**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 didefinisakan sebagai cara menyelesaikan permasalan yang memiliki variabel keputusan minimal dua dengan menggunalkan alat bantu tabel. Metode simpleks dibedakan menjadi dua yaitu, metode simpleks maksimasi untuk mencari keuntungan maksimal dan metode simpleks minimasi untuk mencari biaa minimal.

1. **Tujuan**

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

1. **Manfaat Praktikum**

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

**BAB II**

**PEMBAHASAN**

1. **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 dibuuthkan sebuah metode yang lebih kompleks yaitu dengan menggunakan program komputer QSB ( Quantitative System For Business) atau menggunakan metode simplex. Dalam kenyataanya 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 variable keputusan mulai dari lebih besar atau sama dengan 2 (dua) sampai multivariable. Sedangkan metode grafik hanya dapat digunalan apabila jumlah variable keputusan maksimal 2 (dua) buah. Sehingga dapat disimpulkan bahwa suatu persoalan linear programing 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 harusdikalikan –1.
3. Fungsi kendala dengan tanda “≤ atau ≥” 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:

1. Introduksikan variabel baru sebagai variable dummy dengan singkatan huruf S sebagai singkatan dari Slack (kekurangan) atau Surplus (kelebihan)
2. Variable slack kita introduksikan apabila kita mempunyai bentuk tanda pembatas lebih kecil atau sama dengan (≤)
3. Variabel surplus kita introduksikan apabila kita mempunyai bentuk tanda pembatas lebih besar dari atau sama dengan (≥)
4. Menyiapkan table simplex awal

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Cj** |  |  |  |  |  |  |  |
| **Ci** | **BV** | **X1** | **X2** | **Xn** | **S1** | **S2** | **Sn** | **Bi** |
|  | **S1** |  |  |  |  |  |  |  |
|  | **S2** |  |  |  |  |  |  |  |
|  | **Sn** |  |  |  |  |  |  |  |
|  | **Zj** |  |  |  |  |  |  |  |
| **Cj-Zj** | |  |  |  |  |  |  |  |

Penjelasan penggunaan tabel simplex :

1. Kolom Baris

* Kolom baris selalu ada dan ditempatkan di kolom paling kiri setelah Ci
* 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

1. Kolom Cj

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

1. Kolom diantara kolom Cj 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 juda akan salah.

1. Kolom nilai ruas kanan (NRK atau Bi)

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

1. Jumlah baris

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

1. Baris Zj

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

1. Baris Cj-Zj

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.

1. Memasukan nilai-nilai dan variable dalam formulasi standar ke dalam tabel awal
2. Melakukan proses literasi
3. Tentukan kunci kolom (pivot coloum).

Caranya adalah memilih nilai Cj-Zj yang terbesar dan positif.

1. Tentukan kunci baris (pivot row)

Caranya adalah memilih dasil 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.

1. 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.
2. 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).

1. Apabila sosialisasi optimal belum ditemukan maka kembali ke langkah 5a di atas, sehingga nilai yang terdapat pada baris Cj-Zj ≤ 0
2. Menentukan apakah penyelesaikan optimal sudah tercapai.
3. Membuat kesimpulan jawaban..
4. **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 diliha 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

X1 = Jumlah vanilla yang diproduksi/perhari

X2 = jumlah violette yang diproduksi/hari

**Langkah 1**

Formulasi LP (bentuk standar)

Fungsi tujuan 🡺 Zmax = 40X1 + 30X2

Fungsi kendala 🡺 I. 2X1 + 3X2 ≤ 60

II. 2X2 ≤ 30

III. 2X1 + 1X2 ≤ 40

IV. X1,X2 ≥ 0

Diubah menjadi:

2X1 + 3X2 + S1 + 0S2 + 0S3 = 60

2X2 + 0S1 + S2 + 0S3 = 30

2X1 + 1X2 + 0S1 + 0S2 + S3 = 40

40X1 + 30X2 + 0S1 + 0S2 + 0S3

C1 = 40, C2 = 30, C3 = 0, C4= 0, C5 = 0

**Langkah 2**

Tabel simplex awal masalah PT Yummy Food

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Cj** | 40 | 30 | 0 | 0 | 0 |  |
| **Ci** | **BV** | **X1** | **X2** | **S1** | **S2** | **S3** | **Bi** |
| 0 | **S1** | 2 | 3 | 1 | 0 | 0 | 60 |
| 0 | **S2** | 0 | 2 | 0 | 1 | 0 | 30 |
| 0 | **S3** | 2 | 1 | 0 | 0 | 1 | 40 |
|  | **Zj** | 0 | 0 | 0 | 0 | 0 | 0 |
| **Cj-Zj** | | 40 | 30 | 0 | 0 | 0 |  |

