SKRIPSI

KONVERSI JAVADOC KE LaTeX

Adli Fariz Bonaputra

NPM: 2012730082

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN

«tahun»

UNDERGRADUATE THESIS

«JUDUL BAHASA INGGRIS»

Adli Fariz Bonaputra

NPM: 2012730082

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY

«tahun»

LEMBAR PENGESAHAN

KONVERSI JAVADOC KE LATEX

Adli Fariz Bonaputra

NPM: 2012730082

Bandung, «tanggal» «bulan» «tahun»

Menyetujui,

Pembimbing Utama

Pembimbing Pendamping

Pascal Alfadian, M.Comp.

Ketua Tim Penguji

Anggota Tim Penguji

«penguji 1»

«penguji 2»

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

KONVERSI JAVADOC KE LaTeX

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal «tanggal» «bulan» «tahun»

Meterai
Rp. 6000

Adli Fariz Bonaputra
NPM: 2012730082

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Kata-kata kunci: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Indonesia»

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»
Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Keywords: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Inggris»

«kepada siapa anda mempersembahkan skripsi ini. . . ?»

KATA PENGANTAR

«Tuliskan kata pengantar dari anda di sini . . . »
Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Bandung, «bulan» «tahun»

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Dalam sebuah penelitian, membuat dokumentasi perlu dilakukan. Dokumentasi yang dibuat bisa dalam bentuk *hardcopy* atau *softcopy*, tergantung kebutuhannya. Dokumentasi adalah kegiatan untuk mencatat suatu peristiwa atau aktifitas yang dianggap berharga atau penting. Dokumentasi yang sudah dibuat dapat menjadi referensi untuk memandu dalam melakukan sebuah aktifitas.

Dalam bidang Teknologi Informasi, dokumentasi kode program java umumnya ditulis dalam format *Javadoc*. *Javadoc* adalah sebuah *tools* yang dimiliki oleh Java yang berguna untuk mengekstrak informasi dari sebuah *file* java menjadi sebuah dokumentasi. Umumnya digunakan untuk mendokumentasikan sebuah nama *class*, *interface*, *method* dan *custom tag*. Oleh karena itu, *Javadoc* sangatlah penting karena dapat memuat berbagai informasi dari sebuah *file* java. Informasi tersebut dapat menjelaskan sebuah *class* yang dibuat dalam sebuah dokumentasi perangkat lunak.

Skripsi mahasiswa Program Studi Teknik Informatika Fakultas Teknologi Informasi dan Sains (FTIS) Universitas Katolik Parahyangan (Unpar) adalah membuat perangkat lunak. Perangkat lunak yang dibuat umumnya menggunakan bahasa pemrograman *java*. Seperti yang sudah dijelaskan, bahasa pemrograman *java* memiliki *Javadoc* sebagai informasi dari *class*, *interface*, *method* dan juga *custom tag* yang dibuat, sehingga informasi tersebut dapat digunakan sebagai penjelasan perangkat lunak pada dokumentasi perangkat lunak. Untuk mendokumentasikan perangkat lunak yang dibuat, seluruh mahasiswa diwajibkan untuk menggunakan LaTeX dalam pembuatan sebuah dokumentasi Skripsi. LaTeX merupakan bahasa *markup* untuk menyusun sebuah dokumentasi. LaTeX membuat apa yang ditampilkan sama seperti apa yang yang ditulis. Umumnya bentuk akhir dari dokumen yang dibuat oleh LaTeX biasanya berupa sebuah *file* PDF

Pada salah satu bab dokumentasi Skripsi, terdapat penjelasan dari setiap *class* pada perangkat lunak yang dibuat. Penjelasan tersebut sebenarnya dapat diambil dari *Javadoc* yang telah dibuat pada kelas *java*, namun saat ini berdasarkan pengamatan tersebut masih diketik secara manual dari *Javadoc* ke dalam format LaTeX, sehingga membutuhkan lebih banyak waktu untuk mendokumentasikan setiap *class* pada perangkat lunak yang dibuat.

