

SISTEM PENJADWALAN PERKULIAHAN DENGAN ALOKASI WAKTU DINAMIS BERBASIS ALGORITMA GENETIKA TERBIMBING

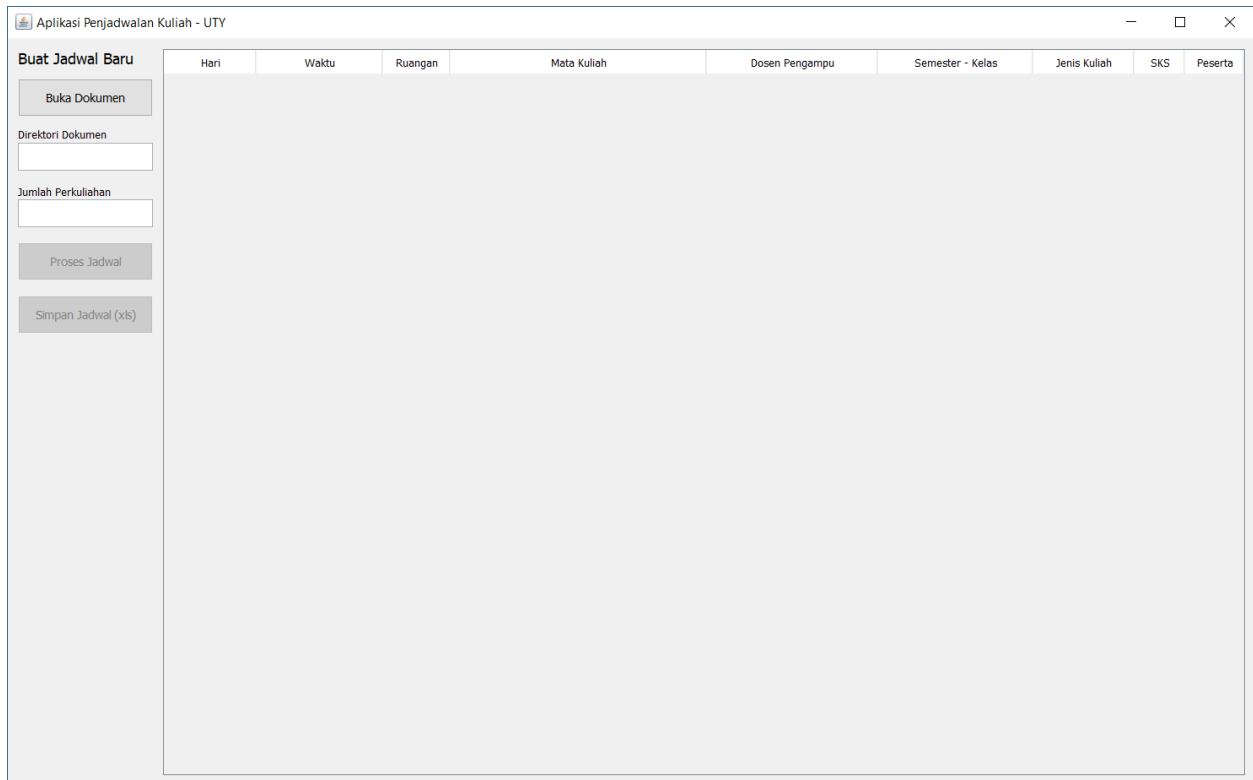
Abstract

Aplikasi ini dirancang untuk menyusun jadwal kuliah secara otomatis di Kampus 1 Universitas Teknologi Yogyakarta. Aplikasi tersebut menerapkan Algoritma Genetika Terbimbing yang menerapkan konsep Guided Creep Mutation pada proses evolusinya dan meniadakan mekanisme rekombinasi untuk mempercepat proses komputasi. Program tersebut mampu menyusun jadwal perkuliahan secara otomatis di tingkat universitas yang terdiri dari 800 hingga lebih dari 1000 pertemuan perkuliahan di tiap semester. Jadwal kuliah yang dihasilkan oleh program tersebut berupa jadwal optimal yang tidak memiliki bentrok pada jadwal dosen, ruangan, dan mahasiswa. Selain itu, program tersebut juga mampu mengakomodasi aturan-aturan penjadwalan lain yang ditetapkan oleh pihak kampus, sehingga jadwal yang dihasilkan sesuai dengan kebutuhan dan kondisi kampus. Jadwal kuliah yang dihasilkan oleh aplikasi berisi hari, waktu, ruang, mata kuliah, dosen, dan kelas dari tiap perkuliahan. Untuk menyusun jadwal

Muhammad Fachrie, S.T., M.Cs., Anita Fira Waluyo, S.Si., M.Sc.
muhammad.fachrie@staff.uty.ac.id, anitafira@staff.uty.ac.id

A. Gambaran Umum Aplikasi

Aplikasi ini dirancang untuk memenuhi kebutuhan penjadwalan kuliah di Kampus 1, Universitas Teknologi Yogyakarta (UTY). Gambar 1 merupakan tampilan utama dari aplikasi yang dibuat. Aplikasi ini sebetulnya adalah pengembangan dari aplikasi penjadwalan kuliah tingkat program studi yang pernah dibuat sebelumnya, namun pada aplikasi yang terbaru ini, penjadwalan dapat dilakukan pada level universitas (mencakup banyak fakultas dan program studi) serta dengan perbaikan algoritma penjadwalan yang lebih efisien.



Gambar 1. Tampilan awal aplikasi penjadwalan kuliah 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* (HC) 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* (SC) 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 HC mengakibatkan jadwal yang dihasilkan tidak

valid, sedangkan pelanggaran pada SC tetap menghasilkan jadwal yang valid. Secara lengkap, Tabel 1 menyajikan data-data HC dan SC pada aplikasi tersebut.

Tabel 1. Data *hard constraint* (HC) dan *soft constraint* (SC) yang diakomodasi oleh aplikasi

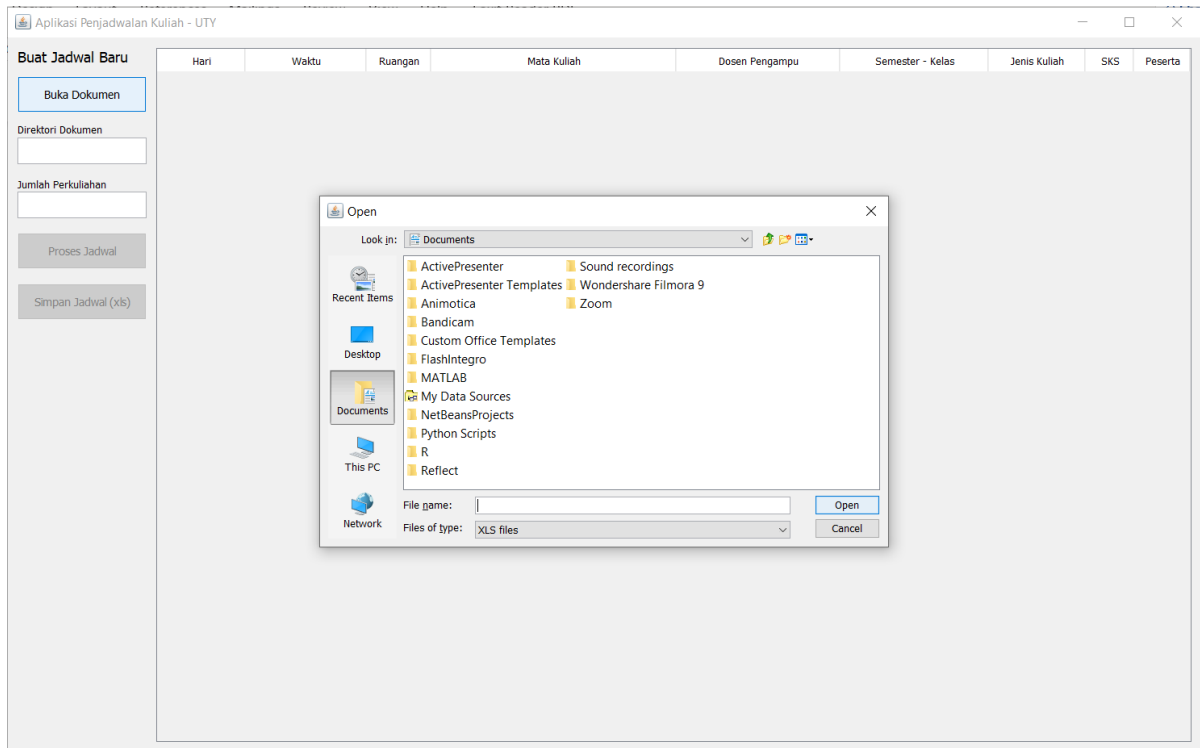
Constraints	Type
No lecturer conflict	HC1
No students group (class) conflict	HC2
No room conflict	HC3
Practical course must be held in laboratory	HC4
Every course meeting must be held in room with enough capacity	HC5
Lecturer with structural position may not teach before 10.30	HC6
A lecturer should not teach more than 6 credits in a day	SC1
Lecturer should not teach after 12.00 PM on Saturday	SC2
Lecturer should not teach in the first time slot if the day before he taught in the last time slot	SC3
Lecturer should not teach more than 7 hours in a day	SC4
The number of lectures on Saturday should be less than on other days	SC5
Students should not attend the class more than 6 credits in a day	SC6

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.

B. Panduan Penggunaan Aplikasi

Penggunaan aplikasi ini tergolong sangat mudah. Cukup dengan menyiapkan dokumen .xls berisi **data dosen, mata kuliah, jenis perkuliahan** (teori atau praktik), beserta **sks, kelas mahasiswa, dan jumlah mahasiswa pada tiap perkuliahan**, 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 Dokumen' 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, jenis perkuliahan, sks, kelas mahasiswa, serta jumlah mahasiswa tiap perkuliahan.