**Langkah 3**

Apakah tabel tersebut sudah optimal?Belum, karena tabel optimal bila nilai yang terdapat pada baris Cj – Zj ≤ 0

**Langkah 4**

Penyelesaian dengan cara iterasi :

1. Menentukan kolom kunci, yaitu kolom yang memiliki nilai Cj-Zj terbesar yaitu kolom X1. Dengan demikian X1 akan masuk dalam basis.
2. Menentukan baris kunci, yaitu baris yang memiliki angka indeks terkecil dan bukan negatif. Dalam hal ini baris S3. Dengan demikian S3 akan keluar dari basis dan tempatnya akan digantikan oleh X1.
3. Menetukan 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.

Angka baru = 40/2, 2/2, ½, 0/2, 0/2, ½

Atau = **20 1 ½ 0 0 ½**

1. Mencari angka baru pada baris lain, yaitu :

* **Baris S1**

Angka lama = [ 60 2 3 1 0 0 ]

Angka baru = [ 20 1 ½ 0 0 ½] **(2)**

Angka baru = **[20 0 2 1 0 -1]**

* **Baris S2**

Angka lama = [ 30 0 2 0 1 0]

Angka baru = [ 20 1 ½ 0 0 1/2] **(0)**

Angka baru = **[ 30 0 2 0 1 0]**

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Cj** | 40 | 30 | 0 | 0 | 0 |  |
| **Ci** | **BV** | **X1** | **X2** | **S1** | **S2** | **S3** | **Bi** |
| 0 | **S1** | 0 | 2 | 1 | 0 | -1 | 20 |
| 0 | **S2** | 0 | 2 | 0 | 1 | 0 | 30 |
| 40 | **X1** | 1 | ½ | 0 | 0 | ½ | 20 |
|  | **Zj** | 40 | 20 | 0 | 0 | 20 |  |
| **Cj-Zj** | | 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 : X2  (masuk dalam basis)
2. Baris kunci : S1 (keluar dari basis)
3. Anka 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.

Angka Baru = 20/2 , 0/2 , 2/2 , ½ , 0/2 , -1/2 ,

Atau = 10 0 1 ½ 0 -1/2

1. Angka baru pada baris lain :
   * + **Baris S2**

Angka Lama = [30 0 2 0 1 0]

Angka Baru = [10 0 1 ½ 0 -½] **(2)**

Angka Baru S2 = [10 0 0 -1 1 1]

* + - **Baris X1**

Angka Lama = [20 1 ½ 0 0 ½ ]

Angka Baru = [10 0 1 ½ 0 -½ ] **(½)**

Angka Baru X1 = [15 1 0 -¼ 0 ¾ ]

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Cj** | 40 | 30 | 0 | 0 | 0 |  |
| **Ci** | **BV** | **X1** | **X2** | **S1** | **S2** | **S3** | **Bi** |
| 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 |
|  | **Zj** | 40 | 30 | 5 | 0 | 15 |  |
| **Cj-ZJ** | | 0 | 0 | -5 | 0 | -15 | 900 |

Solusi optimum tabel iterasi 2 menunjukan bahwa total nilai Z = 900 dengan masing-masing variabel keputusan X1 = 15 dan X2 = 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 :

X1 (vanilla) = 15 unit

X2 (violette) = 10 unit

Z (keuntungan) = Rp 900,00

1. Kendala kedua (bahan baku B) masih tersisa sebanyak 10 Kg yang ditunjukan oleh nilai S2 =10, pada tabel optimal
2. Kendala 1 dan 3 tidak ada sisa (full capacity), yang ditunjukan oleh nilai S1 = S3 = 0 ( variabel nonbasis). Hal ini juga dapat dibuktikan dengan memasukan nilai S1 dan S2 ke dalam kendala 1 dan 3

