APLIKASI PENJADWALAN KULIAH BERBASIS ALGORITMA GENETIKA DI PROGRAM STUDI S1 INFORMATIKA UNIVERSITAS TEKNOLOGI YOGYAKARTA

Abstract

Aplikasi ini dirancang untuk menyusun jadwal kuliah secara otomatis di Program Studi S1 Informatika, Universitas Teknologi Yogyakarta. Aplikasi tersebut menerapkan Algoritma Genetika dengan modifikasi pada proses evolusinya, di mana proses rekombinasi ditiadakan untuk mempercepat komputasi. Program tersebut mampu menyusun jadwal perkuliahan di tingkat Program Studi yang terdiri dari kurang lebih 150 pertemuan perkuliahan tanpa terjadi bentrok pada jadwal dosen, ruangan, dan mahasiswa. Jadwal kuliah yang dihasilkan oleh aplikasi berisi hari, waktu, ruang, mata kuliah, dosen, dan kelas dari tiap perkuliahan, serta dapat langsung disimpan dalam dokumen bertipe .xls.

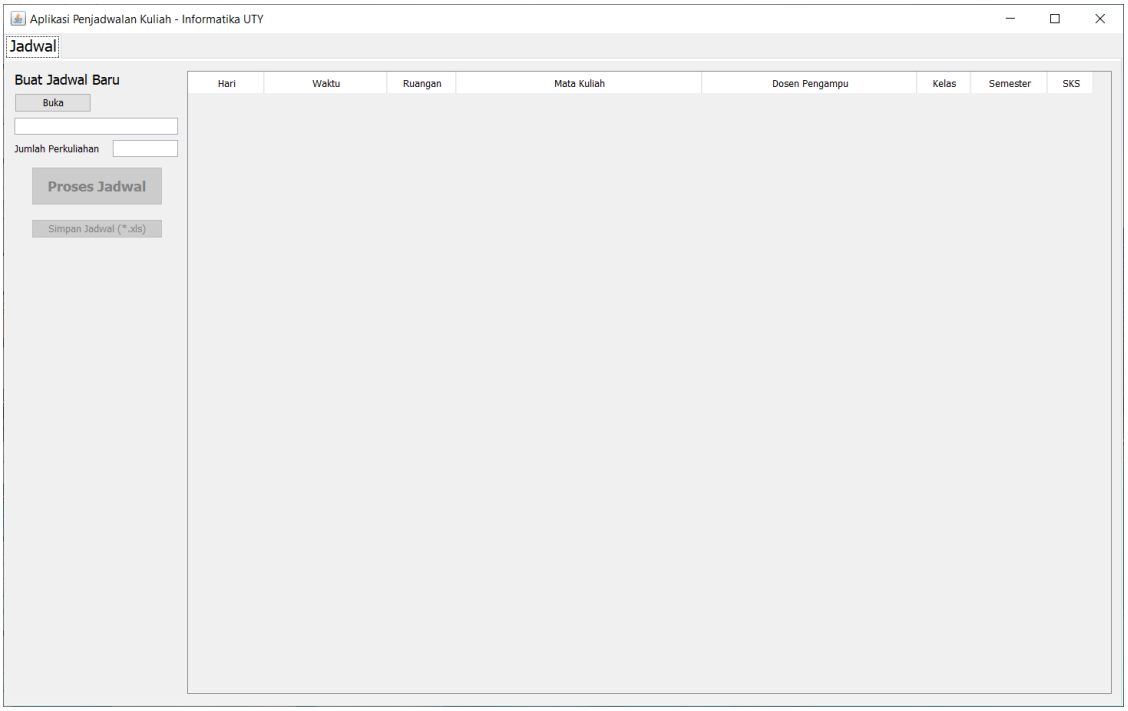
Muhammad Fachrie, S.T., M.Cs.

Muhammad.fachrie@staff.uty.ac.id

A. Penjelasan tentang Aplikasi

Aplikasi ini dirancang untuk memenuhi kebutuhan penjadwalan kuliah di Program Studi S1 Informatika, Universitas Teknologi Yogyakarta (UTY). Gambar 1 merupakan tampilan utama dari aplikasi yang dibuat. Aplikasi ini dapat melakukan penyusunan jadwal kuliah yang meliputi ruang, waktu, dan hari perkuliahan dengan memperhatikan beberapa batasan (*constraint*) yang terbagi menjadi dua macam batasan, yakni *hard constraint* dan *soft constraint*. *Hard constraint* merupakan batasan-batasan yang tidak boleh dilanggar, di antaranya satu dosen hanya boleh mengajar satu mata kuliah pada satu waktu, satu ruangan hanya boleh digunakan oleh satu perkuliahan dalam satu waktu, satu kelas hanya boleh mengikuti satu perkuliahan pada satu waktu. Sedangkan *soft constraint* merupakan batasan-batasan yang sifatnya fleksibel, meski demikian batasan tersebut sebaiknya tidak dilanggar, misalnya satu dosen sebaiknya tidak mengajar lebih dari 6 SKS dalam satu hari, satu dosen sebaiknya mengajar pada jam terakhir tidak lebih dari 2 kali dalam seminggu, satu kelas sebaiknya hanya mengikuti perkuliahan maksimal 6 sks dalam sehari, dll. Pelanggaran terhadap *hard constraint* mengakibatkan jadwal yang dihasilkan tidak valid, sedangkan pelanggaran pada *soft constraint* tetap menghasilkan jadwal yang valid.

Aplikasi ini menerima input berupa dokumen bertipe .xls yang berisi data dosen, mata kuliah, sks, dan kelas mahasiswa yang mengikuti perkuliahan. Pada akhir pemrosesan, aplikasi akan menghasilkan keluaran berupa jadwal perkuliahan yang memenuhi batasan-batasan yang didefinisikan sebelumnya dalam format dokumen .xls.

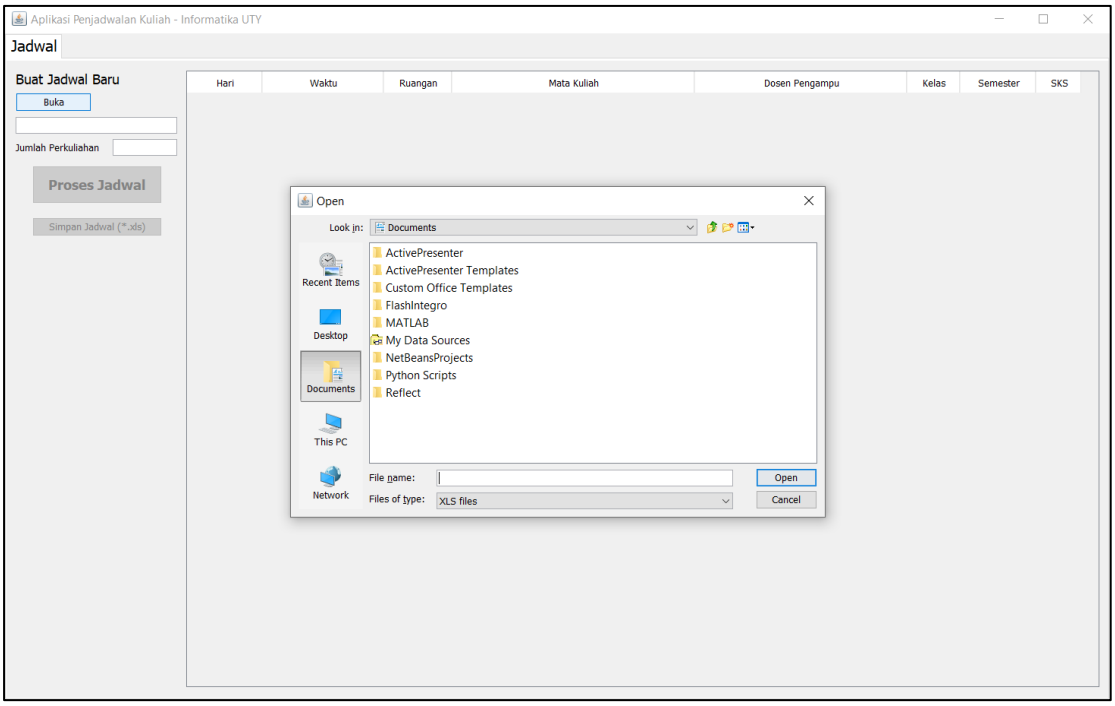


Gambar 1. Tampilan awal aplikasi penjadwalan kuliah yang dibuat

Penggunaan aplikasi ini tergolong sangat mudah. Cukup dengan menyiapkan dokumen .xls berisi data dosen dan mata kuliah yang diajar beserta sks dan kelas mahasiswa yang mengikutinya, kemudian cukup klik satu tombol, kemudian setelah beberapa menit aplikasi akan menampilkan jadwal baru yang

