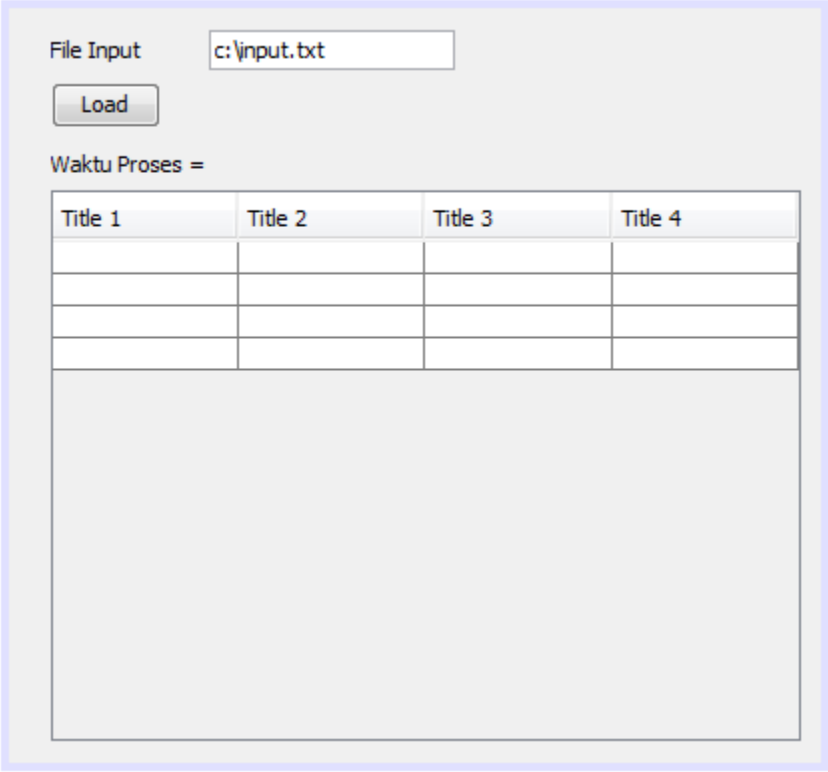
JTable

Untuk dapat menyelesaikan modul ini anda akan perlu untuk membaca Javadoc dari kelas JTable dan AbstractTableModel. Perintah-perintah berwarna merah menunjukkan bahwa anda perlu mencari cara untuk melakukan perintah tersebut dalam Javadoc.

1. Persiapan

Buatlah sebuah kelas dengan nama Antarmuka yang akan berfungsi sebagai GUI dari program kita. Kelas ini akan memiliki bentuk mirip seperti gambar di bawah ini. Anda dapat membuat kelas ini dengan GUI designer pada Netbeans.



Gambar 1 Rancangan GUI

GUI di atas terdiri dari :

- 2 JLabel.
- 1 JTextField.
- 1 JButton.

- 1 JTable. JTable ini akan kita beri nama tabelPasanganTitik.

Jalankanlah program anda dan pastikan bahwa sudah tidak ada error pada tahap ini. (catatan : isi dan header dari JTable tidak harus persis seperti pada contoh. Anda hanya harus memastikan bahwa frame anda sudah memuat sebuah JTable dengan nama tabelPasanganTitik)

2. JTable dan TableModel

Seperti telah dijelaskan di kelas, JTable menggunakan sebuah objek yang meng-*implement* interface TableModel sebagai sumber datanya (**TableModel**-nya). Jumlah kolom, jumlah baris, nama header, dan isi dari sebuah JTable ditentukan oleh objek TableModel milik JTable tersebut. Untuk menunjukkan cara menggunakan TableModel, kita akan memulai dengan membuat sebuah kelas yang merupakan keturunan dari AbstractTableModel. AbstractTableModel adalah superclass dari kelas DefaultTableModel yang ada di slide kuliah.

a. Buatlah kelas dengan nama TableModelPasanganTitik. Kelas ini merupakan subclass dari kelas AbstractTableModel. Pastikan bahwa semua method abstract dari superclass-nya telah anda override.
b. Buatlah agar objek TableModel ini memiliki 3 kolom. **Bacalah dengan baik-baik Javadoc dari method-method yang telah anda override pada poin a.** Perhatikan bahwa cara agar TableModel ini memiliki 3 kolom adalah dengan cara membuat sebuah method di kelas ini mengembalikan nilai 3.
c. Buatlah agar objek TableModel ini memiliki 2 baris.
d. Buatlah agar tiap cell dari TableModel ini berisi tulisan “test”.

