

BAB I

PENDAHULUAN

A. Latar Belakang

Dunia usaha dewasa ini semakin pesat, ditandai dengan banyaknya perusahaan yang bermunculan dengan berbagai macam usaha bahkan dengan usaha sejenis sehingga persaingan yang terjadi diantara pengusaha semakin ketat. Pada dasarnya setiap perusahaan baik perusahaan besar maupun perusahaan kecil bertujuan untuk mencari keuntungan yang sebesar – besarnya dalam menjalankan kegiatan perusahaan, lebih – lebih dalam era globalisasi sekarang ini., maka setiap organisasi dalam dunia bisnis dituntut untuk senantiasa memanfaatkan sumberdaya yang dimiliki seoptimal mungkin. Ketatnya persaingan pada perusahaan yang memproduksi produk yang sejenis akan membuat perusahaan tersebut terpacu untuk menciptakan inovasi – inovasi yang lebih menarik dan beragam serta selektif dalam kualitas produk yang diproduksi. Oleh karena itu, perusahaan dituntut untuk semakin tanggap dalam melihat apa yang diinginkan konsumen.

Hal – hal yang perlu perusahaan perhatikan didalam faktor – faktor produksi yang ada seperti kulit, penjahitan, finishing. Faktor – faktor produksi ini tersedia dalam jumlah terbatas sehingga pengalokasiannya harus direncanakan sebaik mungkin. Perusahaan harus merencanakan dan mengelola perusahaannya dengan baik agar perusahaan dapat memperoleh hasil yang baik dengan memanfaatkan sumberdaya. sumberdaya yang terbatas secara efektif dan efisien serta tercapainya tujuan perusahaan. Dalam penelitian ini, menitik beratkan pada masalah penentuan kombinasi produk yang paling tepat di suatu perusahaan, dalam hal ini adalah perusahaan Cemerlang, sehingga dapat memberikan keuntungan yang maksimal kepada perusahaan tersebut, selain itu juga manajemen perusahaan harus dapat menggunakan kapasitas produksi sebaik baiknya agar dapat memenuhi kebutuhan – kebutuhan konsumen, maka dengan demikian laba atau keuntungan yang optimal dapat ditentukan oleh kombinasi produsen sesuai dengan kapasitas yang ada dalam perusahaan. Sebab dengan mengetahui seberapa besar produksi yang harus dihasilkan dalam kombinasi produk maka perusahaan dapat merencanakan laba yang akan diperolehnya.

Metode simplex digunakan untuk menyelesaikan masalah optimasi yang melibatkan tiga variabel atau lebih yang tidak dapat diselesaikan oleh metode grafik. Metode simpleks adalah metode yang digunakan untuk menyelesaikan permasalahan yang memiliki lebih dari dua variabel. Metode simpleks didefinisikan sebagai cara menyelesaikan permasalahan yang memiliki variabel keputusan minimal dua dengan menggunakan alat bantu tabel. Metode simpleks dibedakan menjadi dua yaitu, metode simpleks maksimasi untuk mencari keuntungan maksimal dan metode simpleks minimasi untuk mencari biaya minimal.

B. Tujuan

- Memperkenalkan konsep Metode Maksimasi Simpleks
- Memperkenalkan pemrograman java Swing berbasis GUI

C. Manfaat Praktikum

Mahasiswa dapat mengerti konsep Metode Maksimasi Simpleks dan dapat mengimplementasikan konsep tersebut ke dalam pemrograman Java berbasis GUI.

BAB II

PEMBAHASAN

A. MATERI

Persoalan program linier tidak selalu sederhana karena melibatkan banyak constraint (pembatas) dan banyak variabel sehingga tidak mungkin diselesaikan dengan metode grafik. Oleh karena itu serangkaian prosedur matematik (aljabar linier) diperlukan untuk mencari solusi dari persoalan yang rumit tersebut. Prosedur yang paling luas digunakan adalah Metode Simplex. Penemuan metode ini merupakan lompatan besar dalam Riset Operasi dan digunakan sebagai prosedur penyelesaian dari setiap program komputer.

Metode simplex merupakan sebuah metode lanjutan dari metode grafik. Metode grafik tidak dapat menyelesaikan persoalan manajemen yang memiliki variabel keputusan yang cukup besar, sehingga untuk menyelesaikannya dibuat sebuah metode yang lebih kompleks yaitu dengan menggunakan program komputer QSB (Quantitative System For Business) atau menggunakan metode simplex. Dalam kenyataannya penggunaan komputer lebih efisien, akan tetapi metode dasar yang digunakan dalam pengoperasian komputer tetap metode simplex.

Metode simplex adalah metode yang dapat digunakan untuk menyelesaikan persoalan manajerial yang telah diformulasikan terlebih dahulu ke dalam persamaan matematika program linear yang mempunyai variabel keputusan mulai dari lebih besar atau sama dengan 2 (dua) sampai multivariabel. Sedangkan metode grafik hanya dapat digunakan apabila jumlah variabel keputusan maksimal 2 (dua) buah. Sehingga dapat disimpulkan bahwa suatu persoalan linear programming yang diselesaikan dengan metode grafik juga dapat diselesaikan dengan metode simpleks, sebaliknya suatu persoalan yang hanya bisa diselesaikan dengan metode simplex tidak dapat diselesaikan dengan metode grafik.

Beberapa ketentuan yang perlu diperhatikan, antara lain:

1. Nilai kanan (NK / RHS) fungsi tujuan harus nol (0).
2. Nilai kanan (RHS) fungsi kendala harus positif. Apabila negatif, nilai tersebut harus dikalikan -1 .

3. Fungsi kendala dengan tanda " \leq atau \geq " harus diubah ke bentuk " $=$ " dengan menambahkan variabel slack/surplus. Variabel slack/ surplus disebut juga variabel dasar

Langkah-langkah yang harus dipahami dalam menggunakan metode simplex, yaitu:

1. Membuat model matrix LP
2. Merubah formulasi LP menjadi formulasi standar

Merubah formulasi biasa ke dalam formulasi standar harus mengikuti kaidah dasar yang berlaku, yaitu:

- a. Introduksikan variabel baru sebagai variable dummy dengan singkatan huruf S sebagai singkatan dari Slack (kekurangan) atau Surplus (kelebihan)
 - b. Variable slack kita introduksikan apabila kita mempunyai bentuk tanda pembatas lebih kecil atau sama dengan (\leq)
 - c. Variabel surplus kita introduksikan apabila kita mempunyai bentuk tanda pembatas lebih besar dari atau sama dengan (\geq)
3. Menyiapkan table simplex awal

	C_j							
C_i	BV	X₁	X₂	X_n	S₁	S₂	S_n	B_i
	S₁							
	S₂							
	S_n							
	Z_j							
C_j-Z_j								

Penjelasan penggunaan tabel simplex :

- a. Kolom Baris
 - Kolom baris selalu ada dan ditempatkan di kolom paling kiri setelah C_i
 - Untuk kolom tabel awal variabel yang pertama kali kita tulis pada kolom ini adalah:

- Variabel tambahan yang bertanda positif seperti slack variable
- Artifisial variabel

Oleh karena itu, surplus variabel tidak pernah kita masukan ke dalam kolom basis pada tabel awal

b. Kolom C_j

Kolom koefisien fungsi tujuan diletakan pada baris pertama tabel awal simplex. Angka koefisien dapat kita lihat pada fungsi tujuan formulasi standar dari persoalan yang dihadapi.

c. Kolom diantara kolom C_j dan kolom paling kanan atau kolom nilai ruas kanan

Jumlah kolom ini bervariasi tergantung berapa jumlah variabel yang ada di dalam fungsi tujuan formulasi standar. Oleh karena itu apabila terjadi kesalahan dalam membuat formulasi standar maka penyelesaian persoalan dengan metode simplex juga akan salah.

d. Kolom nilai ruas kanan (NRK atau B_i)

Pada kolom ini, dituliskan nilai ruas kanan dari setiap batasan yang ada di dalam setiap persoalan yang dihadapi.

e. Jumlah baris

Jumlah baris di antara baris Basic variabel dengan baris Z_j tergantung dari berapa buah batasan yang kita hadapai di dalam persoalan.

f. Baris Z_j

Baris Z_j digunakan untuk mendapat nilai Shadow Price atau Nilai Marginal Value Product dari setiap variabel yang kita hadapi. Angka yang akan kita tuliskan pada baris Z_j ini adalah angka hasil penjumlahan perkalian setiap koefisien dari variabel yang tertera dalam kolom baris dengan angka-angka di dalam Matrix

g. Baris C_j-Z_j

baris ini bermanfaat bagi kita untuk melihat kapan kita berhenti melakukan iterasi atau baris yang dapat membantu kita menentukan apakah penyelesaian optimal telah kita capai.

4. Memasukkan nilai-nilai dan variable dalam formulasi standar ke dalam tabel awal
5. Melakukan proses literasi
 - a. Tentukan kunci kolom (pivot coloum).
Caranya adalah memilih nilai $C_j - Z_j$ yang terbesar dan positif.
 - b. Tentukan kunci baris (pivot row)
Caranya adalah memilih hasil bagi antara NRK dengan angka-angka yang ada dalam kolom kunci, kemudian pilih hasil bagi yang terkecil dan positif. Hasil bagi dengan nilai negative, nol dan tak terhingga tidak dapat dijadikan sebagai kunci baris.
 - c. Cari angka baru yang terdapat pada baris kunci dengan cara membagi semua angka yang terdapat pada baris kunci dengan angka kunci. Angka kunci adalah angka yang terdapat pada persilangan baris kunci dengan kolom kunci.
 - d. Mencari angka baru pada baris yang lain dengan rumus:
Angka baru = nilai pada baris lama – (perkalian koefisien pada kolom kunci dengan angka baru baris kunci).
 - e. Apabila sosialisasi optimal belum ditemukan maka kembali ke langkah 5a di atas, sehingga nilai yang terdapat pada baris $C_j - Z_j \leq 0$
6. Menentukan apakah penyelesaian optimal sudah tercapai.
7. Membuat kesimpulan jawaban..