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

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

Hari	Waktu	Ruangan	Mata Kuliah	Dosen Pengampu	Semester - Kelas	Jenis Kuliah	SKS	Peserta
--	--	--	104205-17 - ALGORITMA DAN STRUKTUR DATA I...	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - D3-SISTEM INFOR...	PRAKTIKUM	2	43
--	--	--	104205-17 - ALGORITMA DAN STRUKTUR DATA I...	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - D3-SISTEM INFOR...	PRAKTIKUM	2	45
--	--	--	20320818 - STRUKTUR DATA PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	41
--	--	--	20321018 - SISTEM OPERASI PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	40
--	--	--	20321018 - SISTEM OPERASI PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	41
--	--	--	20321018 - SISTEM OPERASI PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43
--	--	--	210203-17 - ALGORITMA & STRUKTUR DATA II P...	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-SISTEM INFOR...	PRAKTIKUM	2	40
--	--	--	MK84661205 - ADMINISTRASI JARINGAN PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 6 - S1-TEKNIK ELEKTR...	PRAKTIKUM	2	41
--	--	--	20320918 - SISTEM OPERASI	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	50
--	--	--	20320918 - SISTEM OPERASI	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	55
--	--	--	202204-17 - KEWARGANEGARAAN	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-MANAJEMEN - ...	TEORI	2	58
--	--	--	202204-17 - KEWARGANEGARAAN	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-MANAJEMEN - ...	TEORI	2	57
--	--	--	210601-17 - KEWARGANEGARAAN	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 6 - S1-SISTEM INFOR...	TEORI	2	55
--	--	--	217206-17 - TEORI KOMUNIKASI	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-ILMU KOMUNIK...	TEORI	3	55
--	--	--	217206-17 - TEORI KOMUNIKASI	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-ILMU KOMUNIK...	TEORI	3	55
--	--	--	217405-17 - METODE PENELITIAN SOSIAL	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 4 - S1-ILMU KOMUNIK...	TEORI	3	50
--	--	--	217406-17 - STATISTIK	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 4 - S1-ILMU KOMUNIK...	TEORI	3	55
--	--	--	205602-17 - APLIKASI MOBILE	ADITYA DIMAS DEWANTO, ST, ...	Sem. 6 - S1-SISTEM KOMP...	TEORI	2	48
--	--	--	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	61
--	--	--	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	60
--	--	--	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	65
--	--	--	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	59
--	--	--	205603-17 - APLIKASI MOBILE PRAKTIKUM	ADITYA DIMAS DEWANTO, ST, ...	Sem. 6 - S1-SISTEM KOMP...	PRAKTIKUM	4	43
--	--	--	20320518 - APLIKASI TEKNOLOGI INFORMASI II	ADITYO PERMANA WIBOWO, S,...	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43
--	--	--	20320518 - APLIKASI TEKNOLOGI INFORMASI II	ADITYO PERMANA WIBOWO, S,...	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43
--	--	--	20340518 - SISTEM BASIS DATA LANJUT PRAKTIKUM	ADITYO PERMANA WIBOWO, S,...	Sem. 4 - S1-TEKNIK INFOR...	PRAKTIKUM	2	44
--	--	--	20340518 - SISTEM BASIS DATA LANJUT PRAKTIKUM	ADITYO PERMANA WIBOWO, S,...	Sem. 4 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43

Gambar 3. Isi dokumen .xls berupa data perkuliahan yang belum terjadwal

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	Semester - Kelas	Jenis Kuliah	SKS	Peserta
Jumat	14.40 - 16.20	LK-09	104205-17 - ALGORITMA DAN STRUKTUR DATA I...	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - D3-SISTEM INFOR...	PRAKTIKUM	2	43
Kamis	12.50 - 14.30	LK-10	104205-17 - ALGORITMA DAN STRUKTUR DATA I...	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - D3-SISTEM INFOR...	PRAKTIKUM	2	45
Rabu	08.50 - 10.30	LK-09	20320818 - STRUKTUR DATA PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	41
Sabtu	07.00 - 08.40	LK-09	20321018 - SISTEM OPERASI PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	40
Jumat	08.50 - 10.30	LK-02	20321018 - SISTEM OPERASI PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	41
Rabu	12.50 - 14.30	LK-02	20321018 - SISTEM OPERASI PRAKTIKUM	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43
Selasa	10.40 - 12.20	LK-03	210203-17 - ALGORITMA & STRUKTUR DATA II P...	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-SISTEM INFOR...	PRAKTIKUM	2	40
Kamis	07.00 - 08.40	LK-07	MKB4661205 - ADMINISTRASI JARINGAN PRAKTI...	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 6 - S1-TEKNIK ELEKTR...	PRAKTIKUM	2	41
Senin	07.00 - 09.30	G33	20320918 - SISTEM OPERASI	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	50
Senin	14.40 - 17.10	E24	20320918 - SISTEM OPERASI	ADAM SEKTI AJI, S.KOM, M.KOM,	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	55
Senin	12.50 - 14.30	D02	202204-17 - KEWARGANEGARAAN	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-MANAJEMEN - ...	TEORI	2	58
Selasa	16.30 - 18.10	F02	202204-17 - KEWARGANEGARAAN	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-MANAJEMEN - ...	TEORI	2	57
Jumat	15.30 - 17.10	F04	210601-17 - KEWARGANEGARAAN	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 6 - S1-SISTEM INFOR...	TEORI	2	55
Selasa	08.50 - 11.20	H23	217206-17 - TEORI KOMUNIKASI	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-ILMU KOMUNIK...	TEORI	3	55
Kamis	12.50 - 15.20	H24	217206-17 - TEORI KOMUNIKASI	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 2 - S1-ILMU KOMUNIK...	TEORI	3	55
Kamis	08.50 - 11.20	D33	217405-17 - METODE PENELITIAN SOSIAL	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 4 - S1-ILMU KOMUNIK...	TEORI	3	50
Rabu	07.00 - 09.30	D34	217406-17 - STATISTIK	ADE IRMA SUKMAWATI, S.SOS,...	Sem. 4 - S1-ILMU KOMUNIK...	TEORI	3	55
Kamis	16.30 - 18.10	H33	205602-17 - APLIKASI MOBILE	ADITYA DIMAS DEWANTO, ST, ...	Sem. 6 - S1-SISTEM KOMP...	TEORI	2	48
Selasa	09.40 - 12.10	F03	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	61
Selasa	07.00 - 09.30	D11	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	63
Senin	07.00 - 09.30	E14	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	60
Senin	09.40 - 12.10	H21	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	65
Sabtu	09.40 - 12.10	H23	20320418 - TEKNOLOGI BERBASIS OBJEK	ADITYA DIMAS DEWANTO, ST, ...	Sem. 2 - S1-TEKNIK INFOR...	TEORI	3	59
Jumat	10.40 - 14.00	LK-10	205603-17 - APLIKASI MOBILE PRAKTIKUM	ADITYA DIMAS DEWANTO, ST, ...	Sem. 6 - S1-SISTEM KOMP...	PRAKTIKUM	4	43
Rabu	16.30 - 18.10	LK-08	20320518 - APLIKASI TEKNOLOGI INFORMASI II	ADITYO PERMANA WIBOWO, S,...	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43
Senin	16.30 - 18.10	LK-09	20320518 - APLIKASI TEKNOLOGI INFORMASI II	ADITYO PERMANA WIBOWO, S,...	Sem. 2 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43
Jumat	10.40 - 12.20	LK-05	20340518 - SISTEM BASIS DATA LANUT PRAKTI...	ADITYO PERMANA WIBOWO, S,...	Sem. 4 - S1-TEKNIK INFOR...	PRAKTIKUM	2	44
Kamis	15.30 - 17.10	LK-07	20340518 - SISTEM BASIS DATA LANUT PRAKTI...	ADITYO PERMANA WIBOWO, S,...	Sem. 4 - S1-TEKNIK INFOR...	PRAKTIKUM	2	43

Gambar 4. Aplikasi menampilkan jadwal kuliah hasil pemrosesan

C. Kode Sumber

AlgoritmaGenetika.java

```
package timetable_uty;

import java.util.Random;

public class AlgoritmaGenetika {

    public static int generasi, maxGenerasi;
    private final int jumKromosom;
    private final int jumGen, jumGenPerVar, jumVar;
    private final double pm;
    private final String[][] DataPerkuliahan = BacaDataSiapOlah.DataPerkuliahan;
    public static int maxSKSperHari = 6, minSKSperHari = 3;
    public static boolean lanjutPencarian;

    public AlgoritmaGenetika(int jumKromosom, int maxGenerasi, double pm) {
        this.jumKromosom = jumKromosom;
        AlgoritmaGenetika.maxGenerasi = maxGenerasi;
        this.pm = pm;

        jumVar = MainForm.JumPerkuliahan;
```

```

jumGenPerVar = 3;
jumGen = jumGenPerVar * jumVar;

runAlgoritmaGenetika();
}

private void runAlgoritmaGenetika() {
    lanjutPencarian = true;

    //buat array objek kromosom baru sebanyak jumKromosom
    Kromosom[] krom = new Kromosom[jumKromosom];

    //buat objek bestKromosom untuk menampung kromosom terbaik
    Kromosom bestKromosom = new Kromosom(jumGen, jumGenPerVar);

    for (int i = 0; i < jumKromosom; i++) {
        //Generate kromosom awal
        krom[i] = new Kromosom(jumGen, jumGenPerVar);
        krom[i].generateKromosomRandom(jumGen, jumGenPerVar);
        krom[i].revisiKromosomAwal(jumGen, jumGenPerVar, maxSKSperHari);
    }

    double jumFitness, rerataFitness;
    int bestFitness = 0, idxBestKromosom = 0;
    System.out.println("\nBest Fitness Awal = " + bestFitness);

    //Proses Evolusi
    for (generasi = 1; generasi <= maxGenerasi; generasi++) {
        for (int i = 0; i < jumKromosom; i++) {
            //Evaluasi kromosom
            krom[i].hitungFitness(i, jumGen, jumGenPerVar, maxSKSperHari, minSKSperHari);

            //ganti best kromosom
            if (krom[i].getNilaiFitness() > bestFitness) {
                bestFitness = krom[i].getNilaiFitness();
                idxBestKromosom = i;

                //isi bestKromosom
                for (int idx = 0; idx < jumGen; idx++) {
                    bestKromosom.setGen(idx, krom[idxBestKromosom].getGen(idx));
                }
            }
        }
    }

    //Mutasi kromosom
    for (int i = 0; i < jumKromosom; i++) {
        //Mutasi kromosom
        krom[i].mutasiCreep(i, jumGen, jumGenPerVar, pm);
        //krom[i].mutasiRandom(i, jumGen, jumGenPerVar, pm);
    }

    jumFitness = 0;
    rerataFitness = 0;
    for (int i = 0; i < jumKromosom; i++) {
        jumFitness += krom[i].getNilaiFitness();
    }

    rerataFitness = jumFitness / jumKromosom;

    //Display hasil
    //System.out.println("Generasi ke   = " + generasi);
    //System.out.println("Best Kromosom = " + idxBestKromosom);
    //System.out.println("Best Fitness  = " + bestFitness);
    //System.out.println("=====");
    System.out.println(generasi + ", " + bestFitness + ", " + rerataFitness);
}