Tahap berikutnya adalah menyambungkan sebuah objek dari kelas TableModelPasanganTitik dengan objek tabelPasanganTitik. Langkah-langkah di bawah ini mengasumsikan bahwa anda membuat kelas Antarmuka dengan GUI designer milik Netbeans.

Perhatikanlah isi dari konstruktor kelas Antarmuka!

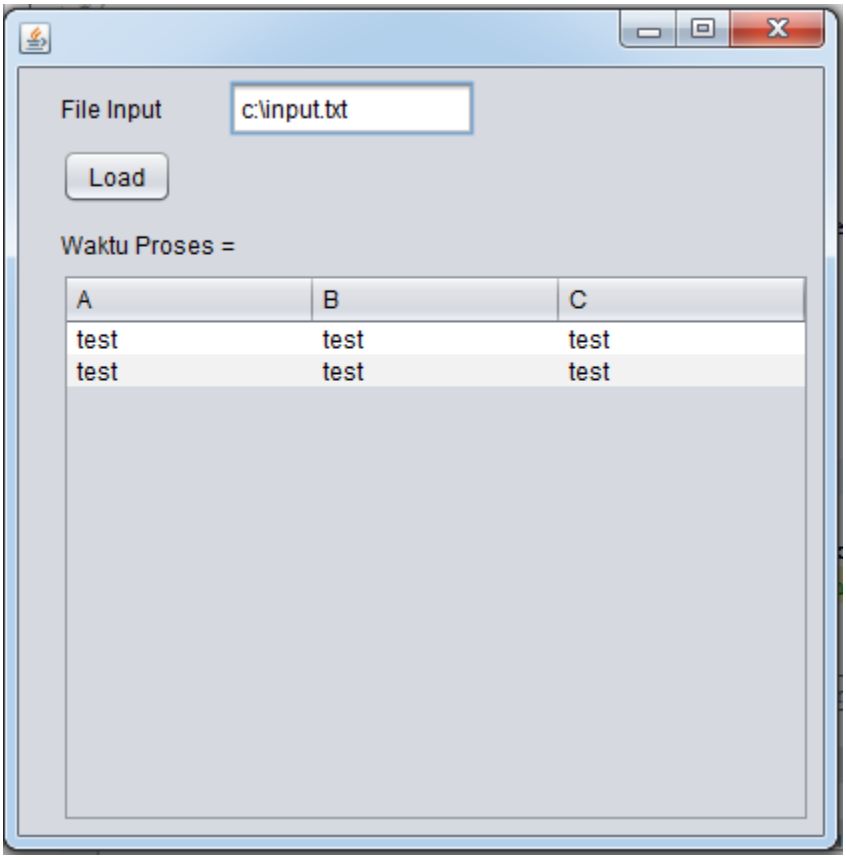
```
public Antarmuka() {
    initComponents();

}
```

Gambar 2 Isi Awal Konstruktor Kelas Antarmuka

initComponents adalah sebuah method yang secara otomatis akan di-*generate* oleh NetBeans untuk menghasilkan GUI sesuai dengan yang anda buat pada GUI designer. Karena Netbeans tidak memperbolehkan kita mengedit method initComponents, maka kita akan menyambungkan objek JTable anda dengan sebuah objek TableModelPasanganTitik **setelah** method initComponents selesai dipanggil.

a. Tambahkanlah sebuah atribut bertipe TableModelPasanganTitik lalu isilah atribut ini dengan sebuah objek dari kelas TableModelPasanganTitik.
b. *Set*-lah model dari JTable anda dengan objek TableModelPasanganTitik tersebut.
c. Jalankan program anda! Bila anda telah melakukannya dengan benar maka JTable anda akan terdiri dari 3 kolom dan 2 baris dengan tiap cell-nya berisi tulisan "test".