B. PEMAHAMAN PRIBADI DENGAN CONTOH SOAL DAN PENYELESAIAN

PT Yummy food memiliki sebuah pabrik yang akan memproduksi dua jenis produk yaitu vanilla dan violette. Untuk memproduksi kedua produk tersebut diperlukan bahan baku A, bahan baku B dan jam tenaga kerja. Maksimum pengerjaan bahan baku A adalah 60kg per hari, bahan baku B 30kg per hari dan tenaga kerja 40jam per hari. Kedua jenis produk memberikan sumbangan keuntungan sebesar Rp40,00 untuk vanilla dan Rp30,00 untuk violette. Masalah yang dihadapi adalah bagaimana

menentukan jumlah unit setiap produk yang akan diproduksi setiap hari. Kebutuhan setiap unit produk akan bahan baku dan jam tenaga kerja dapat dilihat pada tabel berikut ini:

Jenis Bahan Baku dan Tenaga Kerja	Kg Bahan Baku Dan Jam Tenaga Kerja		Maksimum Penyediaan
	Vanilla	Violette	
Bahan baku A	2	3	60Kg
Bahan baku B	-	2	30Kg
Tenaga Kerja	2	1	40jam
Sumbangan keuntungan	Rp40,00	Rp30,00	

Penyelesaian:

Z = Rupiah keuntungan per hari

X_1 = Jumlah vanilla yang diproduksi/perhari

X_2 = jumlah violette yang diproduksi/hari

Langkah 1

Formulasi LP (bentuk standar)

Fungsi tujuan $\rightarrow Z_{\max} = 40X_1 + 30X_2$

Fungsi kendala \rightarrow I. $2X_1 + 3X_2 \leq 60$
 II. $2X_2 \leq 30$
 III. $2X_1 + 1X_2 \leq 40$
 IV. $X_1, X_2 \geq 0$

Diubah menjadi:

$$2X_1 + 3X_2 + S_1 + 0S_2 + 0S_3 = 60$$

$$2X_2 + 0S_1 + S_2 + 0S_3 = 30$$

$$2X_1 + 1X_2 + 0S_1 + 0S_2 + S_3 = 40$$

$$40X_1 + 30X_2 + 0S_1 + 0S_2 + 0S_3$$

$$C_1 = 40, C_2 = 30, C_3 = 0, C_4 = 0, C_5 = 0$$

Langkah 2

Tabel simplex awal masalah PT Yummy Food

	C_j	40	30	0	0	0	
C_i	BV	X₁	X₂	S₁	S₂	S₃	B_i
0	S₁	2	3	1	0	0	60
0	S₂	0	2	0	1	0	30
0	S₃	2	1	0	0	1	40
	Z_j	0	0	0	0	0	0
	C_j-Z_j	40	30	0	0	0	

Langkah 3

Apakah tabel tersebut sudah optimal? Belum, karena tabel optimal bila nilai yang terdapat pada baris $C_j - Z_j \leq 0$

Langkah 4

Penyelesaian dengan cara iterasi :

1. Menentukan kolom kunci, yaitu kolom yang memiliki nilai $C_j - Z_j$ terbesar yaitu kolom X_1 . Dengan demikian X_1 akan masuk dalam basis.
2. Menentukan baris kunci, yaitu baris yang memiliki angka indeks terkecil dan bukan negatif. Dalam hal ini baris S_3 . Dengan demikian S_3 akan keluar dari basis dan tempatnya akan digantikan oleh X_1 .
3. Menentukan angka kunci. Angka kunci adalah angka yang terdapat pada persilangan kolom kunci dengan baris kunci, dalam hal ini angka kunci = 2.
4. Mencari angka baru yang terdapat pada baris kunci, dengan cara membagi semua angka yang terdapat pada baris kunci dengan angka kunci.

$$\text{Angka baru} = 40/2, 2/2, 1/2, 0/2, 0/2, 1/2$$

$$\text{Atau} = \mathbf{20 \quad 1 \quad 1/2 \quad 0 \quad 0 \quad 1/2}$$

5. Mencari angka baru pada baris lain, yaitu :

- **Baris S₁**

$$\text{Angka lama} = [60 \quad 2 \quad 3 \quad 1 \quad 0 \quad 0]$$

$$\text{Angka baru} = [20 \quad 1 \quad 1/2 \quad 0 \quad 0 \quad 1/2] \mathbf{(2) -}$$

$$\text{Angka baru} = [\mathbf{20 \quad 0 \quad 2 \quad 1 \quad 0 \quad -1}]$$

- **Baris S₂**

$$\begin{aligned}
 \text{Angka lama} &= [30 \ 0 \ 2 \ 0 \ 1 \ 0] \\
 \text{Angka baru} &= [20 \ 1 \ \frac{1}{2} \ 0 \ 0 \ 1/2] \text{ (0) } - \\
 \text{Angka baru} &= [30 \ 0 \ 2 \ 0 \ 1 \ 0]
 \end{aligned}$$

Hasil perhitungan di atas, akan nampak pada tabel baru simplex yaitu tabel yang merupakan hasil iterasi pertama.

	C_j	40	30	0	0	0	
C_i	BV	X_1	X_2	S_1	S_2	S_3	B_i
0	S_1	0	2	1	0	-1	20
0	S_2	0	2	0	1	0	30
40	X_1	1	$\frac{1}{2}$	0	0	$\frac{1}{2}$	20
	Z_j	40	20	0	0	20	
	$C_j - Z_j$	0	10	0	0	-20	

Tabel iterasi 1 belum optimal sehingga harus diulang langkah di atas untuk iterasi kedua.

Langkah-langkah iterasi 2 :

1. Kolom kunci : X_2 (masuk dalam basis)
2. Baris kunci : S_1 (keluar dari basis)
3. Angka kunci : 2 (persilangan kolom kunci dan baris kunci)
4. Angka baru pada baris kunci, dengan cara membagi semua angka yang terdapat pada baris kunci dengan angka kunci.

$$\text{Angka Baru} = 20/2, 0/2, 2/2, \frac{1}{2}, 0/2, -1/2,$$

$$\text{Atau} = 10 \ 0 \ 1 \ \frac{1}{2} \ 0 \ -1/2$$

5. Angka baru pada baris lain :

• Baris S_2

$$\text{Angka Lama} = [30 \ 0 \ 2 \ 0 \ 1 \ 0]$$

$$\text{Angka Baru} = [10 \ 0 \ 1 \ \frac{1}{2} \ 0 \ -1/2] \text{ (2) } -$$

$$\text{Angka Baru } S_2 = [10 \ 0 \ 0 \ -1 \ 1 \ 1]$$

• Baris X_1

$$\text{Angka Lama} = [20 \ 1 \ \frac{1}{2} \ 0 \ 0 \ \frac{1}{2}]$$

$$\text{Angka Baru} = [10 \ 0 \ 1 \ \frac{1}{2} \ 0 \ -1/2] \text{ (1/2) } -$$

$$\text{Angka Baru } X_1 = [15 \ 1 \ 0 \ -1/4 \ 0 \ 3/4]$$

Adapun hasil perhitungan di atas, akan Nampak pada table baru yang merupakan hasil iterasi ke dua.

	C_j	40	30	0	0	0	
C_i	BV	X₁	X₂	S₁	S₂	S₃	B_i
30	X₂	0	1	½	0	-1/2	10
0	S₂	0	0	-1	1	1	10
40	X₁	1	0	-1/4	0	¾	15
	Z_j	40	30	5	0	15	
	C_j-Z_j	0	0	-5	0	-15	900

Solusi optimum tabel iterasi 2 menunjukan bahwa total nilai $Z = 900$ dengan masing-masing variabel keputusan $X_1 = 15$ dan $X_2 = 10$.

Variabel Basis	Koefisien Fungsi Tujuan	Nilai Variabel Basis	
X₂	30	10	300
S₂	0	10	0
X₁	40	15	600
JUMLAH			900

KESIMPULAN:

1. Pada tabel iterasi 2 merupakan tabel akhir simplex, dengan solusi optimal adalah :

X_1 (vanilla) = 15 unit
 X_2 (violette) = 10 unit
 Z (keuntungan) = Rp 900,00
2. Kendala kedua (bahan baku B) masih tersisa sebanyak 10 Kg yang ditunjukan oleh nilai $S_2 = 10$, pada tabel optimal
3. Kendala 1 dan 3 tidak ada sisa (full capacity), yang ditunjukan oleh nilai $S_1 = S_3 = 0$ (variabel nonbasis). Hal ini juga dapat dibuktikan dengan memasukan nilai S_1 dan S_2 ke dalam kendala 1 dan 3

