# Tugas 1 (revisi 1)
# *Review* Pemrograman Java dalam Penggunaan *Collections* dan *Generics*

# Struktur Data dan Algoritma
# IKI10400
# Semester Gasal 2010/2011

# Fakultas Ilmu Komputer
# Universitas Indonesia

**Batas waktu pengumpulan kode sumber:**
Senin, 27 September 2010 pukul 21.00 Waktu Server Aren

Kode sumber yang dinilai hanya yang dikumpulkan melalui Aren. Kode sumber yang dikumpulkan melalui mekanisme selain itu akan diabaikan dan dianggap tidak mengumpulkan.

Peringatan: jangan mengumpulkan pekerjaan beberapa menit menjelang batas waktu pengumpulan karena ada kemungkinan pengumpulan gagal dilakukan atau koneksi internet terputus!

Jika tidak dapat mengumpulkan tugas sebelum batas waktu karena suatu atau beberapa hal khusus, mahasiswa yang bersangkutan harus melakukan langkah-langkah dalam SOP Perpanjangan Batas Waktu Pengumpulan Pekerjaan.

# Antrian Rumah Sakit 1011

| | | |
|---|---|---|
| Nama berkas kode sumber | : | `SDA1011.java` |
| Batas waktu eksekusi program | : | 2 detik / kasus uji |
| Batas memori program | : | 32 MiB / kasus uji |

## Latar Belakang

Departemen Sistem Informasi Rumah Sakit 1011 ingin mengembangkan sebuah sistem antrian baru agar pelayanan rumah sakit kepada pasien jauh lebih efisien daripada yang sebelumnya. Banyak keluhan dari pasien atau masyarakat umum mengenai perihal lamanya waktu mengantri yang sampai ke Departemen Pelayanan Umum. Keluhan itu kemudian dikumpulkan untuk dianalisis lebih lanjut. Setelah dianalisis lebih lanjut, ternyata sistem antrian yang lama berdasarkan teknik FCFS (*First Come First Served*) tidaklah efisien karena tidak memperhatikan bebarapa faktor yang cukup krusial. Misalkan tanggal-tanggal sibuk seperti masa pengambilan gaji, masa tutup buku bulanan, masa tutup buku tahunan, atau hari libur nasional; faktor jam-jam sibuk yang cukup mempengaruhi; faktor kondisi pelanggan juga hal yang perlu dipertimbangkan. Tentu saja diinginkan sebuah strategi agar sebanyak mungkin pasien yang dapat terlayani dengan suatu kondisi tertentu. Oleh karena itu, Rumah Sakit 1011 mempercayakan Pak Chanek sebagai Manajer Proyek Sistem Informasi Antrian Rumah Sakit 1011.

Terlepas dari cerita di balik asal-usulnya dari pengajar sampai manajer, singkat cerita Pak Chanek teringat salah satu rekannya yang bekerja di Departemen Sistem Informasi Bank 101110. Pak Chanek bersama timnya melakukan studi banding mengenai sistem antrian Bank 101110 dan kemudian merumuskan beberapa aturan sesuai dengan kondisi Rumah Sakit 1011 yang dispesifikasikan dalam paragraf-paragraf berikut.

## Spesifikasi Antrian Rumah Sakit

Pengembangan sistem ini dibagi menjadi beberapa tahap. Pada tahap awal pengembangan, banyak antrian ditetapkan hanya satu karena sinkronisasi dan aturan banyak antrian sedang dikaji lebih lanjut oleh pihak rumah sakit. Antrian dalam sistem yang dimaksud adalah antrian untuk mengurus hal-hal administratif pasien, pembagian ruang pelayanan (tempat pemeriksaan, konsultasi, Unit Gawat Darurat (UGD), dan sebagainya), dan dokter yang bertugas untuk memeriksa/mengobati/mengoperasi pasien. Antrian-antrian yang terjadi di luar setiap ruang pelayanan belum menjadi bagian dari sistem pada tahap awal ini. Selain itu, waktu operasional rumah sakit tidak diperhitungkan dalam sistem karena rumah sakit tersebut buka 24 jam dan tidak pernah libur walau pada hari libur nasional sekali pun. Namun demikian, sistem dibatasi untuk antrian pada satu hari saja untuk tahap awal pengembangan.

Antrian sistem tersebut memiliki aturan-aturan prioritas yang terurut sebagai berikut.