```

```

        if (!lanjutPencarian) {
            break;
        }
    }

    bestKromosom.infoBestKromosom(0, jumGen, jumGenPerVar, maxSKSperHari, minSKSperHari);
    bestKromosom.hitungJumPakaiRuang(jumGen, jumGenPerVar);
    bestKromosom.tampilJumPakaiRuang();
    bestKromosom.hitungFitness(generasi, jumGen, jumGenPerVar, maxSKSperHari, minSKSperHari);
    bestKromosom.tampilBentrokRuang(jumGen, jumGenPerVar);

    //Ambil kromosom terbaik
    String[][] jadwalMutasi = bestKromosom.getFenotipe(jumVar, jumGen, jumGenPerVar);

    //Masukkan hasil generate awal ke DataPerkuliahan (untuk contoh saja)
    for (int i = 0; i < jumVar; i++) {
        DataPerkuliahan[i][0] = jadwalMutasi[i][0]; //HARI
        DataPerkuliahan[i][1] = jadwalMutasi[i][1]; //WAKTU
        DataPerkuliahan[i][2] = jadwalMutasi[i][2]; //RUANG
    }
}

public String[][] getDataPerkuliahanAwal() {
    return DataPerkuliahan;
}
}

```

Kromosom.java

```

package timetable_uty;

import java.util.Random;

class Kromosom extends Thread {

    private int[] gen;
    private int nilaiFitness;
    private final int maxFitness;
    int[] kuliahBentrokDosen, kuliahBentrokRuang, kuliahBentrokMhs, kuliahSKSharianLebihDosen,
    kuliahSKSharianKurangDosen,
    kuliahSKSharianLebihMhs, ruangKuliahDosenBeda, kuliahDosenLembur;
    int[] jumPakaiRuang;
    private final String[] listPejabat = new String[32];

    public Kromosom(int jumGen, int jumGenPerVar) {
        gen = new int[jumGen];
        nilaiFitness = 0;
        maxFitness = 100 * (jumGen / jumGenPerVar);

        JadwalKuliah jadwal = new JadwalKuliah();
        jumPakaiRuang = new int[jadwal.getIdxRuangPraktikumL()[1] + 1];

        //isi list pejabat yang tidak boleh mengajar di bawah jam 10 karena harus rapat
        listPejabat[0] = "ARIEF HERMAWAN, MT., DR.";
        listPejabat[1] = "BAHAGIA TARIGAN, SE., M.SI., AKT.,";
        listPejabat[2] = "JOKO SUTOPO, ST., MT.,";
        listPejabat[3] = "JUNAIDI , M.SI., AK., CA., CSRS., DR.";
        listPejabat[3] = "SUTARMAN , M.KOM., PH.D.,";
        listPejabat[4] = "ERIK IMAN HERI UJIANTO, , M.KOM., DR.";
        listPejabat[5] = "SEKAR AKROM FARADIZA, SE., M.SC., AKT.,";
        listPejabat[6] = "YULI ASRININGTIAS, S.KOM., M.KOM.,";
        listPejabat[7] = "DWI RATNAWATI, S.PD., M.PD.,";
        listPejabat[8] = "NUR WENING, M.SI, DR";
        listPejabat[9] = "ENNY ITJE SELA, S.SI., M.KOM, DR.";
    }
}

```

```

listPejabat[10] = "M.S. HENDRIYAWAN A., S.T., M.ENG,";
listPejabat[11] = "IKRIMA ALFI, S.T., M.ENG,";
listPejabat[12] = "MURTI RETNOWO, S.KOM., M.CS,";
listPejabat[13] = "IWAN HARTADI TRI UNTORO, ST., M.KOM,";
listPejabat[14] = "AKHMAD AFNAN, S.E., M.ACC,";
listPejabat[15] = "WAWAN SETIAWAN, SE., MSI. AK., CA.,";
listPejabat[16] = "VERA DESY NURMALIA, SE., MM.,";
listPejabat[17] = "ANEKE DEWI RAHAYU, S.PSI, MA, PSI,";
listPejabat[18] = "TIFFANY SETYO PRATIWI, S.I.P., MA.,";
listPejabat[19] = "ANDRI PRASETYO YUWONO, M.I.KOM, ";
listPejabat[20] = "ULFA AMALIA, S.PSI., MA.,";
listPejabat[21] = "VIVIANI, S.PD., M.PD.,,";
listPejabat[22] = "IFAH ROFIQOH, SE., MSI.,";
listPejabat[23] = "SUMARSONO, SE., M.SI,";
listPejabat[24] = "DODI HARIADI, ST., M.ENG,";
listPejabat[25] = "DAMAR PRASETYO, M.KOM., DRS.";
listPejabat[26] = "YUNUS INDRA PURNAMA, SE., M.M., AKT. CA.,";
listPejabat[27] = "UMAR ZAKY, S.KOM, M.CS.,";
listPejabat[28] = "SRI RESPATI ANDAMARI, M.SI., PSI., DRA.";
listPejabat[28] = "LUCITANIA RIZKY, S.IP., MA.,";
listPejabat[29] = "DONNY AVIANTO, ST., MT.,";
listPejabat[30] = "ADITYO PERMANA WIBOWO, S.KOM, M.CS,";
listPejabat[31] = "FADIL INDRA SANJAYA, S.KOM., M.KOM,";
}

public void generateKromosomRandom(int jumGen, int jumGenPerVar) {
    JadwalKuliah jadwal = new JadwalKuliah();
    int[] idxRuangPraktikumS = jadwal.getIdxRuangPraktikumS();
    int[] idxRuangPraktikumL = jadwal.getIdxRuangPraktikumL();
    int[] idxWaktu = jadwal.getIdxWaktu();
    int[] idxHari = jadwal.getIdxHari();

    Random rand = new Random();
    int sks, jumPeserta;
    String jenisKuliah, dosen;

    int idxDataPerkuliahan = 0;

    for (int i = 0; i < jumGen; i += jumGenPerVar) {
        //Baca data SKS dan jeniKuliah dari DataPerkuliahan
        sks = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][7]);
        jenisKuliah = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][6];
        jumPeserta = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][8]);
        dosen = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][4];

        idxDataPerkuliahan++;

        //generate random HARI
        double idxAcak = rand.nextDouble();
        if (idxAcak <= 0.2) {
            gen[i] = 1;
        } else if (idxAcak <= 0.4) {
            gen[i] = 2;
        } else if (idxAcak <= 0.6) {
            gen[i] = 3;
        } else if (idxAcak <= 0.8) {
            gen[i] = 4;
        } else if (idxAcak <= 1) {
            gen[i] = 5;
        }
    }

    //generate random WAKTU
    switch (sks) {
        case 2:
            gen[i + 1] = 1 + rand.nextInt(idxWaktu[1]);
    }
}

```



```

        break;
    case 3:
        gen[i + 1] = 1 + rand.nextInt(idxWaktu[1] - 3);
        break;
    case 4:
        gen[i + 1] = 1 + rand.nextInt(idxWaktu[1] - 3);
        break;
    }

    //cek apakah dosen ini adalah pejabat atau bukan
    for (int j = 0; j < listPejabat.length; j++) {
        //cek adakah spasi di ujung nama pejabat di excel
        if (dosen.substring(dosen.length() - 1).equals(" ")) {
            dosen = dosen.substring(0, dosen.length() - 1);
        }

        if (listPejabat[j].equals(dosen)) {
            //System.out.println(dosen + " - " + listPejabat[j]);
            gen[i + 1] = 4 + rand.nextInt(idxWaktu[1] - 4 + 1);
        }
    }

    //generate random RUANG
    switch (jenisKuliah.toLowerCase()) {
        case "teori":
            if (jumPeserta <= 40) {
                //bisa pakai ruangan S dan M
                gen[i + 2] = jadwal.getIdxRuangTeoriS()[0] + rand.nextInt(jadwal.getIdxRuangTeoriL()[1] -
jadwal.getIdxRuangTeoriS()[0] + 1);

                } else if (jumPeserta <= 60) {
                    //bisa pakai ruangan M saja
                    gen[i + 2] = jadwal.getIdxRuangTeoriM()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriM()[0] + 1);

                } else if (jumPeserta <= 70) {
                    //bisa pakai ruangan L saja
                    gen[i + 2] = jadwal.getIdxRuangTeoriL()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriL()[0] + 1);

                } else {
                    //bisa pakai ruangan L saja
                    gen[i + 2] = jadwal.getIdxRuangTeoriXL()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriXL()[0] + 1);
                }

            break;
        case "praktikum":
            if (jumPeserta <= 45) {
                //khusus pakai lab kecil (praktikum selain akuntansi dan psikologi)
                gen[i + 2] = idxRuangPraktikumS[0] + rand.nextInt(idxRuangPraktikumS[1] - idxRuangPraktikumS[0] + 1);
            } else {
                //khusus untuk praktikum akuntansi dan psikologi
                gen[i + 2] = idxRuangPraktikumL[0] + rand.nextInt(idxRuangPraktikumL[1] - idxRuangPraktikumL[0] + 1);
            }

            break;
    }
}
}

public int getGen(int idxGen) {
    return gen[idxGen];
}