Kendala 1 : $2X_1 + 3X_2 = 60$

$$2(15) + 3(10) = 60$$

$$60 = 60$$

Bahan baku yang digunakan = yang tersedia

Kendala 3 : $2X_1 + 1X_2 = 40$

$$2(15) + 1(10) = 40$$

$$40 = 40$$

Jam kerja yang digunakan = yang tersedia

BAB III

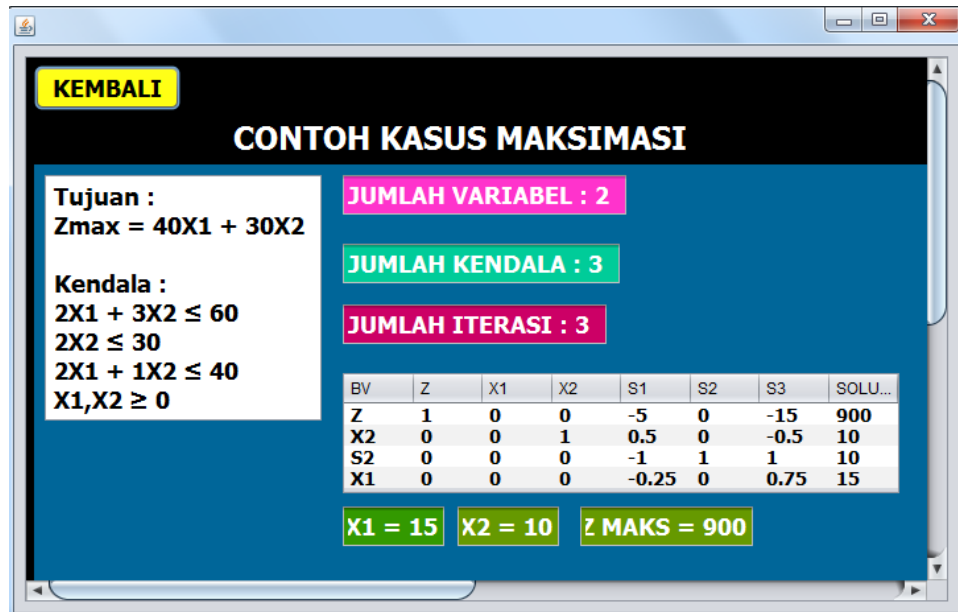
HASIL APLIKASI

Berikut ini beberapa tampilan Antarmuka aplikasi :

- Menu Utama



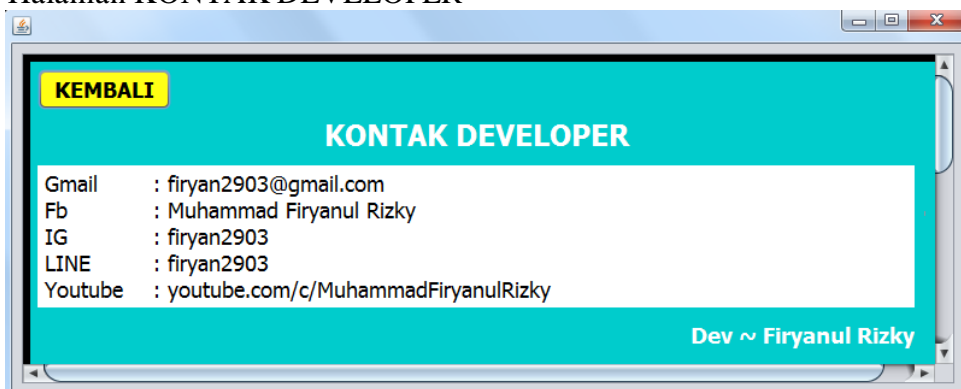
- Halaman CONTOH SOAL YANG BISA DICoba



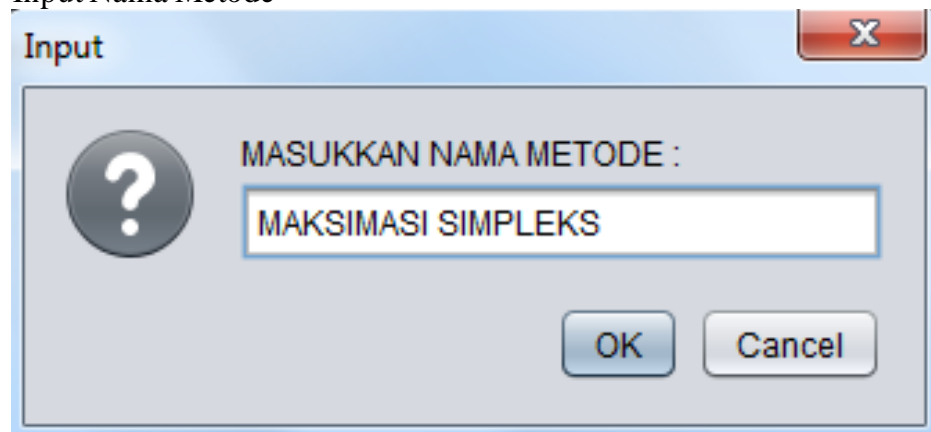
- Halaman KELEBIHAN APLIKASI



- Halaman KONTAK DEVELOPER

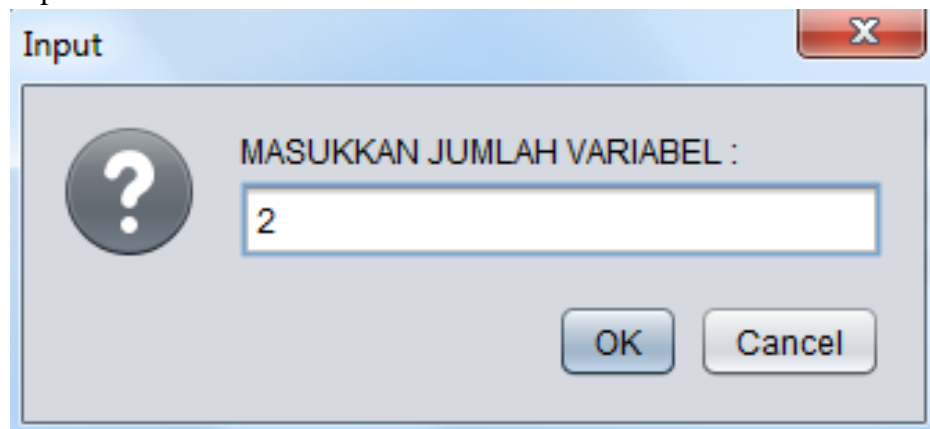


- Input Nama Metode



The screenshot shows a standard Windows-style dialog box titled "Input" with a red close button in the top right corner. Inside the dialog, there is a large question mark icon on the left. To its right, the text "MASUKKAN NAMA METODE :" is displayed. Below this text is a text input field containing the text "MAKSIMASI SIMPLEKS". At the bottom right of the dialog, there are two buttons: "OK" and "Cancel".

- Input Jumlah Variabel



The screenshot shows a standard Windows-style dialog box titled "Input" with a red close button in the top right corner. Inside the dialog, there is a large question mark icon on the left. To its right, the text "MASUKKAN JUMLAH VARIABEL :" is displayed. Below this text is a text input field containing the number "2". At the bottom right of the dialog, there are two buttons: "OK" and "Cancel".

- Input Jumlah Kendala

Input

MASUKKAN JUMLAH KENDALA :

3

OK Cancel

- Tampilan Antarmuka Proses

TUGAS AKHIR RISET OPERASI by : FIRYANUL RIZKY (1708561006) - APLIKASI SIMPLEKS (JAVA v1.0)

PILIHAN MENU

METODE YANG DIGUNAKAN : MAKSIMASI SIMPLEKS

MAKSIMASI SIMPLEKS

JENIS KASUS: Maks [LIHAT ALGORITMA](#)

	X1	X2		
	40	30		
kendala1	2	3	<=	60
kendala2	0	2	<=	30
kendala3	2	1	<=	40

HASIL AKHIR

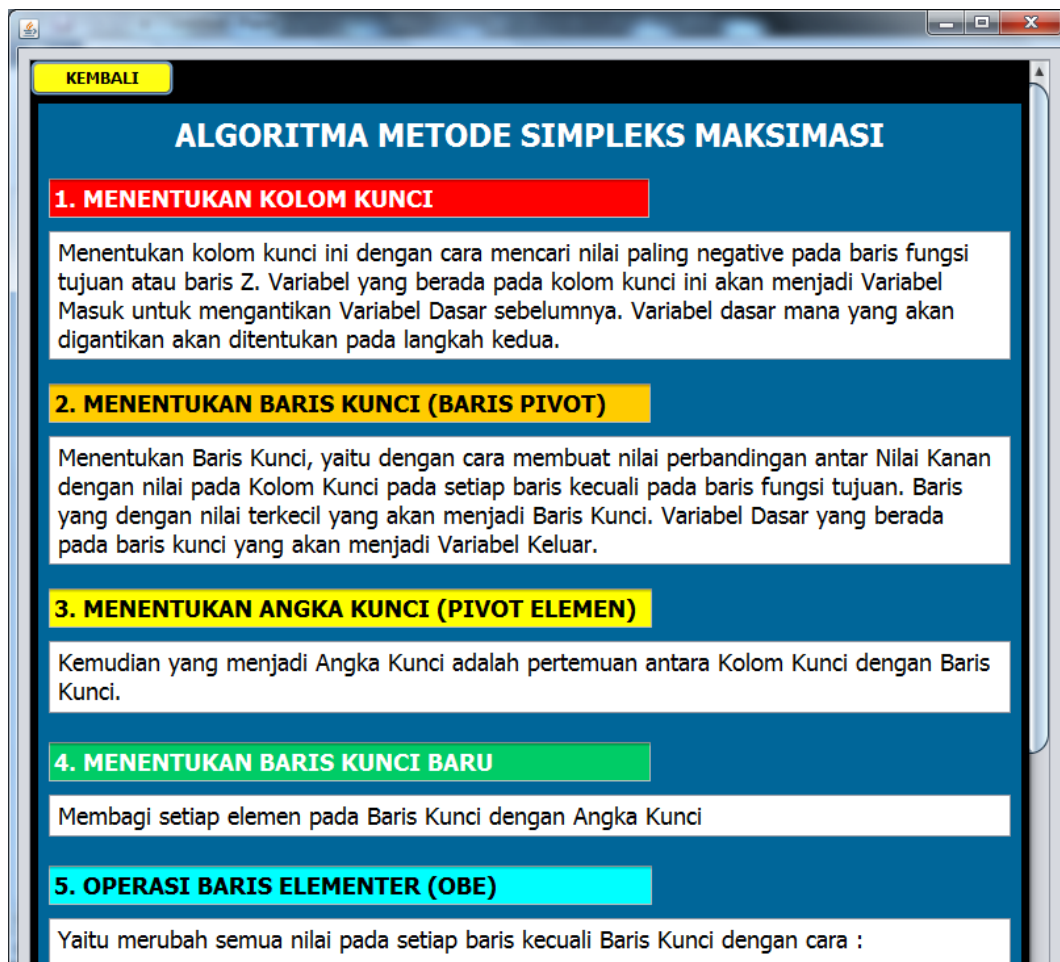
Jumlah Iterasi : 3
 Nilai Optimum Maks Z = 900.0
 Nilai X2 = 10.0
 Nilai X4 = 10.0
 Nilai X1 = 15.0