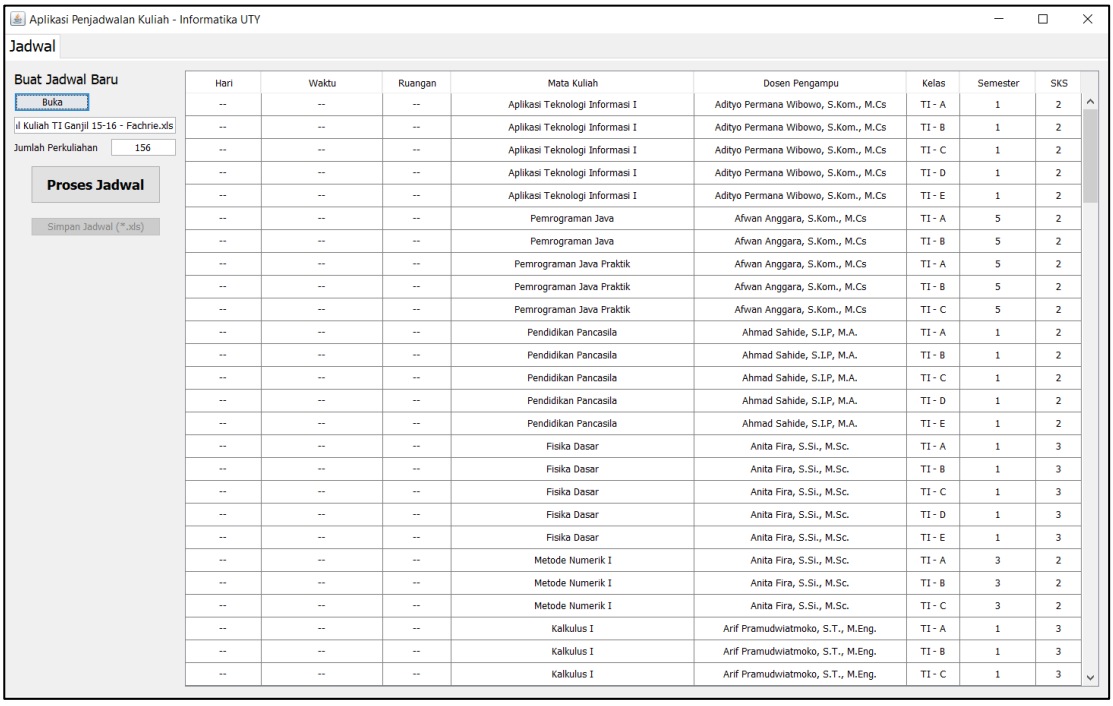
bebas dari bentrok. Berikut ini adalah petunjuk penggunaan aplikasi penjadwalan kuliah yang telah dibuat:

1. Klik tombol ‘Buka’ yang ada di pojok kiri atas aplikasi. Kemudian akan muncul jendela baru seperti yang terlihat pada Gambar 2 untuk mencari dokumen bertipe .xls yang berisi data dosen, mata kuliah, sks, dan kelas mahasiswa yang diajar.



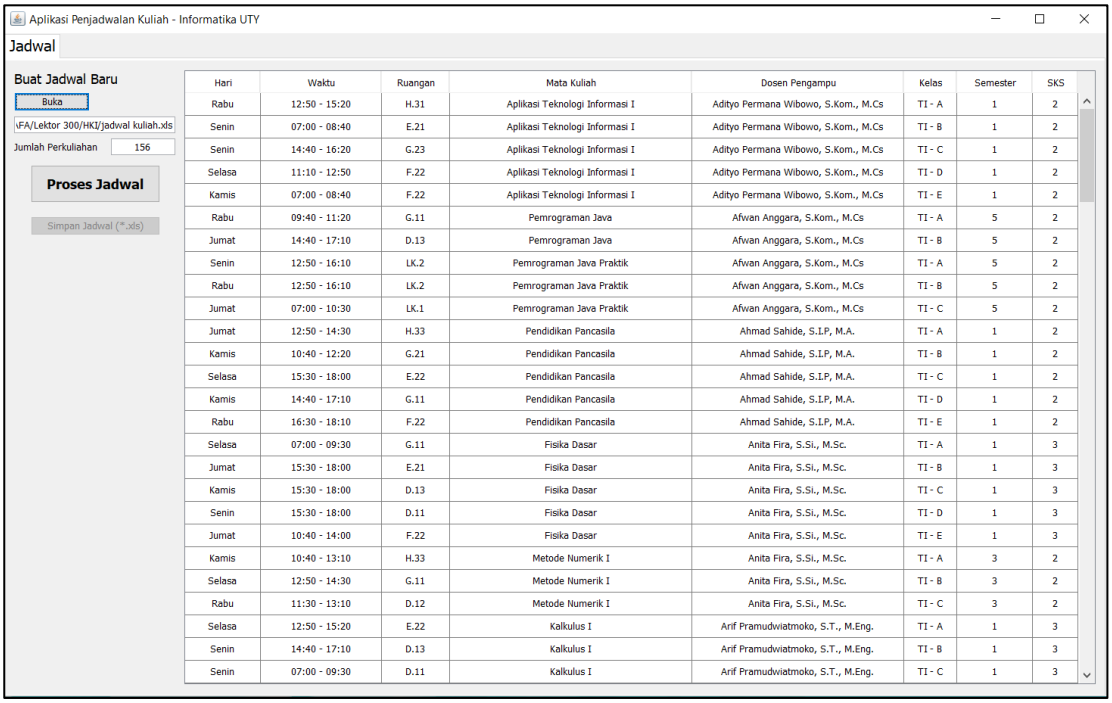
Gambar 2. Tampilan jendela untuk membuka dokumen .xls yang akan disusun jadwalnya

2. Setelah dokumen tersebut ditemukan, maka klik tombol ‘Open’, lalu isi dokumen tersebut akan tampil pada aplikasi seperti yang terlihat pada Gambar 3.



Gambar 3. Isi dokumen .xls yang tampil pada aplikasi

3. Setelah itu, langsung klik tombol ‘Proses Jadwal’ untuk melakukan penjadwalan otomatis. Setelah beberapa menit, aplikasi akan menampilkan jadwal kuliah yang optimal seperti pada Gambar 4. Jadwal kuliah tersebut terbebas dari bentrok dan sesuai dengan batasan-batasan yang didefinisikan sebelumnya.