```

```

public String[][] getFenotipe(int jumVar, int jumGen, int jumGenPerVar) {
    String fenotipe[][] = new String[jumVar][jumGenPerVar];
    JadwalKuliah jadwal = new JadwalKuliah();
    int idxGen = 0;
    int i;

    int sks;

    for (i = 0; i < jumVar; i++) {
        //HARI
        fenotipe[i][0] = jadwal.getHari(gen[idxGen]);

        //WAKTU
        sks = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[i][7]);
        fenotipe[i][1] = jadwal.getWaktu(gen[idxGen + 1], sks);

        //WAKTU
        fenotipe[i][2] = jadwal.getRuang(gen[idxGen + 2]);

        idxGen += jumGenPerVar;
    }

    System.out.println(idxGen + " " + i);
    return fenotipe;
}

public void infoBestKromosom(int idxKrom, int jumGen, int jumGenPerVar, int maxSKSperHari, int minSKSperHari) {
    Constraint cons = new Constraint();
    cons.cekConstraints(gen, jumGen, jumGenPerVar, maxSKSperHari, minSKSperHari);
    int penalty = cons.getPenalty();

    nilaiFitness = (100 * (jumGen / jumGenPerVar)) - penalty;
    System.out.println("Idx Kromosom   = " + idxKrom);
    System.out.println("Bentrok Dosen   = " + cons.getJumBentrokDosen());
    System.out.println("Bentrok Ruang   = " + cons.getJumBentrokRuang());
    System.out.println("Bentrok Mhswa   = " + cons.getJumBentrokMhs());
    System.out.println("SKS lebih dosen = " + cons.getJumSKSlebihDosen());
    System.out.println("SKS kurang dosen = " + cons.getJumSKSKurangDosen());
    System.out.println("SKS lebih Mhswa = " + cons.getJumSKSlebihMhs());
    System.out.println("Fitness awal    = " + 100 * (jumGen / jumGenPerVar));
    System.out.println("Penalty         = " + penalty);
    System.out.println("Fitness akhir   = " + nilaiFitness);
}

public void hitungFitness(int idxKrom, int jumGen, int jumGenPerVar, int maxSKSperHari, int minSKSperHari) {
    //Membuat objek Constraint baru
    Constraint cons = new Constraint();
    cons.cekConstraints(gen, jumGen, jumGenPerVar, maxSKSperHari, minSKSperHari);
    int penalty = cons.getPenalty();

    nilaiFitness = (100 * (jumGen / jumGenPerVar)) - penalty;

    /*if (AlgoritmaGenetika.generasi == AlgoritmaGenetika.maxGenerasi || !AlgoritmaGenetika.lanjutPencarian) {
        System.out.println("Idx Kromosom   = " + idxKrom);
        System.out.println("Bentrok Dosen   = " + cons.getJumBentrokDosen());
        System.out.println("Bentrok Ruang   = " + cons.getJumBentrokRuang());
        System.out.println("Bentrok Mhswa   = " + cons.getJumBentrokMhs());
        System.out.println("SKS lebih dosen = " + cons.getJumSKSlebihDosen());
        System.out.println("SKS lebih Mhswa = " + cons.getJumSKSlebihDosen());
        System.out.println("Fitness awal    = " + 100 * (jumGen / jumGenPerVar));
        System.out.println("Penalty         = " + penalty);
    }*/
    kuliahBentrokDosen = cons.getKuliahBentrokDosen();
    kuliahBentrokRuang = cons.getKuliahBentrokRuang();
    kuliahBentrokMhs = cons.getKuliahBentrokMhs();
}

```

```

kuliahSKSharianLebihDosen = cons.getKuliahSKSharianLebihDosen();
kuliahSKSharianKurangDosen = cons.getKuliahSKSharianKurangDosen();
kuliahSKSharianLebihMhs = cons.getKuliahSKSharianLebihMhs();
kuliahDosenLembur = cons.getKuliahDosenLembur();
}

public void setNilaiFitness(int nilaiFitness) {
    this.nilaiFitness = nilaiFitness;
}

public int getNilaiFitness() {
    return nilaiFitness;
}

public void setGen(int idx, int gen) {
    this.gen[idx] = gen;
}

public void mutasiCreep(int idxKrom, int jumGen, int jumGenPerVar, double pm) {
    JadwalKuliah jadwal = new JadwalKuliah();
    boolean gantiHari, gantiWaktu, gantiRuang;
    String jenisKuliah, dosen;

    int[] tempKrom = new int[jumGen];
    int tempFitness;

    int idxPerkuliahan = 0;
    Random rand = new Random();

    for (int i = 0; i < jumGen; i += jumGenPerVar) {
        jenisKuliah = BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][6];
        dosen = BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][4];
        int sks = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][7]);
        int jumPeserta = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][8]);

        double acak = rand.nextDouble();
        gantiHari = false;
        gantiWaktu = false;
        gantiRuang = false;

        if (acak <= pm) {
            //mutasi gen yang memiliki bentrok jadwal MAHASISWA ==> ubah HARI kuliahnya
            if (kuliahBentrokMhs[idxPerkuliahan] == 1 || kuliahBentrokDosen[idxPerkuliahan] == 1 ||
kuliahDosenLembur[idxPerkuliahan] == 1) {
                //ubah HARI perkuliahannya
                double acakMutasi = rand.nextDouble();

                if (acakMutasi < 0.5) {
                    //ganti HARI
                    gen[i] += 1;

                    //antisipasi jika nilai HARI yg baru melebihi batas atas
                    if (gen[i] > jadwal.getIdxHari()[1]) {
                        gen[i] = jadwal.getIdxHari()[0];
                    }

                    gantiHari = true;
                } else {
                    //ganti WAKTU
                    gen[i + 1] += 1;

                    //antisipasi jika nilai WAKTU yg baru melebihi batas atas
                    switch (sks) {
                        case 2:
                            if (gen[i + 1] > jadwal.getIdxWaktu()[1]) {

```

```

        gen[i + 1] = jadwal.getIdxWaktu()[0];

        //cek apakah dosen ini pejabat atau bukan
        for (int j = 0; j < listPejabat.length; j++) {
            //cek adakah spasi di ujung nama pejabat di excel
            if (dosen.substring(dosen.length() - 1).equals(" ")) {
                dosen = dosen.substring(0, dosen.length() - 1);
            }

            if (dosen.equals(listPejabat[j])) {
                gen[i + 1] = 4;
                break;
            }
        }
        break;
    case 3:
        if (gen[i + 1] > jadwal.getIdxWaktu()[1] - 2) {
            gen[i + 1] = jadwal.getIdxWaktu()[0];

            //cek apakah dosen ini pejabat atau bukan
            for (int j = 0; j < listPejabat.length; j++) {
                //cek adakah spasi di ujung nama pejabat di excel
                if (dosen.substring(dosen.length() - 1).equals(" ")) {
                    dosen = dosen.substring(0, dosen.length() - 1);
                }

                if (dosen.equals(listPejabat[j])) {
                    gen[i + 1] = 4;
                    break;
                }
            }
        }
        break;
    case 4:
        if (gen[i + 1] > jadwal.getIdxWaktu()[1] - 2) {
            gen[i + 1] = jadwal.getIdxWaktu()[0];

            //cek apakah dosen ini pejabat atau bukan
            for (int j = 0; j < listPejabat.length; j++) {
                //cek adakah spasi di ujung nama pejabat di excel
                if (dosen.substring(dosen.length() - 1).equals(" ")) {
                    dosen = dosen.substring(0, dosen.length() - 1);
                }

                if (dosen.equals(listPejabat[j])) {
                    gen[i + 1] = 4;
                    break;
                }
            }
        }
        break;
    }

    gantiWaktu = true;
}

//Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
/*if (gen[i] == 6 && gen[i + 1] > 4) {
    //generate random HARI
    double idxAcak = rand.nextDouble();
    if (idxAcak <= 0.2) {
        gen[i] = 1;
    } else if (idxAcak <= 0.4) {
        gen[i] = 2;
    }
}

```

```

    } else if (idxAcak <= 0.6) {
        gen[i] = 3;
    } else if (idxAcak <= 0.8) {
        gen[i] = 4;
    } else if (idxAcak <= 1) {
        gen[i] = 5;
    }
}*/
}

//mutasi gen (ganti RUANG) yang memiliki bentrok ruangan ATAU gen yang jumPakaiRuangnya >
rerataJumPakaiRuang
if (kuliahBentrokRuang[idxPerkuliahan] == 1) {
    double acakMutasi = rand.nextDouble();

    if (acakMutasi < 0.15) {
        if (!gantiHari) {
            //ganti HARI
            gen[i] += 1;

            //antisipasi jika nilai HARI yg baru melebihi batas atas
            if (gen[i] > jadwal.getIdxHari()[1]) {
                gen[i] = jadwal.getIdxHari()[0];
            }

            //Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
            /*if (gen[i] == 6 && gen[i + 1] > 4) {
                //generate random HARI
                double idxAcak = rand.nextDouble();
                if (idxAcak <= 0.2) {
                    gen[i] = 1;
                } else if (idxAcak <= 0.4) {
                    gen[i] = 2;
                } else if (idxAcak <= 0.6) {
                    gen[i] = 3;
                } else if (idxAcak <= 0.8) {
                    gen[i] = 4;
                } else if (idxAcak <= 1) {
                    gen[i] = 5;
                }
            }*/
            gantiHari = true;
        }
    } else if (acak < 0.3) {
        if (!gantiWaktu) {
            //ganti WAKTU
            gen[i + 1] += 1;

            //antisipasi jika nilai WAKTU yg baru melebihi batas atas
            switch (sks) {
                case 2:
                    if (gen[i + 1] > jadwal.getIdxWaktu()[1]) {
                        gen[i + 1] = jadwal.getIdxWaktu()[0];

                        //cek apakah dosen ini pejabat atau bukan
                        for (int j = 0; j < listPejabat.length; j++) {
                            //cek adakah spasi di ujung nama pejabat di excel
                            if (dosen.substring(dosen.length() - 1).equals(" ")) {
                                dosen = dosen.substring(0, dosen.length() - 1);
                            }

                            if (dosen.equals(listPejabat[j])) {
                                gen[i + 1] = 4;
                                break;
                            }
                        }
                    }
                }
            }
        }
    }
}