1. Prioritas pertama diberikan kepada pasien yang harus dibawa ke UGD karena kecelakaan atau mengalami luka yang sangat parah. Baru setelahnya adalah pasien-pasien yang tidak perlu dibawa ke UGD.
2. Jika terdapat beberapa pasien yang memiliki prioritas sama di muka antrian untuk diproses berikutnya, maka pasien yang datang pertama kali dalam antrian yang memiliki prioritas tertinggi.
3. Jika terdapat beberapa pasien yang datang bersamaan ke dalam antrian, prioritas-prioritas berikutnya secara berurutan dari yang tertinggi diberikan kepada balita, anak-anak, lansia, remaja, lalu yang terakhir dewasa diterapkan. Pasien dikategorikan berdasarkan usia (dibulatkan dalam tahun):
   - balita: < 5 tahun
   - anak-anak: 5 s.d. 12 tahun
   - remaja: 13 s.d. 19 tahun
   - dewasa: 20 s.d. 64 tahun
   - lansia: > 64 tahun

Antrian tersebut diatur oleh Pak Satpam berdasarkan kedatangan calon pasien. Setiap pasien yang saat ini akan diproses keluar dari antrian dan menuju loket administrasi. Proses administrasi untuk pasien remaja, dewasa, dan lansia memerlukan waktu 2 menit. Sedangkan untuk pasien balita dan anak-anak diperlukan waktu 3 menit karena adanya penimbangan berat dan pengukuran tinggi badan, kecuali mereka yang harus dibawa ke UGD hanya 2 menit. Tentu saja pasien dapat ditemani dan/atau diantar oleh keluarga, teman, perawat, atau orang lain untuk mengurus hal-hal tersebut sehingga pasien yang harus dibawa ke UGD sudah dapat dibawa ke UGD segera dan proses administrasi dapat diselesaikan oleh yang lain.

Kali ini Anda sebagai salah satu pengembang perangkat lunak dipercaya Pak Chanek untuk membantu pekerjaannya mengembangkan sistem informasi tersebut. Program yang Anda buat merupakan program tahap awal sehingga nantinya akan diuji coba dengan sekumpulan kasus kedatangan pasien pada hari-hari sebelumnya. Hasil keluaran dari program akan digunakan sebagai data untuk dianalisis lebih lanjut sehingga program yang dibuat harus bisa mensimulasikan deskripsi di atas dengan benar dan berjalan dalam waktu yang singkat karena banyaknya kasus uji dan lagipula Pak Chanek tidak suka menunggu terlalu lama. Selain itu, program yang Anda buat akan dipakai untuk pengembangan tahap selanjutnya. Oleh karena itu, program Anda harus ditulis dengan gaya penulisan yang baik dan diberikan dokumentasi yang jelas. Buatlah program yang dimaksud!

## Format Masukan

Masukan dibaca dari masukan standar. Masukan terdiri dari satu atau beberapa baris. Masukan diproses sampai ditemukan sebuah dan satu-satunya tulisan "`AKHIR DATA`" (selanjutnya tanda kutip hanya untuk kejelasan) pada sebuah baris. Setiap baris yang lain

(banyaknya tidak lebih dari 50000), masing-masing barisnya berisi sebuah informasi mengenai seorang pasien yang memiliki format berikut.

```
Nama,StatusUGD,WaktuKedatangan,Usia
```

*Nama* merupakan nama pasien yang hanya akan terdiri dari huruf kapital A..Z, huruf kecil a..z, dan spasi. Panjang *Nama* minimal 1 karakter dan tidak akan lebih dari 30 karakter. *StatusUGD* merupakan status yang berisi "ya" atau "tidak" yang menyatakan apakah pasien itu harus dibawa ke UGD atau tidak. *WaktuKedatangan* merupakan waktu kedatangan pasien pertama kali ke dalam antrian pada hari itu. Format *WaktuKedatangan* adalah "*hh*:*mm*:*ss*" dengan $0 \le hh < 24$ dan $0 \le mm$ & $ss < 60$. Usia merupakan usia pasti pasien dalam satuan tahun (bilangan bulat nonnegatif) atau perkiraan usia pasien (misalnya pasien merupakan korban kecelakaan yang dibawa warga sekitar lokasi kecelakaan dan pasien tidak membawa kartu identitas). Perkiraan usia pasien memiliki format:

- "<*X*" artinya usia pasien lebih kecil dari *X*.
- "<=*X*" artinya usia pasien lebih kecil atau sama dengan *X*.
- ">*X*" artinya usia pasien lebih besar dari *X*.
- ">=*X*" artinya usia pasien lebih besar atau sama dengan *X*.
- "*X*-*Y*" artinya usia pasien dari *X* s.d. *Y* (termasuk *X* dan *Y*).