Oleh karena itu, perlu dikembangkan sebuah perangkat lunak yang dapat mengekstraksi informasi pada *Javadoc* ke format LaTeX secara otomatis. Perangkat lunak ini mengimplementasikan sebuah *Application Programming Interface* (API) yang digunakan untuk mengambil informasi berupa nama *class*, *interface*, *method* dan juga *custom tag* yang terdapat pada sebuah *file java*

1.2 Rumusan Masalah

Bedasarkan latar belakang yang telah disebutkan di atas, maka dihasilkan beberapa poin yang menjadi rumusan masalah dari masalah ini. Rumusan masalah yang akan dibangun antara lain sebagai berikut:

1. Bagaimana membuat perangkat lunak yang dapat mengonversikan format *Javadoc* ke dalam format LaTeX secara otomatis?

2. Bagaimana antarmuka yang baik untuk perangkat lunak yang akan dibuat?

1.3 Tujuan

Adapun tujuan yang ingin dicapai dari penelitian ini adalah menjawab rumusan masalah di atas, yaitu:

1. Membuat perangkat lunak yang dapat mengonversikan format Javadoc ke format LaTeX secara otomatis.
2. Mempelajari antarmuka yang baik untuk perangkat lunak yang akan dibuat.

1.4 Batasan Masalah

Agar pembahasan masalah tidak terlalu luas, masalah yang akan dikaji di dalam penelitian ini memiliki batasan, yaitu:

1. Perangkat lunak yang dikembangkan menggunakan bahasa pemrograman *Java*
2. Perangkat lunak hanya dapat menerima masukan data berupa sekumpulan *file java*
3. Perangkat lunak hanya menghasilkan *output* berupa format LaTeX yang selanjutnya akan dimasukkan ke dalam file LaTeX

1.5 Metodologi

Untuk menyelesaikan penelitian ini disusunlah tahap-tahap tugas yang perlu dilakukan. Tahap-tahap yang dimaksud adalah sebagai berikut:

1. Melakukan studi literatur untuk mengetahui *syntax* yang terdapat pada LaTeX dan mengetahui apa saja isi dari dokumentasi Javadoc Doclet API.
2. Melakukan survei terhadap format penulisan pada suatu bab pada skripsi yang berisi tentang dokumentasi perangkat lunak yang dibuat. Membutuhkan minimal 3 dokumen skripsi sebagai panduan format penulisan.

1.6 Sistematika Pembahasan

1. Bab 1 Pendahuluan
   Bab ini membahas mengenai latar belakang, rumusan masalah, tujuan, batas masalah, metologi penelitian dan sistematika penulisan.
2. Bab 2 Dasar Teori
3. Bab 3 Analisis
4. Bab 4 Perancangan
5. Bab 5 Implementasi dan Pengujian
6. Bab 6 Kesimpulan dan Saran
   Bab ini akan membahas mengenai kesimpulan dari penelitian yang telah dilakukan dan saran-saran untuk pengembangan lebih lanjut dari penelitian ini.

BAB 2

LANDASAN TEORI

Bab ini akan membahas teori-teori yang akan menjadi dasar dari penelitian ini. Teori yang dibahas yaitu mengenai *Javadoc* dan *Doclet*

2.1 Javadoc

Javadoc adalah sebuah *tools* yang dimiliki oleh *Java* yang berguna untuk mengekstrak informasi dari sekumpulan *source file java* menjadi sebuah dokumentasi. Umumnya *Javadoc* menghasilkan sekumpulan *file* HTML yang mendeskripsikan sebuah *class*, *interface*, *method* dan *custom tag*. *Javadoc* dapat mengekstraksi informasi tersebut dari sebuah *package java*, sebuah *file java* atau keduanya. [1]

2.1.1 *Processing of source files*

Javadoc akan memproses *file* yang memiliki akhiran *".java"* dan keseluruhan *file* yang terdapat di dalam folder yang sama. *Javadoc* dapat mengambil informasi dari 1 atau lebih *file java* dan sebuah *package*.