Kendala 1 : 2X1 + 3X2 = 60

2 (15) + 3 (10) =60

60 = 60

Bahan baku yang digunakan = yang tersedia

Kendala 3 : 2X1 + 1X2 = 40

2 (15) + 1(10) =40

40 = 40

Jam kerja yang digunakan = yang tersedia

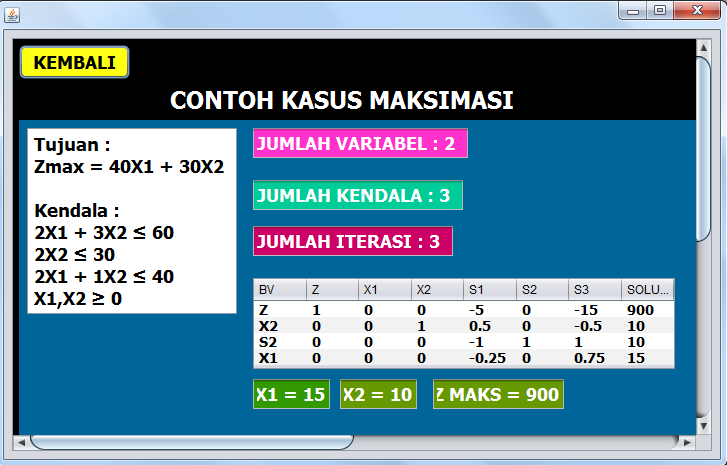
**BAB III**

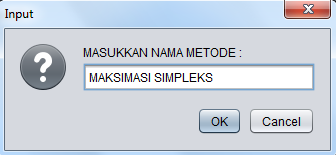
**HASIL APLIKASI**

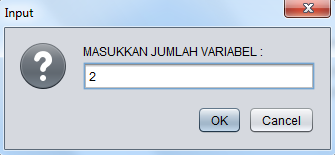
Berikut ini beberapa tampilan Antarmuka aplikasi :

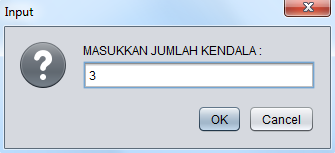
* Menu Utama



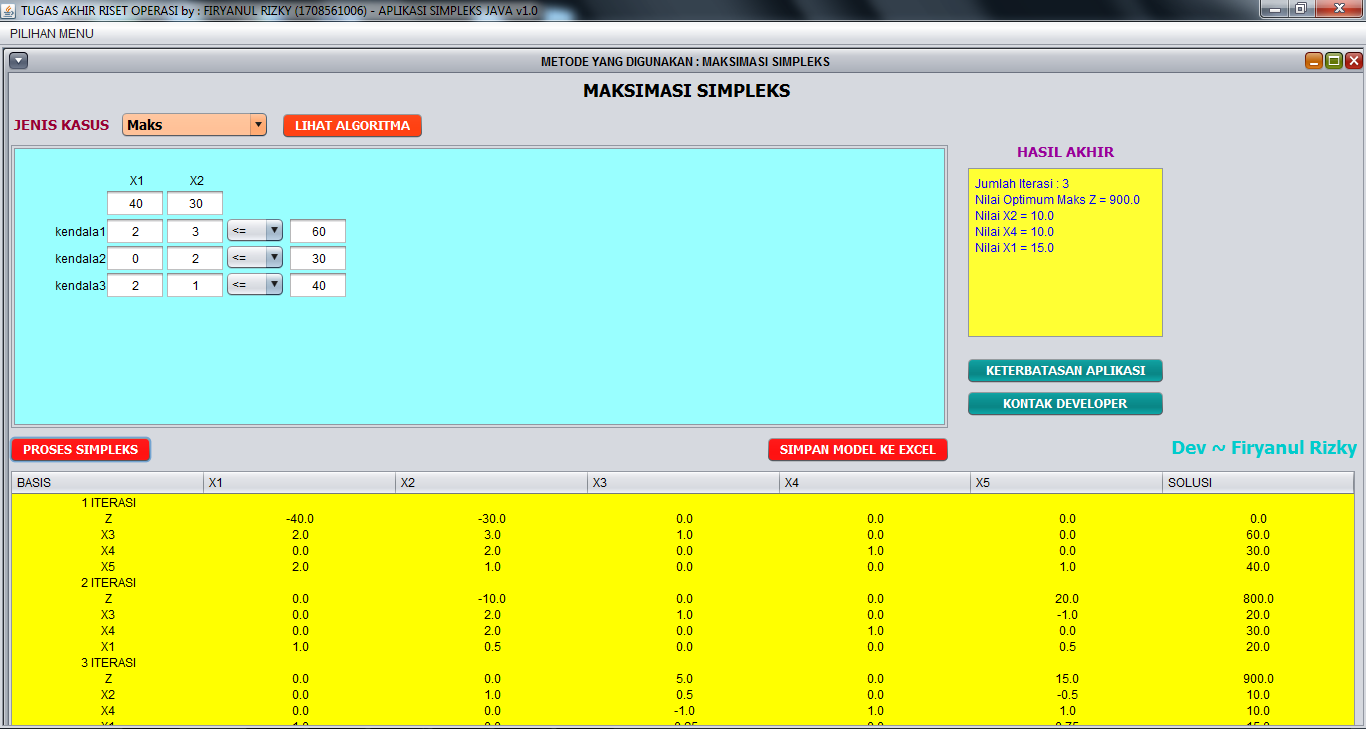
* Halaman CONTOH SOAL YANG BISA DICOBA
* Halaman KELEBIHAN APLIKASI
* Halaman KONTAK DEVELOPER
* Input Nama Metode