Dijamin nama pasien pasti unik. Dijamin perkiraan usia pasien termasuk ke dalam tepat satu kategori usia. Dijamin pula tidak akan ada dua pasien atau lebih yang memiliki prioritas yang sama persis dalam antrian. Waktu kedatangan pasien dalam antrian pada masukan dijamin terurut secara tidak menurun (*nondescending*).

## Format Keluaran

Keluaran ditulis ke keluaran standar. Keluaran berisi daftar pasien (daftar pertama) yang berada dalam antrian yang tidak lebih dari lama mengantri (untuk pasien yang harus dibawa ke UGD paling lama 300 detik, sedangkan pasien lainnya paling lama 30000 detik) diikuti dengan sebuah baris kosong; lalu langsung diikuti daftar pasien (daftar kedua) yang berada dalam antrian yang lebih dari lama mengantri. Setiap baris pada daftar berisi informasi pasien. Jika daftar yang dimaksud tidak berisi pasien, tuliskan "KOSONG" pada sebuah baris.

Setiap informasi pasien pada daftar pertama memiliki format sebagai berikut (sebuah karakter # melambangkan sebuah spasi yang digunakan hanya untuk penegasan).

```
StatusHari#(WaktuPelayanan)#>#Nama#(KategoriUsia):#StatusUGD
```

Setiap informasi pasien pada daftar kedua memiliki format sebagai berikut.

```
Nama#(KategoriUsia):#StatusUGD
```

*StatusHari* akan berisi "Hari ini" atau "Besok" sesuai dengan hari ketika pasien dilayani di loket. *WaktuPelayanan* merupakan waktu pasien dilayani di loket yang memiliki format "*hh*:*mm*:*ss*" dengan *hh*, *mm*, dan *ss* mempunyai batasan yang sama dengan batasan pada masukan. *Nama* merupakan nama pasien yang dilayani dan harus tertulis apa adanya sesuai dengan yang ada pada masukan. *KategoriUsia* berisi salah satu dari "balita", "anak-anak", "remaja", "dewasa", atau "lansia". *StatusUGD* akan berisi "ya" jika pasien harus dibawa ke UGD dan "tidak" jika sebaliknya.

Informasi-informasi pasien pada daftar pertama diurutkan berdasarkan waktu pasien dilayani di loket (tentu saja yang dilayani hari ini akan ditampilkan terlebih dahulu dibandingkan besok). Sedangkan, informasi-informasi pasien pada daftar kedua diurutkan terlebih dahulu berdasarkan *StatusUGD* (yang harus dibawa ke UGD ditampilkan terlebih dahulu dibandingkan yang tidak); lalu jika statusnya sama, berdasarkan *KategoriUsia* (urutannya dari balita, anak-anak, remaja, dewasa, lalu lansia); lalu jika masih sama, berdasarkan *Nama* (gunakan metode `compareTo()` untuk membandingkan `String`, misalnya `nama1.compareTo(nama2)`).

### Contoh Masukan 1

```
Rudi,tidak,08:00:00,19
Linglung,ya,08:00:00,>70
tak bernama,ya,08:00:10,11
Sisatu,ya,08:00:30,30
Sidua,ya,08:01:00,35
AKHIR DATA
```

### Contoh Keluaran 1

```
Hari ini (08:00:00) > Linglung (lansia): ya
Hari ini (08:02:00) > tak bernama (anak-anak): ya
Hari ini (08:04:00) > Sisatu (dewasa): ya
Hari ini (08:06:00) > Sidua (dewasa): ya
Hari ini (08:08:00) > Rudi (remaja): tidak

KOSONG
```

### Contoh Masukan 2

```
tak bernama,ya,23:59:00,30-40
Yandatau,tidak,23:59:05,50
Sikecil,tidak,23:59:05,3
AKHIR DATA
```

### Contoh Keluaran 2

```
Hari ini (23:59:00) > tak bernama (dewasa): ya
```

```
Besok (00:01:00) > Sikecil (balita): tidak
Besok (00:04:00) > Yandatau (dewasa): tidak

KOSONG
```

### Contoh Masukan 3

```
Rudi,tidak,08:00:00,19
Linglung,ya,08:00:00,>70
tak bernama,ya,08:00:10,11
Sisatu,ya,08:00:30,30
Sidua,ya,08:00:59,35
Sitiga,ya,08:00:59,9-11
AKHIR DATA
```