Hari	Waktu	Ruangan	Mata Kuliah	Dosen Pengampu	Kelas	Semester	SKS
Rabu	12:50 - 15:20	H.31	Aplikasi Teknologi Informasi I	Adityo Permana Wibowo, S.Kom., M.Cs	TI - A	1	2
Senin	07:00 - 08:40	E.21	Aplikasi Teknologi Informasi I	Adityo Permana Wibowo, S.Kom., M.Cs	TI - B	1	2
Senin	14:40 - 16:20	G.23	Aplikasi Teknologi Informasi I	Adityo Permana Wibowo, S.Kom., M.Cs	TI - C	1	2
Selasa	11:10 - 12:50	F.22	Aplikasi Teknologi Informasi I	Adityo Permana Wibowo, S.Kom., M.Cs	TI - D	1	2
Kamis	07:00 - 08:40	F.22	Aplikasi Teknologi Informasi I	Adityo Permana Wibowo, S.Kom., M.Cs	TI - E	1	2
Rabu	09:40 - 11:20	G.11	Pemrograman Java	Afwan Anggara, S.Kom., M.Cs	TI - A	5	2
Jumat	14:40 - 17:10	D.13	Pemrograman Java	Afwan Anggara, S.Kom., M.Cs	TI - B	5	2
Senin	12:50 - 16:10	LK.2	Pemrograman Java Praktik	Afwan Anggara, S.Kom., M.Cs	TI - A	5	2
Rabu	12:50 - 16:10	LK.2	Pemrograman Java Praktik	Afwan Anggara, S.Kom., M.Cs	TI - B	5	2
Jumat	07:00 - 10:30	LK.1	Pemrograman Java Praktik	Afwan Anggara, S.Kom., M.Cs	TI - C	5	2
Jumat	12:50 - 14:30	H.33	Pendidikan Pancasila	Ahmad Sahide, S.I.P, M.A.	TI - A	1	2
Kamis	10:40 - 12:20	G.21	Pendidikan Pancasila	Ahmad Sahide, S.I.P, M.A.	TI - B	1	2
Selasa	15:30 - 18:00	E.22	Pendidikan Pancasila	Ahmad Sahide, S.I.P, M.A.	TI - C	1	2
Kamis	14:40 - 17:10	G.11	Pendidikan Pancasila	Ahmad Sahide, S.I.P, M.A.	TI - D	1	2
Rabu	16:30 - 18:10	F.22	Pendidikan Pancasila	Ahmad Sahide, S.I.P, M.A.	TI - E	1	2
Selasa	07:00 - 09:30	G.11	Fisika Dasar	Anita Fira, S.Si., M.Sc.	TI - A	1	3
Jumat	15:30 - 18:00	E.21	Fisika Dasar	Anita Fira, S.Si., M.Sc.	TI - B	1	3
Kamis	15:30 - 18:00	D.13	Fisika Dasar	Anita Fira, S.Si., M.Sc.	TI - C	1	3
Senin	15:30 - 18:00	D.11	Fisika Dasar	Anita Fira, S.Si., M.Sc.	TI - D	1	3
Jumat	10:40 - 14:00	F.22	Fisika Dasar	Anita Fira, S.Si., M.Sc.	TI - E	1	3
Kamis	10:40 - 13:10	H.33	Metode Numerik I	Anita Fira, S.Si., M.Sc.	TI - A	3	2
Selasa	12:50 - 14:30	G.11	Metode Numerik I	Anita Fira, S.Si., M.Sc.	TI - B	3	2
Rabu	11:30 - 13:10	D.12	Metode Numerik I	Anita Fira, S.Si., M.Sc.	TI - C	3	2
Selasa	12:50 - 15:20	E.22	Kalkulus I	Arif Pramudwiatmoko, S.T., M.Eng.	TI - A	1	3
Senin	14:40 - 17:10	D.13	Kalkulus I	Arif Pramudwiatmoko, S.T., M.Eng.	TI - B	1	3
Senin	07:00 - 09:30	D.11	Kalkulus I	Arif Pramudwiatmoko, S.T., M.Eng.	TI - C	1	3

Gambar 4. Aplikasi menampilkan jadwal kuliah hasil pemrosesan

B. Potongan Kode Sumber

MainForm.java

```
01. package timetable_uty;
02. 
03. import java.awt.Toolkit;
04. import java.io.File;
05. import java.io.IOException;
06. import java.util.logging.Level;
07. import java.util.logging.Logger;
08. import javax.swing.JFileChooser;
09. import javax.swing.JTable;
10. import javax.swing.SwingConstants;
11. import javax.swing.UIManager;
12. import static javax.swing.UIManager.getString;
13. import javax.swing.UnsupportedLookAndFeelException;
14. import javax.swing.filechooser.FileNameExtensionFilter;
15. import javax.swing.table.DefaultTableCellRenderer;
16. import javax.swing.table.DefaultTableModel;
17. import javax.swing.table.TableModel;
18. import java.util.*;
19. import javax.swing.DefaultComboBoxModel;
20. import javax.swing.DefaultRowSorter;
21. import javax.swing.JFrame;
22. import javax.swing.JOptionPane;
23. import javax.swing.RowSorter;
24. import javax.swing.table.TableRowSorter;
25. import jxl.Cell;
26. import jxl.Sheet;
27. import jxl.Workbook;
28. import jxl.read.biff.BiffException;
```

```
29.
30. /**
31.  *
32.  * @author fachr
33.  */
34. public class MainForm extends javax.swing.JFrame {
35.
36.     // Boolean isJComboBoxReady --> for jComboBox1ActionPerformed
37.     boolean isJComboBoxReady;
38.     int valid;
39.     static int jumHari = 5;
40.
41.     //Var to state whether user is opening DataKuliah or DataSiapOlah
42.     static String status = "";
43.
44.     //for constraint
45.     static int[] Constraint = new int[8];
46.     static int[] StatusConstraint = new int[8];
47.     static int[] MaxBatas = new int[3];
48.     static int JumPerkuliahan;
49.
50.     static int FixSlot;
51.     static int[] NomorSlot;
52.     static String[][] RuangWaktu;
53.     String[] Hari = {"Senin", "Selasa", "Rabu", "Kamis", "Jumat"};
54.
55.     public MainForm() {
56.         initComponents();
57.         //setExtendedState(MAXIMIZED_BOTH);
58.
59.         //Clear the jTable
60.         DefaultTableModel model1 = (DefaultTableModel) jTable1.getModel();
61.         model1.setRowCount(0);
62.
63.         // Align the header text to the center
64.         DefaultTableCellRenderer renderer1 = (DefaultTableCellRenderer) jTable1.getTableHeader().getDefaultRenderer();
65.         renderer1.setHorizontalAlignment(0);
66.
67.         //Align the cell value center
68.         DefaultTableCellRenderer centerRenderer = new DefaultTableCellRenderer();
69.         centerRenderer.setHorizontalAlignment(SwingConstants.CENTER);
70.         jTable1.getColumnModel().getColumn(0).setCellRenderer(centerRenderer);
71.         jTable1.getColumnModel().getColumn(1).setCellRenderer(centerRenderer);
72.         jTable1.getColumnModel().getColumn(2).setCellRenderer(centerRenderer);
73.         jTable1.getColumnModel().getColumn(3).setCellRenderer(centerRenderer);
74.         jTable1.getColumnModel().getColumn(4).setCellRenderer(centerRenderer);
75.         jTable1.getColumnModel().getColumn(5).setCellRenderer(centerRenderer);
76.         jTable1.getColumnModel().getColumn(6).setCellRenderer(centerRenderer);
77.         jTable1.getColumnModel().getColumn(7).setCellRenderer(centerRenderer);
78.
79.         GenerateSlotJadwal();
80.     }
```