Javadoc dapat memproses sebuah *link* secara otomatis yang mengarah kepada sebuah *package*, *class* dan sebuah nama yang akan didokumentasikan pada saat *Javadoc* memprosesnya. *Link-link* tersebut berada pada beberapa posisi seperti:

1. *Declaration* (*return types*, *argument types*, *field types*)
2. Bagian *"See Also"* yang dihasilkan oleh *tag @see*
3. *In-line text* yang dihasilkan oleh *tag @link*
4. *Exeption* yang dihasilkan oleh *tag @throws*
5. *Link "Specified by"* untuk *member* dari sebuah *interface*
6. *Link "Override"* untuk *member* dari sebuah *class*

Dalam mengekstrak informasi yang terdapat dalam sebuah *package java* atau beberapa *file java* umumnya menghasilkan sebuah dokumentasi standar yang berbentuk *file* HTML dan format penulisan yang mengikuti standar *Javadoc*, akan tetapi untuk menghasilkan sebuah format dokumentasi yang diingin, dapat menggunakan sebuah *doclet* yang disediakan oleh *Javadoc*.

2.1.2 *Terminology*

Terdapat beberapa istilah yang memiliki arti spesifik dalam konteks *Javadoc* sebagai berikut:

- *Generated Document*
  Dokumen yang dihasilkan oleh *Javadoc tools* adalah sebuah *file* HTML dan dibuat oleh *standard doclet*

- *Name*
  Nama dari sebuah perangkat lunak dituliskan dalam Bahasa *java* yaitu nama *package*, *class*, *interface*, *field*, *constructor* atau *method*. Nama tersebut dapat berupa informasi lengkapnya seperti *java.lang.String.equals(java.lang.Object)* atau informasi pendeknya seperti *equals(Object)*

- *Documented Classes*
  Detail dari sebuah *class* dan *interface* akan didokumentasikan pada saat *javadoc* berjalan. Untuk dapat didokumentasikan, *source file* harus tersedia, kemudian nama dari *source file* atau nama dari *package* tersebut harus diletakkan pada *javadoc command-line*

- *Included Classes*
  Class dan *Interface* akan didokumentasikan pada saat *javadoc* berjalan, hal ini sama seperti *Documented Classes*

- *Excluded Classes*
  Class dan *Interface* tidak akan didokumenasikan pada saat *javadoc* berjalan.

- *Referenced Classes*
  Class dan *Interface* yang secara eksplisit disebut oleh *class* dan *interface* lainnya, seperti *return type*, *parameter type*, *cast type*, *extended class*, *implemented interface*, *imported class*, *class* yang digunakan pada *method body*, *@see*, *@link*, *@linkplain* dan *@inheritDoc tag*

- *External Referenced Classes*
  Class yang tidak dihasilkan saat *javadoc* berjalan. Dengan kata lain, *class* tersebut tidak diletakkan pada *javadoc command-line*. *Links* akan dihasilkan jika sebuah *class* mengatakan memiliki *external references* atau *external link*.

2.1.3 *Source Files*

Javadoc akan menghasilkan *output* yang berasal dari beberapa tipe *file*, yaitu sebagai berikut:

- *Class Source Code Files*
  Setiap *class* atau *interface* dapat memiliki dokumentasinya masing-masing yang terdapat pada *file java*

- *Package Comment Files*
  Setiap *package* dapat memiliki dokumentasinya masing-masing yang terdapat pada *root* folder kemudian *Javadoc* akan menggabungkan *file-file* yang terdapat pada *root* menjadi sebuah ringkasan. Untuk membuat dokumentasi tersebut, terdapat 2 pilihan yaitu sebuah *file java* atau sebuah *file* HTML.

```
/**
 * Provides the classes necessary to create an
 * applet and the classes an applet uses
 * to communicate with its applet context.
 * <p>
 * The applet framework involves two entities:
 * the applet and the applet context.
 * An applet is an embeddable window (see the
 * {@link java.awt.Panel} class) with a few extra
 * methods that the applet context can use to
 * initialize, start, and stop the applet.
 *
 * @since 1.0
 * @see java.awt
 */
package java.lang.applet;
```