[KETERBATASAN APLIKASI](#)
[KONTAK DEVELOPER](#)

[PROSES SIMPLEKS](#) [SIMPAN MODEL KE EXCEL](#) Dev ~ Firyanul Rizky

BASIS	X1	X2	X3	X4	X5	SOLUSI
1 ITERASI						
Z	-40.0	-30.0	0.0	0.0	0.0	0.0
X3	2.0	3.0	1.0	0.0	0.0	60.0
X4	0.0	2.0	0.0	1.0	0.0	30.0
X5	2.0	1.0	0.0	0.0	1.0	40.0
2 ITERASI						
Z	0.0	-10.0	0.0	0.0	20.0	800.0
X3	0.0	2.0	1.0	0.0	-1.0	20.0
X4	0.0	2.0	0.0	1.0	0.0	30.0
X1	1.0	0.5	0.0	0.0	0.5	20.0
3 ITERASI						
Z	0.0	0.0	0.0	0.0	15.0	900.0
X2	0.0	1.0	0.5	0.0	-0.5	10.0
X4	0.0	0.0	-1.0	1.0	1.0	10.0
X1	1.0	0.5	0.5	0.5	0.75	15.0

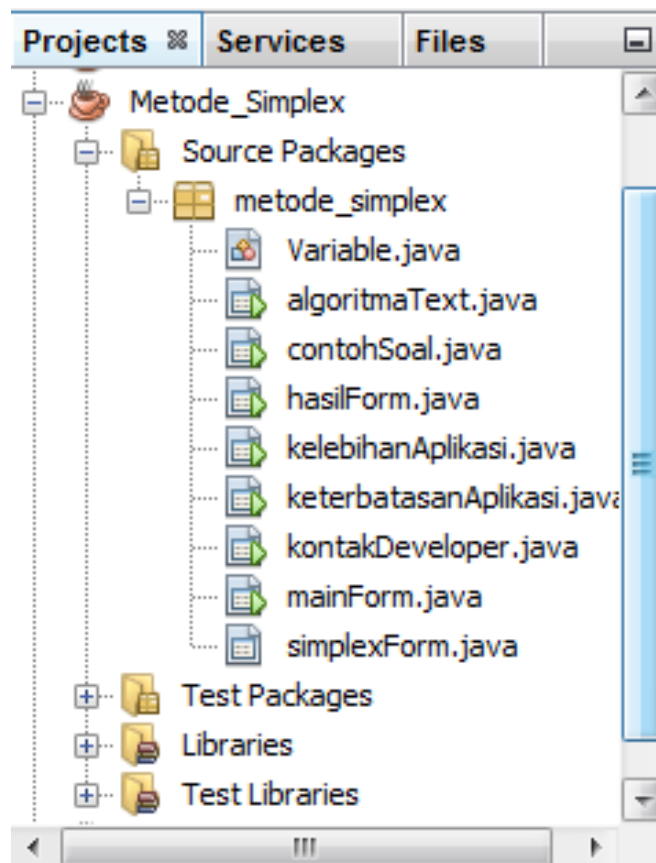
- Halaman Lihat Algoritma



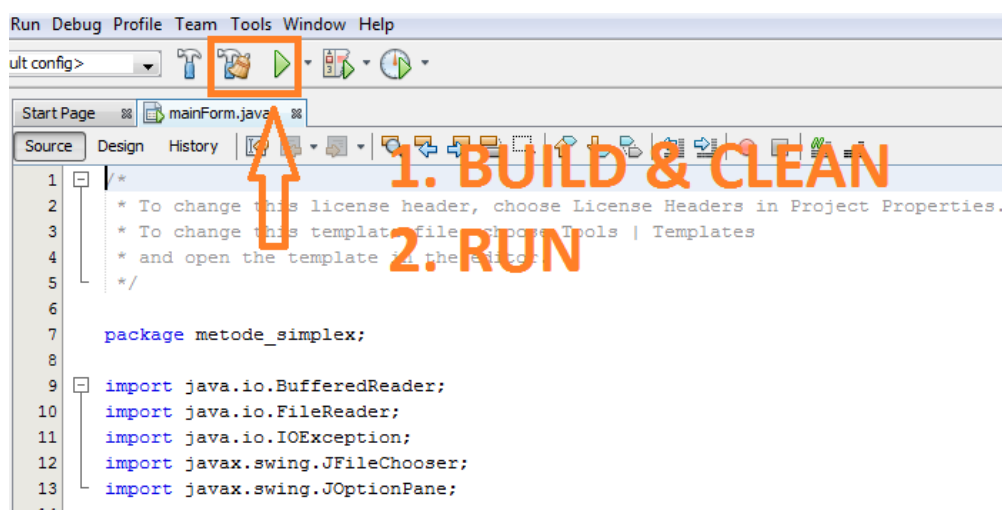
Penjelasan Program :

Sebelum menjalankan program dengan Aplikasi Netbeans yang masih dalam format Jar, ada beberapa hal yang mesti diketahui :

Dalam Jar Metode_Simpleks, kita masuk ke Source package metode_simpleks, disana ada beberapa Class Java seperti gambar berikut :



Program dijalankan dengan cara menjalankan mainForm.java yang akan memunculkan menu utama, klik Build&Clean, lalu klik run.



Penjelasan Class Java yang terdapat dalam package metode_simpleks :

Nama Class	Penjelasan
Variabel.java	: Class Java yang berfungsi mendeklarasi nilai awal dari variabel – variabel yang dipakai pada mainForm yang menyimpan inputan untuk Nama Metode, Jumlah Variabel, dan Jumlah Kendala.
algoritmaText.java	: Class Java Swing yang berisikan GUI dan kodongan untuk menampilkan Halaman LIHAT ALGORITMA yang dapat dipanggil melalui Combo Box LIHAT ALGORITMA pada Halaman Pemrosesan Simpleks.
contohSoal.java	: Class Java Swing yang berisikan GUI dan kodongan untuk menampilkan Halaman CONTOH SOAL yang dapat diakses melalui Combo Box CONTOH SOAL YANG DAPAT DILIHAT pada Menu Utama.
hasilForm.java	: Class Java Swing yang berisikan TABEL, diprogram untuk menampilkan Hasil Iterasi.
kelebihanAplikasi.java	: Class Java Swing yang berisikan GUI dan kodongan untuk menampilkan Halaman KELEBIHAN APLIKASI yang dapat diakses melalui Combo Box KELEBIHAN APLIKASI pada Menu Utama.
keterbatasanAplikasi.java	: Class Java Swing yang berisikan GUI dan kodongan untuk menampilkan Halaman KETERBATASAN APLIKASI yang dapat diakses melalui Combo Box KETERBATASAN APLIKASI pada Menu Utama dan Pemrosesan Simpleks.
kontakDeveloper.java	: Class Java Swing yang berisikan GUI dan kodongan untuk menampilkan Halaman KONTAK DEVELOPER yang dapat diakses melalui Combo Box KONTAK DEVELOPER pada Menu Utama dan Pemrosesan Simpleks.
mainForm.java	: Main Class Java yang berperan menjalankan menu Utama dan set Visible seluruh Class Form berbasis GUI yang ada dalam package metode_simpleks.
simplexForm.java	: Class Java SWING berbasis GUI yang diprogram sebagai induk dari pemrosesan simpleks, pada Class ini akan menerima variabel yang telah di set pada class Variabel.java dan memprosesnya dengan method, konstruktor dan objek tertentu berbasis Object Oriented Programming (OOP).

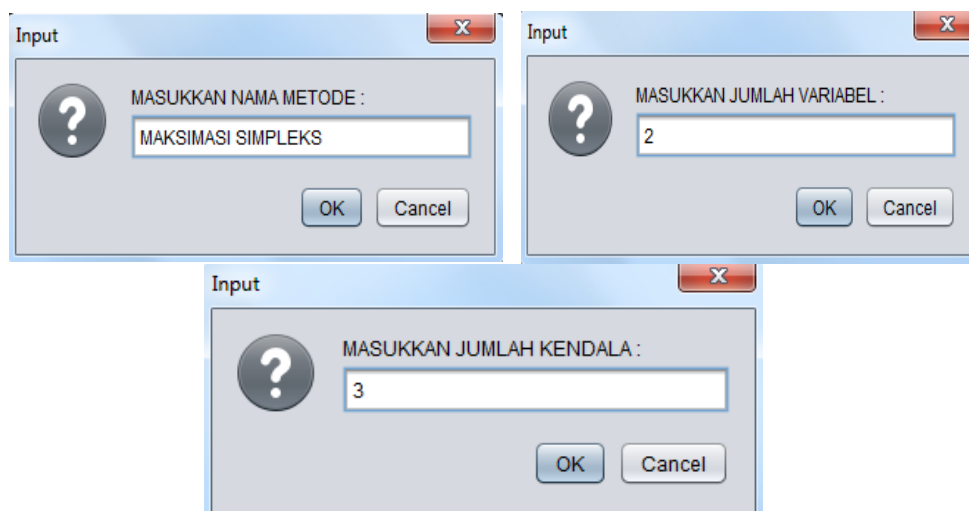
Setelah Di Run, maka akan tampil Menu Utama sebagai berikut :



KLIK pada PILIHAN MENU, disana disuguhkan 2 pilihan :

1. **Pemecahan Model**, ini adalah metode input secara manual, untuk User yang baru menjalankan aplikasi, disarankan untuk memulai dengan pilihan pertama ini.
2. **Import Model dari Excel**, ini adalah metode input otomatis apabila User sudah punya data Eksport berformatkan .csv Excel, File Eksport hanya bisa didapat setelah User melalui input manual dengan Klik Simpan Model ke Excel.