BacaDataSiapOlah.java

```
01. package timetable_uty;
02.
03. import java.io.File;
04. import java.io.IOException;
05. import java.util.ArrayList;
06. import java.util.Arrays;
07. import java.util.Collections;
08. import jxl.Cell;
09. //import jxl.CellType;
10. import jxl.Sheet;
11. import jxl.Workbook;
12. import jxl.read.biff.BiffException;
13. //import java.util.*;
14. import static java.util.Objects.isNull;
15. import javax.swing.JDialog;
16. import javax.swing.JFrame;
17. import javax.swing.JOptionPane;
18.
19. public class BacaDataSiapOlah {
20.
21.     static String[][] DataMentah;
22.     private String inputFile;
23.     static int numRow, numCol, sizeData;
24.     static ArrayList DataHari, DataDosen, DataKuliah, DataRuang, DataSesi;
25.
26.     public void setInputFile(String inputFile) {
27.         this.inputFile = inputFile;
28.     }
29.
```

```
30.     public void read() throws IOException {
31.         File inputWorkbook = new File(inputFile);
32.         Workbook w;
33.         int[] idxData = {0, 0, 0, 0, 0};
34.         ArrayList tempArr = new ArrayList();
35.
36.         try {
37.             w = Workbook.getWorkbook(inputWorkbook);
38.             // Get the first sheet
39.             Sheet sheet = w.getSheet(0);
40.             // Loop over first 10 column and lines
41.
42.             numRow = sheet.getRows();
43.             numCol = 8;
44.             sizeData = numRow * numCol;
45.             //System.out.println(numRow);
46.
47.             DataMentah = new String[numRow][numCol];
48.             DataHari = new ArrayList();
49.             DataDosen = new ArrayList();
50.             DataKuliah = new ArrayList();
51.             DataRuang = new ArrayList();
52.             DataSesi = new ArrayList();
53.
54.             for (int j = 0; j < numCol; j++) {
55.                 tempArr.removeAll(tempArr);
56.                 //System.out.println(tempArr.size());
57.                 for (int i = 1; i < numRow; i++) {  //from 1: row 0 is header
58.                     Cell cell = sheet.getCell(j, i);
59.                     String data = cell.getContents();
60.
61.                     //check if the current row is empty
62.                     int empty = 0;
63.                     for (int jj = 0; jj < numCol; jj++) {
64.                         Cell cekCell = sheet.getCell(jj, i);
65.                         String cekData = cekCell.getContents();
66.
67.                         if (isNull(cekData) || ("".equals(cekData))) {
68.                             empty++;
69.                         }
70.                     }
71.
72.                     System.out.println("empty " + empty);
73.                     if (empty == numCol) {
74.                         numRow = i;   //get the real numRow
75.                         break;
76.                     } else if (isNull(data) || "".equals(data)) {
77.                             DataMentah[i - 1][j] = "--";
78.                             tempArr.add(i - 1, "--");
79.                     } else {
80.                         DataMentah[i - 1][j] = data;
81.                         tempArr.add(i - 1, data);
82.                     }
83.                 }
84.
85.                 //System.out.println("Cek ya " + numRow);
86.                 //sorting tempArr (casting toArray because it is an ArrayList)
87.                 Collections.sort(tempArr);  //this is the method to sort ArrayList. Use it instead of Arrays.sort()
88.
89.                 //Save DataDosen, DataKuliah, DataRuang, DataSesi (for COMBOBOX)
90.                 for (int i = 0; i < tempArr.size(); i++) {
91.                     //System.out.println(tempArr.size());
92.                     switch (j) {
93.                         case 0:
94.                             if (idxData[0] > 0) {
95.                                 if (tempArr.get(i).equals(DataHari.get(idxData[0] - 1)) == false) {
96.                                     DataHari.add(idxData[0], tempArr.get(i));
97.                                     //System.out.println(DataSesi.get(idxData[0]));
98.                                     idxData[0]++;
99.                                 }
100.                            } else {
101.                                DataHari.add(idxData[0], tempArr.get(i));
102.                                //System.out.println(DataSesi.get(idxData[0]));
103.                                idxData[0]++;
104.                            }
105.                            break;
106.                        case 1:
107.                            if (idxData[1] > 0) {
108.                                if (tempArr.get(i).equals(DataSesi.get(idxData[1] - 1)) == false) {
109.                                    DataSesi.add(idxData[1], tempArr.get(i));
110.                                    //System.out.println(DataSesi.get(idxData[0]));
111.                                    idxData[1]++;
112.                                }
113.                            } else {
114.                                DataSesi.add(idxData[1], tempArr.get(i));
115.                                //System.out.println(DataSesi.get(idxData[0]));
116.                                idxData[1]++;
117.                            }
```

```
                            break;
                        case 2:
                            if (idxData[2] > 0) {
                                if (tempArr.get(i).equals(DataRuang.get(idxData[2] - 1)) == false) {
                                    DataRuang.add(idxData[2], tempArr.get(i));
                                    //System.out.println(DataRuang.get(idxData[1]));
                                    idxData[2]++;
                                }
                            } else {
                                DataRuang.add(idxData[2], tempArr.get(i));
                                //System.out.println(DataRuang.get(idxData[1]));
                                idxData[2]++;
                            }
                            break;
                        case 3:
                            if (idxData[3] > 0) {
                                if (tempArr.get(i).equals(DataKuliah.get(idxData[3] - 1)) == false) {
                                    DataKuliah.add(idxData[3], tempArr.get(i));
                                    //System.out.println(DataKuliah.get(idxData[2]));
                                    idxData[3]++;
                                }
                            } else {
                                DataKuliah.add(idxData[3], tempArr.get(i));
                                //System.out.println(DataKuliah.get(idxData[2]));
                                idxData[3]++;
                            }
                            break;
                        case 4:
                            if (idxData[4] > 0) {
                                if (tempArr.get(i).equals(DataDosen.get(idxData[4] - 1)) == false) {
                                    DataDosen.add(idxData[4], tempArr.get(i));
                                    //System.out.println(DataDosen.get(idxData[3]));
                                    idxData[4]++;
                                }
                            } else {
                                DataDosen.add(idxData[4], tempArr.get(i));
                                //System.out.println(DataDosen.get(idxData[3]));
                                idxData[4]++;
                            }
                            break;
                    }
                }
                //System.out.println();
            }
        } catch (BiffException e) {
            e.printStackTrace();
        }
    }
}
```

InformedGA.java

```
package timetable_uty;

import java.util.*;
import javax.swing.JProgressBar;

public class InformedGA {

    int UkPop = 3;
    int JumGen = BacaDataKuliah.numRow - 1;
    int[][] Kromosom = new int[UkPop][JumGen];
    int[] BestKromosom = new int[JumGen];
    int MaxGenerasi = 1000;
    int jumSesi, jumRuang, MaxSlot;
    int DurasiPerSKS = 50;  //durasi kuliah 1 sks dalam satuan menit
    int MaxSlotMalamPerDosen = 2; //maksimal 1 kali per pekan per dosen/ per kelas
    int MaxSlotMalamPerKelas = 2; //maksimal 1 kali per pekan per kelas/ per kelas
    int[] fitness = new int[UkPop];
    int MaxSKSPerHari = 7, MinKuliahPerHari = 2;
    boolean Mulai = true;
    int[] GenBermasalah;
    String[] Hari = {"Senin", "Selasa", "Rabu", "Kamis", "Jumat"};

    public void MainInformedGA() {
        //Timer starts
        long startTime = System.currentTimeMillis();
        int BestFitness = 99999;
        int IdxBestFitness = 0;
        double PM = 0.3;