Gambar 2.1: Package-info.java

```
<HTML>
<BODY>
Provides the classes necessary to create an applet and the
classes an applet uses to communicate with its applet context.
<p>
The applet framework involves two entities: the applet
and the applet context. An applet is an embeddable
window (see the {@link java.awt.Panel} class) with a
few extra methods that the applet context can use to
initialize, start, and stop the applet.

@since 1.0
@see java.awt
</BODY>
</HTML>
```

Gambar 2.2: Package.html

Ketika *Javadoc* memproses *package* tersebut, *Javadoc* akan melakukan beberapa langkah yaitu sebagai berikut:

1. Menyalin informasi untuk diproses. Jika *file* berupa HTML maka pada bagian *<body>* hingga *</body>* akan disalin.
2. Memproses semua *tag* pada *package* yang ada.
3. Memasukan teks yang sudah diproses tersebut pada bagian bawah halaman dokumentasi yang dihasilkan.
4. Salin kalimat pertama pada *package* tersebut pada bagian atas halaman dokumentasi

- *Overview Comment Files*
  Setiap aplikasi atau sekumpulan *package* yang akan didokumentasikan akan memiliki dokumentasi *overview*. Dokumentasi tersebut dapat dibuat lebih dari 1, jika pada saat pembuatan perangkat lunak menggunakan sekumpulan *package* yang berbeda. Untuk membuat sebuah dokumentasi ini, perlu membuat sebuah *file* HTML yang umumnya bernama *overview.html*. Kemudian *Javadoc* akan memproses seperti pada *Package Comment Files*

- *Miscellaneous Unprocessed Files*
  File tersebut dapat berubah sebuah *graphic files*, *file java* dan sebuah *file* HTML.

2.1.4 *Generated Files*

Secara *default*, *javadoc* akan menggunakan *standard doclet* yang akan menghasilkan sebuah dokumentasi berformat HTML. Doclet tersebu akan menghasilkan *file* HTML secara terpisah. Terdapat 3 grup yang masing-masing grup memiliki kriterianya sendiri, 3 grup tersebut adalah sebagai berikut:

- *Basic Content Pages*
  - sebuah halaman *class* atau *interface* (*classname*.html) untuk masing-masing *class* atau *interface* yang akan didokumentasikan
  - sebuah halaman *package* (*package-summary.html*) untuk masing-masing *package* yang akan didokumentasikan
  - sebuah halaman *overview* (*overview-summary.html*) untuk keseluruhan sekumpulan *package*. Halaman ini adalah halaman utama yang dihasilkan.

- *Cross-Reference Pages*
  - sebuah halaman hirarki dari *class* untuk sekumpulan dari semua *package* (*overview-tree.html*)

  - sehalaman hirarki dari *class* untuk setiap *package* (*package-tree.html*)
  - sehalaman *"use"* (*package-use.html*) yang berisikan *package*, *classes*, *methods*, *constructors* atau *interface*. Jika diberikan sebuah *class* bernama A, makan halaman tersebut akan berisikan *subclasses* dari A, *methods* yang memiliki *return* A dan *methods* atau *constructors* dengan parameter bertipe A.
  - sebuah halaman *deprecated API* (*deprecated-list.html*). Halaman ini adalah halaman dari sekumpulan nama yang tidak direkomendasikan untuk digunakan.
  - sebuah halaman sekumpulan nilai *constant* (*constant-values.html*) untuk sekumpulan nilai *static*.
  - sebuah halaman *serialized form* (*serialized-form.html*)
  - sebuah halaman *index* (*index-*.html*).

