 0 & 0.25 & 0.91421 & 0.25 & 0 & 0.25 & 0.91421 & 16.13 \\ 0.04289 & 0 & 0 & 0.75 & 0.04289 & 0 & 0.61396 & 0.75 & 0.04289 & 7.04 \end{bmatrix}$$

Solusi:

```

***** SISTEM PERSAMAAN LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
../test/studikasuk3b.txt

SUB-MENU SPL
1. Metode Eliminasi Gauss
2. Metode Eliminasi Gauss-Jordan
3. Metode Matriks Balikan
4. Kaidah Cramer

Masukkan nomor opsi yang diinginkan = 2

Matrix setelah Eliminasi Gauss-Jordan:

Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

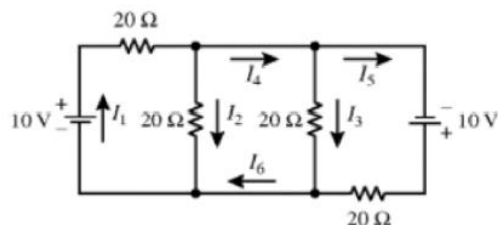
[1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
[0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
[0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
[0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
[0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0]
[0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0]
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 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, 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, 0.0, 1.0]
[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, 0.0, 0.0, 0.0, 0.0]
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 13.0]
[0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 15.0]
[1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 8.0]
[0.0, 0.0, 0.04289, 0.0, 0.04289, 0.75, 0.04289, 0.75, 0.61396, 14.79]
[0.0, 0.25, 0.91421, 0.25, 0.91421, 0.25, 0.91421, 0.25, 0.0, 14.31]
[0.61396, 0.75, 0.04289, 0.75, 0.04289, 0.0, 0.04289, 0.0, 0.0, 14.79]
[0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 18.0]
[0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 12.0]
[1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 6.0]
[0.04289, 0.75, 0.61396, 0.0, 0.04289, 0.75, 0.0, 0.0, 0.04289, 10.5]
[0.91421, 0.25, 0.0, 0.25, 0.91421, 0.25, 0.0, 0.25, 0.91421, 16.13]
[0.04289, 0.0, 0.0, 0.75, 0.04289, 0.0, 0.61396, 0.75, 0.04289, 7.04]
Solusi SPL:
SPL tidak memiliki solusi

Apakah masih ingin melanjutkan perhitungan (Y/N)?

```

4. Tentukan arus yang mengalir pada rangkaian listrik di bawah ini:



Dengan KVL, dapat diperoleh persamaan-persamaan berikut dalam matrix augmented, solusi:



```

[20.0, 20.0, 0.0, 0.0, 0.0, 0.0, 10.0]
[0.0, 0.0, -20.0, 0.0, 20.0, 0.0, 10.0]
[1.0, -1.0, 0.0, -1.0, 0.0, 0.0, 0.0]
[0.0, 0.0, -1.0, 1.0, -1.0, 0.0, 0.0]
[0.0, 0.0, -1.0, 0.0, -1.0, 1.0, 0.0]
[1.0, -1.0, 0.0, 0.0, 0.0, -1.0, 0.0]

```

Solusi SPL:

$$x_0 = 0.25 + (0.5a)$$

$$x_1 = 0.25 + (-0.5a)$$

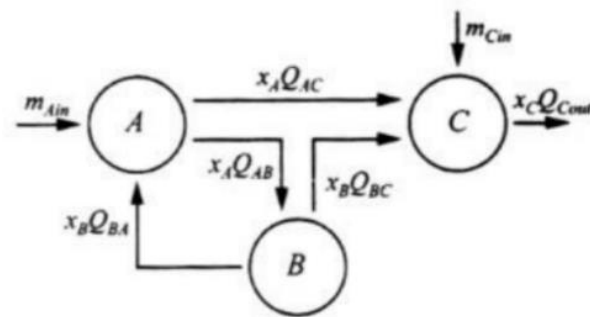
$$x_2 = -0.25 + (0.5a)$$

$$x_3 = 0.0 + (1.0a)$$

$$x_4 = 0.25 + (0.5a)$$

$$x_5 = a$$

5. Lihatlah sistem reaktor pada gambar berikut



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

$$A: m_{Ain} + Q_{BA}x_B - Q_{AB}x_A - Q_{AC}x_A = 0$$

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

$$C: m_{Cin} + Q_{AC}x_A - Q_{BC}x_B - Q_{Cout}x_C = 0$$

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  $m_{Cin} = 200 mg/s$ .

