**TUGAS BESAR 1 IF2123**

**ALJABAR LINEAR DAN GEOMETRI**

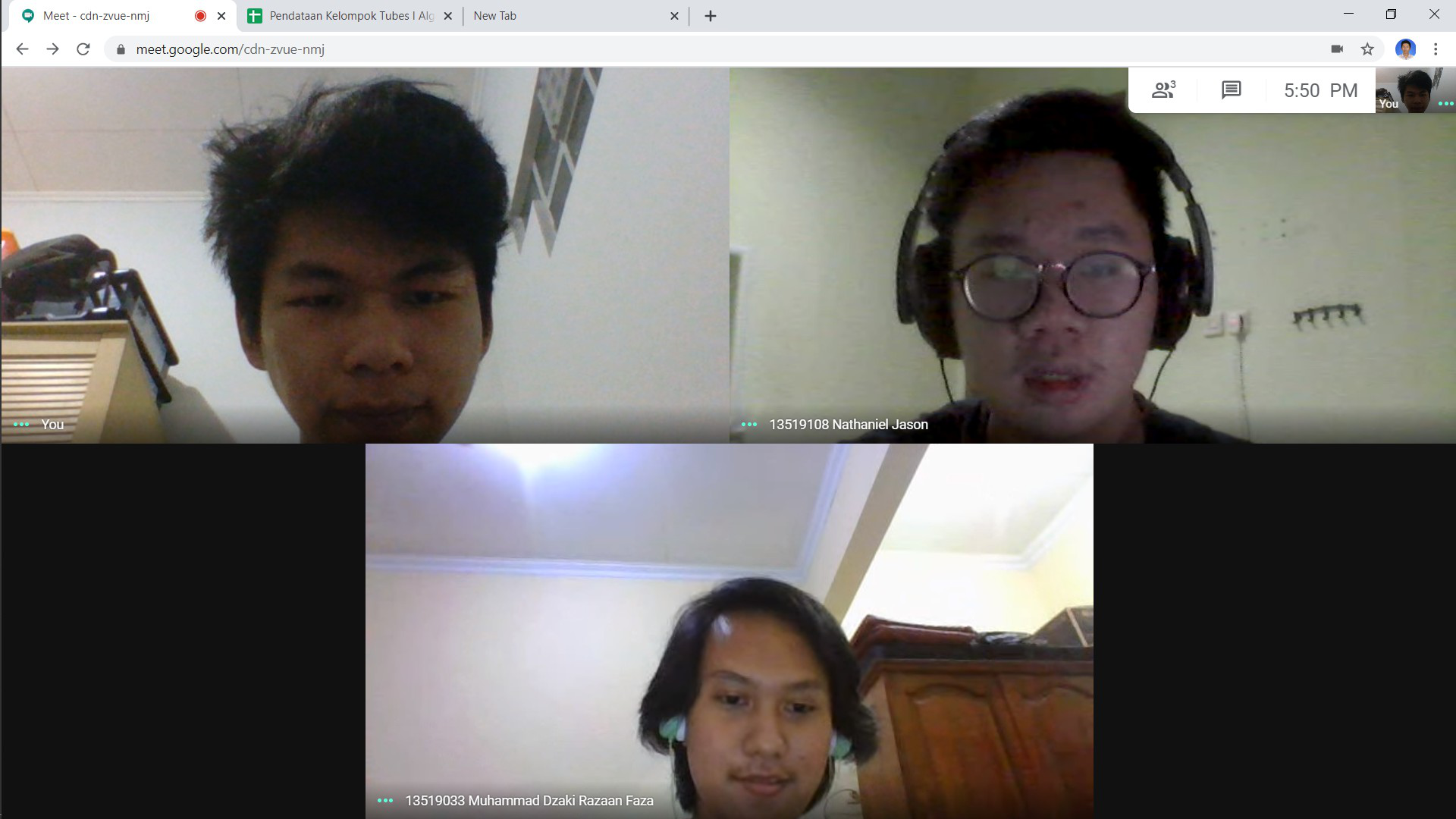
**2020/2021**

Oleh

Muhammad Dzaki Razaan Faza 13519033

Ariya Adinatha 13519048

Nathaniel Jason 13519108



**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2020**

**BAB I**

**PENDAHULUAN**

**1.1 Latar Belakang**

Pada mata kuliah IF2123 Aljabar Linier dan Geometri diberikan materi mengenai teori dan algoritma pada matriks yang dapat digunakan untuk menyelesaikan permasalahan pada aljabar linier, seperti mencari solusi pada sebuah sistem persamaan linier, mencari determinan sebuah matriks, mencari hasil balikan dari sebuah matriks, dan juga melakukan interpolasi polinomial. Langkah - langkah dan algoritma yang dapat digunakan pada adalah Operasi Baris Elementer (OBE), metode eliminasi Gauss, metode eliminasi Gauss Jordan, metode matriks balikan, dan kaidah Cramer. Banyak masalah di dunia nyata yang dapat diselesaikan melalui aljabar linier. Salah satunya adalah ketika mencari nilai arus pada sebuah rangkaian listrik, maupun dalam teknik sipil saat ingin mencari perhitungan mekanika yang melibatkan banyak variabel. Dalam aljabar linier, sistem persamaan linier diubah bentuknya ke dalam persamaan matriks, sehingga dapat diselesaikan oleh metode - metode yang telah disebutkan sebelumnya, karena melibatkan banyak variabel, diperlukan sebuah mesin yang dapat menghitung. Sehingga kemungkinan salah hitung dan ketidaktelitian manusia dapat diperkecil. Oleh karena itu perlunya implementasi algoritma kedalam sebuah mesin agar dapat otomasi dalam perhitungan.

Dibuat program menggunakan bahasa Java yang dapat melakukan hal - hal berikut :

1. Menghitung solusi SPL dengan menggunakan metode eliminasi Gauss, metode eliminasi Gauss Jordan, metode matriks balikan, dan kaidah Cramer.
2. Menyelesaikan persoalan interpolasi polinomial.
3. Menghitung determinan sebuah matriks dengan beberapa cara yang disebutkan pada poin pertama, ditambah dengan menggunakan matriks kofaktor serta matriks adjoin dari matriks.

Program ini memiliki spesifikasi seperti berikut :

1. Program dapat menerima masukan dari keyboard maupun dari file text.
2. Pada sistem persamaan linier (SPL) dapat menerima masukan dari *keyboard* berupa *m, n,* (ukuran matriks), nilai dari koefisien *aij* dan *bij*. Sedangkan, masukan dari text dapat berupa matriks *augmented* tanpa ada tanda kurung, dan setiap elemen matriks dipisah oleh spasi.
3. Untuk menghitung determinan dan matriks balikan, masukan dari keyboard adalah *n* (ukuran matriks persegi) dan *aij*, entri dari matriks perseginya. Untuk masukan dari file, setiap elemen dipisahkan oleh tanda spasi.
4. Untuk persoalan interpolasi, masukannya dari keyboard adalah , yakni derajat dari polinomial yang akan menginterpolasi, serta *n + 1* titik *x,y* yang akan ditaksir oleh polinomial tersebut. Sedangkan, jika masukannya dari file, titik tersebut dinyatakan pada tiap baris tanpa koma dan tanda kurung.
5. Untuk SPL, keluaran program adalah solusi dari SPL tersebut. Jika solusinya tunggal, maka nilainya ditulis. Jika tidak ada solusi, ditulis tidak ada solusi. Sedangkan jika solusinya banyak, ditulis dalam bentuk parametrik.
6. Untuk persoalan determinan, matriks balikan, matriks kofaktor, dan matriks adjoin, maka keluarannya sesuai dengan persoalan masing-masing.
7. Untuk persoalan interpolasi polinom, keluarannya adalah persamaan polinom dan taksiran nilai fungsi pada nilai yang diberikan.
8. Keluaran program harus dapat ditampilkan pada layar ataupun disimpan ke file eksternal.

**BAB II**

**LANDASAN TEORI**

**2.1 Eliminasi Gauss**

Eliminasi Gauss ditemukan oleh **Carl Friedrich Gauss**, metode ini dapat dimanfaatkan untuk memecahkan sistem persamaan linier yang diubah ke bentuk matriks, matriks tersebut lalu diubah ke bentuk **Eselon Baris** melalui Operasi Baris Elementer (OBE), dan diselesaikan dengan substitusi balik. Algoritma ini bertujuan untuk menghasilkan sebuah matriks eselon, yaitu sebuah matriks yang dimulai oleh angka 1 pada setiap barisnya dengan kolomnya yang semakin menjorok ke dalam.

**2.2 Eliminasi Gauss Jordan**

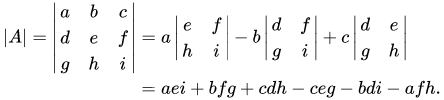
Eliminasi Gauss-Jordan adalah prosedur pemecahan sistem persamaan linear dengan mengubahnya menjadi bentuk matriks **eselon baris tereduksi** dengan OBE. Bentuk eselon baris tereduksi merupakan sebuah bentuk matriks eselon baris yang lebih disederhanakan sehingga pencarian solusi dapat ditemukan lebih mudah. Eselon baris tereduksi bersifat 1 utama, yaitu hanya memiliki angka 1 pada setiap bagian kolom dan barisnya.

**2.3 Determinan**

Determinan adalah nilai yang dapat dihitung dari unsur suatu matriks persegi. Determinan A dapat dinotasikan dengan det(A). Pada matriks berukuran 2 x 2, determinan dapat ditemukan dengan menggunakan rumus berikut :



Sedangkan untuk ukuran 3 x 3 dan lebih besar daripada itu, dapat dihitung dengan ekspansi kofaktor, baik pada baris manapun.



**2.4 Matriks Balikan**

Matriks balikan dari suatu matriks persegi A yang berukuran *n x n* adalah suatu matriks B yang memenuhi *AB = BA = I* , dengan *I* adalah matriks identitas berukuran *n x n*. Matriks B dapat ditulis sebagai . Syarat sebuah matriks dapat memiliki balikan adalah determinannya tidak nol dan berbentuk persegi. Matriks balikan dirumuskan sebagai berikut



Dengan ad - bc tidak bernilai 0.

**2.5 Matriks Kofaktor**

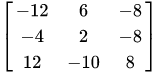
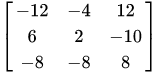
Kofaktor adalah hasil perkalian minor dengan suatu angka yang besarnya mengikuti aturan dimana *i* adalah baris dan *j* adalah kolom. Kofaktor suatu elemen baris ke-*i* dan kolom ke-*j* dari matriks A dilambangkan dengan C*ij*.

C*ij* *=* M*ij*

Matriks kofaktor merupakan matriks yang terdiri dari kofaktor matriks itu sendiri. Susunan matriks kofaktor tetap mengikuti susunan kofaktor - kofaktornya.

**2.6 Matriks Adjoin**

Adjoin matriks merupakan transpose dari matriks kofaktor. Adjoin A dapat dinotasikan dengan Adj (A). Transpose merupakan pertukaran elemen pada baris menjadi kolom. Adjoin matriks digunakan dalam menentukan invers matriks. Berikut merupakan matriks A yang diubah menjadi Adj (A).

**2.7 Kaidah Cramer**

Kaidah Cramer adalah rumus yang dapat digunakan untuk menyelesaikan sistem persamaan linear. Metode ini menggunakan determinan suatu matriks dan matriks lain yang diperoleh dengan mengganti salah satu kolom dengan vektor yang terdiri dari angka di sebelah kanan persamaannya. Jika Ax = b adalah sebuah sistem linear *n* yang tidak diketahui dan det(A) tidak bernilai 0 maka persamaan tersebut mempunyai penyelesaian yang unik

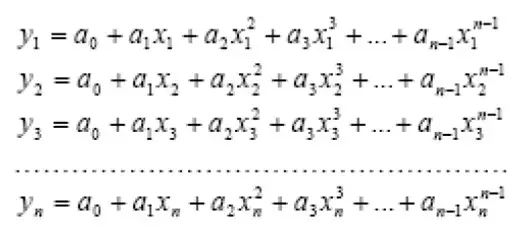


**2.8 Interpolasi Polinom**

Interpolasi polinomial digunakan untuk mencari titik-titik antara dari *n* buah titik , , , ..., dengan menggunakan pendekatan fungsi polinomial berpangkat n - 1:

****

Dengan memasukkan nilai dari setiap titik ke dalam persamaan polinomial di atas dan diperoleh persamaan simultan dengan *n* persamaan dan *n* variabel bebas:



**2.9 Regresi Linier Berganda**

Regresi Linier Berganda adalah model regresi linier atau prediksi dengan melibatkan lebih dari satu variabel bebas. Model regresi linear berganda dinotasikan dengan persamaan berikut

****

Dengan Y sebagai variabel terikat, X sebagai variabel bebas, sebagai konstanta, dan sebagai koefisien estimate.