- *Support Files*
  - sebuah halaman bantuan (*help-doc.html*).
  - sebuah halaman *index* (*index.html*) yang membuat sebuah HTML *frames*.
  - beberapa *frame file* (**-frame.html*) yang berisi sekumpulan *packages*, *class* dan *interface* dan digunakan pada saat HTML *frames* ditampilkan
  - sebuah *file* teks *package list* (*package-list*).
  - sebuah *style sheet file* (*stylesheet.css*) untuk mengontrol warna, jenis *font*, ukuran *font* dan posisi dari halamanan yang dihasilkan
  - sebuah *doc-files* yang berisikan gambar dan beberapa contoh *file java*

Javadoc akan menghasilkan 2 atau 3 HTML *frame*. *Javadoc* akan membuat minimum *frame* yang dibutuhkan. Jika hanya terdapat 1 *package*, maka *javadoc* akan membuat 1 *frame* yang berisi dari sekumpulan *class* pada *package* tersebut. Jika terdapat lebih dari 2 *package*, maka *javadoc* akan membuat 3 *frame* dari sekumpulan *package*. Jika *class* yang digunakan adalah *java.applet.Applet* dan semua dokumentasi yang dihasilkan akan berada pada folder yang bernama *apidocs*, struktur *file* yang dihasilkan adalah sebagai berikut:

```
apidocs                          Top directory
    index.html                   Initial page that sets up HTML frames
  * overview-summary.html        Lists all packages with first sentence summaries
    overview-tree.html           Lists class hierarchy for all packages
    deprecated-list.html         Lists deprecated API for all packages
    constant-values.html         Lists values of static fields for all packages
    serialized-form.html         Lists serialized form for all packages
  * overview-frame.html          Lists all packages, used in upper-left frame
    allclasses-frame.html        Lists all classes for all packages, used in lower-left frame
    help-doc.html                Lists user help for how these pages are organized
    index-all.html               Default index created without -splitindex option
    index-files                  Directory created with -splitindex option
        index-<number>.html      Index files created with -splitindex option
    package-list                 Lists package names, used only for resolving external refs
    stylesheet.css               HTML style sheet for defining fonts, colors and positions
    java                         Package directory
        applet                   Subpackage directory
            Applet.html          Page for Applet class
            AppletContext.html   Page for AppletContext interface
            AppletStub.html      Page for AppletStub interface
            AudioClip.html       Page for AudioClip interface
          * package-summary.html Lists classes with first sentence summaries for this package
          * package-frame.html   Lists classes in this package, used in lower left-hand frame
          * package-tree.html    Lists class hierarchy for this package
            package-use          Lists where this package is used
            doc-files            Directory holding image and example files
            class-use            Directory holding pages API is used
                Applet.html      Page for uses of Applet class
                AppletContext.html Page for uses of AppletContext interface
                AppletStub.html  Page for uses of AppletStub interface
                AudioClip.html   Page for uses of AudioClip interface
    src-html                     Source code directory
        java                     Package directory
            applet               Subpackage directory
                Applet.html      Page for Applet source code
                AppletContext.html Page for AppletContext source code
                AppletStub.html  Page for AppletStub source code
                AudioClip.html   Page for AudioClip source code
```

Gambar 2.3: Struktur *file* yang dihasilkan

2.2 Doclet

Doclet yang terdapat pada *Javadoc* dapat digunakan untuk menghasilkan sebuah *output Javadoc* yang dapat disesuaikan. Standar *doclet* yang dihasilkan oleh *Javadoc* adalah dokumentasi dengan format HTML. Selain menghasilkan *output* yang dapat disesuaikan, Doclet juga dapat mengekstrak informasi secara spesifik. [2]

2.2.1 *Method-Method* pada Doclet

Berikut adalah beberapa *interface* yang terdapat pada *Doclet*:

- *RootDoc* sebuah *interface* yang menyatakan sebuah *root* dari perangkat lunak yang dibuat. Dari *root* tersebut semua informasi dapat diekstrak. *Method-method* yang digunakan adalah sebagai berikut
  - *classes() Method* ini akan mengembalikan sejumlah *class* dan *interface* pada *package*
- *ClassDoc* sebuah *interface* yang menyatakan informasi dari sebuah *class*. Informasi tersebut dapat berupa nama *class*, nama *method* dan *tag*. *Method-method* yang digunakan adalah sebagai berikut
  - *name() Method* ini akan mengembalikan sebuah nama *class* atau *interface* pada *package*
  - *commentText() Method* ini akan mengembalikan sebuah informasi dari deskripsi *class*
  - *methods() Method* ini akan mengembalikan sebuah *array of methods*
- *MethodDoc* sebuah *interface* yang menyatakan informasi dari sebuah *method*. *Method-method* yang digunakan adalah sebagai berikut
  - *name() Method* ini akan mengembalikan sebuah nama *method*
  - *modifiers() Method* ini akan mengembalikan sebuah *access modifier* dari sebuah *method*
  - *returnType() Method* ini akan mengembalikan sebuah *return type* dari sebuah *method*
  - *flatSignature() Method* ini akan mengembalikan *signature* dari sebuah *method*. Jika terdapat *Method* dengan parameter (String x, int y), maka akan mengembalikan (String, int)
- *ParamTag* sebuah *interface* yang menyatakan informasi dari sebuah *Tag* parameter. *Method-method* yang digunakan adalah sebagai berikut
  - *name() Method* ini akan mengembalikan sebuah *tag @param*
  - *parameterName() Method* ini akan mengembalikan sebuah nama parameter dari sebuah *method*
  - *parameterComment() Method* ini akan mengembalikan sebuah deskripsi dari parameter yang terdapat pada *method*

DAFTAR REFERENSI

[1] Oracle (1993) javadoc - the java api documentation generator. http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/javadoc.html#public. 27 September 2017.

[2] Oracle (1993) Javadoc doclet api. http://docs.oracle.com/javase/7/docs/jdk/api/javadoc/doclet/index.html. 5 oktober 2017.

LAMPIRAN A

KODE PROGRAM

Listing A.1: MyCode.c

```c

// This does not make algorithmic sense,
// but it shows off significant programming characters.

#include<stdio.h>

void myFunction( int input, float* output ) {
    switch ( array[i] ) {
        case 1: // This is silly code
            if ( a >= 0 || b <= 3 && c != x )
                *output += 0.005 + 20050;
            char = 'g';
            b = 2^n + ~right_size - leftSize * MAX_SIZE;
            c = (--aaa + &daa) / (bbb++ - ccc % 2 );
            strcpy(a,"hello $@?");
    }
    count = ~mask | 0x00FF00AA;
}

// Fonts for Displaying Program Code in LATEX
// Adrian P. Robson, nepsweb.co.uk
// 8 October 2012
// http://nepsweb.co.uk/docs/progfonts.pdf
```

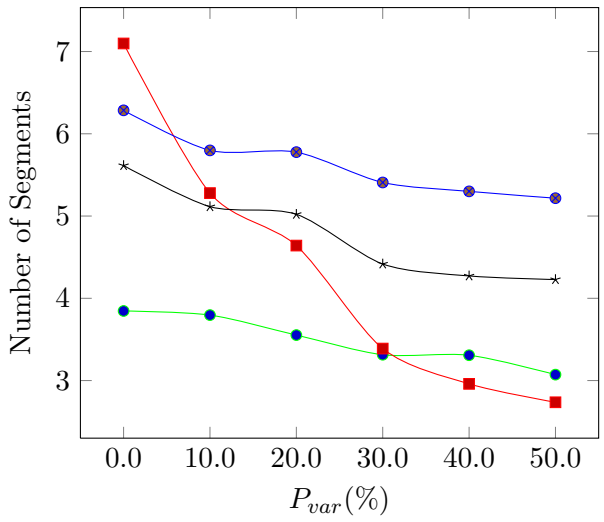
Listing A.2: MyCode.java

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.HashSet;

//class for set of vertices close to furthest edge
public class MyFurSet {
    protected int id;                                   //id of the set
    protected MyEdge FurthestEdge;                      //the furthest edge
    protected HashSet<MyVertex> set;                    //set of vertices close to furthest edge
    protected ArrayList<ArrayList<Integer>> ordered;    //list of all vertices in the set for each trajectory
    protected ArrayList<Integer> closeID;               //store the ID of all vertices
    protected ArrayList<Double> closeDist;              //store the distance of all vertices
    protected int totaltrj;                             //total trajectories in the set