```

```
        UbahSKSPraktik(Mulai);
        RandomInitialization();
        //GreedyInitialization();
        for (int g = 0; g <= MaxGenerasi; g++) {
            //Calculate the fitness value for each chromosome
            for (int i = 0; i < UkPop; i++) {
                GenBermasalah = new int[JumGen];   //GenBermasalah is set as empty at the beginning investigation of each chromosom

                fitness[i] = HitungFitness(Kromosom[i], MainForm.MaxBatas, g);

                if (fitness[i] < BestFitness) {
                    BestFitness = fitness[i];
                    IdxBestFitness = i;
                }

                Mutasi_v2(Kromosom[i], i, PM);
            }

            if ((g % 50) == 0) {
                System.out.println(g + 1 + ". Best Fitness " + BestFitness + " Kromosom ke- " + IdxBestFitness);
                //Mutate the best chromosome using mutasi versi 1
                //Mutasi(Kromosom[IdxBestFitness],1);
            }

            //save the best chromosome
            System.arraycopy(Kromosom[IdxBestFitness], 0, BestKromosom, 0, JumGen);
        }

        //Show the best chromosome
        for (int i = 0; i < JumGen; i++) {
            System.out.print(Kromosom[IdxBestFitness][i] + " ");
        }
        System.out.println();

        //Timer stops
        long endTime = System.currentTimeMillis();
        long totalTime = endTime - startTime;
        System.out.println("Total Waktu Eksekusi " + totalTime);

        //kembalikan lagi nilai SKS praktik ke awal
        Mulai = false;
        UbahSKSPraktik(Mulai);
        //Save the result into a new 2D matrix
    }

    //untuk mengubah SKS praktik dari 2 menjadi 4
    private void UbahSKSPraktik(boolean Mulai) {
        for (int i = 0; i < JumGen; i++) {
            String Kuliah = BacaDataKuliah.DataMentah[i][3];
            String JenisKuliah = Kuliah.substring(Kuliah.length() - 7); //7 adalah length dari kata "Praktik"

            if (JenisKuliah.toLowerCase().equals("praktik")) {
                //Ubah SKS praktik menjadi 4
                if (Mulai) {
                    BacaDataKuliah.DataMentah[i][7] = "4";
                } else {
                    BacaDataKuliah.DataMentah[i][7] = "2";
                }
            }
        }
    }

    private void RandomInitialization() {
        for (int i = 0; i < UkPop; i++) {
            for (int j = 0; j < JumGen; j++) {
                //create discrete random number from 1-MaxSlot
                Random rand = new Random();
                int slot = rand.nextInt(MainForm.FixSlot - 1) + 1;

                //Add value of slot into Kromosom[i][j]
                Kromosom[i][j] = slot;

                System.out.print(Kromosom[i][j] + " ");
            }
            System.out.println();
        }
    }

    private int CekBentrokJadwal(int[] Krom) {
        //to check if a lecture is not held in the same schedule (day,time,room)
        int JumBentrok = 0;

        for (int i = 0; i < JumGen; i++) {
            for (int j = i; j < JumGen; j++) {
                if (i != j) {
                    if (Krom[i] == Krom[j]) {
                        JumBentrok++;
                        //System.out.println(i + " " + j);
                        //Note the conflict gene
                        GenBermasalah[i]++;
                    }
                }
            }
        }
```

```
        //System.out.println("Bentrok Jadwal = " + JumBentrok);
        return JumBentrok;
    }

    private int CekBentrok(int[] Krom) {
        int JumBentrok = 0, idxBentrok = -1;
        String CurrKelas, CekKelas;
        String CurrDosen, CekDosen;
        String CurrRuang, CekRuang;
        double temp1, temp2;

        for (int i = 0; i < JumGen; i++) {
            int CurrIdxSlot = Krom[i];

            for (int j = i; j < JumGen; j++) {
                int CekIdxSlot = Krom[j];
                boolean Bentrok = false;

                if (i != j) {
                    String CurrHari = MainForm.RuangWaktu[CurrIdxSlot - 1][0];
                    String CekHari = MainForm.RuangWaktu[CekIdxSlot - 1][0];

                    temp1 = Double.parseDouble(MainForm.RuangWaktu[CurrIdxSlot - 1][2].substring(0, 2));
                    temp2 = Double.parseDouble(MainForm.RuangWaktu[CurrIdxSlot - 1][2].substring(3, 5));
                    double CurrWktMulai = temp1 + (0.01 * temp2);

                    temp1 = Double.parseDouble(MainForm.RuangWaktu[CurrIdxSlot - 1][2].substring(8, 10));
                    temp2 = Double.parseDouble(MainForm.RuangWaktu[CurrIdxSlot - 1][2].substring(11));
                    double CurrWktSelesai = temp1 + (0.01 * temp2);

                    temp1 = Double.parseDouble(MainForm.RuangWaktu[CekIdxSlot - 1][2].substring(0, 2));
                    temp2 = Double.parseDouble(MainForm.RuangWaktu[CekIdxSlot - 1][2].substring(3, 5));
                    double CekWktMulai = temp1 + (0.01 * temp2);

                    temp1 = Double.parseDouble(MainForm.RuangWaktu[CekIdxSlot - 1][2].substring(8, 10));
                    temp2 = Double.parseDouble(MainForm.RuangWaktu[CekIdxSlot - 1][2].substring(11));
                    double CekWktSelesai = temp1 + (0.01 * temp2);
                    //System.out.println("CekWktMulai " + CekWktMulai);

                    //System.out.println(CurrHari + " " + CekHari  + " " + CurrRuang + " " + CekRuang + " " + CurrSesi + " " + CekSesi);
                    if (CurrHari.equals(CekHari) && (CurrWktMulai == CekWktMulai || CurrWktSelesai == CekWktSelesai)) {
                        Bentrok = true;
                        idxBentrok = j;
                    } else if (CurrHari.equals(CekHari) && (CurrWktMulai > CekWktMulai && CurrWktMulai < CekWktSelesai)) {
                        Bentrok = true;
                        idxBentrok = j;
                    } else if (CurrHari.equals(CekHari) && (CekWktMulai > CurrWktMulai && CekWktMulai < CurrWktSelesai)) {
                        Bentrok = true;
                        idxBentrok = j;
                    } else if (CurrHari.equals(CekHari) && (CurrWktSelesai > CekWktMulai && CurrWktSelesai < CekWktSelesai)) {
                        Bentrok = true;
                        idxBentrok = j;
                    } else if (CurrHari.equals(CekHari) && (CekWktSelesai > CurrWktMulai && CekWktSelesai < CurrWktSelesai)) {
                        Bentrok = true;
                        idxBentrok = j;
                    }

                    //Cek Bentrok Kelas
                    if (Bentrok) {
                        CurrKelas = (BacaDataKuliah.DataMentah[i][5] + BacaDataKuliah.DataMentah[i][6]);
                        CekKelas = (BacaDataKuliah.DataMentah[j][5] + BacaDataKuliah.DataMentah[j][6]);

                        CurrDosen = BacaDataKuliah.DataMentah[i][4];
                        CekDosen = BacaDataKuliah.DataMentah[j][4];

                        CurrRuang = MainForm.RuangWaktu[CurrIdxSlot - 1][1];
                        CekRuang = MainForm.RuangWaktu[CekIdxSlot - 1][1];

                        if (CurrKelas.equals(CekKelas) || CurrDosen.equals(CekDosen) ||
                            CurrRuang.equals(CekRuang)) {
                            JumBentrok++;
                            //System.out.println(CurrKelas + " " + CekKelas);
                            //Note the conflict gene
                            if (idxBentrok != -1) {
                                GenBermasalah[idxBentrok]++;
                            }
                        }
                    }
                }
            }
        }
        //System.out.println("Bentrok Kelas = " + JumBentrok);
        return JumBentrok;
    }

    private int CekDurasiKuliah(int[] Krom) {
        int JumPelanggaran = 0;

        for (int i = 0; i < JumGen; i++) {
            int sks = Integer.parseInt(BacaDataKuliah.DataMentah[i][7]);
            int JamMulai = Integer.parseInt(MainForm.RuangWaktu[Krom[i] - 1][2].substring(0, 2));
            int MenitMulai = Integer.parseInt(MainForm.RuangWaktu[Krom[i] - 1][2].substring(3, 5));
            int JamSelesai = Integer.parseInt(MainForm.RuangWaktu[Krom[i] - 1][2].substring(8, 10));
            int MenitSelesai = Integer.parseInt(MainForm.RuangWaktu[Krom[i] - 1][2].substring(11, 13));

            int DurasiSesi = (JamSelesai * 60 + MenitSelesai) - (JamMulai * 60 + MenitMulai); // *60 to convert to minute
            int DurasiKuliah = sks * DurasiPerSKS;