**BAB III**

**IMPLEMENTASI PROGRAM DALAM JAVA**

**3.1 Implementasi Eliminasi Gauss dan Gauss Jordan (Gauss.java)**

1. Public static double[ ][ ] splGauss(double[ ][ ] arr)

Mengembalikan matriks eselon baris dengan menggunakan metode operasi baris elementer (OBE). Fungsi ini umumnya digunakan dengan parameter berupa matriks augmented.

1. Public static double[ ][ ] splGaussJordan(double[ ][ ] arr)

Mengembalikan matriks dalam bentuk matriks eselon baris tereduksi dengan menggunakan metode operasi baris elementer (OBE). Fungsi ini umumnya digunakan dengan parameter berupa matriks augmented. Fungsi ini menggunakan fungsi lain yaitu splGauss**.**

1. public static String[ ] solveSPLGauss(double[ ][ ] arr)

Mengembalikkan solusi dari sistem persamaan linear dengan metode Gauss. Solusi yang dikembalikan adalah dalam bentuk *array* *of String*.

1. public static String[ ] solveSPLGaussJordan(double[ ][ ] arr)

Mengembalikkan solusi dari sistem persamaan linear dengan metode GaussJordan. Solusi yang dikembalikan adalah dalam bentuk *array* *of String*.

**3.2 Implementasi Determinan (determinan.java)**

1. static void cofactor(double[ ][ ] M, int N, double[ ][ ], int a, int b)

Prosedur akan menyalin elemen matriks yang tidak terdapat pada baris ke-a dan kolom ke-b. Dengan menggunakan algoritma *for loop* elemen-elemen tersebut akan disalin ke dalam matriks baru berupa minor dari Mab yang nantinya akan dikalikan dengan (-1)a+b untuk membentuk kofaktor.

1. public static double determinanCofactorWay(double M[][], int N)

Fungsi akan mengembalikan nilai determinan yang dicari menggunakan cara kofaktor. Dengan memanfaatkan prosedur *cofactor* dan algoritma rekursif maka dapat dicari determinan dengan cara memecah matriks hingga hanya memiliki satu elemen dan dikembalikan lagi untuk dikalikan dengan elemen matriks dan tanda yang benar.

1. static tukerRow(double M[][], int N, int p, int q)

Prosedur akan menukar baris ke-p dengan baris ke-q. Tujuannya supaya dalam fungsi determinan menggunakan reduksi baris dapat menerapkan operasi baris elementer yaitu penukaran baris.

1. static int rowReduction(double M[][], int N)

Fungsi akan melakukan prosedur pada matriks M dan membuatnya menjadi matriks segitiga atas. Untuk melakukan ini maka harus diterapkan operasi baris elementer. Prosedur dilakukan dengan cara mengecek elemen pertama untuk setiap baris. Apabila elemen tersebut adalah nol, baris akan ditukar dengan baris lain yang elemen awalnya tidak nol. Untuk setiap penukaran baris akan menambah nilai *count.* Nilai ini yang nantinya akan di-return oleh fungsi. Apabila tidak ada penukaran, maka *loop*  akan dilanjutkan ke iterasi berikutnya karena artinya untuk kolom tersebut semua elemennya sudah nol. Untuk elemen pertama baris yang tidak nol maka akan dilakukan pembagian ke baris dibawahnya untuk dilakukan pengurangan terhadap pengalian dari elemen baris elemen pertama sehingga terbentuk matriks segitiga.

1. public static double determinanReductionRowWay(double M[][], int N)

Fungsi ini hanya tinggal memanfaatkan kedua fungsi sebelumnya. Oleh karena itu, dengan sudah terbentuknya matriks segitiga atas maka perhitungan determinan dapat dilakukan dengan mengalikan setiap elemen pada diagonal utama. Selain itu, hasil determinan juga akan dikalikan dengan -1 pangkat seberapa banyak pertukaran baris dilakukan sehingga terbentuklah matriks dengan metode reduksi baris.

**3.3 Implementasi Matriks (Matriks.java)**

1. public static double[ ][ ] bacaMatriks(int brs, int kol)

Fungsi ini menerima parameter berupa jumlah baris dan kolom dari matriks yang ingin di input oleh pengguna. Program ini akan menerima input pengguna berupa matriks. Fungsi ini akan mengembalikkan matriks dalam tipe data *matrix of double*.

1. public static void tulisMatriks(double[ ][ ] arr)

Prosedur ini menuliskan matriks ke layar.

1. public static double[ ][ ] transposeMatriks(double[ ][ ] arr)

Mengembalikan matriks yang ada pada parameter dalam keadaan sudah di transpose. Cara kerja nya pun dengan membuat matriks baru yang elemen - elemennya diisi secara “pantulan” dari matriks pada parameter fungsi ini. Yang dimaksud dengan “pantulan” adalah matriks yang sudah di transpose sedemikian rupa sehingga elemen pada baris *i* dan kolom *j* itu sama dengan elemen pada matriks yang ada di parameter pada baris *j* dan kolom *i.*

1. public static double[ ][ ] kaliMatriks(double[ ][ ] m1, double[ ][ ] m2)

Mengembalikan matriks hasil perkalian matriks m1 dan m2. Fungsi ini memiliki prekondisi jumlah kolom pada matriks m1 itu sama dengan jumlah baris pada matriks m2.

1. public static double[ ][ ] matriksKofaktor(double[ ][ ] arr)

Mengembalikan matriks kofaktor dari matriks pada parameter fungsi ini.

1. public static double[ ][ ] matriksAdjoint(double[ ][ ] arr)

Mengembalikan matriks adjoint dari sebuah matriks. Matriks adjoint merupakan hasil transpose dari matriks kofaktor.

**3.4 Implementasi Inverse Matriks (Inverse.java)**

1. public static double[ ][ ] inverseMatriks(double[ ][ ] arr)

Mengembalikan matriks yang sudah di-*inverse.* Cara kerjanya adalah dengan membagi semua elemen dari matriks adjoint dengan determinan matriks.

1. public static double[ ][ ] solveSPLInverse(double[ ][ ] arr)

Mengembalikan solusi dari sistem persamaan linear dalam tipe data *matrix* *of double* dengan besar kolomnya sama dengan 1. Proses penyelesaian menggunakan kaidah matriks balikan.

**3.6 Implementasi Kaidah Cramer**

1. public static double[] splCramer(double X[][],int N )

Dengan memanfaatkan *method*  dalam *class* determinan maka dapat dicari determinan dari matriks sistem persamaan linear X. *Input* matriks sudah diasumsikan pasti berbentuk *nxn.* Lalu, dengan menggunakan fungsi gantiX dan *for loop* dapat dicari determinan untuk masing-masing substitusi kolom dan dicari solusinya dengan membagi determinan matriks substitusi dengan matriks awal. Solusi disimpan dalam sebuah array berdimensi satu. Fungsi akan mengembalikan nilai array tersebut.

1. static double[][] gantiX(X[][], double B[][], int N, int j)

Fungsi bertujuan untuk membuat matriks sementara baru yang merupakan substitusi matriks hasil B ke dalam kolom j dari matriks sistem persamaan linear X.

**3.7 Implementasi Interpolasi Polinom**

1. static double[][] inputXY(int N )

Fungsi akan mengembalikan matriks yang sudah di-*input*  elemen-elemennya. Matriks masukan berupa matriks augmented.

1. static double[][] buatPolinomX(double XY[][], int N)

Fungsi akan memecah matriks masukan menjadi komponen x-nya saja dan juga mengubahnya menjadi matriks polinomial, Pn(x).

1. static double[][] buatPolinomY(double XY[][], int N)

Fungsi akan mengembalikan matriks hasil dari polimonial komponen x yang sudah dibuat pada fungsi buatPolinomX.

1. public static double[] interpol(double XY[][], int N)

Fungsi akan mengembalikan solusi dari sistem persamaan linear yang terbentuk dari dua fungsi sebelumnya dengan menggunakan metode *cramer.* Solusi dimasukkan ke dalam sebuah *array* satu dimensi.

1. static double hitungPerkiraan(double a[], int N, double X)

Fungsi akan memberikan taksiran nilai X *input* dari pengguna dari fungsi polinomial yang didapat dari hasil interpolasi.

1. static int idxFirst(double Solusi[], int N)

Fungsi akan mengembalikan indeks pertama dari *array* solusi yang nantinya akan digunakan untuk memudahkan dalam mencetak fungsi yang didapat dari hasil interpolasi.

1. public static void cetakFungsi(double Solusi[], int N)

Prosedur akan mencetak fungsi yang didapat dari hasil interpolasi polinomial.

**3.8 Implementasi Regresi Linier Berganda**

1. public static double[ ][ ] regLinBerganda(double[ ][ ] arr)

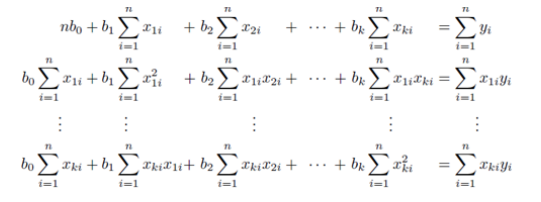
Mengembalikan nilai nilai variabel B dari persamaan dibawah ini.



Nilai nilai variabel B tersebut dikembalikan dalam tipe data *matrix of double*, dengan banyaknya kolom adalah 1. Salah satu alasan mengapa tipe data untuk data yang dikembalikan fungsi ini adalah *matrix* adalah karena proses penyelesaian untuk menemukan nilai nilai variabel B adalah melalui sistem persamaan linear (SPL) yang menggunakan metode - metode penyelesaian yang berhubungan dengan *matrix*. Sehingga hasil akhirnya pun juga *matrix*. Tahapan cara kerja dari fungsi ini adalah sebagai berikut.

Tahapan pertama adalah membagi input *matrix* dari fungsi ini menjadi 2 *matrix* yang berbeda, yang satu berisi *value* dari setiap variabel peubah, sedangkan yang satunya lagi berisi *value* dari hasil.

Lalu selanjutnya adalah membuat 2 *matrix* baru, sebut saja *A* dan *B*, yang kemudian akan diisi dengan data sebagai berikut.



*Matrix* *A* berisi segala yang ada di sebelah kiri tanda “=” pada gambar diatas, sedangkan *matrix B* berisi segala sesuatu yang ada di sebelah kanan tanda “=”. Proses pengisian menggunakan konsep *loop*.

Tahap selanjutnya adalah menggabungkan *matrix A* dan *B* agar menjadi *augmented matrix*, yang dapat digunakan pada fungsi penyelesaian SPL yang sudah dibuat terlebih dahulu.

**3.9 Implementasi Manipulasi File (MatriksFile.java)**

Pada implementasi ini digunakan beberapa *Class* bawaan *Java* yang krusial untuk keberjalanan program. *Class*nya adalah sebagai berikut.

1. java.io.File

*Class* ini adalah representasi yang terdapat di *Java* dari sebuah *file* atau *directory*, karena pada dasarnya *file* dan *directory* memiliki format penamaan yang berbeda - beda tergantung pada *platform* dan *operating system* yang digunakan sehingga diperlukan representasi yang seragam sehingga program lebih fleksibel dalam bidang *platform* atau *operating system* yang dapat menjalankannya. Ada beberapa metode yang dimiliki *Class* ini, tetapi pada *project* ini sebagian besar *Class* ini digunakan untuk membuat objek baru bertipe *File* untuk keperluan pembacaan data dari *file* ataupun menyimpan hasil perhitungan ke *file*.

1. java.io.FileNotFoundException dan java.io.IOException

*Class* ini secara garis besar digunakan untuk *error handling* pada saat pembuatan, penyimpanan, dan pembacaan *file*.

1. java.io.FileWriter

*Class* ini digunakan untuk menuliskan data ke *file* dalam bentuk *character*.

Ada beberapa fungsi dan prosedur yang ada pada implementasi adalah sebagai berikut.