Solusi:

Setelah dilakukan substitusi, persamaannya menjadi:

$$A: 120x_A - 60x_B = 1300$$

$$B: 40x_A - 80x_B = 0$$

$$C: 80x_A - 20x_B - 150x_C = -200$$

$$\text{Matriks Augmented} = \begin{bmatrix} 120 & -60 & 0 & 1300 \\ 40 & -80 & 0 & 0 \\ 80 & -20 & -150 & -200 \end{bmatrix}$$

```
[1.0, 0.0, 0.0, 14.444443]
[0.0, 1.0, 0.0, 7.222229]
[0.0, 0.0, 1.0, 8.074074]
```

```
[120.0, -60.0, 0.0, 1300.0]
[40.0, -80.0, 0.0, 0.0]
[80.0, -20.0, -150.0, -200.0]
Solusi SPL:
x0 = 14.444443
x1 = 7.222229
x2 = 8.074074
```

## 6. Studi Kasus Interpolasi

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

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

Lakukan pengujian pada nilai-nilai default berikut:

$$x = 0.2 \quad f(x) = ?$$

$$x = 0.55 \quad f(x) = ?$$

$$x = 0.85 \quad f(x) = ?$$

$$x = 1.28 \quad f(x) = ?$$

Solusi:

Solusi persamaan interpolasi polinom yang didapat adalah

$$y = -0.02297638 + 0.23999657x + 0.19741657x^2 - 0.000057586807x^3 \\ + 0.026122008x^4 - 0.00005440668x^5 + 0.0000141696555x^6$$

Sehingga didapatkan

$$f(0.2) = 0.032960914$$

$$f(0.55) = 0.17111865$$

$$f(0.85) = 0.33723584$$

$$f(1.28) = 0.677542$$

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

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

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

$$\text{tanggal(desimal)} = \text{bulan} + (\text{tanggal} / \text{jumlah hari pada bulan tersebut})$$

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

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

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

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

Solusi:

Dengan menggunakan semua data di atas dan dilakukan interpolasi polinom pada program yang telah dibuat, didapatkan persamaan  $y = -32766406700 +$

$$26603251700x - 8626530300x^2 + 1391142140x^3 - 1.16609912x^4 + 7850794.0x^5 - 1298199.5x^6 + 183236.7x^7 - 12223.826x^8 + 307.50186x^9$$

Namun, apabila disubstitusikan dengan persoalan tanggal-tanggal di atas, didapatkan angka negatif

```
***** INTERPOLASI POLINOM *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-20007/test/studikassus6.2.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

-3.27664067E10 + 2.66032517E10x - 8.6265303E9x^2 + 1.39114214E9x^3 - 1.16609912E8x^4 + 7850794.0x^5 - 1298199.5x^6 + 183236.7x^7 - 12223.826x^8 + 307.50186x^9
Hasil taksiran nilai untuk x = 8.226 adalah -2781184.0
Apakah masih ingin melanjutkan perhitungan (Y/N)? Y
```

Dari sini, kami berasumsi bahwa interpolasi polinom pada program kami memiliki limit, apalagi jika jumlah data/titik yang dimasukkan cukup banyak. Sehingga untuk persoalan-persoalan tersebut, akan diambil 5 data yang paling dekat dengan nilai yang dicari.

a. Tanggal (desimal) =  $7 + (16/31) = 7.516$

Range data yang diambil: 30/06/2021 sampai 26/07/2021

Persamaan yang didapat adalah  $y = 4697577000 - 2.529585150x + 5.10358656x^2 - 4.5723228x^3 + 1534795.2x^4$

Sehingga perkiraan jumlah covid pada tanggal 16/07/2021 adalah 64000

```
***** INTERPOLASI POLINOM *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-20007/test/studikassus6.2.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

4.697577E9 - 2.52958515E9x + 5.10358656E8x^2 - 4.5723228E7x^3 + 1534795.2x^4
Hasil taksiran nilai untuk x = 7.516 adalah 64000.0
Apakah masih ingin melanjutkan perhitungan (Y/N)? Y
```

b. Tanggal (desimal) =  $8 + (10/31) = 8.323$

Range data yang diambil: 26/07/2021 sampai 30/08/2021

Persamaan yang didapat adalah  $y = -457819712 + 210342640x - 36165144x^2 + 2758211.2x^3 - 78739.92x^4$

Sehingga perkiraan jumlah covid pada tanggal 10/08/2021 adalah 28960

```
***** INTERPOLASI POLINOM *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-2007/test/studikasuk6.2.2.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

-4.57819712E8 + 2.1034264E8x - 3.6165144E7x^2 + 2758211.2x^3 - 78739.92x^4
Hasil taksiran nilai untuk x = 8.323 adalah 28960.0
```

c. Tanggal (desimal) =  $9 + (5/30) = 9.167$

Range data yang diambil: 26/07/2021 sampai 30/08/2021

Persamaan yang didapat adalah  $y = -457819712 + 210342640x - 36165144x^2 + 2758211.2x^3 - 78739.92x^4$