```

```
            if (DurasiSesi < DurasiKuliah || (Math.abs(DurasiKuliah - DurasiSesi) >= DurasiKuliah)) {
                JumPelanggaran++;
                //System.out.println(i+1);
                //Note the conflict gene
                GenBermasalah[i]++;
            }
        }

        return JumPelanggaran;
    }

    private int CekMaxNgajar(int[] _Krom, int Max) {
        //to check if a lecturer does not exceed the maximum number of teaching in a day
        int JumPelanggaran = 0, JumNgajar;
        int[] Krom = new int[JumGen];
        System.arraycopy(_Krom, 0, Krom, 0, JumGen);

        for (int i = 0; i < JumGen; i++) {
            if (Krom[i] != -1) {
                String CekDosen = BacaDataKuliah.DataMentah[i][4];
                String CekHari = MainForm.RuangWaktu[Krom[i] - 1][0];
                JumNgajar = 0;
                for (int j = i; j < JumGen; j++) {
                    if (Krom[j] != -1) {
                        String NamaDosen = BacaDataKuliah.DataMentah[j][4];
                        String HariNgajar = MainForm.RuangWaktu[Krom[j] - 1][0];
                        if (NamaDosen.equals(CekDosen) && HariNgajar.equals(CekHari)) {
                            //System.out.println(CekDosen + " " + NamaDosen + " " + " " + CekHari + " " + HariNgajar);
                            JumNgajar++;
                            //Change the value of Krom[j] into -1 to mark as read
                            Krom[j] = -1;
                        }
                    }
                }
                if (JumNgajar > Max) {
                    JumPelanggaran++;
                    //Note the conflict gene
                    GenBermasalah[i]++;
                }
            }
        }
        //System.out.println("Jumlah pelanggaran SC1 = " + JumPelanggaran);
        return JumPelanggaran;
    }

    private int CekMinNgajar(int[] _Krom, int Min) {
        //to check if a lecturer does not least than the minimum number of teaching in a day

        int JumPelanggaran = 0, JumNgajar;
        int[] Krom = new int[JumGen];
        int[] HariKerja;
        int JumSatuKaliNgajar;
        int[] CatatJumSKSPerHari;
        System.arraycopy(_Krom, 0, Krom, 0, JumGen);

        for (int i = 0; i < JumGen; i++) {
            if (Krom[i] != -1) {
                String CekDosen = BacaDataKuliah.DataMentah[i][4];
                String CekHari = MainForm.RuangWaktu[Krom[i] - 1][0];
                JumNgajar = 0;
                JumSatuKaliNgajar = 0;
                HariKerja = new int[MainForm.jumHari];
                CatatJumSKSPerHari = new int[Hari.length];

                for (int j = i; j < JumGen; j++) {
                    if (Krom[j] != -1) {
                        String NamaDosen = BacaDataKuliah.DataMentah[j][4];
                        String HariNgajar = MainForm.RuangWaktu[Krom[j] - 1][0];
                        int sks = Integer.parseInt(BacaDataKuliah.DataMentah[j][7]);
                        if (NamaDosen.equals(CekDosen)) {
                            //note the day he give a lecture
                            for (int h = 0; h < Hari.length; h++) {
                                if (HariNgajar.equals(Hari[h])) {
                                    HariKerja[h]++;
                                    CatatJumSKSPerHari[h] += sks;
                                }
                            }

                            if (HariNgajar.equals(CekHari)) {
                                JumNgajar++;
                                //Change the value of Krom[j] into -1 to mark as read
                                Krom[j] = -1;
                            }
                        }
                    }
                }
                // catat berapa kali dia punya jadwal ngajar cuma sehari dan itu tidak melebihi batas maks sks harian
                for (int h = 0; h < Hari.length; h++) {
                    if (HariKerja[h] == 1 && CatatJumSKSPerHari[h] * Min <= MaxSKSPerHari) {
                        JumSatuKaliNgajar++;
                    }
                }

                if (JumNgajar < Min && JumSatuKaliNgajar > 1) {
                    JumPelanggaran++;
                    //Note the conflict gene
                    GenBermasalah[i]++;
                }
            }
        }
```

```
        //System.out.println("Jumlah pelanggaran SC1 = " + JumPelanggaran);
        return JumPelanggaran;
    }

    private int CekJumKuliahMhs(int[] _Krom, int Max) {
        //to check if a class of students does not exceed the maximum number of attending lecture in a day
        int JumPelanggaran = 0, JumKuliahMhs;
        int[] Krom = new int[JumGen];
        System.arraycopy(_Krom, 0, Krom, 0, JumGen);

        for (int i = 0; i < JumGen; i++) {
            if (Krom[i] != -1) {
                String CekKelas = BacaDataKuliah.DataMentah[i][5];
                String CekSemester = BacaDataKuliah.DataMentah[i][6];
                String CekHari = MainForm.RuangWaktu[Krom[i] - 1][0];
                JumKuliahMhs = 0;
                for (int j = i; j < JumGen; j++) {
                    if (Krom[j] != -1) {
                        String Kelas = BacaDataKuliah.DataMentah[j][5];
                        String Semester = BacaDataKuliah.DataMentah[j][6];
                        String HariKuliah = MainForm.RuangWaktu[Krom[j] - 1][0];
                        if (Kelas.equals(CekKelas) && Semester.equals(CekSemester) && HariKuliah.equals(CekHari)) {
                            JumKuliahMhs++;
                            //Change the value of Krom[j] into -1 to mark as read
                            Krom[j] = -1;
                        }
                    }
                }
                if (JumKuliahMhs > Max) {
                    JumPelanggaran++;
                    //Note the conflict gene
                    GenBermasalah[i]++;
                }
            }
        }
        //System.out.println("Jumlah pelanggaran SC2 = " + JumPelanggaran);
        return JumPelanggaran;
    }

    private int CekJumNgajarBerurutan(int[] _Krom, int Max) {
        //to check if a lecturer does not exceed the maximum number of teaching in a row in a day
        int JumPelanggaran = 0, JumNgajarBerurutan, idxBerurutan = -1;
        int[] Krom = new int[JumGen];
        System.arraycopy(_Krom, 0, Krom, 0, JumGen);

        for (int i = 0; i < JumGen; i++) {
            String CekDosen = BacaDataKuliah.DataMentah[i][4];
            String CekHari = MainForm.RuangWaktu[Krom[i] - 1][0];
            int IdxSesi1 = Integer.parseInt(MainForm.RuangWaktu[Krom[i] - 1][3]);

            JumNgajarBerurutan = 0;

            for (int j = i; j < JumGen; j++) {
                String NamaDosen = BacaDataKuliah.DataMentah[j][4];
                String HariNgajar = MainForm.RuangWaktu[Krom[j] - 1][0];
                int IdxSesi2 = Integer.parseInt(MainForm.RuangWaktu[Krom[j] - 1][3]);

                if (i != j) {
                    if (NamaDosen.equals(CekDosen) && HariNgajar.equals(CekHari)) {
                        //Kuliah 1
                        int JamMulai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(0, 2));
                        int MenitMulai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(3, 5));
                        int JamSelesai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(8, 10));
                        int MenitSelesai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(11, 13));

                        //Kuliah 2
                        int JamMulai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(0, 2));
                        int MenitMulai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(3, 5));
                        int JamSelesai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(8, 10));
                        int MenitSelesai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(11, 13));

                        //Calculate the difference between Starting Time and Ending Time
                        int SelisihWaktu = 0;
                        if (JamMulai1 >= JamSelesai2) {
                            SelisihWaktu = (JamMulai1 * 60 + MenitMulai1) - (JamSelesai2 * 60 + MenitSelesai2); // *60 to convert to minute
                        } else if (JamMulai2 >= JamSelesai1) {
                            SelisihWaktu = (JamMulai2 * 60 + MenitMulai2) - (JamSelesai1 * 60 + MenitSelesai1); // *60 to convert to minute
                        }

                        SelisihWaktu = Math.abs(SelisihWaktu);

                        if (SelisihWaktu <= (DurasiPerSKS + 10)) {  //selisih jadwalnya = 1
                            //System.out.println(JamMulai + ":" + MenitMulai + " " + CekJamSelesai + ":" + CekMenitSelesai + " TOTAL SELISIH "
                            JumNgajarBerurutan++;

                            //CekDosen and CekHari is changed into current NamaDosen and current HariNgajar --> berantai
                            CekDosen = NamaDosen;
                            CekHari = HariNgajar;
                            idxBerurutan = j;
                        }
                    }
                }
            }
            if (JumNgajarBerurutan > (Max - 1)) {   //i.e.: jika Max = 3, maka tidak boleh ada 3 kuliah berurutan, berarti ada dua kali pengece
                JumPelanggaran++;
                if (idxBerurutan != -1) {
                    //Note the conflict gene
                    GenBermasalah[idxBerurutan]++;
                }
            }
        }
        //System.out.println("Jumlah pelanggaran SC3 = " + JumPelanggaran);
        return JumPelanggaran;
    }
```

```
427.     private int CekJumNgajarMalam(int[] _Krom, int Max) {
428.         //to check how many lectures are held at night in order to minimize the amount of it
429.         int JumNgajarMalam, JumPelanggaran = 0, idxPelanggaran = -1;
430.         int[] Krom = new int[JumGen];