Gambar 3 Tampilan Program Pada Akhir Bagian 2

d. Buatlah agar baris Header dari JTable ini berisi "Titik" untuk kolom 1 dan "Pasangan Terdekat" untuk kolom 2.

3. Menggabungkan GUI dengan kelas Peta

Pada bagian ini kita akan menggabungkan GUI yang telah kita buat dengan kelas bernama Peta. **Anda tidak perlu membuat kelas ini**. Kelas Peta berfungsi untuk menyimpan objek-objek dari kelas Titik. Kelas Titik merepresentasikan sebuah titik dalam bidang 2 dimensi. Kelas Peta memiliki dua buah method yaitu :

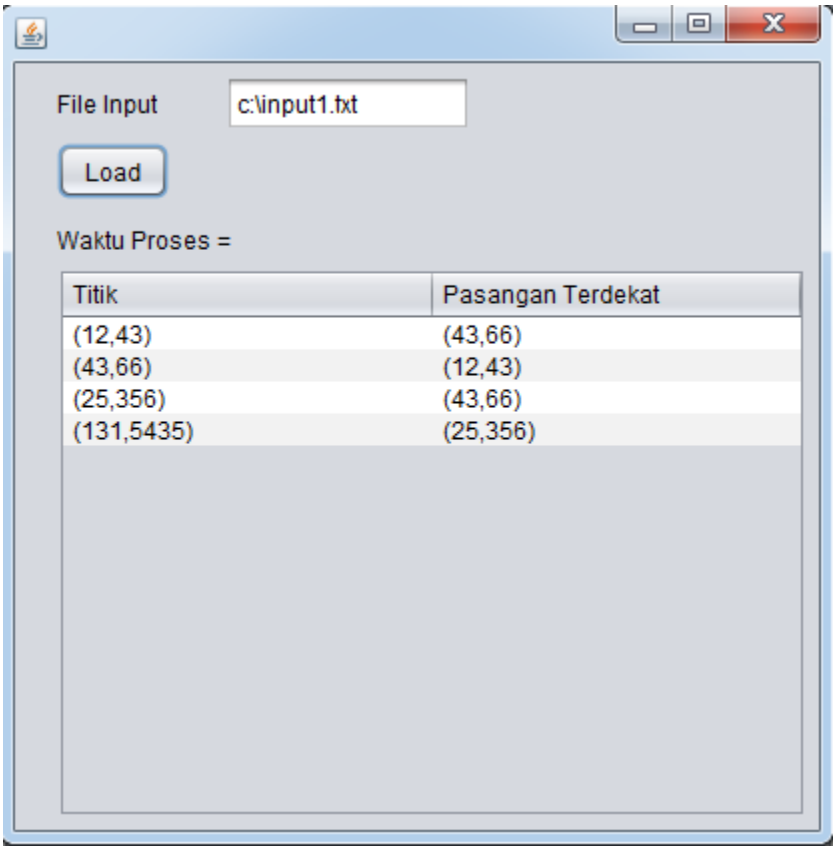
- loadFromFile : membaca dan menyimpan data titik-titik yang disimpan dalam sebuah file teks. Tiap baris dalam file ini akan memuat data dari satu buah titik. Tiap titik terdiri dari

komponen X dan komponen Y. Komponen X adalah nilai pertama dalam sebuah baris dan komponen Y adalah nilai kedua dalam baris tersebut. Komponen X dan komponen Y dari sebuah titik akan dipisahkan oleh sebuah spasi.

- getPasanganTitik : mencari pasangan terdekat dari tiap titik.

Tujuan kita adalah agar kelas Antarmuka berfungsi sebagai GUI untuk kelas Peta. Pada saat tombol "Load" ditekan, sebuah objek dari kelas Peta akan membaca file yang *path*-nya ditulis dalam JTextField File Input. Setelah itu method getPasanganTitik dari objek itu akan dipanggil dan hasilnya akan ditampilkan dalam tabelPasanganTitik.