Sehingga perkiraan jumlah covid pada tanggal 5/09/2021 adalah 28960

```
***** INTERPOLASI POLINOM *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-2007/test/studikasuk6.2.3.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

-4.57819712E8 + 2.1034264E8x - 3.6165144E7x^2 + 2758211.2x^3 - 78739.92x^4
Hasil taksiran nilai untuk x = 9.167 adalah 13248.0
Apakah masih ingin melanjutkan perhitungan (Y/N)? Y
```

d. Tanggal 07/08/2021

Desimal =  $8 + (7/31) = 8.226$

Range data yang diambil: 26/07/2021 sampai 30/08/2021

Persamaan yang didapat adalah  $y = -457819712 + 210342640x - 36165144x^2 + 2758211.2x^3 - 78739.92x^4$

Sehingga perkiraan jumlah covid pada tanggal 07/08/2021 adalah 33536

```
***** INTERPOLASI POLINOM *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-2007/test/studikasuk6.2.4.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

-4.57819712E8 + 2.1034264E8x - 3.6165144E7x^2 + 2758211.2x^3 - 78739.92x^4
Hasil taksiran nilai untuk x = 8.226 adalah 33536.0
```

c. Sederhanakan fungsi

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

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

Solusi:

Dengan mengambil  $n = 5$ , didapat titik-titik:

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

Dengan interpolasi polinom derajat 5 didapatkan persamaan  $y = 0.29034984 + 0.37786934x - 0.15031594x^2 + 0.023867622x^3 - 0.003694738x^4$

```
***** INTERPOLASI POLINOM *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

Masukkan jumlah titik (n)
n = 5
Masukkan input dengan format x y
0.4 0.41888
0.8 0.50715
1.2 0.56092
1.6 0.58368
2 0.57665
Nilai x yang akan ditaksir = 1.8
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

0.29034984 + 0.37786934x - 0.15031594x^2 + 0.023867622x^3 - 0.003694738x^4
Hasil taksiran nilai untuk x = 1.8 adalah 0.5839011
Apakah masih ingin melanjutkan perhitungan (Y/N)?
```

## 7. Studi Kasus Regresi Linier Berganda

Diberikan sekumpulan data sesuai pada tabel berikut ini.

Table 12.1: Data for Example 12.1

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

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

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

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

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

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

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

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

Solusi:

Berdasarkan program yang telah dibuat, solusi persamaan linear dari persoalan regresi linear adalah:

$$-3.511651 - 0.002624384x_1 + 0.00079925x_2 + 0.15428x_3$$

Dari persamaan linear tersebut, didapatkan estimasi nilai Nitrous Oxide dengan mensubstitusikan nilai  $x_1 = 50$  (humidity 50%),  $x_2 = 76$  (temperatur 76°F), dan  $x_3 = 29.3$  (tekanan udara 29.30), yaitu sebesar 0.9384268

```
***** REGRESI LINIER *****
Jenis input apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 2

Path File (Contoh: C:/Users/Asus/Documents/matrix.txt)
D:/IF Semester 3/Aljabar Linier dan Geometri/Algeo01-20007/test/studikasus7.txt
Jenis output apa yang ingin diberikan:
1. Keyboard
2. File
Masukkan nomor opsi yang diinginkan = 1

-3.511651 - 0.002624384x1 + 7.9925405E-4x2 + 0.15428507x3
Nilai taksiran untuk x [50.0, 76.0, 29.3] adalah 0.9384258
Apakah masih ingin melanjutkan perhitungan (Y/N)? Y
```



## **BAB 5 KESIMPULAN DAN SARAN**

Dari pengerjaan Tugas Besar 1 Aljabar Linear dan Geometri ini, telah berhasil dibuat suatu program yang dapat menyelesaikan permasalahan sistem persamaan linear, determinan matriks, matriks balikan, interpolasi polinom, dan regresi linear berganda dengan menggunakan metode eliminasi Gauss, Gauss Jordan, matriks inverse, dan kaidah Cramer.

Akan tetapi, masih diperlukan pengembangan lebih lanjut lagi agar program ini bisa dapat digunakan dengan efektif pada semua kasus masalah. Program ini belum bisa menyelesaikan beberapa kasus, terutama pada interpolasi polinom berderajat lebih dari 10 dengan integer yang terlalu besar, seperti pada studi kasus 6 bagian b. Atau mungkin, apabila bertemu dengan kasus menaksir sebuah nilai yang memiliki jumlah data yang cukup banyak (lebih dari 10) akan lebih baik dan efektif menggunakan metode regresi linear, dibandingkan dengan interpolasi polinom. Selain itu, secara keseluruhan, untuk metode Gauss, Gauss Jordan, determinan, dan kaidah Cramer sudah bisa berjalan dengan baik pada semua test case yang telah dicoba.

## DAFTAR REFERENSI

Anton, Howard dan Chris Rorres. 2014. *Elementary Linear Algebra 11<sup>th</sup> Edition*. Canada: Anton Textbooks, Inc.

*Determinan*. <https://id.wikipedia.org/wiki/Determinan> (dakses tanggal 27 September 2021)