### Contoh Keluaran 3

```
Hari ini (08:00:00) > Linglung (lansia): ya
Hari ini (08:02:00) > tak bernama (anak-anak): ya
Hari ini (08:04:00) > Sisatu (dewasa): ya
Hari ini (08:06:00) > Rudi (remaja): tidak

Sitiga (anak-anak): ya
Sidua (dewasa): ya
```

### Petunjuk

Sejauh mungkin, manfaatkanlah fasilitas yang tersedia dalam Java Collections API. Berikut adalah contoh potongan kode sumber yang dapat dipelajari untuk membantu penyelesaian Tugas 1. Ingat berikut ini adalah contoh, bukan solusi!

```
import java.util.Comparator;
import java.util.PriorityQueue;


/** SDA1011 ...
    @author ...
  */
public class SDA1011 {
    /** main ...
      * @param args ...
      */
    public static void main(String[] args) {
        // MAKS_KAPASITAS_ANTRIAN ...
        final int MAKS_KAPASITAS_ANTRIAN = 50000;

        // antrian ...
        PriorityQueue<Pasien> antrian = new PriorityQueue<Pasien>(
            MAKS_KAPASITAS_ANTRIAN, new PakSatpam());

        antrian.add(new Pasien("A", 50));
        antrian.add(new Pasien("C", 40));
```

```
        antrian.add(new Pasien("B", 40));

        while (antrian.size() != 0) {
            System.out.println(antrian.poll());
        }
    }
}


/** PakSatpam ...
  * @author ...
  *
  * Perbandingan dilakukan berdasarkan usia terlebih dahulu.
  * Usia yang lebih muda mendapat prioritas yang lebih tinggi dibandingkan
  * usia yang lebih tua.
  * Jika usia kedua pasien sama, perbandingan dilakukan terhadap nama
  * mengikuti aturan urutan ASCII pada metode compareTo().
  */
class PakSatpam implements Comparator<Pasien> {
    /** compare ...
      * @param pasien1 ...
      * @param pasien2 ...
      * @return ...
      */
    public int compare(Pasien pasien1, Pasien pasien2) {
        // result ...
        int result = pasien1.getUsia() - pasien2.getUsia();

        if (result == 0) {
            // Lihat penjelasan metode compareTo() pada java.lang.String.
            result = pasien1.getNama().compareTo(pasien2.getNama());
        }

        return result;
    }
}


/** Pasien ...
  * @author ...
  */
class Pasien {
    /** nama ... */
    private String nama;

    /** usia ... */
    private int usia;

    //Constructor(s)

    /** Pasien ...
      * @param nama ...
      * @param usia ...
      */
    public Pasien(String nama, int usia) {
        setNama(nama);
```

```
        setUsia(usia);
    }

    //Method(s)

    /** setNama ...
      * @param nama ...
      */
    public void setNama(String nama) {
        this.nama = nama;
    }

    /** setUsia ...
      * @param usia ...
      */
    public void setUsia(int usia) {
        this.usia = usia;
    }

    /** getNama ...
      * @return ...
      */
    public String getNama() {
        return nama;
    }

    /** getUsia ...
      * @return ...
      */
    public int getUsia() {
        return usia;
    }

    /** toString ...
      * @return ...
      */
    public String toString() {
        return getNama() + " (" + getUsia() + ")";
    }
}
```

## Kriteria Penilaian

Terdapat tiga bagian penilaian, yaitu:

- penilaian penilai otomatis Aren.
- penilaian *white-box review* (detail lihat di bawah).
- penilaian dokumentasi dan gaya penulisan pemrograman (detail lihat di SCeLE).

Komponen-komponen penilaian *white-box review* adalah sebagai berikut.

- Penggunaan ADT yang tepat, termasuk penggunaan struktur data dan pembanding yang tepat dan benar.
- Kode, termasuk terdapat implementasi untuk membaca masukan dan struktur data (kelas) yang menyimpan data pasien.

- Modularitas, termasuk terdapat kelas terpisah untuk fungsi masukan dan keluaran, kelas terpisah yang menyatakan pasien, dan kelas terpisah yang menyatakan pembanding.
- Enkapsulasi, termasuk ADT-nya yang tepat terdefinisi pada kelas yang menyatakan antrian dan semua detail mengenai keluaran terdefinisi pada kelas yang menyatakan keluaran.
- *Generics.*