```

```

    }
    }
    break;
case 3:
    if (gen[i + 1] > jadwal.getIdxWaktu()[1] - 2) {
        gen[i + 1] = jadwal.getIdxWaktu()[0];

        //cek apakah dosen ini pejabat atau bukan
        for (int j = 0; j < listPejabat.length; j++) {
            //cek adakah spasi di ujung nama pejabat di excel
            if (dosen.substring(dosen.length() - 1).equals(" ")) {
                dosen = dosen.substring(0, dosen.length() - 1);
            }

            if (dosen.equals(listPejabat[j])) {
                gen[i + 1] = 4;
                break;
            }
        }
    }
    break;
case 4:
    if (gen[i + 1] > jadwal.getIdxWaktu()[1] - 2) {
        gen[i + 1] = jadwal.getIdxWaktu()[0];

        //cek apakah dosen ini pejabat atau bukan
        for (int j = 0; j < listPejabat.length; j++) {
            //cek adakah spasi di ujung nama pejabat di excel
            if (dosen.substring(dosen.length() - 1).equals(" ")) {
                dosen = dosen.substring(0, dosen.length() - 1);
            }

            if (dosen.equals(listPejabat[j])) {
                gen[i + 1] = 4;
                break;
            }
        }
    }
    break;
}

//Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
/*if (gen[i] == 6 && gen[i + 1] > 4) {
    //generate random HARI
    double idxAcak = rand.nextDouble();
    if (idxAcak <= 0.2) {
        gen[i] = 1;
    } else if (idxAcak <= 0.4) {
        gen[i] = 2;
    } else if (idxAcak <= 0.6) {
        gen[i] = 3;
    } else if (idxAcak <= 0.8) {
        gen[i] = 4;
    } else if (idxAcak <= 1) {
        gen[i] = 5;
    }
}
*/
}
} else {
    switch (jenisKuliah.toLowerCase()) {
        case "teori":
            gen[i + 2] += 1 + rand.nextInt(10);

            //antisipasi jika RUANG yg baru melebihi batas nilai
            if (jumPeserta <= 40 && gen[i + 2] > jadwal.getIdxRuangTeoriL()[1]) {

```

```

//bisa pakai ruangan S dan M
gen[i + 2] = jadwal.getIdxRuangTeoriS()[0] + rand.nextInt(jadwal.getIdxRuangTeoriL()[1] -
jadwal.getIdxRuangTeoriS()[0] + 1);

} else if (jumPeserta <= 60 && gen[i + 2] > jadwal.getIdxRuangTeoriXL()[1]) {
//bisa pakai ruangan M saja
gen[i + 2] = jadwal.getIdxRuangTeoriM()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriM()[0] + 1);

} else if (jumPeserta <= 70 && gen[i + 2] > jadwal.getIdxRuangTeoriXL()[1]) {
//bisa pakai ruangan L saja
gen[i + 2] = jadwal.getIdxRuangTeoriL()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriL()[0] + 1);

} else if (jumPeserta <= 80 && gen[i + 2] > jadwal.getIdxRuangTeoriXL()[1]) {
//bisa pakai ruangan L saja
gen[i + 2] = jadwal.getIdxRuangTeoriXL()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriXL()[0] + 1);
}

//gen[i + 2] = jadwal.getIdxRuangTeoriS()[0] + rand.nextInt(jadwal.getIdxRuangTeoriL()[1] -
jadwal.getIdxRuangTeoriS()[0] + 1);
break;

case "praktikum":
gen[i + 2] += 1;

//antisipasi jika RUANG yg baru melebihi batas nilai
if (jumPeserta <= 45 && gen[i + 2] > jadwal.getIdxRuangPraktikumS()[1]) {
//khusus pakai lab kecil (praktikum selain akuntansi dan psikologi)
gen[i + 2] = jadwal.getIdxRuangPraktikumS()[0] + rand.nextInt(jadwal.getIdxRuangPraktikumS()[1] -
jadwal.getIdxRuangPraktikumS()[0] + 1);
} else if (jumPeserta <= 70 && gen[i + 2] > jadwal.getIdxRuangPraktikumL()[1]) {
//khusus untuk praktikum akuntansi dan psikologi
gen[i + 2] = jadwal.getIdxRuangPraktikumL()[0] + rand.nextInt(jadwal.getIdxRuangPraktikumL()[1] -
jadwal.getIdxRuangPraktikumL()[0] + 1);
}
break;
}
}
gantiRuang = true;
}
}

//mutasi gen yang memiliki SKS harian dosen atau mahasiswa lebih dari maxSKSperHari
if (kuliahSKSharianLebihDosen[idxPerkuliahan] == 1 /*|| kuliahSKSharianKurangDosen[idxPerkuliahan] == 1*/ ||
kuliahSKSharianLebihMhs[idxPerkuliahan] == 1) {
if (!gantiHari) {
//ubah HARI perkuliahannya
gen[i] += 1;

//antisipasi jika nilai HARI yg baru melebihi batas atas
if (gen[i] > jadwal.getIdxHari()[1]) {
gen[i] = jadwal.getIdxHari()[0];
}

//Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
/*if (gen[i] == 6 && gen[i + 1] > 4) {
//generate random HARI
double idxAcak = rand.nextDouble();
if (idxAcak <= 0.2) {
gen[i] = 1;
} else if (idxAcak <= 0.4) {
gen[i] = 2;
} else if (idxAcak <= 0.6) {

```

```

        gen[i] = 3;
    } else if (idxAcak <= 0.8) {
        gen[i] = 4;
    } else if (idxAcak <= 1) {
        gen[i] = 5;
    }
    }*/
}
}

//Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
if (gen[i] == 6 && gen[i + 1] > 4) {
    //generate random HARI
    double idxAcak = rand.nextDouble();
    if (idxAcak <= 0.2) {
        gen[i] = 1;
    } else if (idxAcak <= 0.4) {
        gen[i] = 2;
    } else if (idxAcak <= 0.6) {
        gen[i] = 3;
    } else if (idxAcak <= 0.8) {
        gen[i] = 4;
    } else if (idxAcak <= 1) {
        gen[i] = 5;
    }
}

/*if (gen[i] == 6) {
    acak = rand.nextDouble();

    if (acak < 0.5) {
        //generate random HARI
        double idxAcak = rand.nextDouble();
        if (idxAcak <= 0.2) {
            gen[i] = 1;
        } else if (idxAcak <= 0.4) {
            gen[i] = 2;
        } else if (idxAcak <= 0.6) {
            gen[i] = 3;
        } else if (idxAcak <= 0.8) {
            gen[i] = 4;
        } else if (idxAcak <= 1) {
            gen[i] = 5;
        }
    }
}*/

//update idxPerkuliahan
idxPerkuliahan++;
}
}

public void mutasiRandom(int idxKrom, int jumGen, int jumGenPerVar, double pm) {
    JadwalKuliah jadwal = new JadwalKuliah();
    boolean gantiHari, gantiWaktu, gantiRuang;
    String jenisKuliah, dosen;

    int[] tempKrom = new int[jumGen];
    int tempFitness;

    int idxPerkuliahan = 0;
    Random rand = new Random();

    for (int i = 0; i < jumGen; i += jumGenPerVar) {
        jenisKuliah = BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][6];

```



```

dosen = BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][4];
int sks = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][7]);
int jumPeserta = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxPerkuliahan][8]);

double acak = rand.nextDouble();
gantiHari = false;
gantiWaktu = false;
gantiRuang = false;

if (acak <= pm) {
    //mutasi gen yang memiliki bentrok jadwal MAHASISWA ==> ubah HARI kuliahnya
    if (kuliahBentrokMhs[idxPerkuliahan] == 1 || kuliahBentrokDosen[idxPerkuliahan] == 1 ||
    kuliahDosenLembur[idxPerkuliahan] == 1) {
        //ubah HARI perkuliahannya
        double acakMutasi = rand.nextDouble();

        if (acakMutasi < 0.5) {
            //generate random HARI
            double idxAcak = rand.nextDouble();
            if (idxAcak <= 0.2) {
                gen[i] = 1;
            } else if (idxAcak <= 0.4) {
                gen[i] = 2;
            } else if (idxAcak <= 0.6) {
                gen[i] = 3;
            } else if (idxAcak <= 0.8) {
                gen[i] = 4;
            } else if (idxAcak <= 0.95) {
                gen[i] = 5;
            }
        }

        gantiHari = true;
    } else {
        //generate random WAKTU
        switch (sks) {
            case 2:
                gen[i + 1] = 1 + rand.nextInt(jadwal.getIdxWaktu()[1]);
                break;
            case 3:
                gen[i + 1] = 1 + rand.nextInt(jadwal.getIdxWaktu()[1] - 3);
                break;
            case 4:
                gen[i + 1] = 1 + rand.nextInt(jadwal.getIdxWaktu()[1] - 3);
                break;
        }

        //cek apakah dosen ini adalah pejabat atau bukan
        for (int j = 0; j < listPejabat.length; j++) {
            //cek adakah spasi di ujung nama pejabat di excel
            if (dosen.substring(dosen.length() - 1).equals(" ")) {
                dosen = dosen.substring(0, dosen.length() - 1);
            }

            if (listPejabat[j].equals(dosen)) {
                //System.out.println(dosen + " - " + listPejabat[j]);
                gen[i + 1] = 4 + rand.nextInt(jadwal.getIdxWaktu()[1] - 4 + 1);
            }
        }

        gantiWaktu = true;
    }

    //Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
    if (gen[i] == 6 && gen[i + 1] > 4) {
        //generate random HARI
    }
}