1. Public static String currDir( )

Mengembalikan alamat dari *directory* yang akan digunakan untuk menyimpan *file - file* yang merupakan salah satu output dari program utama.

1. public static int[ ] ukuranMatriksFile(String filename)

Mengembalikan ukuran matriks yang terdapat dalam file yang dibaca, output dikeluarkan dalam bentuk *array* yang sudah pasti formatnya. Formatnya adalah jumlah baris diletakkan pada array indeks “0”, dan jumlah kolom diletakkan pada array indeks “1”. Proses perhitungan ukuran matriks, jumlah baris dan kolom menggunakan konsep *loop*.

1. Public static double[ ][ ] fileToMatriks(String filename)

Mengembalikkan sebuah matriks yang elemen - elemennya adalah hasil pembacaan dari sebuah *file*. Cara yang digunakan untuk menentukan ukurannya adalah dengan menggunakan fungsi ukuranMatriksFile.

1. public static int menuInputMatriks()

Mengembalikkan pilihan pengguna dari menu input. Fungsi akan menampilkan juga menu input yang dapat dipilih oleh pengguna, akan terus mengulang sampai pengguna memasukkan pilihan yang valid, yaitu yang terdapat pada menu input yang ditampilkan.

1. public static boolean fileExist(String filename)

Mengembalikkan boolean yang menunjukan keberadaan suatu *file*. Fungsi akan mengembalikkan t*rue* jika *file* dengan nama *filename* (parameter fungsi) ini ada, mengembalikkan *false* jika tidak ada.

1. public static String inputFileName( )

Mengembalikkan nama *file* dari input pengguna. Fungsi ini juga akan menampilkan menu peng-input-an nama *file* dari pengguna, akan terus mengulang sampai pengguna memasukkan nama *file* yang ada. Proses validasi keberadaan nama *file*  yang dimasukkan pengguna adalah menggunakan fungsi fileExist.

1. public static boolean saveToFile( )

Fungsi ini akan mengembalikkan *boolean*, *true* jika pengguna ingin menyimpan hasil perhitungan ke sebuah *file*, *false* jika tidak. Fungsi ini juga menampilkan menu pilihan penyimpanan file, dan akan mengulang terus sampai pengguna memasukkan pilihan yang valid.

1. public static String createFile( )

Fungsi ini akan mengembalikkan nama *file* yang diinput pengguna. Fungsi ini juga menampilkan menu peng-input-an nama *file*. Akan terus mengulang jika nama *file* yang dimasukkan ternyata sudah ada.

1. public static void matriksToFile(String filename, double[ ][ ] arr)

Prosedur ini digunakan untuk menuliskan matriks kedalam *file*.

1. public static void splCramerToFile(String filename, double[ ][ ] arr)

Prosedur ini digunakan untuk menuliskan solusi sistem persamaan linear yang menggunakan metode Cramer ke dalam *file*.

1. public static void splInverseToFile(String filename, double[ ][ ] arr)

Prosedur ini digunakan untuk menuliskan solusi sistem persamaan linear yang menggunakan metode matriks balikan ke dalam *file*.

1. public static void determinanToFile(String filename, double detEK, double detRB)

Prosedur ini digunakan untuk menuliskan solusi determinan yang menggunakan metode ekspansi kofaktor dan reduksi baris ke dalam *file*.

1. public static void regLinToFile(String filename, double[ ][ ] b, double hasil)

Prosedur ini digunakan untuk menuliskan solusi persamaan linear dan hasil taksiran regresi linear berganda ke dalam *file*.

1. public static void interpolToFile(String filename, double hasil)

Prosedur ini digunakan untuk menuliskan solusi persamaan dan hasil interpolasi polinom ke dalam *file*.

1. public static void splGaussToFile(String filename, double hasil)

Prosedur ini digunakan untuk menuliskan solusi untuk sistem persamaan linear dengan metode Gauss ataupun GaussJordan ke dalam *file*.

**3.10 Implementasi Program Utama**

Pada implementasi program utama, terdapat beberapa metode untuk membantu keberjalanan program, sebagai berikut.

1. public static void menu( )

Menampilkan menu fitur utama.

1. public static void submenu1( )

Menampilkan submenu pada bagian sistem persamaan linear.

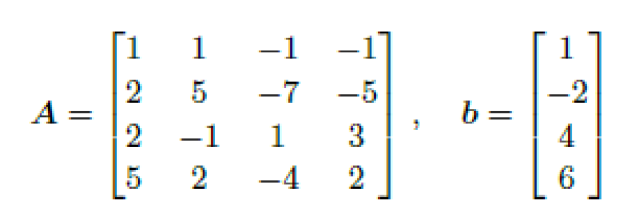
Cara kerja dari program utama ini adalah dengan menggunakan beberapa tahapan yang diulang terus menerus sampai pengguna memasukkan pilihan “keluar” pada menu fitur utama. Tahapan - tahapannya adalah sebagai berikut. Pertama - tama program akan menampilkan menu fitur utama yang kemudian pengguna akan memasukkan pilihannya pada menu tersebut. Lalu program akan menampilkan menu input, jika pengguna memilih untuk menginput menggunakan *file* maka pengguna akan memasukkan nama *file*. Jika pengguna memilih untuk menginput dengan *keyboard* maka program akan meminta beberapa data dan meminta pengguna untuk memasukkan data dalam bentuk matriks. Setelah selesai melakukan perhitungan program akan menampilkan menu input file, dimana pengguna dapat memilih untuk menyimpan hasil perhitungan ke *file* atau tidak.

**BAB IV**

**EKSPERIMEN**

**4.1 Solusi SPL Ax = b**

1. Soal



Hasil eksekusi program :

Matriks dibaca lewat file maupun dimasukkan melalui keyboard.

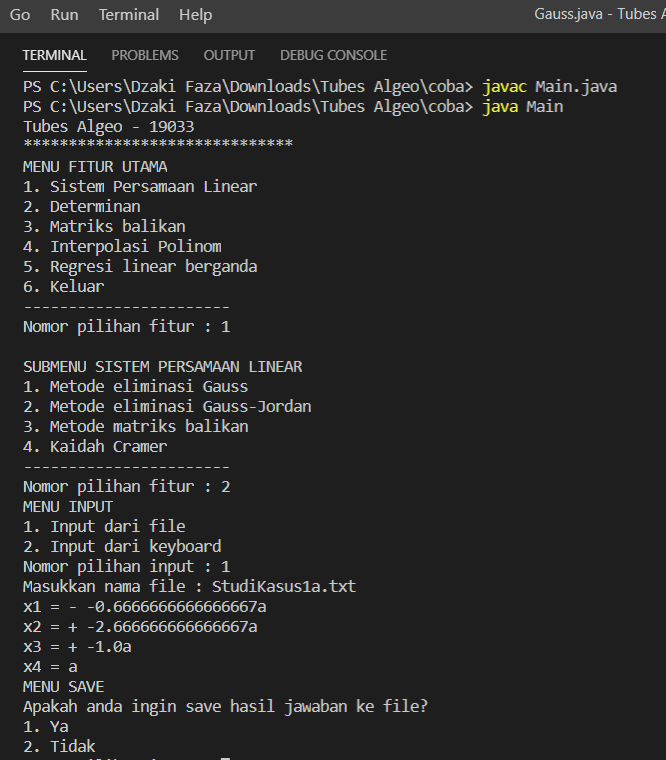
Hasil dari metode yang telah dibuat :

1. Metode Eliminasi Gauss



**Gambar 4.1.1 Eksekusi Program SPL Metode Eliminasi Gauss**

1. Metode Eliminasi Gauss-Jordan



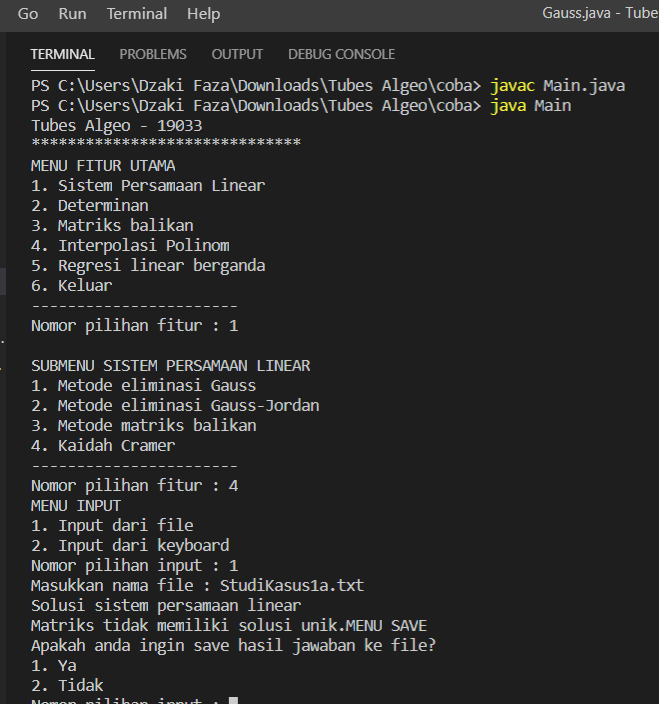
**Gambar 4.1.2 Eksekusi Program SPL Metode Eliminasi Gauss-Jordan**

1. Metode Matriks Balikan



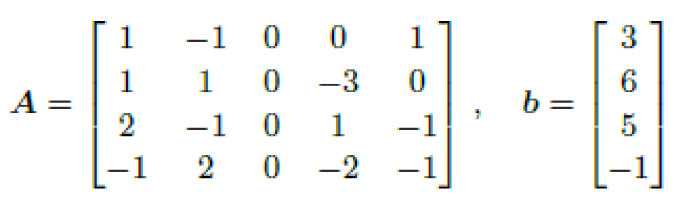
**Gambar 4.1.3 Eksekusi Program SPL Metode Matriks Balikan**

1. Kaidah Cramer



**Gambar 4.1.4 Eksekusi Program SPL Metode Cramer**

1. Soal

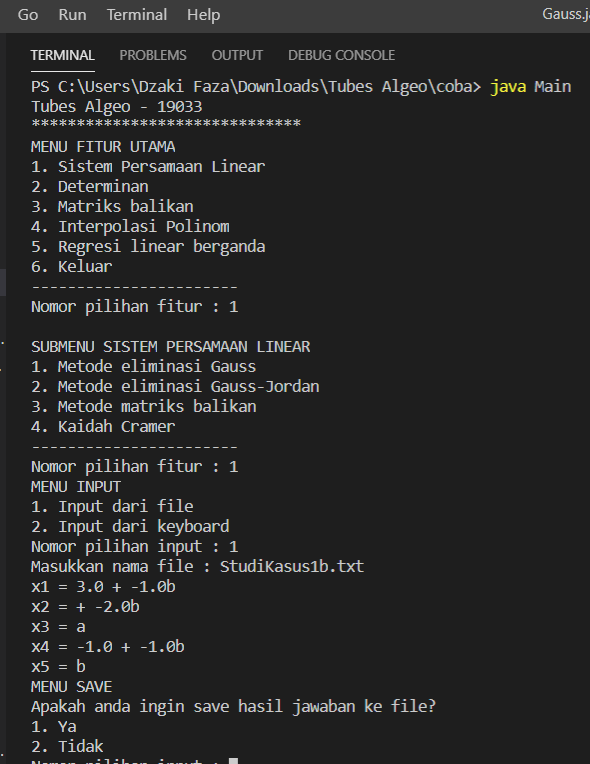


Hasil eksekusi program :

Matriks dibaca lewat file maupun dimasukkan melalui keyboard.