Jika User memilih **Pemecahan Model**, maka user akan diperintah untuk menginput data – data meliputi :



Inputan tersebut pada awalnya akan diset sebagai public Variabel dengan nilai awal 0 dengan jenis Variabel int, String “ ” dan boolean false pada Class Varibel.java dengan source Code sebagai berikut :

```

1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6
7  package metode_simplex;
8
9  import javax.swing.JOptionPane;
10
11 /**
12  *
13  * @author Firyanul Rizky
14  */
15 public final class Variable {
16     public static int DP_KENDALA = 0;
17     public static int DP_VARIABEL = 0;
18     public static String DP_PROBLEM = "";
19     public static boolean DP_FILE = false;
20 }
21

```

Penampilan Antarmuka Input form adalah tugas dari Class mainForm.java dengan memanfaatkan Java Swing Option Pane, pada class ini Varibel awal yang diset pada class Variabel.java diberikan variabel baru untuk mendapatkan hasil output yang dinamis, setelah diberi variabel proses akan memanggil simplexForm.java dan diset sebagai konstruktor, Source Code seperti dibawah ini :

```

34  @SuppressWarnings("unchecked")
35  Generated Code
160  public void error (String pesan) {
161      JOptionPane.showMessageDialog(this, pesan, "ERROR", JOptionPane.ERROR_MESSAGE);
162  }
163  private void jMenuItemActionPerformed(java.awt.event.ActionEvent evt) {
164      String subyek = "" + JOptionPane.showInputDialog("MASUKKAN NAMA METODE :");
165      int variabel = Integer.parseInt("0" + JOptionPane.showInputDialog("MASUKKAN JUMLAH VARIABEL :"));
166      int kendala = Integer.parseInt("0" + JOptionPane.showInputDialog("MASUKKAN JUMLAH KENDALA :"));
167      if ((variabel > 1) && (kendala >= 1)) {
168          Variable.DP_KENDALA=kendala;
169          Variable.DP_VARIABEL=variabel;
170          Variable.DP_PROBLEM=subyek;
171          simplexForm DP = new simplexForm();
172          DP.setVisible(true);
173          jDesktopPane1.add(DP);
174      } else {
175          error("Kesalahan dalam Input Jumlah Kendala dan Jumlah variabel");
176      }
177  }

```

Setelah User selesai menginputkan data yang diminta, selanjutnya akan dialihkan ke Halaman Pemrosesan Simpleks, pada halaman ini, User diminta untuk menginputkan nilai pada masing – masing nilai Z dan kendala Kasus :

Untuk Memperoleh tampilan yang dinamis seperti gambar diatas, dilakukan pembentukan method GUI Internal Frame yang didalamnya juga menginisialisasi method dan object baru bersamaan dengan pemanggilan object GUI seperti textField, label dan opsi pilihan Combo pada Class simplexForm.java dengan Source Code seperti berikut ini :

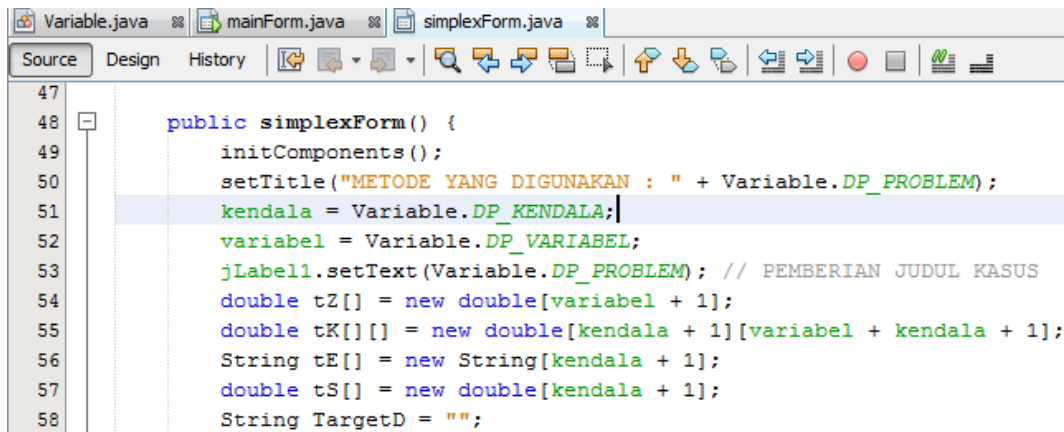
```

27  *
28  * @author Firyanul Rizky
29  */
30  public class simplexForm extends javax.swing.JInternalFrame {
31
32      /**
33       * Creates new form formDP
34       */
35      int kendala = Variable.DP_KENDALA;
36      int variabel = Variable.DP_VARIABEL;
37      // MATRIKS
38      public double Z[] = new double[variabel + kendala + 2];
39      public double K[][] = new double[kendala + 1][variabel + kendala + 1];
40      public int KDS[] = new int[kendala + 1]; //INDEKS VARIABEL
41      public String E[] = new String[kendala + 1];
42      public double S[] = new double[kendala + 1];
43
44      protected JTextField textField;
45      protected JLabel label;
46      protected JComboBox kombo;

```

Pemanggilan objek GUI seperti textField, label dan kombo dilakukan secara dinamis dengan konsep Array, dibentuk variabel Array baru untuk menempatkan textField dan Combo Box seperti gambar kedua diatas.

Pada Judul paling atas kita set pada Title Internal Form dengan menampilkan variabel inputan User, sedangkan judul dibawahnya diset pada Label, Berikut Source Code untuk penampilan Judul/Title sesuai dengan Input yang berasal dari User :

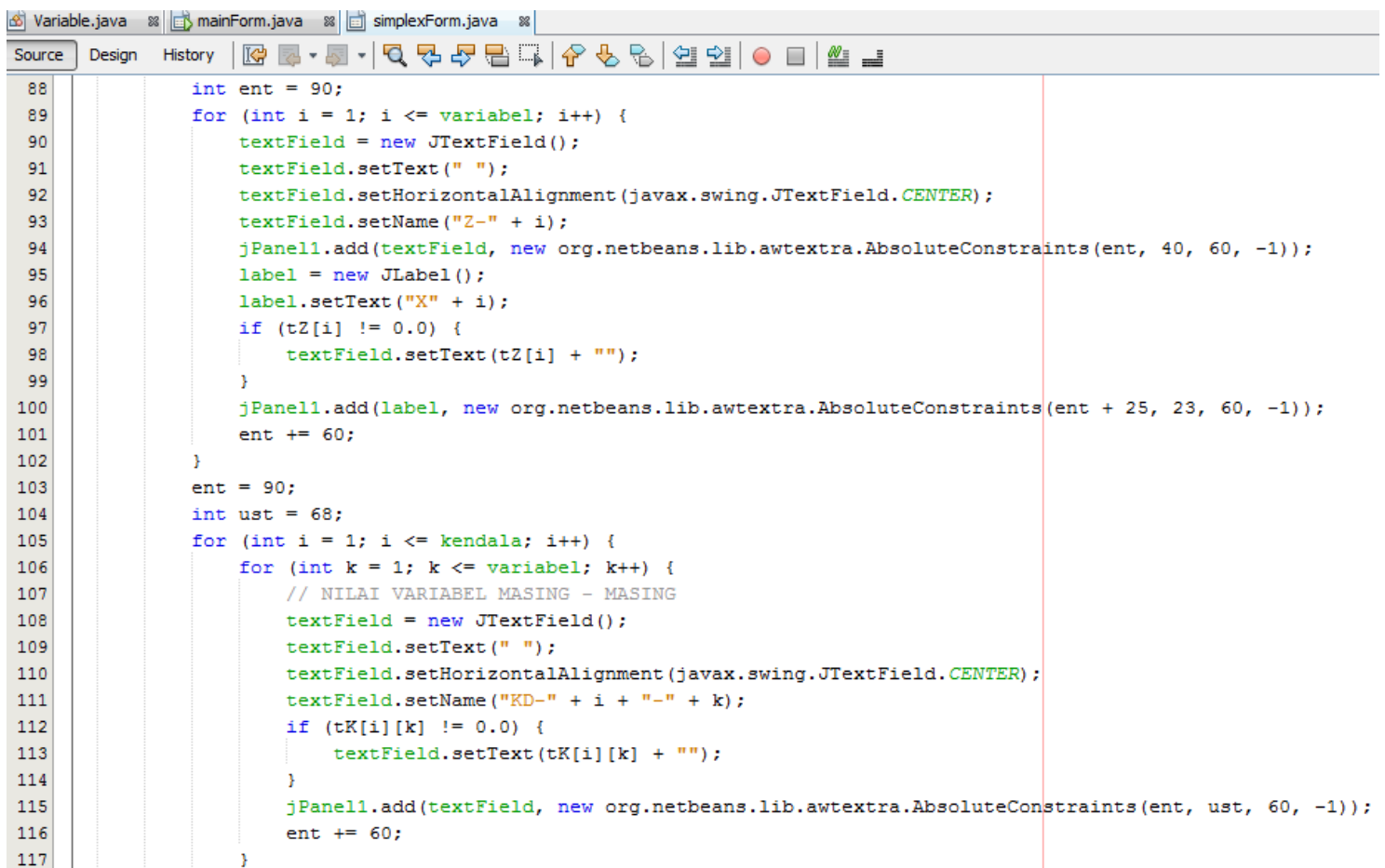


```

47
48     public simplexForm() {
49         initComponents();
50         setTitle("METODE YANG DIGUNAKAN : " + Variable.DP_PROBLEM);
51         kendala = Variable.DP_KENDALA;
52         variabel = Variable.DP_VARIABEL;
53         jLabel1.setText(Variable.DP_PROBLEM); // PEMBERIAN JUDUL KASUS
54         double tZ[] = new double[variabel + 1];
55         double tK[][] = new double[kendala + 1][variabel + kendala + 1];
56         String tE[] = new String[kendala + 1];
57         double tS[] = new double[kendala + 1];
58         String TargetD = "";