```

```

double idxAcak = rand.nextDouble();
if (idxAcak <= 0.2) {
    gen[i] = 1;
} else if (idxAcak <= 0.4) {
    gen[i] = 2;
} else if (idxAcak <= 0.6) {
    gen[i] = 3;
} else if (idxAcak <= 0.8) {
    gen[i] = 4;
} else if (idxAcak <= 1) {
    gen[i] = 5;
}
}
}

```

//mutasi gen (ganti RUANG) yang memiliki bentrok ruangan ATAU gen yang jumPakaiRuangnya > rerataJumPakaiRuang

```

if (kuliahBentrokRuang[idxPerkuliahan] == 1) {
    double acakMutasi = rand.nextDouble();

    if (acakMutasi < 0.15) {
        if (!gantiHari) {
            //generate random HARI
            double idxAcak = rand.nextDouble();
            if (idxAcak <= 0.2) {
                gen[i] = 1;
            } else if (idxAcak <= 0.4) {
                gen[i] = 2;
            } else if (idxAcak <= 0.6) {
                gen[i] = 3;
            } else if (idxAcak <= 0.8) {
                gen[i] = 4;
            } else if (idxAcak <= 0.95) {
                gen[i] = 5;
            }

            gantiHari = true;
        }
    } else if (acak < 0.3) {
        if (!gantiWaktu) {
            //generate random WAKTU
            switch (sks) {
                case 2:
                    gen[i + 1] = 1 + rand.nextInt(jadwal.getIdxWaktu()[1]);
                    break;
                case 3:
                    gen[i + 1] = 1 + rand.nextInt(jadwal.getIdxWaktu()[1] - 3);
                    break;
                case 4:
                    gen[i + 1] = 1 + rand.nextInt(jadwal.getIdxWaktu()[1] - 3);
                    break;
            }

            //cek apakah dosen ini adalah pejabat atau bukan
            for (int j = 0; j < listPejabat.length; j++) {
                //cek adakah spasi di ujung nama pejabat di excel
                if (dosen.substring(dosen.length() - 1).equals(" ")) {
                    dosen = dosen.substring(0, dosen.length() - 1);
                }

                if (listPejabat[j].equals(dosen)) {
                    //System.out.println(dosen + " - " + listPejabat[j]);
                    gen[i + 1] = 4 + rand.nextInt(jadwal.getIdxWaktu()[1] - 4 + 1);
                }
            }
        }
    }
}

```

```

    }
    } else {
        //generate random RUANG
        switch (jenisKuliah.toLowerCase()) {
            case "teori":
                if (jumPeserta <= 40) {
                    //bisa pakai ruangan S dan M
                    gen[i + 2] = jadwal.getIdxRuangTeoriS()[0] + rand.nextInt(jadwal.getIdxRuangTeoriL()[1] -
jadwal.getIdxRuangTeoriS()[0] + 1);

                } else if (jumPeserta <= 60) {
                    //bisa pakai ruangan M saja
                    gen[i + 2] = jadwal.getIdxRuangTeoriM()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriM()[0] + 1);

                } else if (jumPeserta <= 70) {
                    //bisa pakai ruangan L saja
                    gen[i + 2] = jadwal.getIdxRuangTeoriL()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriL()[0] + 1);

                } else {
                    //bisa pakai ruangan L saja
                    gen[i + 2] = jadwal.getIdxRuangTeoriXL()[0] + rand.nextInt(jadwal.getIdxRuangTeoriXL()[1] -
jadwal.getIdxRuangTeoriXL()[0] + 1);
                }

                break;
            case "praktikum":
                if (jumPeserta <= 45) {
                    //khusus pakai lab kecil (praktikum selain akuntansi dan psikologi)
                    gen[i + 2] = jadwal.getIdxRuangPraktikumS()[0] + rand.nextInt(jadwal.getIdxRuangPraktikumS()[1] -
jadwal.getIdxRuangPraktikumS()[0] + 1);
                } else {
                    //khusus untuk praktikum akuntansi dan psikologi
                    gen[i + 2] = jadwal.getIdxRuangPraktikumL()[0] + rand.nextInt(jadwal.getIdxRuangPraktikumL()[1] -
jadwal.getIdxRuangPraktikumL()[0] + 1);
                }

                break;
        }
    }
    gantiRuang = true;

    //Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
    if (gen[i] == 6 && gen[i + 1] > 4) {
        //generate random HARI
        double idxAcak = rand.nextDouble();
        if (idxAcak <= 0.2) {
            gen[i] = 1;
        } else if (idxAcak <= 0.4) {
            gen[i] = 2;
        } else if (idxAcak <= 0.6) {
            gen[i] = 3;
        } else if (idxAcak <= 0.8) {
            gen[i] = 4;
        } else if (idxAcak <= 1) {
            gen[i] = 5;
        }
    }
}
}

//mutasi gen yang memiliki SKS harian dosen atau mahasiswa lebih dari maxSKSperHari
if (kuliahSKSharianLebihDosen[idxPerkuliah] == 1 || /*kuliahSKSharianKurangDosen[idxPerkuliah] == 1 ||*/
kuliahSKSharianLebihMhs[idxPerkuliah] == 1) {

```

```

if (!gantiHari) {
    //ubah HARI perkuliahannya
    gen[i] += 1;

    //antisipasi jika nilai HARI yg baru melebihi batas atas
    if (gen[i] > jadwal.getIdxHari()[1]) {
        gen[i] = jadwal.getIdxHari()[0];
    }

    //Jika HARI = "Sabtu", tidak ada yang boleh mengajar di atas jam 10.40
    if (gen[i] == 6 && gen[i + 1] > 4) {
        //generate random HARI
        double idxAcak = rand.nextDouble();
        if (idxAcak <= 0.2) {
            gen[i] = 1;
        } else if (idxAcak <= 0.4) {
            gen[i] = 2;
        } else if (idxAcak <= 0.6) {
            gen[i] = 3;
        } else if (idxAcak <= 0.8) {
            gen[i] = 4;
        } else if (idxAcak <= 1) {
            gen[i] = 5;
        }
    }
}
}

//update idxPerkuliahan
idxPerkuliahan++;
}

public void hitungJumPakaiRuang(int jumGen, int jumGenPerVar) {
    //reset isi jumPakaiRuang
    for (int i = 0; i < jumGen; i += jumGenPerVar) {
        //System.out.println(this.gen[i + 2]);
        jumPakaiRuang[gen[i + 2]] = 0;
    }

    for (int i = 0; i < jumGen; i += jumGenPerVar) {
        //System.out.println(this.gen[i + 2]);
        jumPakaiRuang[gen[i + 2]]++;
    }
}

public void tampilJumPakaiRuang() {
    JadwalKuliah jadwal = new JadwalKuliah();
    //System.out.println(jumPakaiRuang.length);
    for (int i = 1; i < jumPakaiRuang.length; i++) {
        System.out.println("Ruang " + jadwal.getRuang(i) + ": " + jumPakaiRuang[i]);
    }
}

public void tampilBentrokRuang(int jumGen, int jumGenPerVar) {
    for (int i = 0; i < jumGen; i += jumGenPerVar) {
        if (kuliahBentrokRuang[i / jumGenPerVar] == 1) {
            System.out.println("Ruang Bentrok: " + gen[i + 2]);
        }
    }
}
}
}