431.         System.arraycopy(_Krom, 0, Krom, 0, JumGen);
432.
433.         for (int i = 0; i < JumGen; i++) {
434.             if (Krom[i] != -1) {
435.                 String CurrDosen = BacaDataKuliah.DataMentah[i][4];
436.                 JumNgajarMalam = 0;
437.                 for (int j = i; j < JumGen; j++) {
438.                     if (Krom[j] != -1) {
439.                         String CekDosen = BacaDataKuliah.DataMentah[j][4];
440.                         int JamSelesai = Integer.parseInt(MainForm.RuangWaktu[Krom[j] - 1][2].substring(8, 10));
441.
442.                         if (CurrDosen.equals(CekDosen)) {
443.                             if (JamSelesai > 18) {
444.                                 JumNgajarMalam++;
445.                                 //Change the value of Krom[j] into -1 to mark as read
446.                                 Krom[j] = -1;
447.                                 //System.out.println("Jam selesai " + JamSelesai + " " + JumNgajarMalam);
448.                                 idxPelanggaran = j;
449.                             }
450.                         }
451.                     }
452.                 }
453.                 if (JumNgajarMalam > Max) {
454.                     JumPelanggaran = JumPelanggaran + (JumNgajarMalam - Max); //-Max: because subtracted with Maximum number of night lecture
455.                     if (idxPelanggaran != -1) {
456.                         //Note the conflict gene
457.                         GenBermasalah[idxPelanggaran]++;
458.                     }
459.                 }
460.             }
461.         }
462.         //System.out.println("Jumlah pelanggaran SC4 = " + JumPelanggaran);
463.         return JumPelanggaran;
464.     }
465.
466.     private int CekJumKuliahMalam(int[] _Krom, int Max) {
467.         //to check how many lectures are held at night in order to minimize the amount of it
468.         int JumKuliahMalam, JumPelanggaran = 0, idxPelanggaran = -1;
469.         int[] Krom = new int[JumGen];
470.         System.arraycopy(_Krom, 0, Krom, 0, JumGen);
471.
472.         for (int i = 0; i < JumGen; i++) {
473.             if (Krom[i] != -1) {
474.                 String CurrKelas = (BacaDataKuliah.DataMentah[i][5] + BacaDataKuliah.DataMentah[i][6]);
475.                 JumKuliahMalam = 0;
476.                 for (int j = i; j < JumGen; j++) {
477.                     if (Krom[j] != -1) {
478.                         String CekKelas = (BacaDataKuliah.DataMentah[j][5] + BacaDataKuliah.DataMentah[j][6]);
479.                         int JamSelesai = Integer.parseInt(MainForm.RuangWaktu[Krom[j] - 1][2].substring(8, 10));
480.
481.                         if (CurrKelas.equals(CekKelas)) {
482.                             if (JamSelesai > 18) {
483.                                 JumKuliahMalam++;
484.                                 //Change the value of Krom[j] into -1 to mark as read
485.                                 Krom[j] = -1;
486.                                 idxPelanggaran = j;
487.                             }
488.                         }
489.                     }
490.                 }
491.                 if (JumKuliahMalam > Max) {
492.                     JumPelanggaran++;
493.                     if (idxPelanggaran != -1) {
494.                         //Note the conflict gene
495.                         GenBermasalah[idxPelanggaran]++;
496.                     }
497.                 }
498.             }
499.         }
500.         //System.out.println("Jumlah pelanggaran SC5 = " + JumPelanggaran);
501.         return JumPelanggaran;
502.     }
503.
504.     private int CekRuangYangSama(int[] _Krom) {
505.         //this method aims to check whether the TWO OR MORE consecutive lectures of a lecturer is held in the same room
506.         int JumPelanggaran = 0, JumBedaRuang;
507.         int[] Krom = new int[JumGen];
508.         System.arraycopy(_Krom, 0, Krom, 0, JumGen);
509.
510.         for (int i = 0; i < JumGen; i++) {
511.             String Hari1 = MainForm.RuangWaktu[Krom[i] - 1][0];
512.             String Dosen1 = BacaDataKuliah.DataMentah[i][4];
513.             String Ruang1 = MainForm.RuangWaktu[Krom[i] - 1][2];
514.             int IdxSesi1 = Integer.parseInt(MainForm.RuangWaktu[Krom[i] - 1][3]);
515.
516.             JumBedaRuang = 0;
517.             for (int j = i; j < JumGen; j++) {
518.                 String Hari2 = MainForm.RuangWaktu[Krom[j] - 1][0];
519.                 String Dosen2 = BacaDataKuliah.DataMentah[j][4];
520.                 String Ruang2 = MainForm.RuangWaktu[Krom[j] - 1][2];
521.                 int IdxSesi2 = Integer.parseInt(MainForm.RuangWaktu[Krom[j] - 1][3]);
522.
523.                 if (i != j) {
524.                     if (Dosen1.equals(Dosen2) && Hari1.equals(Hari2)) {
525.                         //Kuliah 1
526.                         int JamMulai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(0, 2));
527.                         int MenitMulai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(3, 5));
```

```
                        int JamSelesai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(8, 10));
                        int MenitSelesai1 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi1 - 1][2].substring(11, 13));

                        //Kuliah 2
                        int JamMulai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(0, 2));
                        int MenitMulai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(3, 5));
                        int JamSelesai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(8, 10));
                        int MenitSelesai2 = Integer.parseInt(MainForm.RuangWaktu[IdxSesi2 - 1][2].substring(11, 13));

                        //Calculate the difference between Starting Time and Ending Time
                        int SelisihWaktu = 0;
                        if (JamMulai1 >= JamSelesai2) {
                            SelisihWaktu = (JamMulai1 * 60 + MenitMulai1) - (JamSelesai2 * 60 + MenitSelesai2); // *60 to convert to minute
                        } else if (JamMulai2 >= JamSelesai1) {
                            SelisihWaktu = (JamMulai2 * 60 + MenitMulai2) - (JamSelesai1 * 60 + MenitSelesai1); // *60 to convert to minute
                        }

                        SelisihWaktu = Math.abs(SelisihWaktu);

                        //IF "kuliah1 AND kuliah2 berurutan" THEN ...
                        if (SelisihWaktu <= (DurasiPerSKS + 10)) {  //selisih jadwalnya = 1
                            if (!Ruang1.equals(Ruang2)) {
                                JumBedaRuang++;
                                //Note the conflict gene
                                GenBermasalah[j]++;
                            }
                        }
                    }
                }
            }
            if (JumBedaRuang > 0) {
                JumPelanggaran++;
            }
        }
        //System.out.println("Jumlah pelanggaran SC6 = " + JumPelanggaran);
        return JumPelanggaran;
    }

    private int CekKuliahPagiDanMalam(int[] Krom) {
        //a lecturer that teach in the first session (07.00) should not give a lecture till night (18.30) or in the last session
        int JumPelanggaran = 0, idxGenBermasalah = -1;
        boolean NgajarPagi, NgajarMalam;
        int JamPertama = 8;
        int JamTerakhir = 18;

        for (int i = 0; i < JumGen; i++) {
            if (Krom[i] != -1) {
                String Dosen1 = BacaDataKuliah.DataMentah[i][4];
                String Hari1 = BacaDataKuliah.DataMentah[i][0];
                NgajarPagi = false;
                NgajarMalam = false;
                for (int j = i; j < JumGen; j++) {
                    if (Krom[j] != -1) {
                        String Dosen2 = BacaDataKuliah.DataMentah[j][4];
                        String Hari2 = BacaDataKuliah.DataMentah[j][0];
                        int JamMasuk = Integer.parseInt(MainForm.RuangWaktu[Krom[j] - 1][2].substring(0, 2));
                        int JamPulang = Integer.parseInt(MainForm.RuangWaktu[Krom[j] - 1][2].substring(8, 10));

                        if (Dosen1.equals(Dosen2) && Hari1.equals(Hari2)) {
                            if (JamMasuk <= JamPertama) {
                                NgajarPagi = true;
                            } else if (JamPulang >= JamTerakhir) {
                                NgajarMalam = true;
                                //note the idx of gene
                                idxGenBermasalah = j;
                            }
                        }
                    }
                }
                if (NgajarPagi && NgajarMalam) {
                    JumPelanggaran++;
                    if (idxGenBermasalah != -1) {
                        //Note the conflict gene
                        GenBermasalah[idxGenBermasalah]++;
                    }
                }
            }
        }
        return JumPelanggaran;
    }

    private int CekJumSKSHarian(int[] _Krom, int Max) {
        //aims to check whether a lecturer exceed the maximum number of SKS in a day
        //if DAY == SATURDAY, then MAX = 4;
        int JumPelanggaran = 0, JumSKSPerHari;
        int[] Krom = new int[JumGen];
        System.arraycopy(_Krom, 0, Krom, 0, JumGen);

        for (int i = 0; i < JumGen; i++) {
            if (Krom[i] != -1) {
                String Hari1 = MainForm.RuangWaktu[Krom[i] - 1][0];
                String Dosen1 = BacaDataKuliah.DataMentah[i][4];
                JumSKSPerHari = 0;

                for (int j = i; j < JumGen; j++) {
                    if (Krom[j] != -1) {
                        String Hari2 = MainForm.RuangWaktu[Krom[j] - 1][0];
                        String Dosen2 = BacaDataKuliah.DataMentah[j][4];
                        int SKS = Integer.parseInt(BacaDataKuliah.DataMentah[j][7]);