    /*
     * Constructor
     * @param id : id of the set
     * @param totaltrj : total number of trajectories in the set
     * @param FurthestEdge : the furthest edge
     */
    public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
        this.id = id;
        this.totaltrj = totaltrj;
        this.FurthestEdge = FurthestEdge;
        set = new HashSet<MyVertex>();
        ordered = new ArrayList<ArrayList<Integer>>();
        for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
        closeID = new ArrayList<Integer>(totaltrj);
        closeDist = new ArrayList<Double>(totaltrj);
        for (int i = 0;i <totaltrj;i++) {
            closeID.add(-1);
            closeDist.add(Double.MAX_VALUE);
        }
    }

}
```

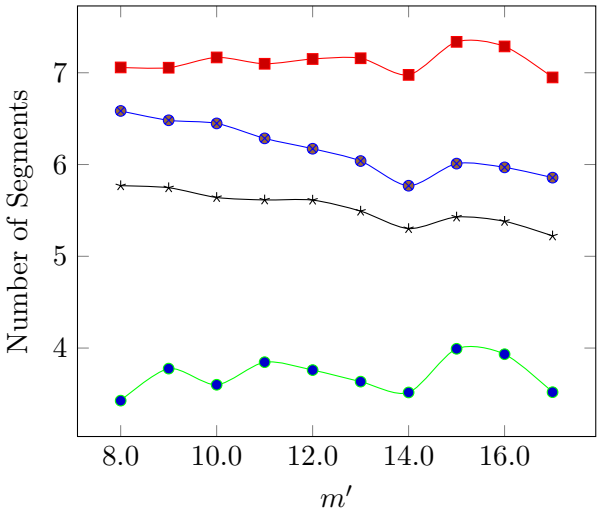
LAMPIRAN B

HASIL EKSPERIMEN

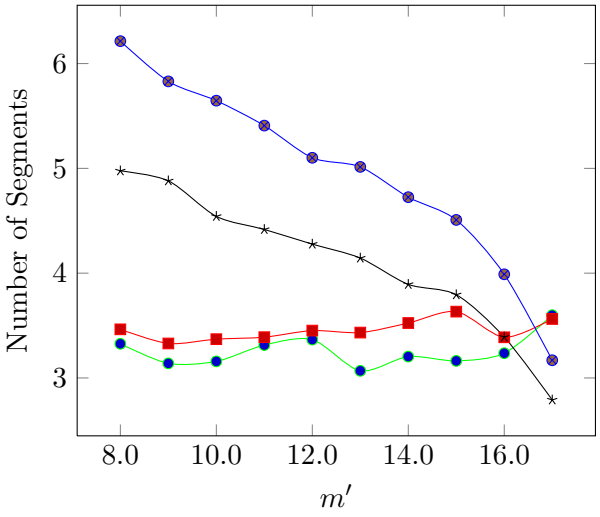
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



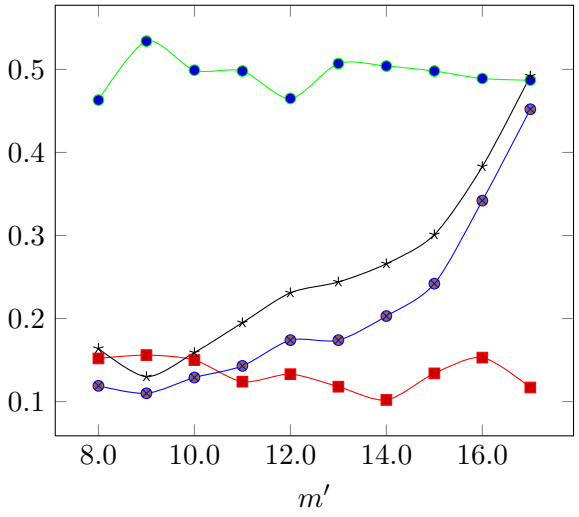
Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4