```

Constraint.java

```
package timetable_uty;

public class Constraint {

    private int totPenalty;
    private final int[] penalty = {100, 50, 10, 5};
    private int[] kuliahBentrokDosen, kuliahBentrokRuang, kuliahBentrokMhs, kuliahSKSharianLebihDosen,
    kuliahSKSharianLebihMhs, kuliahSKSharianKurangDosen, kuliahDosenLembur;
    public int jumBentrokDosen, jumBentrokRuang, jumBentrokMhs, jumSKSlebihDosen, jumSKSkurangDosen,
    jumSKSlebihMhs, jumRuangKuliahDosenBeda, jumDosenLembur;
    //private final String[] jenisHardConstraint;
    //private final String[] jenisSoftConstraint;

    public Constraint() {
        totPenalty = 0;
        jumBentrokDosen = 0;
        jumBentrokRuang = 0;
        jumBentrokMhs = 0;
        jumSKSlebihDosen = 0;
        jumSKSlebihMhs = 0;
        jumRuangKuliahDosenBeda = 0;
        jumDosenLembur = 0;
    }

    public int getPenalty() {
        return totPenalty;
    }

    public int[] getKuliahBentrokDosen() {
        return kuliahBentrokDosen;
    }

    public int[] getKuliahBentrokRuang() {
        return kuliahBentrokRuang;
    }

    public int[] getKuliahBentrokMhs() {
        return kuliahBentrokMhs;
    }

    public int[] getKuliahSKSharianLebihDosen() {
        return kuliahSKSharianLebihDosen;
    }

    public int[] getKuliahSKSharianKurangDosen() {
        return kuliahSKSharianKurangDosen;
    }

    public int[] getKuliahSKSharianLebihMhs() {
        return kuliahSKSharianLebihMhs;
    }

    public int[] getKuliahDosenLembur(){
        return kuliahDosenLembur;
    }

    public int getJumBentrokDosen() {
        return jumBentrokDosen;
    }

    public int getJumBentrokRuang() {
        return jumBentrokRuang;
    }
}
```

```

public int getJumBentrokMhs() {
    return jumBentrokMhs;
}

public int getJumSKSlebihDosen() {
    return jumSKSlebihDosen;
}

public int getJumSKSkurangDosen() {
    return jumSKSkurangDosen;
}

public int getJumSKSlebihMhs() {
    return jumSKSlebihMhs;
}

public int getJumRuangKuliahDosenBeda() {
    return jumRuangKuliahDosenBeda;
}

public void cekConstraints(int[] gen, int jumGen, int jumGenPerVar, int maxSKSperHari, int minSKSperHari) {
    String dosen1, dosen2, kelas1, kelas2, jenisKuliah1, jenisKuliah2;
    int hari1, hari2, waktu1, waktu2, ruang1, ruang2, sks1, sks2;
    boolean bentrokDosen, bentrokRuang, bentrokMhs, dosenLembur, ruangDosenBeda;

    kuliahBentrokDosen = new int[jumGen / jumGenPerVar];
    kuliahBentrokRuang = new int[jumGen / jumGenPerVar];
    kuliahBentrokMhs = new int[jumGen / jumGenPerVar];
    kuliahSKSharianLebihDosen = new int[jumGen / jumGenPerVar];
    kuliahSKSharianKurangDosen = new int[jumGen / jumGenPerVar];
    kuliahSKSharianLebihMhs = new int[jumGen / jumGenPerVar];
    kuliahDosenLembur = new int[jumGen / jumGenPerVar];

    int idxDataPerkuliahan;
    int jumSKSperHariMhs, jumSKSperHariDosen;

    JadwalKuliah jadwal = new JadwalKuliah();

    //variabel untuk mencatat hariMengajar dosen
    String prevDosen = "";

    //array untuk mencatat hari mengajar dosen
    int[] hariMengajar = new int[jadwal.getIdxHari()][1];

    for (int i = 0; i < (jumGen - jumGenPerVar + 1); i += jumGenPerVar) {
        idxDataPerkuliahan = i / jumGenPerVar;
        //System.out.println(idxDataPerkuliahan);

        dosen1 = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][4];
        kelas1 = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][5];
        sks1 = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][7]);
        jenisKuliah1 = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan][6];
        hari1 = gen[i];
        waktu1 = gen[i + 1];
        ruang1 = gen[i + 2];
        jumSKSperHariMhs = sks1;
        jumSKSperHariDosen = sks1;

        //System.out.println(dosen1);
        //baca waktu perkuliahan
        int jamMulai1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(0, 2));
        int menitMulai1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(3, 5));
        int jamAkhir1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(8, 10));
        int menitAkhir1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(11));
    }
}

```

```

bentrokDosen = false;
bentrokRuang = false;
bentrokMhs = false;
ruangDosenBeda = false;
dosenLembur = false;

//System.out.println(dosen1 + " - " + kelas1 + " " + waktu1);
for (int j = (i + jumGenPerVar); j < jumGen; j += jumGenPerVar) {
    dosen2 = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan + 1][4];
    kelas2 = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan + 1][5];
    sks2 = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan + 1][7]);
    jenisKuliah2 = BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan + 1][6];
    hari2 = gen[j];
    waktu2 = gen[j + 1];
    ruang2 = gen[j + 2];

    //System.out.println(dosen2 + " - " + kelas2 + " " + waktu2);
    //baca waktu perkuliahan 2
    int jamMulai2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(0, 2));
    int menitMulai2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(3, 5));
    int jamAkhir2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(8, 10));
    int menitAkhir2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(11));

    //ubah waktu menjadi bilangan bulat
    int waktuMulai1 = (100 * jamMulai1) + menitMulai1;
    int waktuAkhir1 = (100 * jamAkhir1) + menitAkhir1;
    int waktuMulai2 = (100 * jamMulai2) + menitMulai2;
    int waktuAkhir2 = (100 * jamAkhir2) + menitAkhir2;

    //cek apakah satu dosen mengajar beberapa kuliah di hari dan waktu yang sama
    if (dosen1.toLowerCase().equals(dosen2.toLowerCase())) {
        if (hari1 == hari2) {
            if ((waktuMulai2 <= waktuAkhir1 && waktuMulai2 >= waktuMulai1)
                || waktuAkhir2 <= waktuAkhir1 && waktuAkhir2 >= waktuMulai1) {
                bentrokDosen = true;
            }
        }
    }

    //cek apakah satu ruangan dipakai oleh beberapa dosen di hari dan waktu yang sama
    if (ruang1 == ruang2) {
        if (hari1 == hari2) {
            if ((waktuMulai2 <= waktuAkhir1 && waktuMulai2 >= waktuMulai1)
                || (waktuAkhir2 <= waktuAkhir1 && waktuAkhir2 >= waktuMulai1)) {
                bentrokRuang = true;
            }
        }
    }

    //cek apakah satu kelas mahasiswa mengikuti beberapa kuliah di hari dan waktu yang sama
    if (kelas1.toLowerCase().equals(kelas2.toLowerCase())) {
        if (hari1 == hari2) {
            if ((waktuMulai2 <= waktuAkhir1 && waktuMulai2 >= waktuMulai1)
                || (waktuAkhir2 <= waktuAkhir1 && waktuAkhir2 >= waktuMulai1)) {
                bentrokMhs = true;
            }
        }
    }

    //cek apakah satu dosen memberikan kuliah lebih dari sks maksimum atau kurang dari sks minimum yang diizinkan
    if (dosen1.toLowerCase().equals(dosen2.toLowerCase())) {
        if (hari1 == hari2) {
            jumSKSperHariDosen += sks2;
        }
    }
}

```

```

    }

    //cek apakah satu mahasiswa mengikuti kuliah lebih dari sks maksimum yang diizinkan
    if (kelas1.toLowerCase().equals(kelas2.toLowerCase())) {
        if (hari1 == hari2) {
            jumSKSperHariMhs += sks2;
        }
    }

    //cek apakah RUANG dari satu DOSEN dalam satu HARI sama atau tidak
    if (dosen1.toLowerCase().equals(dosen2) && hari1 == hari2 &&
jenisKuliah1.toLowerCase().equals(jenisKuliah2.toLowerCase())) {
        if (ruang1 != ruang2) {
            ruangDosenBeda = true;
        }
    }

    //cek apakah satu DOSEN memiliki jam mengajar paling awal dan paling akhir
    if (dosen1.toLowerCase().equals(dosen2) && hari1 == hari2) {
        if (waktu1 == 1 && waktu2 > 6){
            dosenLembur = true;
        }
    }

    //update idxDataPerkuliah
    if (idxDataPerkuliah < MainForm.JumPerkuliah - 1) {
        idxDataPerkuliah++;
    }
}

//beri penalti jika bentrokDosen = true
if (bentrokDosen) {
    totPenalty += penalty[0];
    kuliahBentrokDosen[i / jumGenPerVar] = 1;
    jumBentrokDosen++;
}

//beri penalti jika bentrokRuang = true
if (bentrokRuang) {
    totPenalty += penalty[0];
    kuliahBentrokRuang[i / jumGenPerVar] = 1;
    jumBentrokRuang++;
}

//beri penalti jika bentrokMhs = true
if (bentrokMhs) {
    totPenalty += penalty[1];
    kuliahBentrokMhs[i / jumGenPerVar] = 1;
    jumBentrokMhs++;
}

//beri penalti jika jumSKSperHariDosen > maxSKSperHari
if (jumSKSperHariDosen > maxSKSperHari) {
    totPenalty += penalty[2];
    kuliahSKSharianLebihDosen[i / jumGenPerVar] = 1;
    jumSKSlebihDosen++;
}

//beri penalti jika jumSKSperHariDosen < minSKSperHari
if (jumSKSperHariDosen < minSKSperHari) {
    totPenalty += penalty[2];
    kuliahSKSharianKurangDosen[i / jumGenPerVar] = 1;
    jumSKSkurangDosen++;
}
}

```



```

//beri penalti jika jumSKSperHariMhs < maxSKSperHari
if (jumSKSperHariMhs > maxSKSperHari) {
    totPenalty += penalty[3];
    kuliahSKSharianLebihMhs[i / jumGenPerVar] = 1;
    jumSKSlebihMhs++;
}

//beri penalti jika dosenLembur == true
if (dosenLembur) {
    totPenalty += penalty[2];
    kuliahDosenLembur[i / jumGenPerVar] = 1;
    jumDosenLembur++;
}
}

//update
if (jumBentrokDosen == 0 && jumBentrokMhs == 0 && jumBentrokRuang == 0
    && jumSKSlebihDosen == 0 && jumSKSlebihMhs == 0 && jumDosenLembur == 0) {
    AlgoritmaGenetika.lanjutPencarian = false;

    //penalty dijadikan 0 agar kromosom tersebut menjadi kromosom terbaik
    //totPenalty = 0;
}
}

public boolean isBentrokRuang(int idxGen, int[] gen, int jumGen, int jumGenPerVar) {
    boolean bentrokRuang = false;
    int hari1, hari2, waktu1, waktu2, ruang1, ruang2, sks1, sks2;
    int idxDataPerkuliahan1, idxDataPerkuliahan2;

    idxDataPerkuliahan1 = idxGen / jumGenPerVar;
    idxDataPerkuliahan2 = 0;

    sks1 = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[idxDataPerkuliahan1][7]);
    hari1 = gen[idxGen];
    waktu1 = gen[idxGen + 1];
    ruang1 = gen[idxGen + 2];

    //baca waktu perkuliahan 1
    JadwalKuliah jadwal = new JadwalKuliah();
    int jamMulai1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(0, 2));
    int menitMulai1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(3, 5));
    int jamAkhir1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(8, 10));
    int menitAkhir1 = Integer.parseInt(jadwal.getWaktu(waktu1, sks1).substring(11));

    for (int j = 0; j < jumGen; j += jumGenPerVar) {
        //System.out.println("idxGen: " + idxGen + " - idxDataPerkuliahan2: " + idxDataPerkuliahan2);
        sks2 = Integer.parseInt(BacaDataSiapOlah.DataPerkuliahan[j / jumGenPerVar][7]);
        hari2 = gen[j];
        waktu2 = gen[j + 1];
        ruang2 = gen[j + 2];

        //baca waktu perkuliahan 2
        int jamMulai2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(0, 2));
        int menitMulai2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(3, 5));
        int jamAkhir2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(8, 10));
        int menitAkhir2 = Integer.parseInt(jadwal.getWaktu(waktu2, sks2).substring(11));

        //ubah waktu menjadi bilangan bulat
        int waktuMulai1 = (100 * jamMulai1) + menitMulai1;
        int waktuAkhir1 = (100 * jamAkhir1) + menitAkhir1;
        int waktuMulai2 = (100 * jamMulai2) + menitMulai2;
        int waktuAkhir2 = (100 * jamAkhir2) + menitAkhir2;
    }
}