```

Penempatan textField dilakukan secara dinamis pada suatu JPanel GUI yang diprogram sedemikian rupa dengan memanggil variabel Array di method sebelumnya, Source Code seperti berikut :

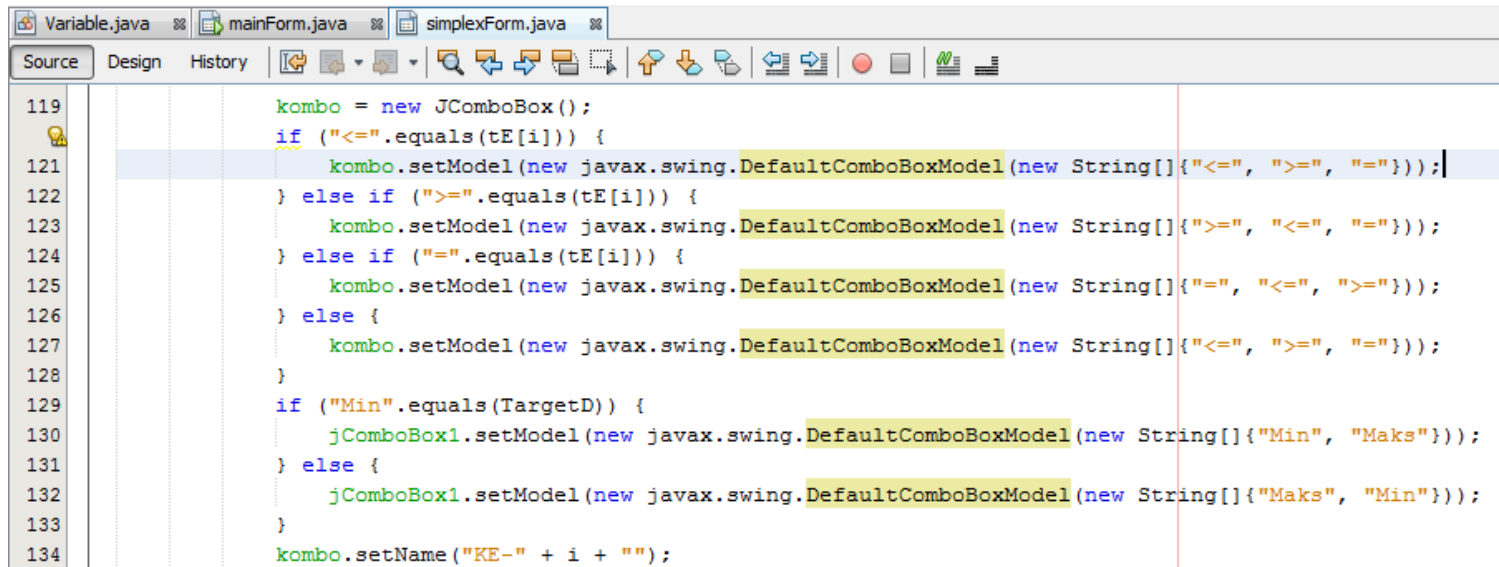


```

88     int ent = 90;
89     for (int i = 1; i <= variabel; i++) {
90         textField = new JTextField();
91         textField.setText(" ");
92         textField.setHorizontalAlignment(javax.swing.JTextField.CENTER);
93         textField.setName("Z-" + i);
94         jPanel1.add(textField, new org.netbeans.lib.awtextra.AbsoluteConstraints(ent, 40, 60, -1));
95         label = new JLabel();
96         label.setText("X" + i);
97         if (tZ[i] != 0.0) {
98             textField.setText(tZ[i] + " ");
99         }
100        jPanel1.add(label, new org.netbeans.lib.awtextra.AbsoluteConstraints(ent + 25, 23, 60, -1));
101        ent += 60;
102    }
103    ent = 90;
104    int ust = 68;
105    for (int i = 1; i <= kendala; i++) {
106        for (int k = 1; k <= variabel; k++) {
107            // NILAI VARIABEL MASING - MASING
108            textField = new JTextField();
109            textField.setText(" ");
110            textField.setHorizontalAlignment(javax.swing.JTextField.CENTER);
111            textField.setName("KD-" + i + "-" + k);
112            if (tK[i][k] != 0.0) {
113                textField.setText(tK[i][k] + " ");
114            }
115            jPanel1.add(textField, new org.netbeans.lib.awtextra.AbsoluteConstraints(ent, ust, 60, -1));
116            ust += 60;
117        }

```

Penempatan ComboBox dilakukan dengan dinamis. melibatkan variabel Array, koding programnya ditulis setelah finish state dari Source Code diatas, Source Codenya sebagai berikut

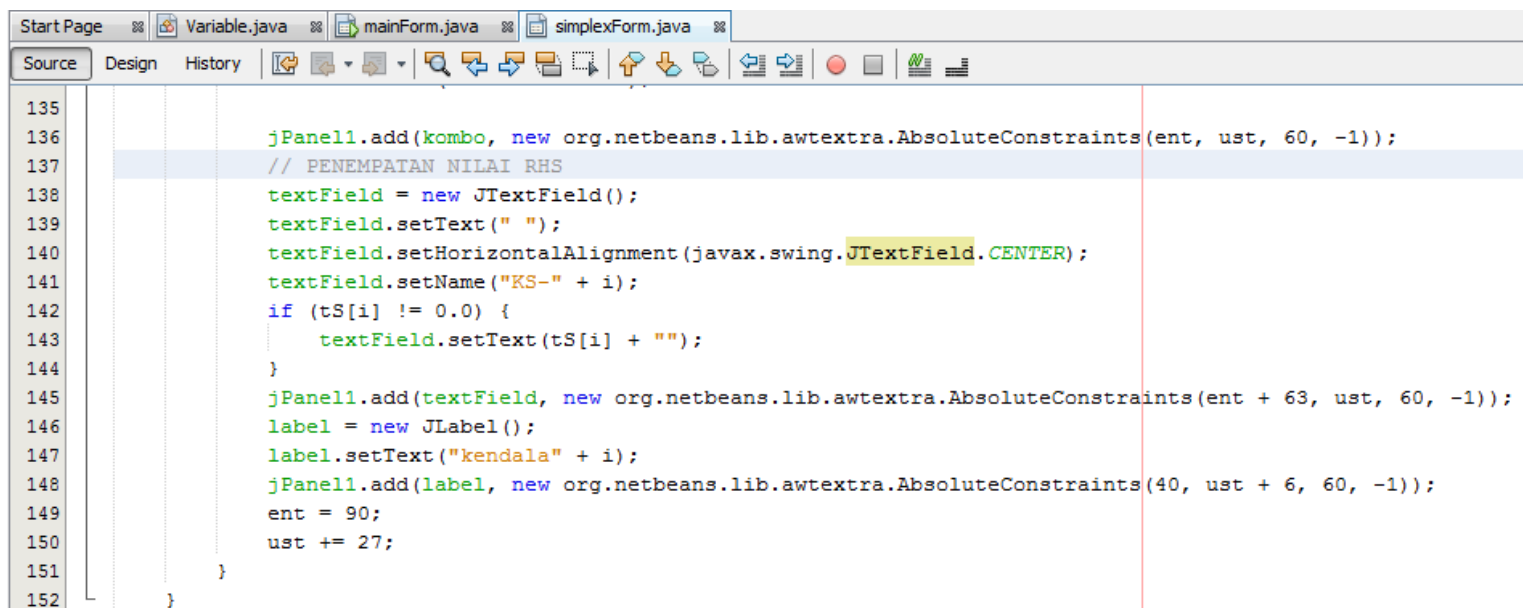


```

119     kombo = new JComboBox();
120     if ("<=".equals(tE[i])) {
121         kombo.setModel(new javax.swing.DefaultComboBoxModel(new String[]{"<=", ">=", "="}));
122     } else if (">=".equals(tE[i])) {
123         kombo.setModel(new javax.swing.DefaultComboBoxModel(new String[]{">=", "<=", "="}));
124     } else if ("=".equals(tE[i])) {
125         kombo.setModel(new javax.swing.DefaultComboBoxModel(new String[]{"=", "<=", ">="}));
126     } else {
127         kombo.setModel(new javax.swing.DefaultComboBoxModel(new String[]{"<=", ">=", "="}));
128     }
129     if ("Min".equals(TargetD)) {
130         jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(new String[]{"Min", "Maks"}));
131     } else {
132         jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(new String[]{"Maks", "Min"}));
133     }
134     kombo.setName("KE-" + i + "");

```

Berikutnya adalah penentuan Jumlah Textfield untuk Input Nilai RHS yang disesuaikan dengan array variabel yang dimasukkan User, setelah finish state looping dari string “Kendala” secara dinamis akan dilakukan dengan melibatkan variabel kendala yang berasal dari inputan User.



```

135
136     jPanel1.add(kombo, new org.netbeans.lib.awtextra.AbsoluteConstraints(ent, ust, 60, -1));
137     // PENEMPATAN NILAI RHS
138     textField = new JTextField();
139     textField.setText("");
140     textField.setHorizontalAlignment(javax.swing.JTextField.CENTER);
141     textField.setName("KS-" + i);
142     if (tS[i] != 0.0) {
143         textField.setText(tS[i] + "");
144     }
145     jPanel1.add(textField, new org.netbeans.lib.awtextra.AbsoluteConstraints(ent + 63, ust, 60, -1));
146     label = new JLabel();
147     label.setText("kendala" + i);
148     jPanel1.add(label, new org.netbeans.lib.awtextra.AbsoluteConstraints(40, ust + 6, 60, -1));
149     ent = 90;
150     ust += 27;
151 }
152