```

```
                        if (Hari1.equals(Hari2) && Dosen1.equals(Dosen2)) {
                            JumSKSPerHari = JumSKSPerHari + SKS;
                            //Mark Krom[j] as read --> -1
                            Krom[j] = -1;
                        }

                        if (JumSKSPerHari > Max) {
                            JumPelanggaran++;
                            //Note the conflict gene
                            GenBermasalah[i]++;
                        }
                    }
                }
            }
        }

        return JumPelanggaran;
    }

    private int CekRuangPraktik(int[] _Krom) {
        int JumPelanggaran = 0;
        int[] Krom = new int[JumGen];
        System.arraycopy(_Krom, 0, Krom, 0, JumGen);

        for (int i = 0; i < JumGen; i++) {
            String Kuliah = BacaDataKuliah.DataMentah[i][3];
            String JenisKuliah = Kuliah.substring(Kuliah.length() - 7); //7 adalah length dari kata "Praktik"

            if (JenisKuliah.toLowerCase().equals("praktik")) {
                String Ruang = MainForm.RuangWaktu[Krom[i] - 1][1];
                String SubRuang = Ruang.substring(0, 2);

                if (!SubRuang.toLowerCase().equals("lk")) {
                    //System.out.println(Kuliah + " " + JenisKuliah + " " + Ruang + " " + SubRuang);
                    GenBermasalah[i]++;
                    JumPelanggaran++;
                }
            } else if (!JenisKuliah.toLowerCase().equals("praktik")) {
                String Ruang = MainForm.RuangWaktu[Krom[i] - 1][1];
                String SubRuang = Ruang.substring(0, 2);

                if (SubRuang.toLowerCase().equals("lk")) {
                    //System.out.println(Kuliah + " " + JenisKuliah + " " + Ruang + " " + SubRuang);
                    GenBermasalah[i]++;
                    JumPelanggaran++;
                }
            }
        }
        return JumPelanggaran;
    }

    private void Mutasi_v2(int[] Krom, int idxKrom, double PM) {
        for (int i = 0; i < JumGen; i++) {
            if (GenBermasalah[i] > 0) {
                int temp = Krom[i];
                Random rand = new Random();
                Krom[i] = rand.nextInt(MainForm.FixSlot) + 1;

                int tempFitness = HitungFitness(Krom, MainForm.MaxBatas, 1);

                //jika fitness lebih kecil dari sebelumnya, maka batalkan mutasi
                if (tempFitness > fitness[idxKrom]) {
                    Krom[i] = temp;
                } else {
                    GenBermasalah[i]--;
                }
            } else {
                double acak = Math.random();
                if (acak < PM) {
                    int temp = Krom[i];
                    Random rand = new Random();
                    Krom[i] = rand.nextInt(MainForm.FixSlot) + 1;

                    int tempFitness = HitungFitness(Krom, MainForm.MaxBatas, 1);

                    //jika fitness lebih kecil dari sebelumnya, maka batalkan mutasi
                    if (tempFitness > fitness[idxKrom]) {
                        Krom[i] = temp;
                    } else {
                        GenBermasalah[i]--;
                    }
                }
            }
            //System.out.println(Krom[i] + " ");
        }
    }

    private int HitungFitness(int[] _Krom, int[] Max, int generasi) {
        int TempFitness;
        int HC_penalty = 100;
        int SC_penalty1 = 5;
        int SC_penalty2 = 5;
        int[] pelanggaran = new int[18];
        int[] Krom = new int[JumGen];
        System.arraycopy(_Krom, 0, Krom, 0, JumGen);

        pelanggaran[0] = CekBentrokJadwal(Krom);
        pelanggaran[1] = CekBentrok(Krom);
        //pelanggaran[1] = CekBentrokKelas(Krom);
        //pelanggaran[2] = CekBentrokDosen(Krom);
```

```
728.          pelanggaran[3] = CekMaxNgajar(Krom, Max[0]);
729.          pelanggaran[4] = CekJumKuliahMhs(Krom, Max[1]);
730.          pelanggaran[5] = CekJumNgajarBerurutan(Krom, Max[2]);
731.          pelanggaran[6] = CekJumNgajarMalam(Krom, 1);
732.          pelanggaran[7] = CekJumKuliahMalam(Krom, 2);
733.          pelanggaran[8] = CekRuangYangSama(Krom);
734.          //pelanggaran[9] = CekJumKuliahWeekend(Krom);
735.          pelanggaran[10] = CekDurasiKuliah(Krom);
736.          pelanggaran[11] = CekKuliahPagiDanMalam(Krom);
737.          pelanggaran[12] = CekJumSKSHarian(Krom, MaxSKSPerHari);
738.          //pelanggaran[13] = CekBentrokRuang(Krom);
739.          pelanggaran[14] = CekMinNgajar(Krom, MinKuliahPerHari);
740.          //pelanggaran[15] = CekMatkulPilihan(Krom);
741.          //pelanggaran[16] = CekMinPemakaianRuang_v2(Krom);
742.          pelanggaran[17] = CekRuangPraktik(Krom);
743.
744.          int fitnessHC = HC_penalty * (pelanggaran[0] + pelanggaran[1] + pelanggaran[2] +
745. pelanggaran[10] + pelanggaran[13] + pelanggaran[17]);
746.          int fitnessSC1 = SC_penalty1 * (pelanggaran[3] + pelanggaran[6] + pelanggaran[5] +
747. pelanggaran[8] + pelanggaran[9] + pelanggaran[11] + pelanggaran[12] + pelanggaran[14] + pelanggaran[15]);
748.          int fitnessSC2 = SC_penalty2 * (pelanggaran[4] + pelanggaran[7] + pelanggaran[16]);
749.
750.          TempFitness = fitnessHC + fitnessSC1 + fitnessSC2;
751.
752.          if (generasi > (MaxGenerasi - 1)) {
753.              System.out.println("Jumlah pelanggaran HC1  = " + pelanggaran[0]);
754.              System.out.println("Jumlah pelanggaran HC2  = " + pelanggaran[1]);
755.              System.out.println("Jumlah pelanggaran HC3  = " + pelanggaran[2]);
756.              System.out.println("Jumlah pelanggaran HC4  = " + pelanggaran[10]);
757.              System.out.println("Jumlah pelanggaran HC5  = " + pelanggaran[13]);
758.              System.out.println("Jumlah pelanggaran SC1  = " + pelanggaran[3]);
759.              System.out.println("Jumlah pelanggaran SC2  = " + pelanggaran[4]);
760.              System.out.println("Jumlah pelanggaran SC3  = " + pelanggaran[5]);
761.              System.out.println("Jumlah pelanggaran SC4  = " + pelanggaran[6]);
762.              System.out.println("Jumlah pelanggaran SC5  = " + pelanggaran[7]);
763.              System.out.println("Jumlah pelanggaran SC6  = " + pelanggaran[8]);
764.              System.out.println("Jumlah pelanggaran SC7  = " + pelanggaran[9]);
765.              System.out.println("Jumlah pelanggaran SC8  = " + pelanggaran[11]);
766.              System.out.println("Jumlah pelanggaran SC9  = " + pelanggaran[12]);
767.              System.out.println("Jumlah pelanggaran SC10  = " + pelanggaran[14]);
768.              System.out.println("Jumlah pelanggaran SC11  = " + pelanggaran[15]);
769.              System.out.println("Jumlah pelanggaran SC12  = " + pelanggaran[16]);
770.              System.out.println("Jumlah pelanggaran SC13  = " + pelanggaran[17]);
771.
772.              for (int i = 0; i < GenBermasalah.length; i++) {
773.                  if (GenBermasalah[i] == 1) {
774.                      System.out.println(i + " " + Krom[i]);
775.                  }
776.              }
777.          }
778.          return TempFitness;
779.      }
780. }
```

TimeTableUTY.java

```
01. package timetable_uty;
02.
03. public class TimeTable_UTY {
04.
05.     /**
06.      * @param args the command line arguments
07.      */
08.     public static void main(String[] args) {
09.         try {
10.             for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
11.                 if ("Windows".equals(info.getName())) {
12.                     javax.swing.UIManager.setLookAndFeel(info.getClassName());
13.                     break;
14.                 }
15.             }
16.         } catch (ClassNotFoundException | InstantiationException | IllegalAccessException | javax.swing.UnsupportedLookAndFeelException ex) {
17.             java.util.logging.Logger.getLogger(TimeTable_UTY.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
18.         }
19.
20.         java.awt.EventQueue.invokeLater(() -> {
21.             new MainForm().setVisible(true);
22.         });
23.     }
24. }
```