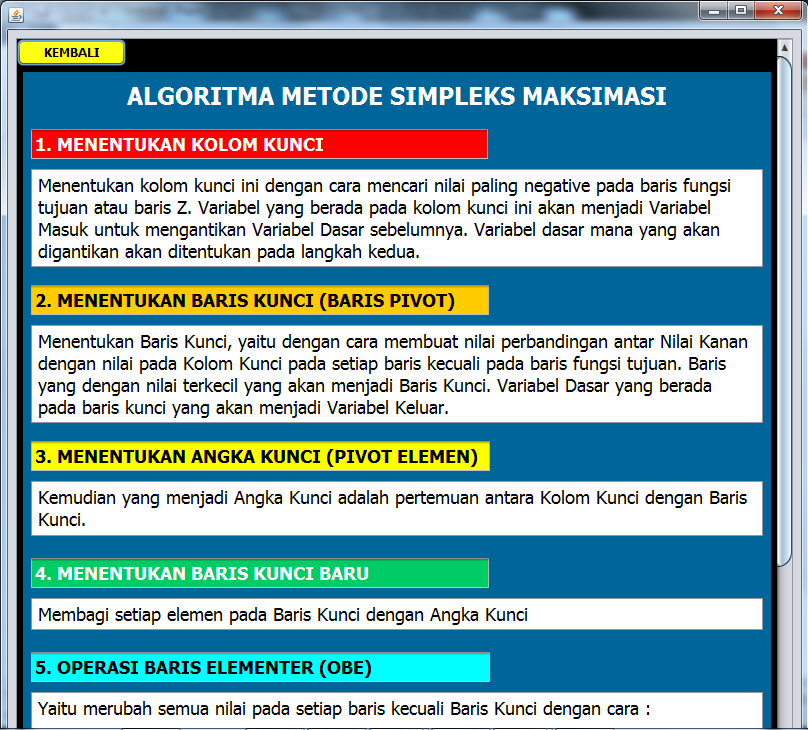


* Input Jumlah Variabel
* Input Jumlah Kendala



* Tampilan Antarmuka Proses

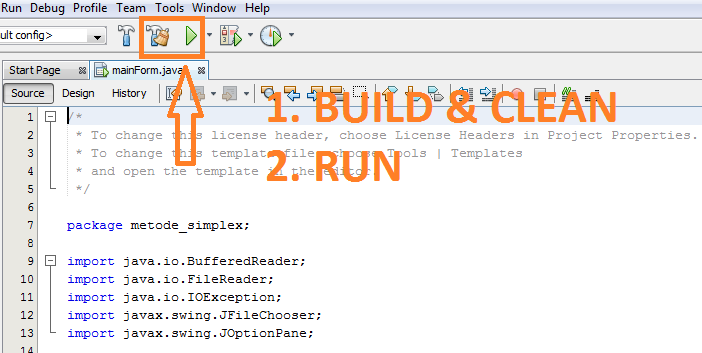


* Halaman Lihat Algoritma

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.