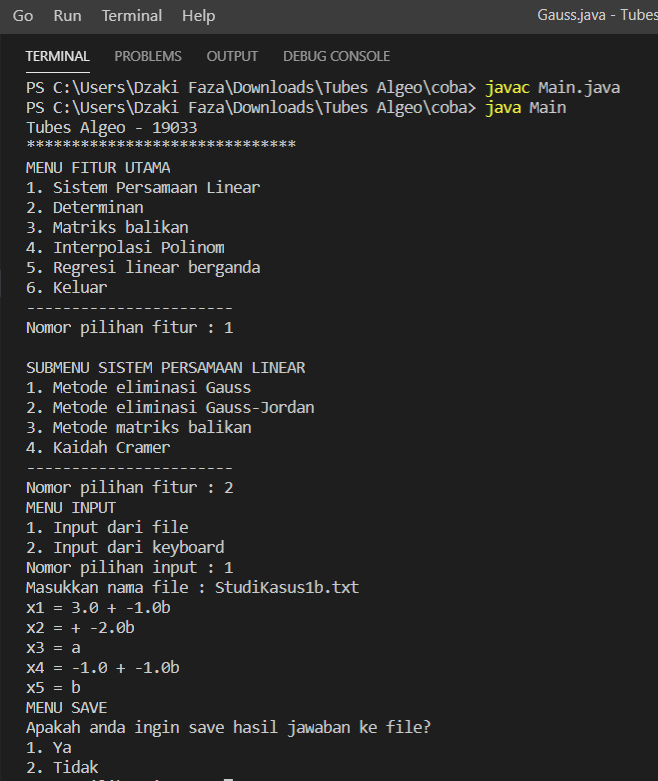
Hasil dari metode yang telah dibuat :

1. Metode Eliminasi Gauss



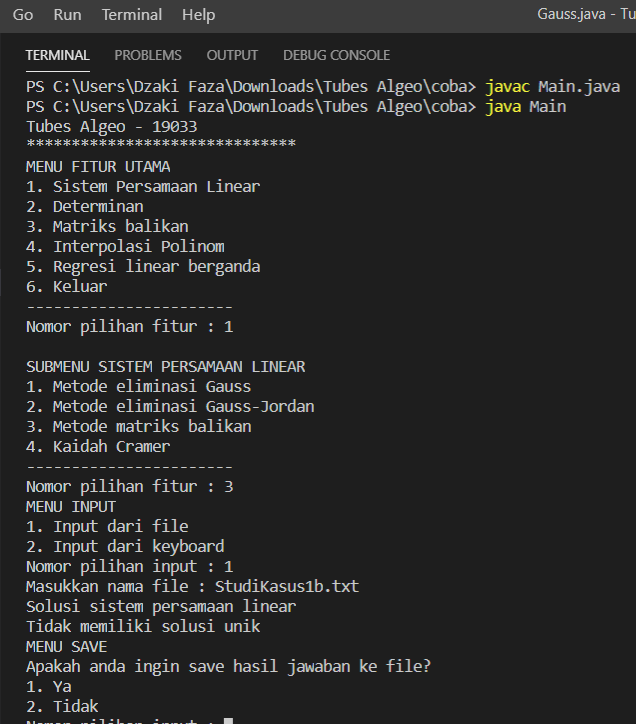
**Gambar 4.1.5 Eksekusi Program SPL Metode Eliminasi Gauss**

1. Metode Eliminasi Gauss-Jordan



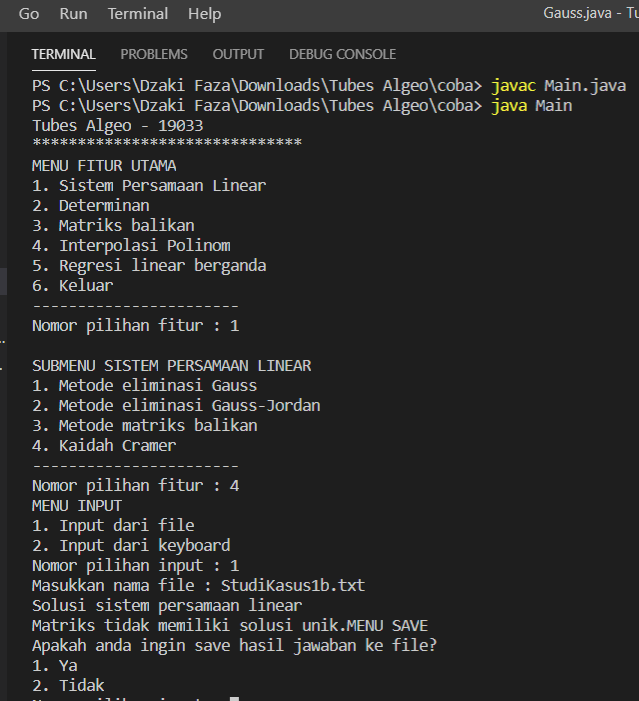
**Gambar 4.1.6 Eksekusi Program SPL Metode Eliminasi Gauss-Jordan**

1. Metode Matriks Balikan



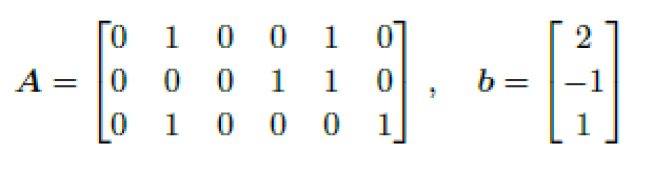
**Gambar 4.1.7 Eksekusi Program SPL Metode Matriks Balikan**

1. Kaidah Cramer



**Gambar 4.1.8 Eksekusi Program SPL Metode Cramer**

1. Soal

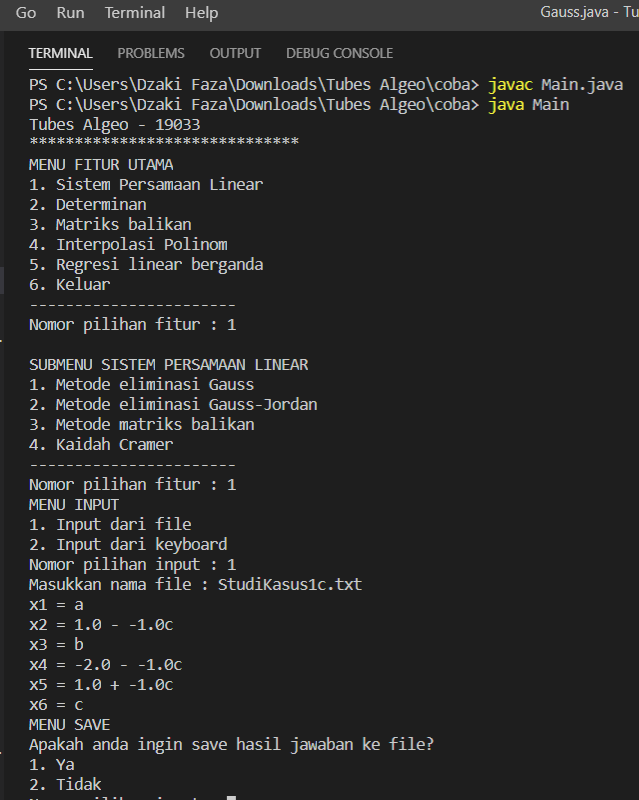


Hasil eksekusi program :

Matriks dibaca lewat file maupun dimasukkan melalui keyboard.

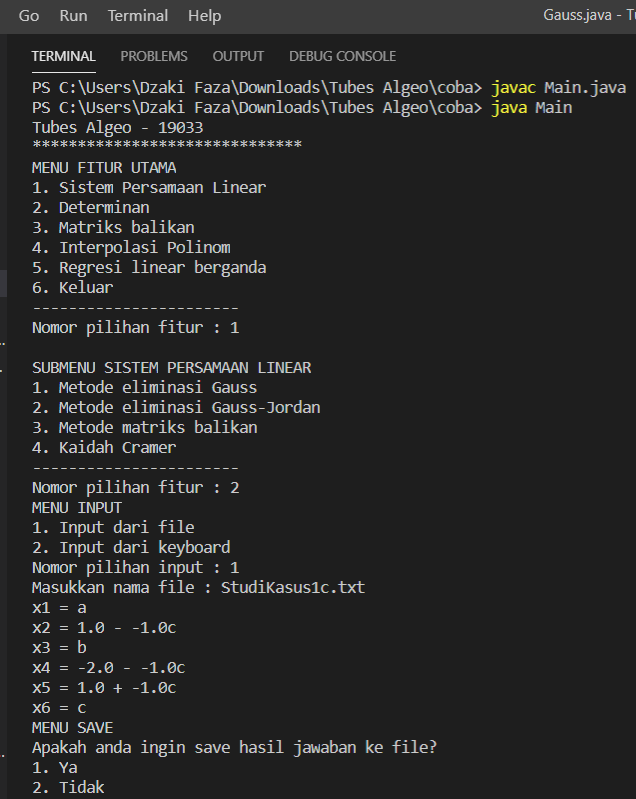
Hasil dari metode yang telah dibuat :

1. Metode Eliminasi Gauss



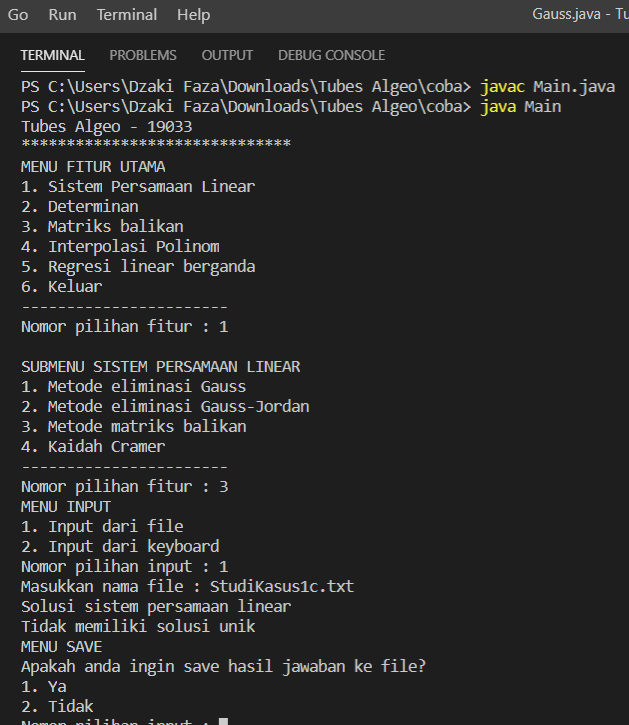
**Gambar 4.1.9 Eksekusi Program SPL Eliminasi Gauss**

1. Metode Eliminasi Gauss-Jordan



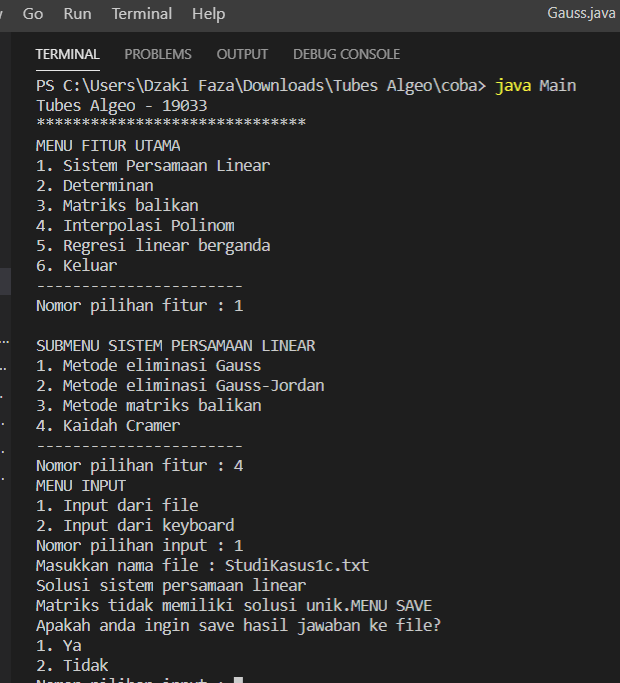
**Gambar 4.1.10 Eksekusi Program SPL Eliminasi Gauss-Jordan**

1. Metode Matriks Balikan



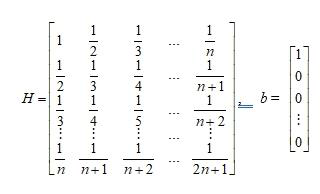
**Gambar 4.1.11 Eksekusi Program SPL Metode Eliminasi**

1. Kaidah Cramer



**Gambar 4.1.12 Eksekusi Program SPL Metode Cramer**

1. Soal



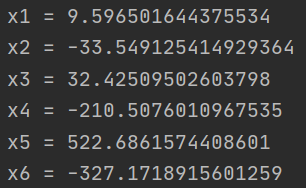
Hasil eksekusi program :

Matriks dibaca lewat file maupun dimasukkan melalui keyboard.