```

```

//cek apakah satu ruangan dipakai oleh beberapa dosen di hari dan waktu yang sama
if (idxGen != j) {
    if (ruang1 == ruang2) {
        if (hari1 == hari2) {
            if ((waktuMulai2 <= waktuAkhir1 && waktuMulai2 >= waktuMulai1)
                || (waktuAkhir2 <= waktuAkhir1 && waktuAkhir2 >= waktuMulai1)) {
                bentrokRuang = true;
                break;
            }
        }
    }
}

//update idxPerkuliahan
if (idxDataPerkuliahan2 < MainForm.JumPerkuliahan - 1) {
    idxDataPerkuliahan2++;
}

if (bentrokRuang) {
    break;
}

return bentrokRuang;
}
}

```

JadwalKuliah.java

```

package timetable_uty;

public class JadwalKuliah {

    private String ruang, waktu, hari;
    private int kuota;
    private final int[] idxRuangTeoriS = {1, 1};
    private final int[] idxRuangTeoriM = {2, 16};
    private final int[] idxRuangTeoriL = {17, 48};
    private final int[] idxRuangTeoriXL = {49, 53};
    private final int[] idxRuangPraktikumS = {54, 64};
    private final int[] idxRuangPraktikumL = {65, 66};
    private final int[] idxWaktu = {1, 8};
    private final int[] idxHari = {1, 6};

    public String getHari(int kode) {
        //method untuk mengecek Hari perkuliahan berdasarkan kode hari pada gen
        hari = "";
        switch (kode) {
            case 1:
                hari = "Senin";
                break;
            case 2:
                hari = "Selasa";
                break;
            case 3:
                hari = "Rabu";
                break;
            case 4:
                hari = "Kamis";
                break;
            case 5:
                hari = "Jumat";
                break;
            case 6:

```

```

        hari = "Sabtu";
        break;
    }

    return hari;
}

public int[] getIdXRuangTeoriS() {
    return idxRuangTeoriS;
}

public int[] getIdXRuangTeoriM() {
    return idxRuangTeoriM;
}

public int[] getIdXRuangTeoriL() {
    return idxRuangTeoriL;
}

public int[] getIdXRuangTeoriXL() {
    return idxRuangTeoriXL;
}

public int[] getIdXRuangPraktikumS() {
    return idxRuangPraktikumS;
}

public int[] getIdXRuangPraktikumL() {
    return idxRuangPraktikumL;
}

public int[] getIdXWaktu() {
    return idxWaktu;
}

public int[] getIdXHari() {
    return idxHari;
}

public String getWaktu(int kode, int sks) {
    //method untuk mengecek Waktu/ jam perkuliahan berdasarkan kode waktu dan sks pada gen
    waktu = "";

    //waktu urutanny diubah berdasarkan waktu mulai, tidak usah dikelompokkan berdasarkan sks
    switch (kode) {
        case 1:
            waktu = "07.00"; //selisih dgn jam berikutnya = 110 menit
            break;
        case 2:
            waktu = "08.50"; //selisih dgn jam berikutnya = 50 menit
            break;
        case 3:
            waktu = "09.40"; //selisih dgn jam berikutnya = 60 menit
            break;
        case 4:
            waktu = "10.40"; //selisih dgn jam berikutnya = 130 menit
            break;
        case 5:
            waktu = "12.50"; //selisih dgn jam berikutnya = 110 menit
            break;
        case 6:
            waktu = "14.40"; //selisih dgn jam berikutnya = 70 menit
            break;
        case 7:
            waktu = "15.30"; //selisih dgn jam berikutnya = 60 menit. tidak cocok untuk matkul sks = 3 dan 4
    }
}

```

```

        break;
    case 8:
        waktu = "16.30"; //tidak cocok untuk matkul sks = 3 dan 4
        break;
    }

    //hitung waktu akhir kuliah
    String waktuAkhir;
    int jamMulai, menitMulai, jamAkhir = 0, menitAkhir = 0;

    jamMulai = Integer.parseInt(waktu.substring(0, 2));
    menitMulai = Integer.parseInt(waktu.substring(3));

    switch (sks) {
        case 2:
            jamAkhir = jamMulai + 1;
            menitAkhir = menitMulai + 40;
            break;
        case 3:
            jamAkhir = jamMulai + 2;
            menitAkhir = menitMulai + 30;
            break;
        case 4:
            jamAkhir = jamMulai + 3;
            menitAkhir = menitMulai + 20;
            break;
    }

    //perbaiki jamAkhir dan menitAkhir
    if (menitAkhir >= 60) {
        jamAkhir += 1;
        menitAkhir -= 60;
    }

    //menuliskan waktuAkhir dalam format jj.mm
    if (jamAkhir >= 10) {
        waktuAkhir = String.valueOf(jamAkhir).concat(".");
    } else {
        waktuAkhir = "0" + String.valueOf(jamAkhir).concat(".");
    }

    if (menitAkhir >= 10) {
        waktuAkhir = waktuAkhir.concat(String.valueOf(menitAkhir));
    } else {
        waktuAkhir = waktuAkhir.concat("0" + String.valueOf(menitAkhir));
    }

    waktu = waktu.concat(" - " + waktuAkhir);

    return waktu;
}

public String getRuang(int kode) {
    //method untuk mengecek Ruang perkuliahan berdasarkan kode ruang pada gen
    ruang = "";

    switch (kode) {
        //untuk matkul teori
        case 1:
            ruang = "G31";
            break;
        case 2:
            ruang = "D01";
            break;
        case 3:

```

```
    ruang = "D03";
    break;
case 4:
    ruang = "D04";
    break;
case 5:
    ruang = "E02";
    break;
case 6:
    ruang = "E03";
    break;
case 7:
    ruang = "E04";
    break;
case 8:
    ruang = "E11";
    break;
case 9:
    ruang = "E12";
    break;
case 10:
    ruang = "E13";
    break;
case 11:
    ruang = "E14";
    break;
case 12:
    ruang = "F05";
    break;
case 13:
    ruang = "F25";
    break;
case 14:
    ruang = "G23";
    break;
case 15:
    ruang = "G25";
    break;
case 16:
    ruang = "G33";
    break;
case 17:
    ruang = "D11";
    break;
case 18:
    ruang = "D12";
    break;
case 19:
    ruang = "D13";
    break;
case 20:
    ruang = "D14";
    break;
case 21:
    ruang = "D21";
    break;
case 22:
    ruang = "D22";
    break;
case 23:
    ruang = "D23";
    break;
case 24:
    ruang = "D24";
    break;
```

```
case 25:
    ruang = "D31";
    break;
case 26:
    ruang = "D32";
    break;
case 27:
    ruang = "D33";
    break;
case 28:
    ruang = "D34";
    break;
case 29:
    ruang = "E21";
    break;
case 30:
    ruang = "E22";
    break;
case 31:
    ruang = "E23";
    break;
case 32:
    ruang = "E24";
    break;
case 33:
    ruang = "E31";
    break;
case 34:
    ruang = "E32";
    break;
case 35:
    ruang = "E33";
    break;
case 36:
    ruang = "E34";
    break;
case 37:
    ruang = "F01";
    break;
case 38:
    ruang = "F02";
    break;
case 39:
    ruang = "F03";
    break;
case 40:
    ruang = "F04";
    break;
case 41:
    ruang = "H21";
    break;
case 42:
    ruang = "H22";
    break;
case 43:
    ruang = "H23";
    break;
case 44:
    ruang = "H24";
    break;
case 45:
    ruang = "H31";
    break;
case 46:
    ruang = "H32";
```

```
        break;
    case 47:
        ruang = "H33";
        break;
    case 48:
        ruang = "H34";
        break;
    case 49:
        ruang = "D02";
        break;
    case 50:
        ruang = "E01";
        break;
    case 51:
        ruang = "E15";
        break;
    case 52:
        ruang = "F26";
        break;
    case 53:
        ruang = "F36";
        break;

    //untuk matkul praktik
    case 54:
        ruang = "LK-01";
        break;
    case 55:
        ruang = "LK-02";
        break;
    case 56:
        ruang = "LK-03";
        break;
    case 57:
        ruang = "LK-04";
        break;
    case 58:
        ruang = "LK-05";
        break;
    case 59:
        ruang = "LK-06";
        break;
    case 60:
        ruang = "LK-07";
        break;
    case 61:
        ruang = "LK-08";
        break;
    case 62:
        ruang = "LK-09";
        break;
    case 63:
        ruang = "LK-10";
        break;
    case 64:
        ruang = "LK-11";
        break;
    case 65:
        ruang = "LK-12";
        break;
    case 66:
        ruang = "LK-13";
        break;
}
```

```
    return ruang;
}

public int getKuota(int kode) {
    //method untuk mengecek Ruang perkuliahan berdasarkan kode ruang pada gen
    int kuota;

    if (kode == 1) {
        kuota = 40;
    } else if (kode <= 16) {
        kuota = 60;
    } else if (kode <= 48){
        kuota = 70;
    } else {
        kuota = 90;
    }

    return kuota;
}
}
```