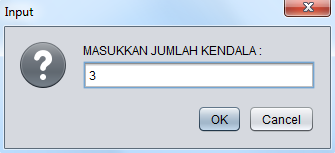
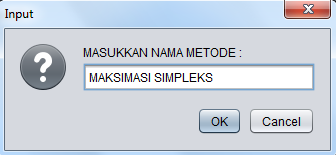
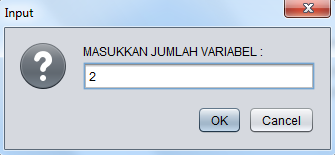
Penjelasan Class Java yang terdapat dalam package metode\_simpleks :

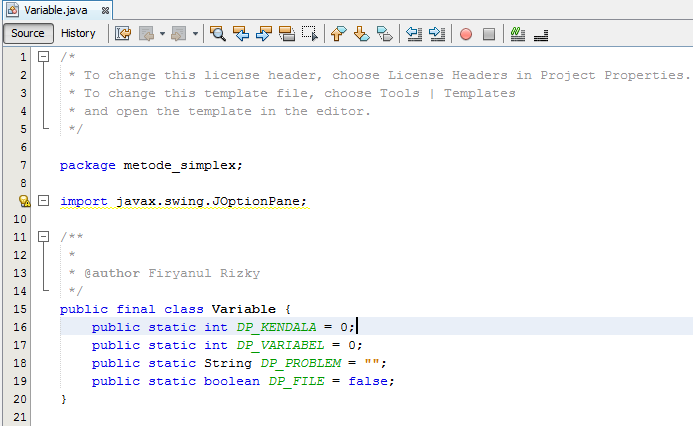
|  |  |  |
| --- | --- | --- |
| Nama Class | | Penjelasan |
| Variabel.java | : | Class Java yang berfungsi mendeklarasi nilai awal dari variabel – variabel yang dipakai pada mainForm yang menyimpan inputan untuk Nama Metode, Jumlah Variabel, dan Jumlah Kendala. |
| algoritmaText.java | : | Class Java Swing yang berisikan GUI dan kodingan 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 kodingan 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 kodingan 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 kodingan untuk menampilkan Halaman KETERBATASAN APLIKASI yang dapat diakses melalui Combo Box KETERBATASAN APLIKASI pada Menu Utama dan Pemrosesasn Simpleks. |
| kontakDeveloper.java | : | Class Java Swing yang berisikan GUI dan kodingan untuk menampilkan Halaman KONTAK DEVELOPER yang dapat diakses melalui Combo Box KONTAK DEVELOPER pada Menu Utama dan Pemrosesasn 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). |

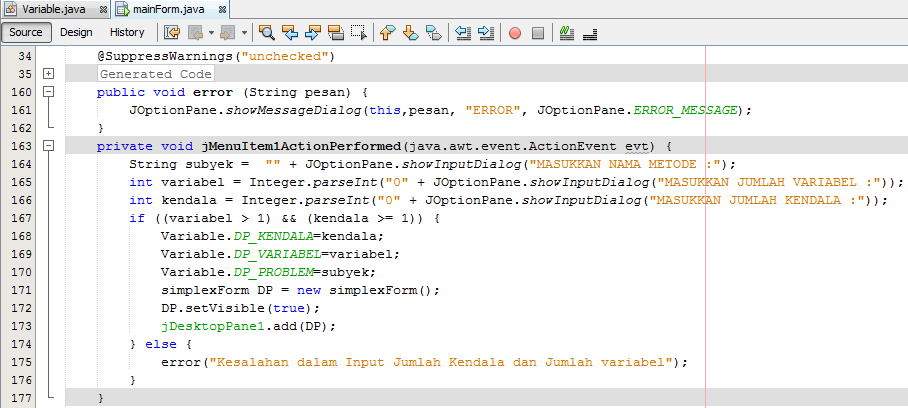
Setelah Di Run, maka akan tampil Menu Utama sebagai berikut :

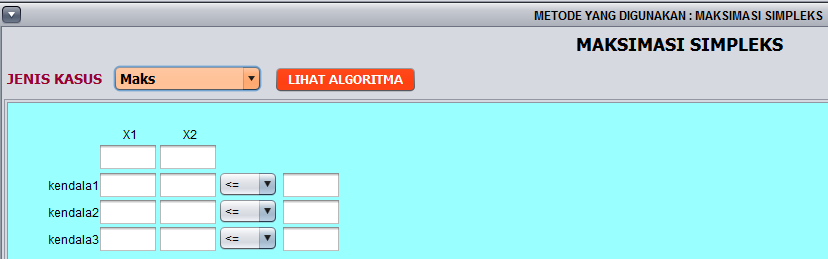
KLIK pada PILIHAN MENU, disana disuguhkan 2 pilihan :