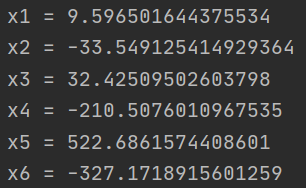
n = 6

Hasil dari metode yang telah dibuat :

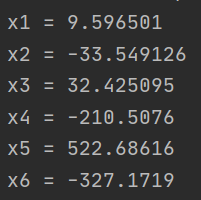
1. Metode Eliminasi Gauss



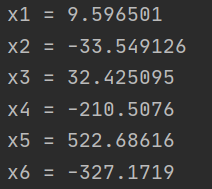
1. Metode Eliminasi Gauss-Jordan



1. Metode Matriks Balikan

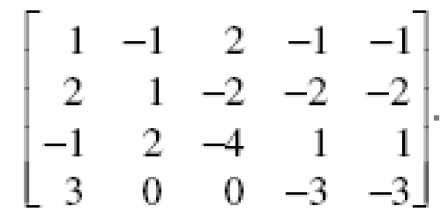


1. Kaidah Cramer



**4.2 SPL Berbentuk Matriks *Augmented***

1. Soal

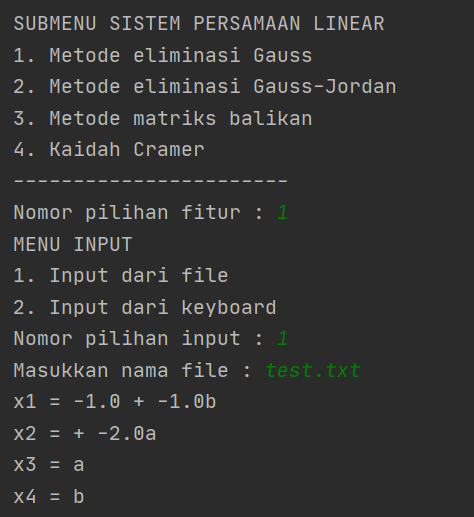


Hasil eksekusi program :

Matriks dibaca lewat file maupun dimasukkan melalui keyboard.

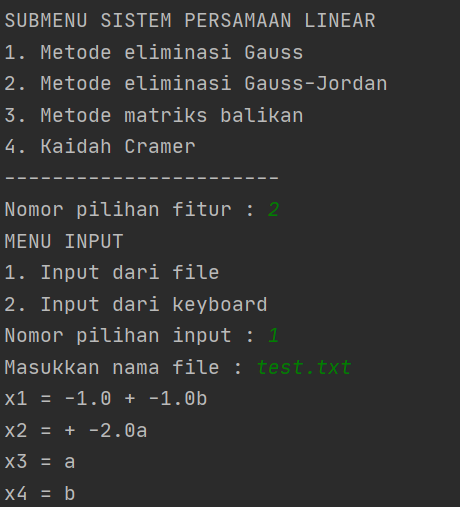
Hasil dari metode yang telah dibuat :

1. Metode Eliminasi Gauss



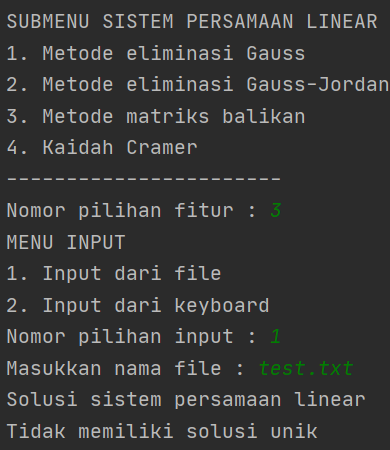
**Gambar 4.2.11 Eksekusi Program SPL Metode Eliminasi Gauss**

1. Metode Eliminasi Gauss- Jordan



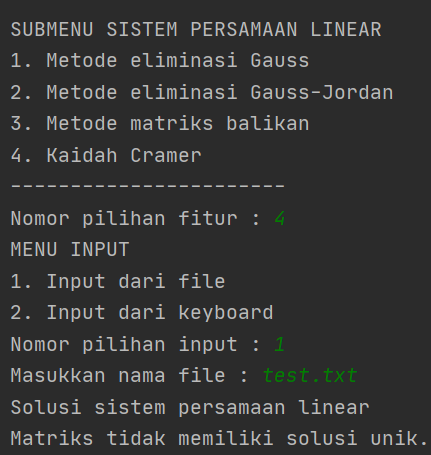
**Gambar 4.2.12 Eksekusi Program SPL Metode Eliminasi Gauss Jordan**

1. Metode Matriks Balikan



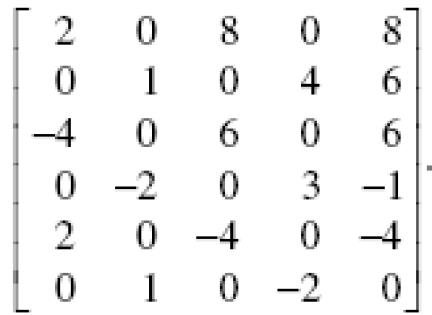
**Gambar 4.2.13 Eksekusi Program SPL Metode Matriks Balikan**

1. Kaidah Cramer



**Gambar 4.2.14 Eksekusi Program SPL Metode Cramer**

1. Soal

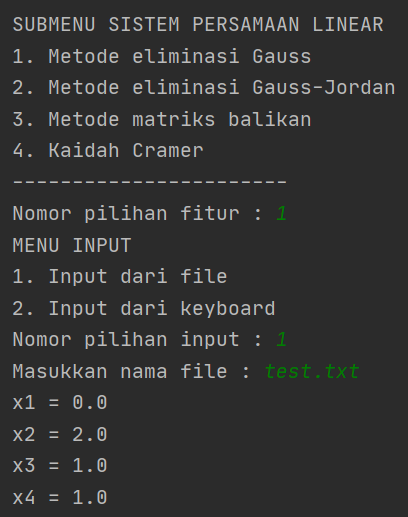


Hasil eksekusi program :

Matriks dibaca lewat file maupun dimasukkan melalui keyboard.

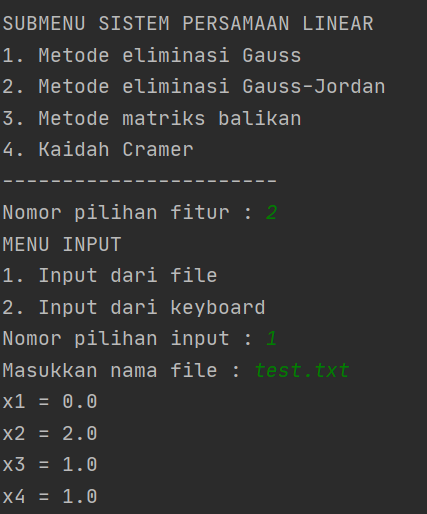
Hasil dari metode yang telah dibuat :

1. Metode Eliminasi Gauss



**Gambar 4.2.21 Eksekusi Program SPL Metode Eliminasi Gauss**

1. Metode Eliminasi Gauss- Jordan



**Gambar 4.2.22 Eksekusi Program SPL Metode Eliminasi Gauss Jordan**

**4.3 SPL**

1. Soal

8+ + 3 + 2 = 0

2+ 9 - - 2 = 1

+ 3 + 2 - = 2

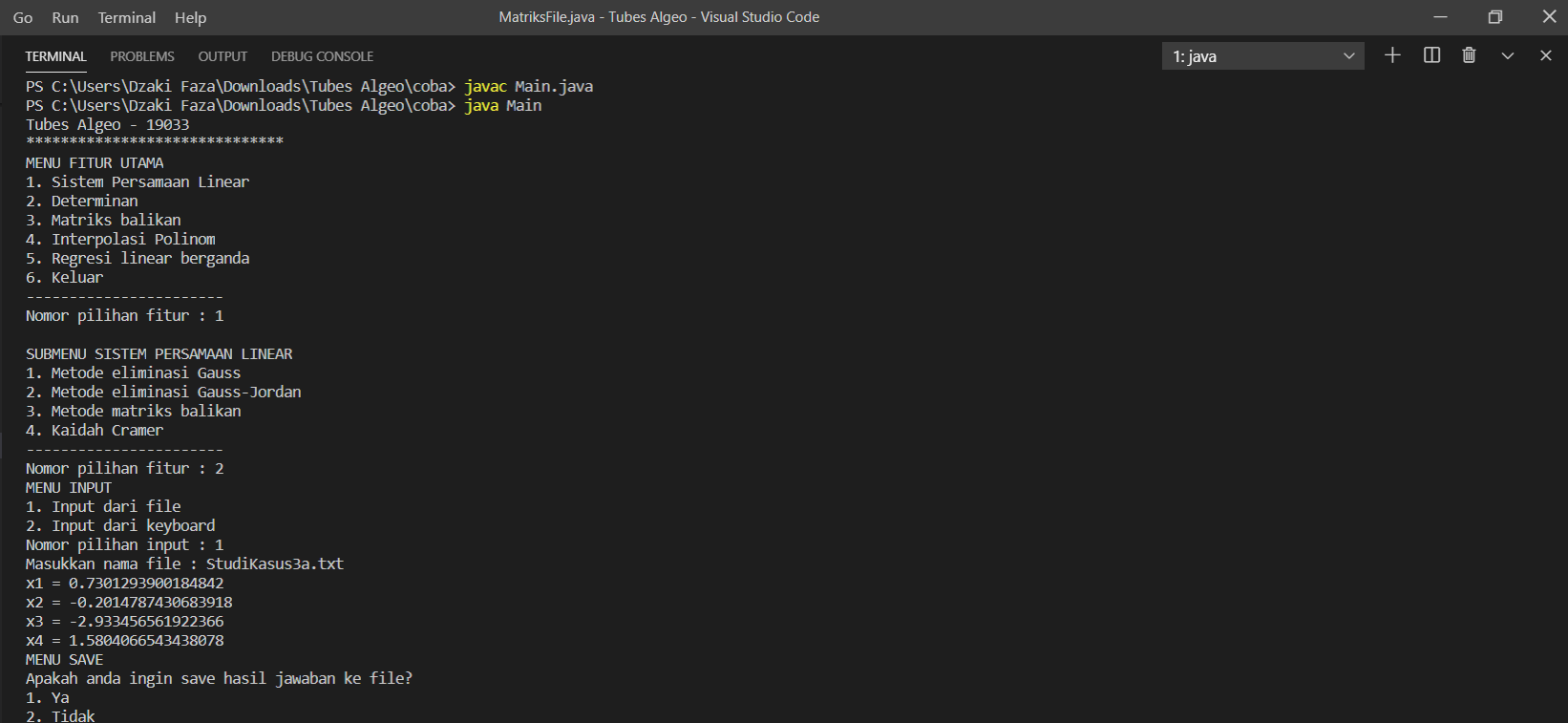
+ 6 + 4 = 3

Hasil eksekusi program

Matriks dibaca lewat file maupun dimasukkan melalui keyboard.