Di bawah ini adalah contoh hasil akhir yang diinginkan setelah kita menjalankan file input1.txt :



Gambar 4 Tampilan Program

Untuk mencapai hasil akhir tersebut, lakukanlah langkah-langkah berikut ini :

a. Tambahkan atribut bertipe Peta pada kelas Antarmuka dan buatlah objeknya dalam konstruktor kelas Antarmuka.
b. Tambahkanlah event handler pada JButton load file untuk event mouseClicked. Handler ini akan memanggil method loadFromFile dari objek Peta dengan parameternya adalah isi dari JTextField File Input. Pastikan bahwa program anda dapat membaca sebuah file input yang valid tanpa menghasilkan exception!

Bila anda mendapatkan java.io.FileNotFoundException, periksalah apakah dua hal ini terpenuhi :

- Path yang ada dalam JTextField valid dan mengarah ke file input yang valid.
- Anda mengerti bagaimana cara menulis 1 karakter "\" dalam sebuah literal String. Untuk menguji pemahaman anda, cobalah untuk menulis "c:\input.txt" ke layar menggunakan System.out.println. Bila anda belum berhasil, berarti anda belum mengerti caranya :p

4. Memodifikasi kelas TableModelPasanganTitik

Seperti telah dibahas sebelumnya, untuk mengubah isi dari sebuah JTable kita harus mengubah *behaviour* dari objek TableModel yang digunakan oleh JTable tersebut.

1. Kita perlu menyimpan data-data yang dikembalikan oleh method getPasanganTitik agar TableModel ini dapat menentukan isi dari sebuah cell (lihat perintah 4.d). Tambahkanlah sebuah atribut pada kelas TableModelPasanganTitik untuk menyimpan data-data tersebut.
2. Perbaikilah method-method yang menentukan jumlah kolom, jumlah baris, dan isi cell agar *behaviour*-nya sesuai dengan kebutuhan kita.
3. Tambahkanlah sebuah method dengan header :

   public void updateData(Titik[][] newData)

   pada kelas TableModelPasanganTitik. Method ini berfungsi sebagai setter untuk atribut data sekaligus untuk memberitahu kepada objek JTable-nya bahwa isi TableModel ini telah berubah dan JTable tersebut harus meng-*update* tampilannya.

   Pemberitahuan ini dapat dilakukan dengan memanggil method fireTableDataChanged dari kelas AbstractTableModel setelah atribut datanya di-set. Pemanggilan method fireTableDataChanged akan men-*trigger* sebuah event yang akan di-*handle* oleh objek JTable yang terhubung dengan AbstractTableModel ini. Event ini di-*handle* dengan cara menggambar ulang tampilan JTable tersebut. Bila anda belum memahami apa arti "event" dan apa yang dimaksud dengan "di-*handle*" bacalah kembali catatan kuliah anda! (sayangnya pada saat anda melakukan ini biasanya terjadi null pointer exception…)

```
public void updateData(Titik[][] input)
{
    this.data=input;
    this.fireTableDataChanged();
}
```

Gambar 5 Isi Method updateData

4. Modifikasilah handler mouseClicked dari JButton Load File sehingga setelah method loadFromFile selesai dilakukan, method ini akan memanggil method getPasanganTitik dari kelas Peta dan memberikan hasilnya ke TableModelPasanganTitik melalui method updateData.

5. Ujilah program anda dan perbaiki kesalahan-kesalahan yang masih ada☺
6. Hitunglah waktu yang dibutuhkan oleh sebuah file input dan tampilkan hasilnya (dalam satuan milidetik pada JTextLabel Waktu Proses.
7. Ujilah program anda dengan file input3.txt. Perhatikan bahwa GUI kita pada saat ini akan berhenti pada saat proses masih berjalan. Pada minggu depan kita akan mempelajari cara untuk membuat program kita lebih responsif

Kumpulkanlah tugas ini sebagai R04xxyyy.jar!