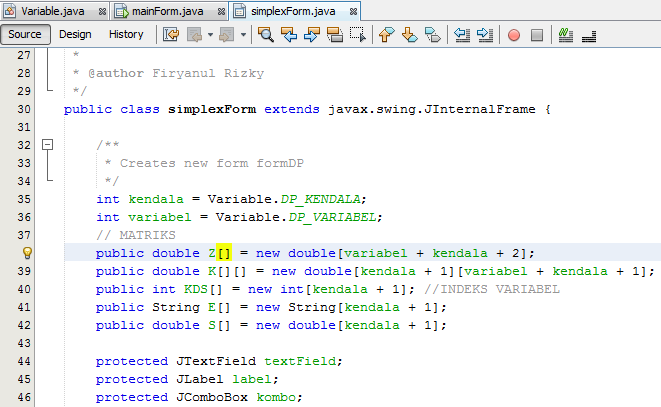
1. **Pemecahan Model**, ini adalah metode input secara manual, untuk User yang baru menjalankan aplikasi, disarankan untuk memulai dengan pilihan pertama ini.
2. **Import Model dari Excel**, ini adalah metode input otomatis apabila User sudah punya data Eksport berformatkan .csv Excel, File Eksport hanya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

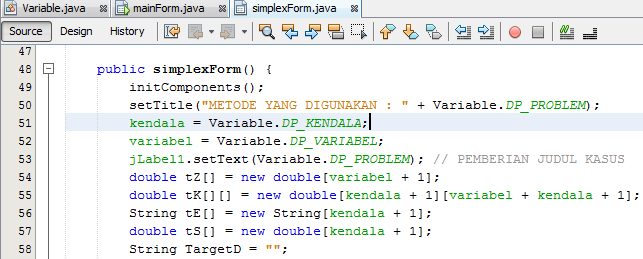
Inputan tersebut pada awalnya akan diset sebagai public Variabel dengan nilai awal 0 dengan jenis Variabel int, String “ “ dan boolean false pada Class Varibel.java dengan source Code sebagai berikut :

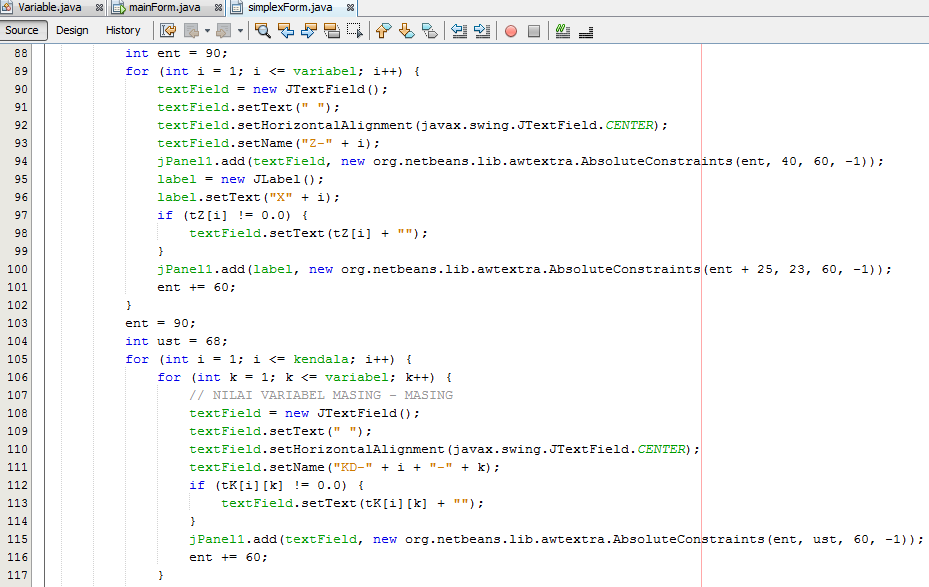
Penampilan Antarmuka Input form adalah tugas dari Class mainForm.java dengan memanfaatkan Java Swing Option Pane, pada class ini Varibel awal yang diset pada class Variabel.java diberikan variabel baru untuk mendapatkan hasil output yang dinamis, setelah diberi variabel proses akan memanggil simplexForm.java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