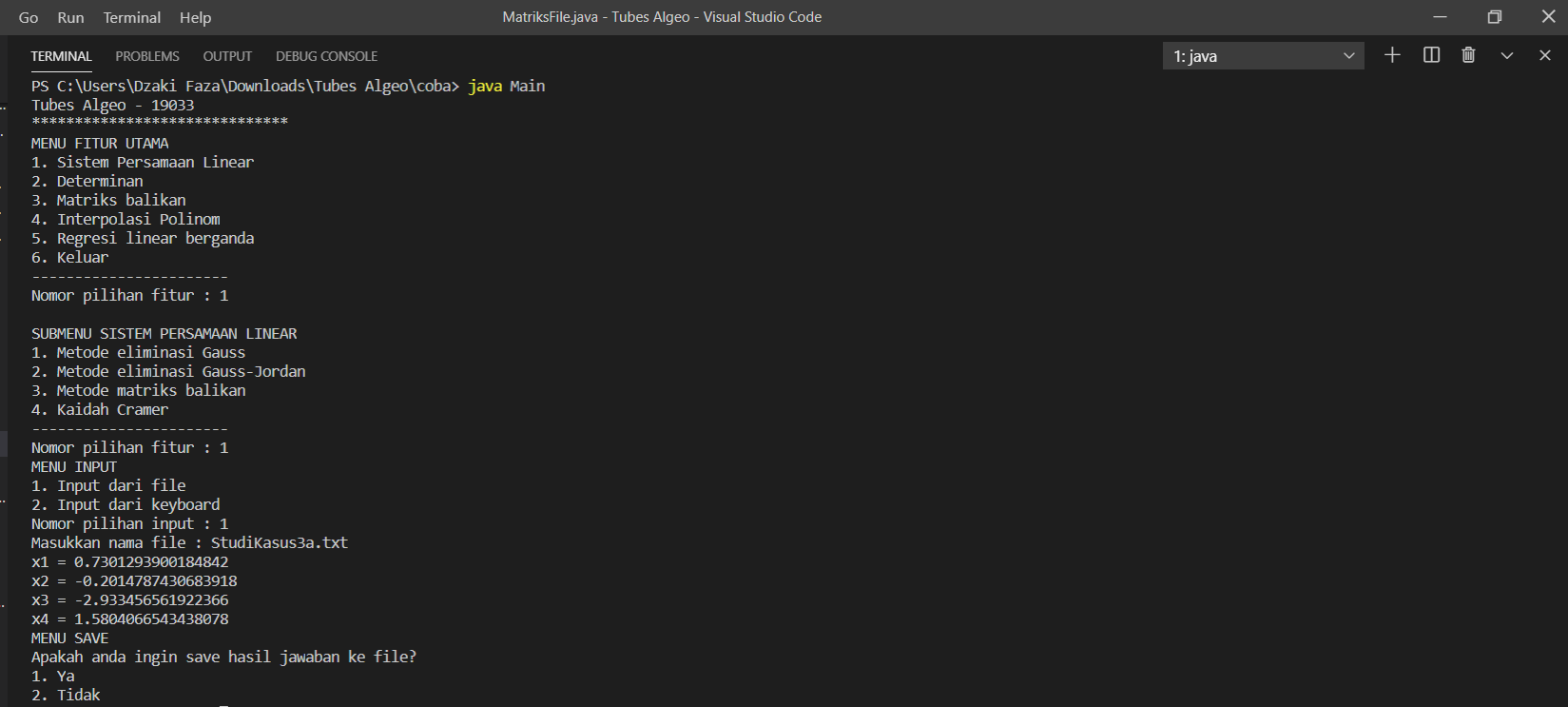
Hasil dari metode yang telah dibuat :

1. Metode Eliminasi Gauss



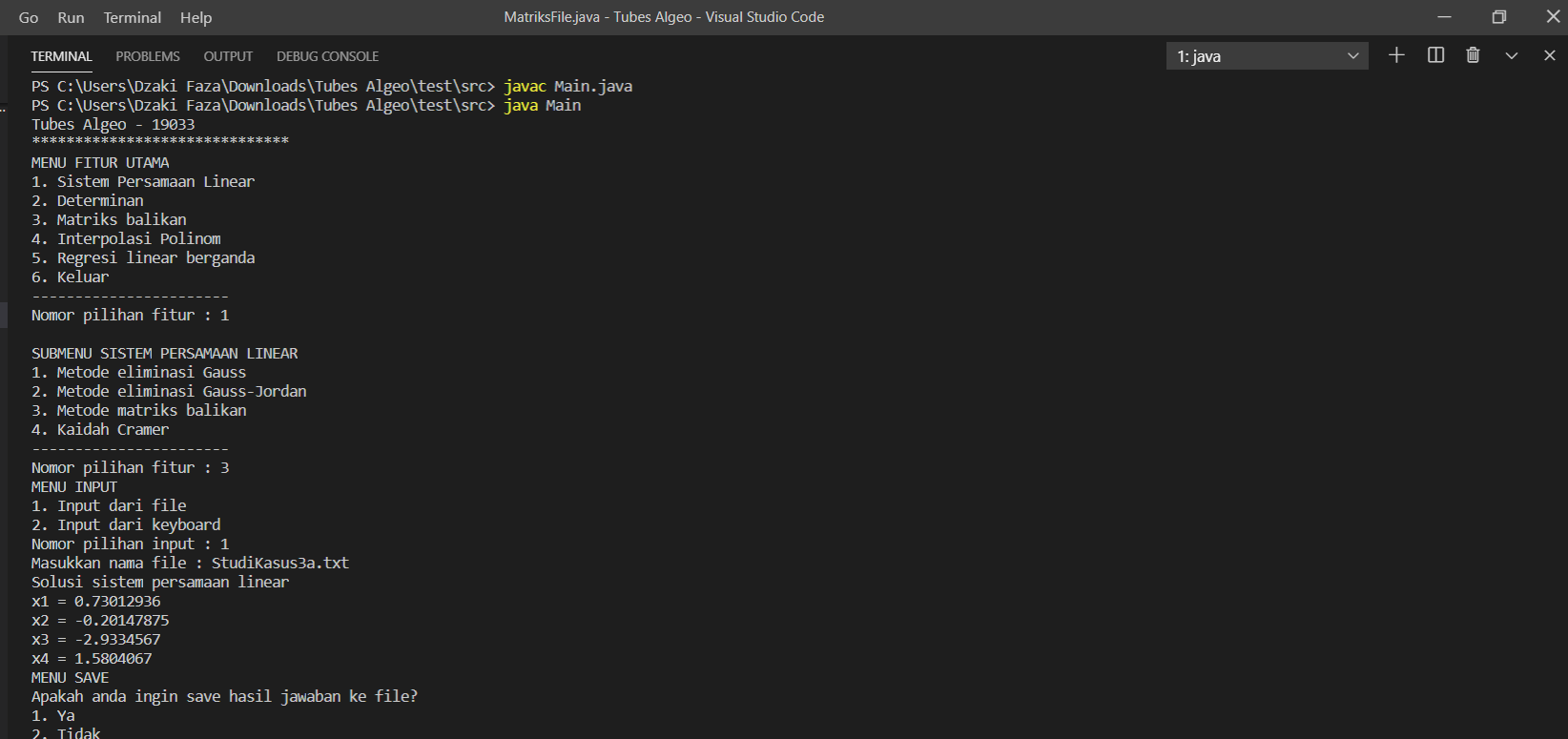
**Gambar 4.3.1 Eksekusi Program SPL Metode Eliminasi Gauss**

1. Metode Eliminasi Gauss- Jordan

****

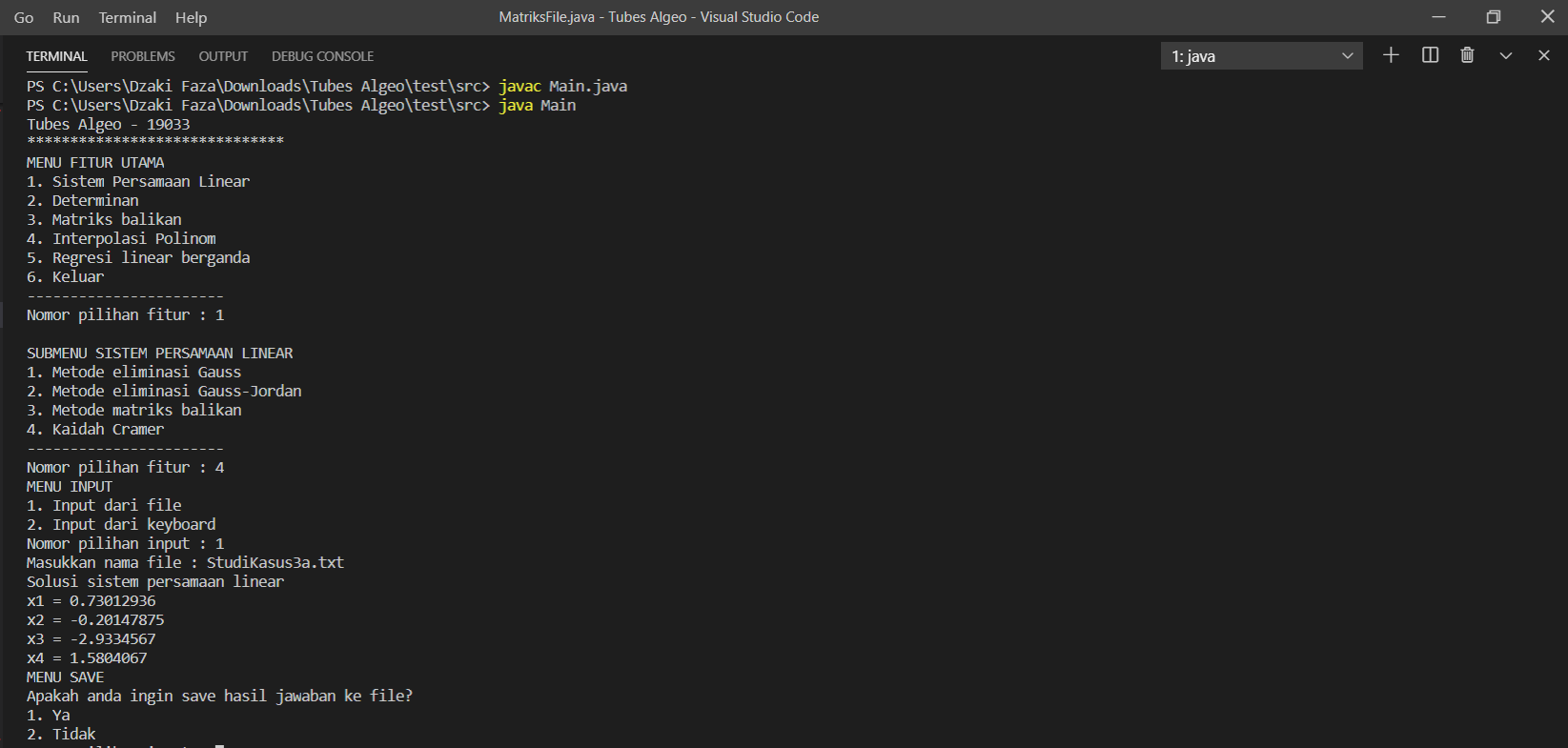
**Gambar 4.3.2 Eksekusi Program SPL Metode Eliminasi Gauss-Jordan**

1. Metode Matriks Balikan

****

**Gambar 4.3.3 Eksekusi Program SPL Metode Matriks Balikan**

1. Kaidah Cramer



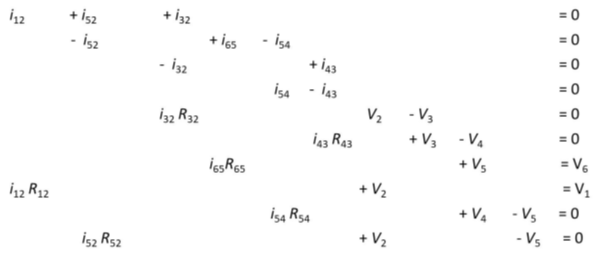
**Gambar 4.3.4 Eksekusi Program SPL Metode Cramer**