```

Kemudian lanjut ke proses inialisasi variabel putaran untuk memuat Jumlah iterasi matriks, yang digunakan adalah method Math.round dengan variabel bertipe data double (bisa include bil. Bulat dan pecahan) dengan rumus (variabel deklarasi $x * 100 / 100$), Source Code nya sebagai berikut :


```

Start Page  Variable.java  mainForm.java  simplexForm.java
Source  Design  History
159  @SuppressWarnings("unchecked")
160  Generated Code
369
370  public double putaran(double x) {
371      return (double) Math.round((x * 100)) / 100;
372  }
373
374  public void error(String pesan) {
375      JOptionPane.showMessageDialog(this, pesan, "ERROR", JOptionPane.ERROR_MESSAGE);
376  }

```

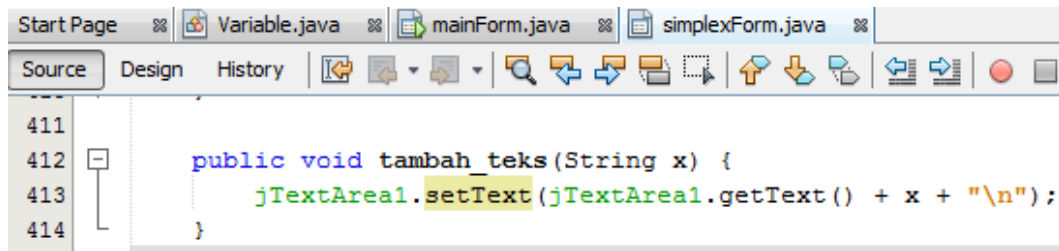
Kemudian proses akan memuat keseluruhan matriks yang telah diinputkan User meliputi Kolom Diagonal, Baris Horizontal, dan RHS, serta akan mengklarifikasi bahwa jika salah satu element tidak terisi maka akan diberi tanda “-” dan tidak akan dihitung sistem, Source Codenya sebagai berikut :

```

Start Page  Variable.java  mainForm.java  simplexForm.java
Source  Design  History
377
378  public void muat_matriks() {
379      Component[] cList = jPanel11.getComponents();
380      for (int x = 0; x < cList.length; x++) {
381          if (cList[x] instanceof javax.swing.JTextField) {
382              String value = ((javax.swing.JTextField) cList[x]).getName();
383              String[] parts = value.split("-");
384              if (parts[0].contains("Z")) {
385                  int t = Integer.parseInt(parts[1]);
386                  Z[t] = -1 * Double.parseDouble(((javax.swing.JTextField) cList[x]).getText());
387              }
388              if (parts[0].contains("KD")) {
389                  int k = Integer.parseInt(parts[1]);
390                  int d = Integer.parseInt(parts[2]);
391                  K[k][d] = Double.parseDouble(((javax.swing.JTextField) cList[x]).getText());
392              }
393              if (parts[0].contains("KS")) {
394                  int k = Integer.parseInt(parts[1]);
395                  S[k] = Double.parseDouble(((javax.swing.JTextField) cList[x]).getText());
396                  KDS[k] = variabel + k;
397              }
398          }
399          if (cList[x] instanceof javax.swing.JComboBox) {
400              String value = ((javax.swing.JComboBox) cList[x]).getName();
401              if (value != null) {
402                  String[] parts = value.split("-");
403                  if (parts[0].contains("KE")) {
404                      int k = Integer.parseInt(parts[1]);
405                      E[k] = ((javax.swing.JComboBox) cList[x]).getSelectedItem().toString();
406                  }
407              }
408          }
409      }

```


Kemudian kita deklarasikan method `tambah_teks` dengan data String yang memuat variabel dari iterasi :



Dibawah ini adalah Proses penting dalam Class `simplexForm.java` yang akan mengeksekusi method – method sebelumnya dan meng-create method baru untuk bisa menghasilkan Hasil Iterasi dan Hasil Optimal Akhir dari Simpleks Maksimasi dalam bentuk Tabel Form, Proses ini dilakukan dengan method Button PROSES SIMPLEKS, method ini mewakili keseluruhan algoritma Simpleks dalam mencari hasil optimal, Source Codenya seperti berikut :

```

private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
    // MATRIKS PENCIPTAAN DARI VARIABEL
    muat_matriks();
    // FORMASI MATRIKS VARIABEL
    String Target = jComboBox1.getSelectedItem().toString();
    jTable1.setModel(new DefaultTableModel());
    JTextArea1.setText("");
    boolean Maks = false;
    for (int k = 1; k <= kendala; k++) {
        if (E[k] != "<=") {
            error("SEMUA KENDALA HARUS <=");
            return;
        }
        if (S[k] < 0) {
            error("NILAI RHS TIDAK BOLEH KURANG DARI 0");
            return;
        }
    }
    if (Target == "Maks") {
        Maks = true;
    } else if (Target == "Min") {
        Maks = false;
    }
    // MENAMBAH SLAK VARIABEL
    for (int i = 1; i <= kendala; i++) {
        for (int k = 1; k <= kendala; k++) {
            if (i == k) {
                K[i][variabel + k] = 1.00;
            } else {
                K[i][variabel + k] = 0.00;
            }
        }
        Z[variabel + i] = 0.00;
    }
}

```

```

    }
    // NILAI RHS Z
    Z[kendala + variabel + 1] = 0.0;
    // MEMULAI ITERASI
    boolean optimal = false;
    int itr = 1;

    // TABEL ITERASI
    DefaultTableCellRenderer centerRenderer = new
DefaultTableCellRenderer();
    centerRenderer.setHorizontalAlignment(JLabel.CENTER);
    jTable1.setDefaultRenderer(String.class,
centerRenderer);

    DefaultTableModel dtm;
    dtm = new DefaultTableModel() {
        @Override
        public boolean isCellEditable(int row, int column) {
            return column < 0;
        }
    };
    jTable1.setModel(dtm);
    dtm.addColumn("BASIS");
    for (int k = 1; k <= (variabel + kendala); k++) {
        dtm.addColumn("X" + k);
    }

    dtm.addColumn("SOLUSI");
    for (int i = 0; i <= (variabel + kendala + 1); i++) {
jTable1.getColumnModel().getColumn(i).setCellRenderer(centerRend
erer);
    }
    do {
        Vector rowx = new Vector();
        rowx.add(itr + " ITERASI");
        dtm.addRow(rowx);
        Vector row = new Vector();
        row.add("Z");
        for (int k = 1; k <= (variabel + kendala + 1); k++) {
            row.add(putaran(Z[k]));
            dtm.isCellEditable(1, k);
        }
        dtm.addRow(row);
        for (int i = 1; i <= (kendala); i++) {
            Vector row2 = new Vector();
            row2.add("X" + KDS[i]);
            for (int k = 1; k <= (variabel + kendala); k++) {
                row2.add(putaran(K[i][k]));
                dtm.isCellEditable(i, k);
            }
            row2.add(putaran(S[i]));
            dtm.addRow(row2);
        }

        if (Maks) {
            // JIKA YANG DIPILIH MODEL MAKSIMASI
            int PC = 0; // KOLOM UTAMA
            double PCV = 0; // NILAI BARIS UTAMA Z

```

```

for (int c = 1; c <= (variabel + kendala); c++) {
    boolean dasarkoneksi = false;
    for (int i = 1; i <= kendala; i++) {
        if (dasarkoneksi) {

            } else {
                if (KDS[i] == c) {
                    dasarkoneksi = true;
                }
            }
        }
        double absolut = 0 - Z[c];
        if ((absolut > PCV) && (!dasarkoneksi)) {
            PCV = absolut;
            PC = c;
        }
    }
    int PR = 0;
    double PRV = 0;
    for (int k = 1; k <= kendala; k++) {
        double absolut = S[k] / K[k][PC];
        if (k == 1) {
            PRV = absolut;
        }
        if ((absolut > 0) && (absolut <= PRV)) {
            PRV = absolut;
            PR = k;
        }
    }
}

System.out.println(PR + "-" + PC);
if (PC == 0) {
    optimal = true;
    tambah_teks("Jumlah Iterasi : " + itr);
    tambah_teks("Nilai Optimum Maks Z = " +
putaran(Z[variabel + kendala + 1]));
    for (int k = 1; k <= kendala; k++) {
        tambah_teks("Nilai X" + KDS[k] + " = " +
putaran(S[k]));
    }
}

if (!optimal) { // JIKA TABEL OPTIMAL
    KDS[PR] = PC;
    // PENENTUAN PIVOT BARU
    double pivot = 0.00;
    for (int k = 1; k <= (variabel + kendala);
k++) {
        if (k == 1) {
            pivot = K[PR][PC];
        }
        double sel = (K[PR][k] / pivot);
        K[PR][k] = sel;
    }
    S[PR] = S[PR] / pivot;
    // Garis Baru Lainnya
    for (int i = 1; i <= (kendala); i++) {
        if (i != PR) {

```

```

        pivot = K[i][PC];
        for (int k = 1; k <= variabel +
kendala; k++) {
            K[i][k] = K[i][k] - (pivot *
K[PR][k]);
        }
        S[i] = S[i] - (pivot * S[PR]);
    }
    // NILAI Z BARU
    pivot = Z[PC];
    for (int k = 1; k <= (variabel + kendala);
k++) {
        Z[k] = Z[k] - (pivot * K[PR][k]);
    }
    // NILAI SOLUSI Z
    int cz = variabel + kendala + 1;
    Z[cz] = Z[cz] - pivot * S[PR];
}
}
if (!Maks) {
    // SELAIN MODEL MAKSIMASI
    int PC = 0; // KOLOM UTAMA
    double PCV = 0; // PENENTUAN NILAI Z UTAMA
    for (int c = 1; c <= (variabel + kendala); c++) {
        boolean dasarkoneksi = false;
        for (int i = 1; i <= kendala; i++) {
            if (dasarkoneksi) {

            } else {
                if (KDS[i] == c) {
                    dasarkoneksi = true;
                }
            }
        }
        double absolut = Z[c];
        if ((absolut > PCV) && (!dasarkoneksi)) {
            PCV = absolut;
            PC = c;
        }
    }
    int PR = 0;
    double PRV = 0;
    for (int k = 1; k <= kendala; k++) {
        double absolut = S[k] / K[k][PC];
        if (k == 1) {
            PRV = absolut;
        }
        if ((absolut > 0) && (absolut <= PRV)) {
            PRV = absolut;
            PR = k;
        }
    }

    if (PC == 0) {
        optimal = true;
    }
    if (PC == 0) {