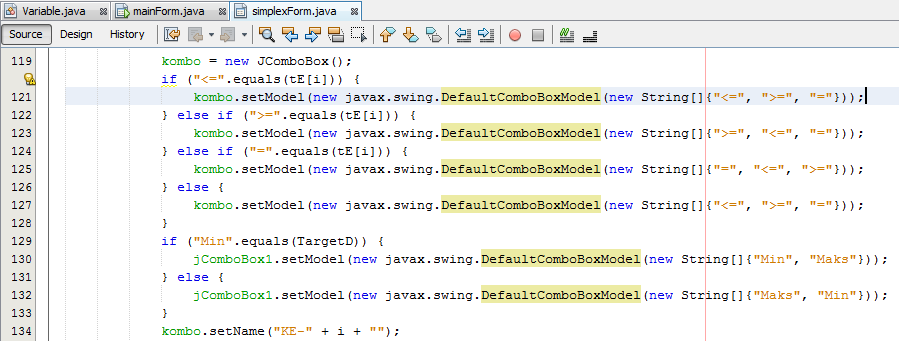
Setelah User selesai menginputkan data yang diminta, selanjutnya akan dialihkan ke Halaman Pemrosesan Simpleks, pada halaman ini, User diminta untuk menginputkan nilai pada masing – masing nilai Z dan kendala Kasus :

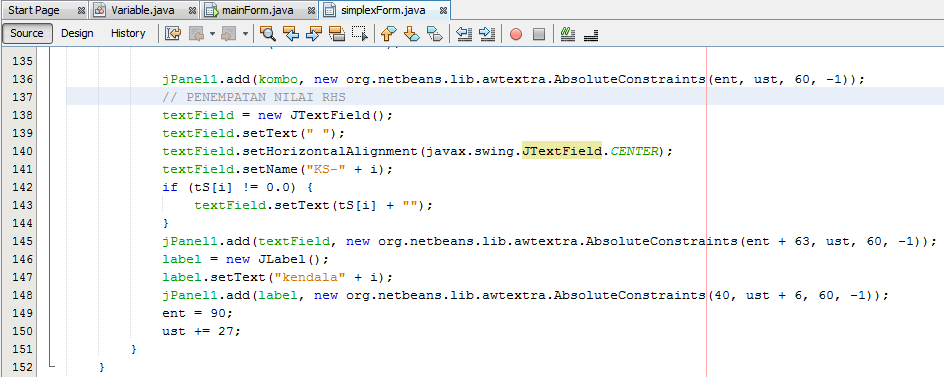
Untuk Memperoleh tampilan yang dinamis seperti gambar diatas, dilakukan pembentukan method GUI Internal Frame yang didalamnya juga menginisialisasi method dan object baru bersamaan dengan pemanggilan object GUI seperti textField, label dan opsi pilihan Combo pada Class simplexForm.java dengan Source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

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.

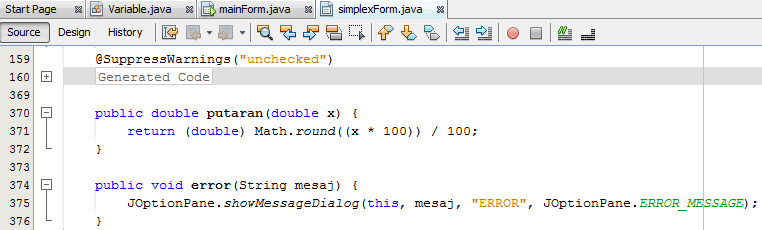
Pada Judul paling atas kita set pada Title Internal Form dengan menampilkan variabel inputan User, sedangkan judul dibawahnya diset pada Label, Berikut Source Code untuk penampilan Judul/Title sesuai dengan Input yang berasal dari User :