1. Soal

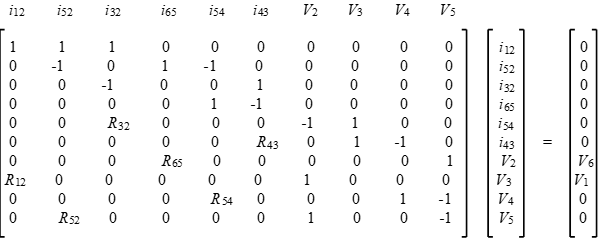


**4.4 Sistem Persamaan Linear Pada Arus Rangkaian**

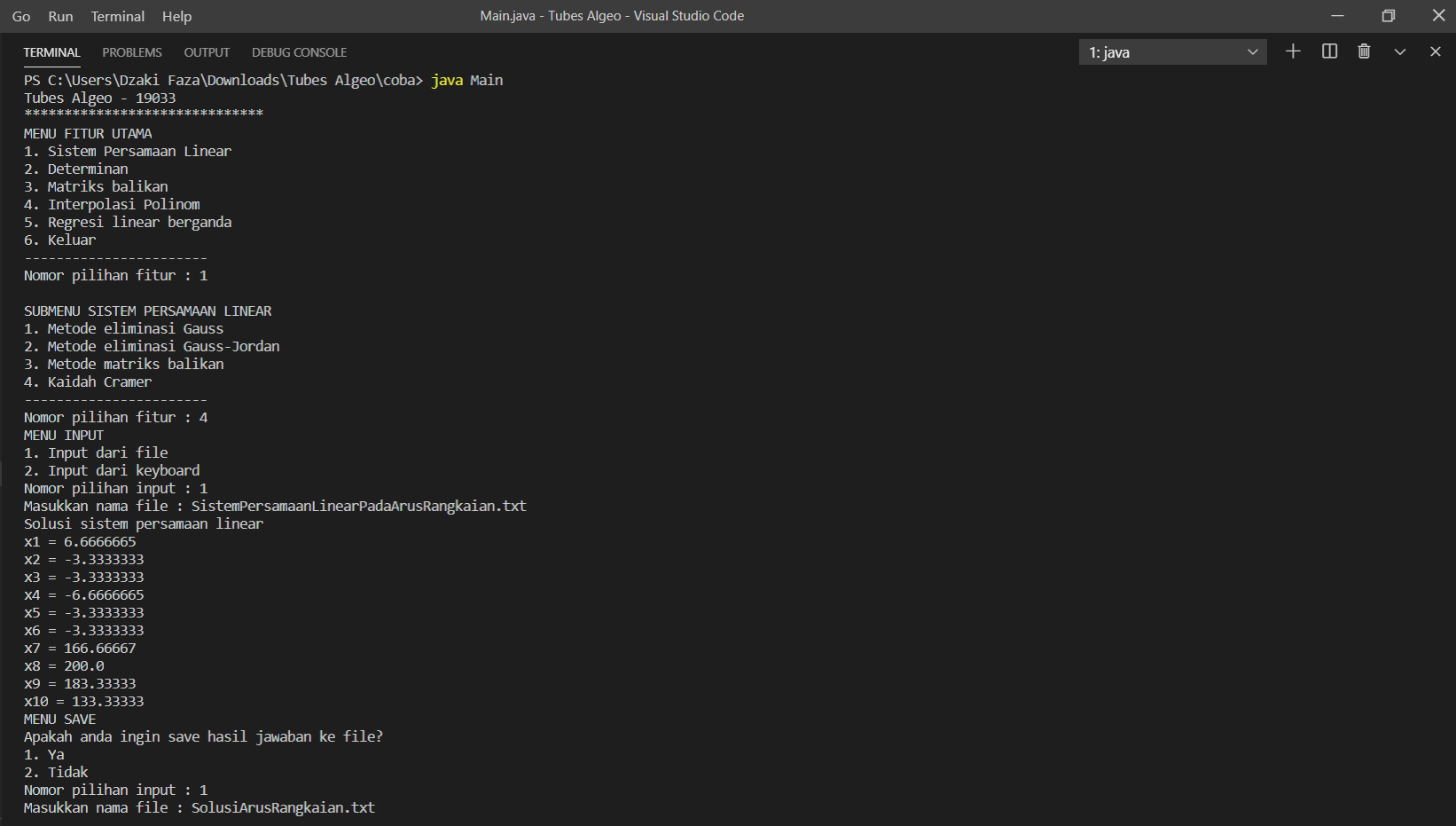
Diketahui sistem persamaan linear yang didapat dari persamaan hukum kirchoff dan hukum ohm sebagai berikut:



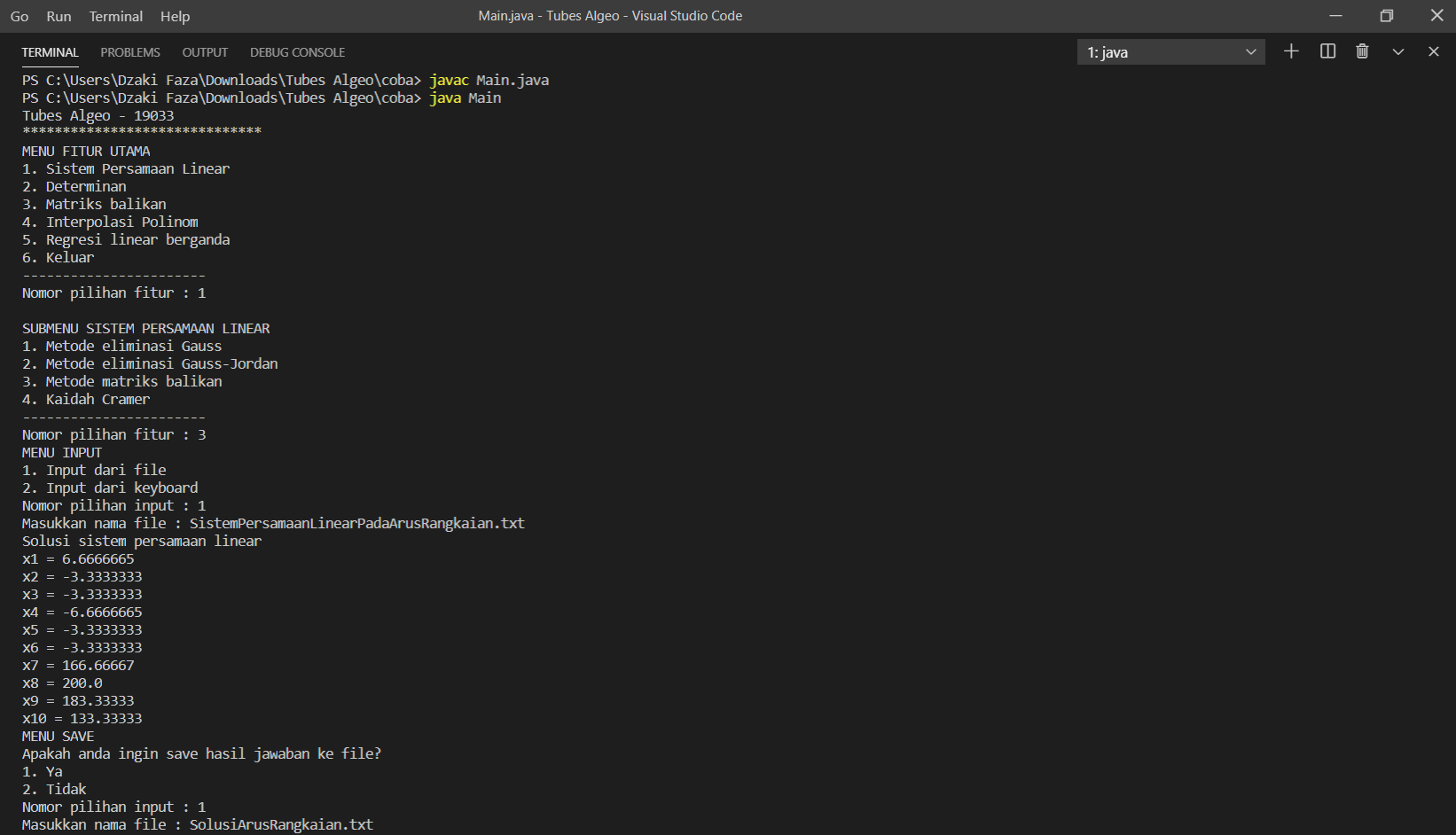
Dengan R12 = 5 ohm; R52 = 10 ohm; R32 = 10 ohm; R65 = 20 ohm; R54 = 15 ohm; R34 = 5 ohm; V1 = 200 volt; V6 = 0 volt. Supaya lebih mudah memahami bentuk sistem persamaan linear tersebut dapat diubah menjadi sebagai berikut:



Untuk menyelesaikan persamaan tersebut maka matriks akan dimasukkan dalam program untuk dihitung masing-masing nilai arus(I) dan tegangan(V) yang belum diketahui. Input dalam program berupa matriks augmented yang nantinya akan diolah sesuai dengan metode penyelesaian yang diinginkan. Berikut adalah hasil eksekusi program:



**Gambar 4.4.1 Eksekusi Program SPL Pada Arus Rangkaian Solusi Cramer**

****

**Gambar 4.4.2 Eksekusi Program SPL Pada Arus Rangkaian Solusi Matriks Balikan**

Dari keempat cara penyelesaian didapat nilai:

1. x1 = i12 = 6,6666665 A
2. x2 = i52 = -3,3333333 A
3. x3 = i32 = -3,3333333 A
4. x4 = i65 = -6,6666665 A
5. x5 = i54 = -3,3333333 A
6. x6 = i43 = -3,3333333 A
7. x7 = V2 = 166,66667 volt
8. x8 = V3 = 200,0 volt
9. x9 = V4 = 183,33333 volt
10. x10 = V5 = 133,33333 volt

Dari keempat metode penyelesaian ditemukan hasil yang sama sehingga dapat dinyatakan bahwa program berhasil.

**4.5 Interpolasi Pada Fungsi**

Diketahui nilai beberapa titik x dari sebuah fungsi sebagai berikut:

**Tabel 4.5 Data Nilai Fungsi**

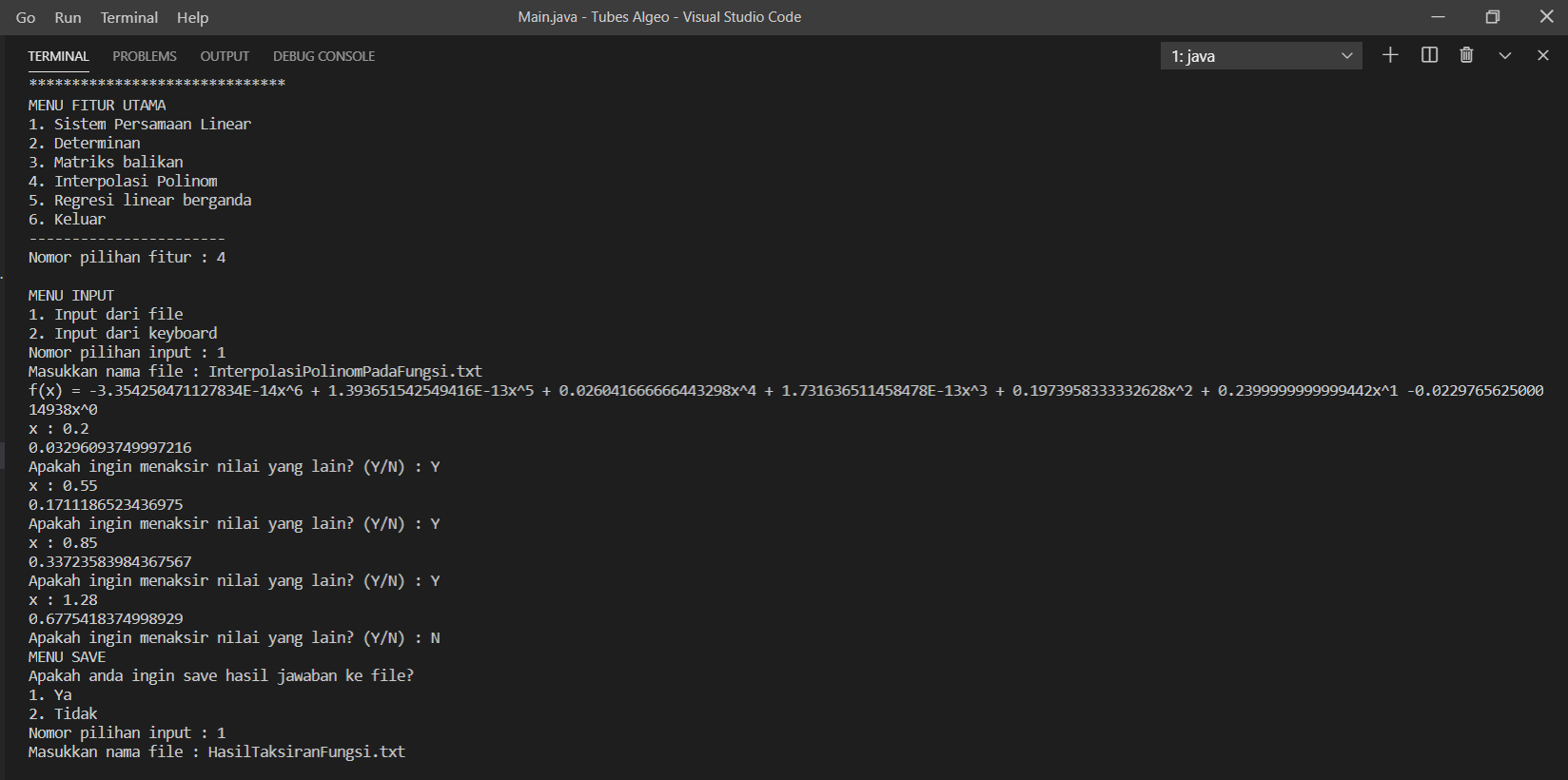
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 | 1.1 | 1.3 |
| f(x) | 0.003 | 0.067 | 0.148 | 0.248 | 0.370 | 0.518 | 0.697 |