```

```

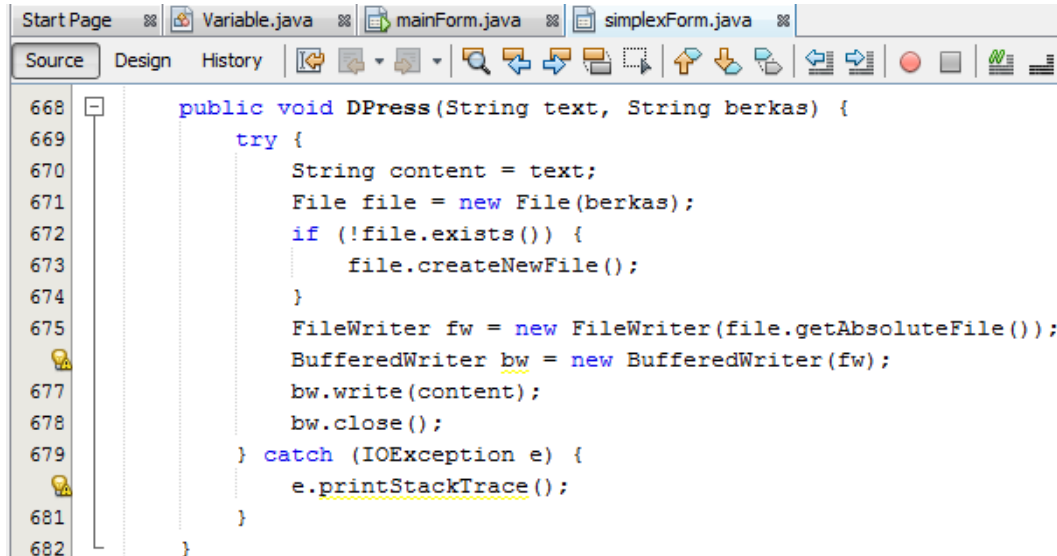
        optimal = true;
        tambah_teks("Jumlah Iterasi : " + itr);
        tambah_teks("Nilai Optimum Min Z = " +
putaran(Z[variabel + kendala + 1]));
        for (int k = 1; k <= kendala; k++) {
            tambah_teks("Nilai X" + KDS[k] + " = " +
putaran(S[k]));
        }
    }
    if (!optimal) { // JIKA TABEL OPTIMAL
        KDS[PR] = PC;
        // BARIS PIVOT BARU
        double pivot = 0.00;
        for (int k = 1; k <= (variabel + kendala);
k++) {
            if (k == 1) {
                pivot = K[PR][PC];
            }
            double sel = (K[PR][k] / pivot);
            K[PR][k] = sel;
        }
        S[PR] = S[PR] / pivot;
        // GARIS BARU LAINNYA
        for (int i = 1; i <= (kendala); i++) {
            if (i != PR) {
                pivot = K[i][PC];
                for (int k = 1; k <= variabel +
kendala; k++) {
                    K[i][k] = K[i][k] - (pivot *
K[PR][k]);
                }
                S[i] = S[i] - (pivot * S[PR]);
            }
        }
        // NILAI Z BARU
        pivot = Z[PC];
        for (int k = 1; k <= (variabel + kendala);
k++) {
            Z[k] = Z[k] - (pivot * K[PR][k]);
        }
        // NILAI SOLUSI Z
        int cz = variabel + kendala + 1;
        Z[cz] = Z[cz] - pivot * S[PR];
    }
    }
    itr++;
} while (!optimal);
// TABEL MODEL

}

public String spasi(int x) {
    String rtr = "";
    for (int i = 1; i <= x; i++) {
        rtr = rtr + " ";
    }
    return rtr;
}

```

Kemudian kita akan bahas Method yang fungsinya bisa Eksport File dari Model yang diinputkan User, prosedur yang pertama, kita deklarasikan Method FileWriter dengan dua Parameter String Text dan berkas :



```

668 public void DPRESS(String text, String berkas) {
669     try {
670         String content = text;
671         File file = new File(berkas);
672         if (!file.exists()) {
673             file.createNewFile();
674         }
675         FileWriter fw = new FileWriter(file.getAbsolutePath());
676         BufferedWriter bw = new BufferedWriter(fw);
677         bw.write(content);
678         bw.close();
679     } catch (IOException e) {
680         e.printStackTrace();
681     }
682 }

```

Lalu, User dapat mengakses Eksport model dengan Method Button SIMPAN MODEL KE EXCEL yang akan memanggil method diatas sebagai konstruktor, file akan disimpan dengan format .csv Exceld, Source Code :

```

private void jButton5ActionPerformed(java.awt.event.ActionEvent
evt) {
    JFileChooser FileChooser = new JFileChooser();
    FileChooser.setDialogTitle("Pilih tempat menyimpan
model");
    int userSelection = FileChooser.showSaveDialog(this);
    String File = "";
    if (userSelection == JFileChooser.APPROVE_OPTION) {
        File FileToSave = FileChooser.getSelectedFile();
        File = FileToSave.getAbsolutePath() + ".csv";
    }
    muat_matriks();
    String Target = jComboBox1.getSelectedItem().toString();
    String text = "D=" + variabel + "=K=" + kendala + ";";
    for (int k = 1; k <= (variabel); k++) {
        text = text + "X" + k + ";";
    }
    text = text + "SAMA DENGAN;SOLUSI;\n";
    text = text + Target + ";";
    for (int k = 1; k <= (variabel); k++) {
        text = text + (-1 * Z[k]) + ";";
    }
    text = text + ";;\n";
    for (int i = 1; i <= (kendala); i++) {
        text = text + "KENDALA;";
        for (int k = 1; k <= (variabel); k++) {
            text = text + putaran(K[i][k]) + ";";
        }
        text = text + E[i] + ";";
    }
}

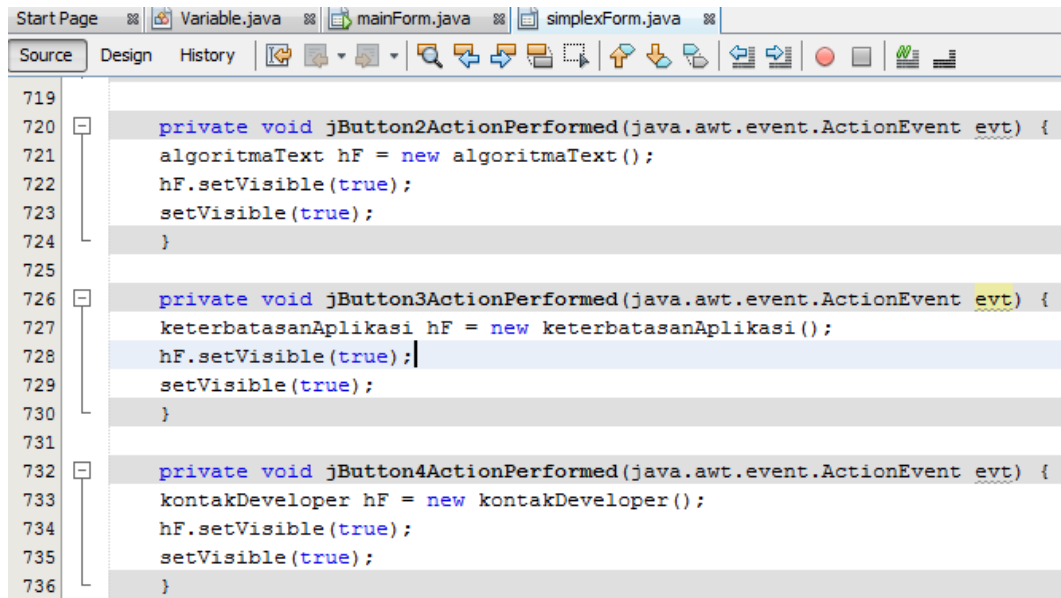
```

```

        text = text + S[i] + ";";
        text = text + "\n";
    }
    DPress(text, File);
}

```

Method – Method yang terlibat dalam proses inti Algoritma Simpleks sudah selesai, dibawah ini adalah method pelengkap yang gunanya adalah memanggil Class – Class diluar mainForm.java dan simplex.java :



Demikianlah Beberapa Source Code yang kiranya penting untuk dipahami menyangkut keutuhan sistem aplikasi, sebenarnya masih ada beberapa lagi yang bisa kita bahas yakni source kode diluar Class mainForm dan simplexForm, namun karena Class – Class tersebut berfungsi hanya sebagai pelengkap, maka cukup sekian saja yang kiranya bisa penulis jabarkan.

BAB IV

PENUTUP

A. Simpulan

Kesimpulan yang didapat adalah dalam menentukan Kasus Maksimasi suatu PL dalam simpleks perlu dipertimbangkan jumlah Variabel dan Jumlah Kendala dalam kasus tersebut dan batasan dari kendala haruslah Kurang Dari Sama Dengan " \leq ". Bila kendala bukan " \leq " Sistem akan menganggapnya tidak valid.

B. Saran

Saran yang dapat disampaikan adalah program yang penulis rancang kiranya bisa dikembangkan lagi untuk kasus Minimasi dan kiranya bisa pula di update lagi untuk meniadakan keterbatasan – keterbatasan yang ada dalam sistem aplikasi.

DAFTAR PUSTAKA

aimprof. 2016. “Metode Simpleks – Maksimasi”. <
<https://aimprof08.wordpress.com/2016/08/20/metode-simpleks-maksimasi-2/>>.
Diakses 20 Mei 2019.