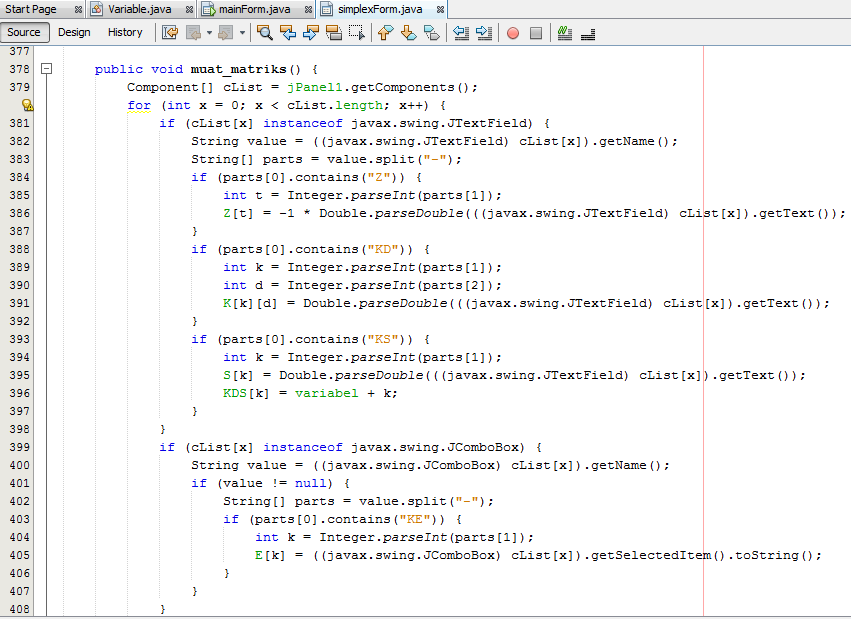
Penempatan textField dilakukan secara dinamis pada suatu Jpanel GUI yang diprogram sedemikian rupa dengan memanggil variabel Array di method sebelumnya, Source Code seperti berikut :

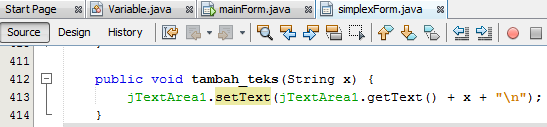
Penempatan ComboBox dilakukan dengan dinamis. melibatkan variabel Array, koding programnya ditulis setelah finish state dari Source Code diatas, Source Codenya sebagai berikut

Berikutnya adalah penentuan Jumlah Textfield untuk Input Nilai RHS yang disesuailkan 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.

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



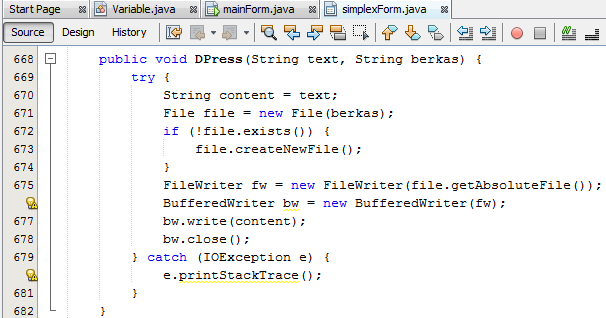
Kemudian proses akan memuat keseluruhan matriks yang telah diinputkan User meliputi Kolom Diagonal, Baris Horizontal, dan RHS, serta akan mengklarifikasi bahwa jika salah satu element tidak terisi maka akan diberi tanda “-“ dan tidak akan dihitung sistem, Source Codenya sebagai berikut :

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 :

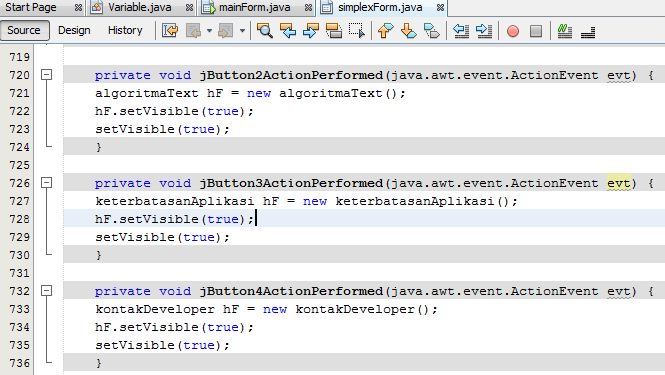
|  |
| --- |
| private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  // MATRIKS PENCIPTAAN DARI VARIABEL  muat\_matriks();  // FORMASI MATRIKS VARIABEL  String Target = jComboBox1.getSelectedItem().toString();  jTable1.setModel(new DefaultTableModel());  jTextArea1.setText("");  boolean 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(centerRenderer);  }  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 :



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**

1. **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 “<=”. Bila kendala bukan “<=” Sistem akan menganggapnya tidak valid.

1. **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 – keterbatasn 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.