Dengan menggunakan interpolasi polinomial maka dapat dibuat sebuah fungsi polinomial baru yang nilainya mendekati f(x). Input dalam program berupa matriks Nx2 dengan n adalah banyaknya titik yang diketahui nilainya, dalam kasus ini N = 7. Oleh karena itu, f(x) akan didekati dengan polinomial pangkat tujuh, P7(x).

|  |  |
| --- | --- |
| 0.1 | 0.003 |
| 0.3 | 0.067 |
| 0.5 | 0.148 |
| 0.7 | 0.248 |
| 0.9 | 0.370 |
| 1.1 | 0.518 |
| 1.3 | 0.697 |

**Gambar 4.5.1 Matriks Input Interpolasi Polinomial**

Lalu, program akan memproses input dan memberikan koefisien untuk masing-masing x pangkat. Setelah ditemukan polinomial yang mendekati nilai fungsi f(x), akan ditaksir beberapa titik x agar diketahui nilainya. Nilai-nilai X tersebut adalah x = 0.2; x = 0.55; x = 0.85; x = 1.28. Berikut adalah hasil eksekusi program interpolasi:



**Gambar 4.5.2 Eksekusi Program Interpolasi Polinom Pada Fungsi**

Program akan menyelesaikan persamaan dengan berbagai pilihan metode. Setelah mendapatkan nilainya program akan menuliskan fungsi polinomial hasil interpolasi. Lalu, dengan memasukkan input x akan ditaksir nilai f(x) melalui hasil dari pendekatan fungsi. Hasil tersebut adalah:

1. f(0.2) = 0,03296093749997216
2. f(0.55) = 0,1711186523436975
3. f(0.85) = 0,33723583984367567
4. f(1.28) = 0,6775418374998929

**4.6 Polinom Interpolasi Pada Data Covid-19**

Diketahui data jumlah kasus Covid-19 di Indonesia mulai tanggal 24 April 2020 sampai 10 September 2020 sebagai berikut:

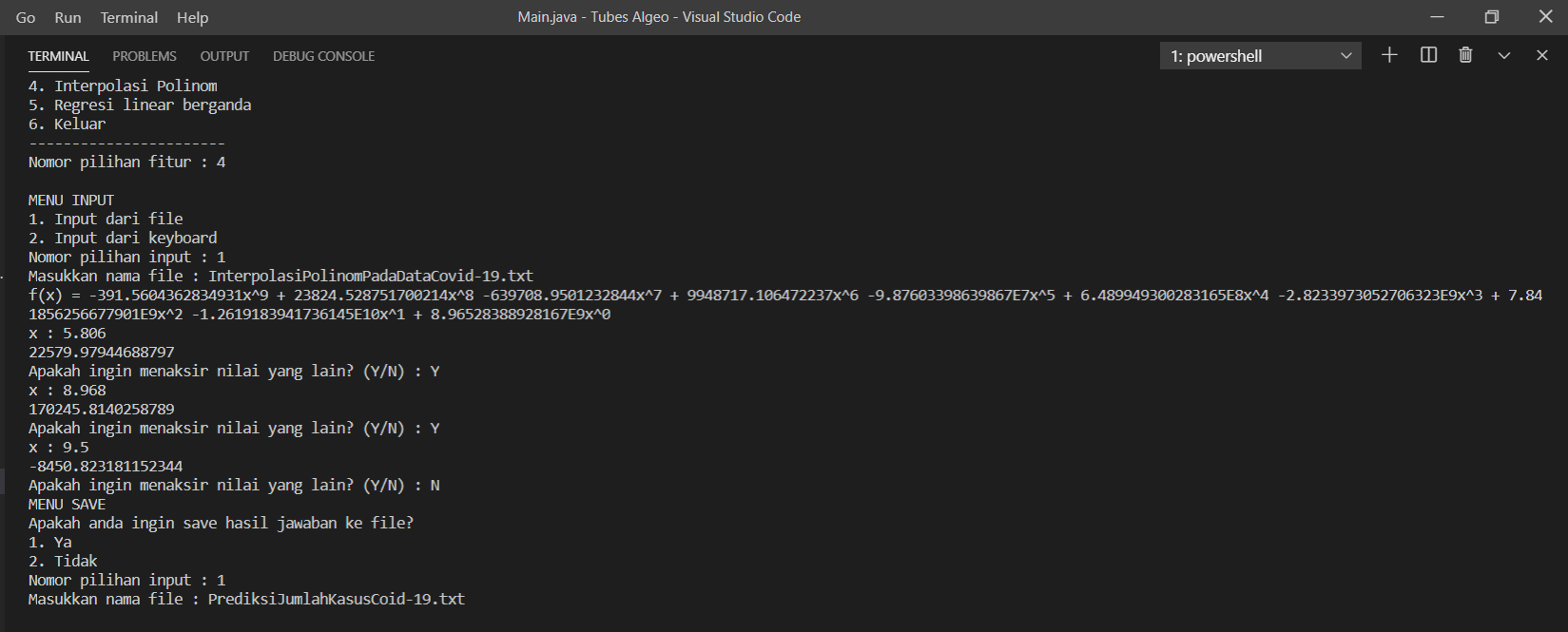


**Gambar 4.6.1 Data Kasus Covid-19**

Dengan menggunakan interpolasi polinom maka akan diprediksi jumlah kasus pada tanggal-tanggal berikut 25/05/2020, 30/08/20, 15/09/20. Dengan menerapkan rumus yang sudah ditentukan maka ketiga tanggal tersebut diubah ke dalam bentuk decimal sebelum dimasukkan sebagai program yaitu sebagai berikut:

1. 25/05/2020 = 5,806
2. 30/08/2020 = 8,968
3. 5/09/2020 = 9,500

Berikut adalah hasil eksekusi program interpolasi:



**Gambar 4.6.2 Eksekusi Program Interpolasi Polinom Pada Data Covid-19**

Program akan menyelesaikan persamaan dengan berbagai pilihan metode. Setelah mendapatkan nilainya program akan menuliskan fungsi polinomial hasil interpolasi. Lalu, dengan memasukkan input tanggal(desimal) akan ditaksir jumlah kasus melalui hasil dari pendekatan. Hasil tersebut adalah sebanyak 22.580 kasus pada tanggal 25/05/2020; 17.0246 kasus pada tanggal 30/08/2020; -8.451 kasus atau artinya sudah tidak ada lagi kasus pada tanggal 15/09/2020.

**4.7 Penyederhanaan Fungsi**

Diketahui fungsi sebagai berikut:

f(x) =

Dengan menggunakan interpolasi fungsi akan disederhanakan menjadi polinom pangkat lima dengan rentang titik yang diambil adalah [0,2]. Berikut adalah input yang diperlukan:

**Tabel 4.7 Input Interpolasi Polinomial**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | 0 | 0.4 | 0.8 | 1.2 | 1.6 |
| f(x) | 0 | 0.41888 | 0.50716 | 0.56092 | 0.58368 |

Melalui metode penyelesaian sistem persamaan linear akan dicari setiap koefisien x pangkat sehingga dapat ditentukan fungsi polinomial pangkat lima hasil pendekatan dari f(x). Hasil ekesekusi program adalah sebagai berikut:



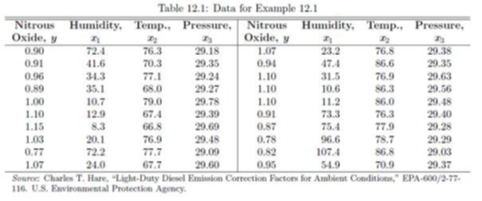
**Gambar 4.7.1 Eksekusi Program Interpolasi Polinom Pada Penyederhanaan Fungsi**

Jadi, hasil penyederhanaan fungsi f(x) dalam bentuk polinomial pangkat lima adalah:

f(x) = -0.4761x4 + 1.9138x3 -2.7964x2 + 1.89x

**4.8 Regresi Linier Berganda Pada Tabel Kelembaban Udara**

Diketahui data sebagai berikut:



**Gambar 4.8.1 Tabel Data**

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

20b0 + 863.1b1 + 1530.4b2 + 587.84b3 = 19.42

863.1b0 + 54876.89b1 + 67000.09b2 + 25283.395b3 = 779.477

1530.4b0 + 67000.09b1 + 117912.32b2 + 44976.867b3 = 1483.437

587.84b0 + 25283.395b1 + 44976.867b2 + 17278.5086b3 = 571.1219

Dengan menggunakan metode penyelesaian sistem persamaan linear dapat ditentukan nilai dari bn untuk setiap variabel. Fungsi ini adalah pendekatan nilai menggunakan metode regresi linear. Setelah itu, akan dicari nilai Nitrous Oxide(y) apabila Humidity(x1) bernilai 50%, temperature(x2) 76°F, dan tekanan udara(x3) sebesar 29.30 menggunakan fungsi yang dihasilkan. Dengan input tersebut berikut adalah hasil ekesekusi program:



**Gambar 4.8.2 Eksekusi Program Regresi Linear Variabel Banyak Pada Tabel Kelembaban Udara**

Hasil pendekatan nilai Nitrous Oxide adalah sebesar 1.

**BAB V**

**PENUTUP**

**5.1 Kesimpulan**

Dengan menggunakan teori matematika yang ada dalam aljabar linier dan menerapkannya ke dalam bahasa pemrograman Java, didapatkan sebuah aplikasi yang dapat menerima masukan SPL maupun matrix *augmented* melalui keyboard maupun file luar berformat txt, yang kemudian akan dihasilkan solusi untuk persamaan linier dan juga matriks yang telah dimasukkan oleh pengguna ke dalam aplikasi dengan menggunakan metode - metode seperti Gauss, Gauss - Jordan, Determinan, Adjoin, Invers, Metode Cramer, Polinom Interpolasi, Regresi Linier Berganda.

**5.2 Saran**

Penulis menyadari bahwa dalam pengerjaan program masih banyak yang dapat dikembangkan, sebagai berikut.

1. Program dapat menjadi lebih baik jika ditambahkan Graphical User Interface
2. Untuk mempermudah dalam pembacaan kode, variabel dan fungsi yang tidak digunakan dapat dihapus.
3. Untuk itu meningkatkan *reusability* pada program, perlu dibuat class berdasarkan fungsional masing - masing fungsi. Sehingga dapat mengurangi penulisan kode dengan fungsi yang sama berkali - kali
4. Program yang ada memiliki komentar yang hanya menjelaskan beberapa fungsi, terbatas pada beberapa bagian dan tidak menjelaskan langkah - langkah yang dilakukan oleh program. Diperlukan penjelasan mengenai fungsi dan prosedur untuk memperjelas program.

**5.3 Refleksi**

Melalui tugas ini, penulis belajar untuk bekerja sama dalam kelompok, tidak menunda - nunda pekerjaan, berbagi tugas, dan juga membuat program dalam bahasa Java untuk pertama kalinya.

**DAFTAR PUSTAKA**

<https://www.profematika.com/eliminasi-gauss-dan-contoh-penerapannya/>

<https://www.profematika.com/eliminasi-gauss-jordan-beserta-contoh-penerapannya/#:~:text=Eliminasi%20Gauss%2DJordan%20adalah%20prosedur,tereduksi%20dengan%20Operasi%20Baris%20Elementer>

<https://blog.ruangguru.com/cara-mencari-determinan-dan-invers-matriks>

<https://id.wikipedia.org/wiki/Determinan#:~:text=Dalam%20bidang%20aljabar%20linear%2C%20determinan,transformasi%20yang%20digambarkan%20oleh%20matriks>

<https://id.wikipedia.org/wiki/Aljabar_linear#:~:text=Matriks%20Balikan%20(Invers),-Orde%202x2&text=Jika%20tidak%20ditemukan%20matriks%20B,dari%20A%20maka%20B%20%3D%20C.&text=Karena%20AB%20%E2%89%A0%20BA%20%E2%89%A0,matriks%20B%20disebut%20matriks%20tunggal>

<https://www.madematika.net/2017/08/pengertian-minor-kofaktor-matriks.html>

<https://id.wikipedia.org/wiki/Kaidah_Cramer>

<https://www.dosenpendidikan.co.id/rumus-interpolasi/>

<https://www.statistikian.com/2018/01/penjelasan-tutorial-regresi-linear-berganda.html#:~:text=Regresi%20Linear%20Berganda%20adalah%20model,disebut%20dengan%20multiple%20linear%20regression>

<https://www.geeksforgeeks.org/file-class-in-java/>

<https://www.javatpoint.com/java-filewriter-class>