

SKRIPSI

KONVERSI JAVADOC KE  $\text{\LaTeX}$



Adli Fariz Bonaputra

NPM: 2012730082

PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS  
UNIVERSITAS KATOLIK PARAHYANGAN  
2018



UNDERGRADUATE THESIS

JAVADOC TO L<sup>A</sup>T<sub>E</sub>X CONVERSION



Adli Fariz Bonaputra

NPM: 2012730082

DEPARTMENT OF INFORMATICS  
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES  
PARAHYANGAN CATHOLIC UNIVERSITY  
2018



## ABSTRAK

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 kelas, *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  $\text{\LaTeX}$  dalam pembuatan sebuah dokumentasi Skripsi.  $\text{\LaTeX}$  merupakan bahasa *markup* untuk menyusun sebuah dokumentasi.  $\text{\LaTeX}$  membuat apa yang ditampilkan sama seperti apa yang ditulis. Umumnya bentuk akhir dari dokumen yang dibuat oleh  $\text{\LaTeX}$  biasanya berupa sebuah *file* PDF.

Pada salah satu bab dokumentasi Skripsi, terdapat penjelasan dari setiap kelas 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  $\text{\LaTeX}$ , sehingga membutuhkan lebih banyak waktu untuk mendokumentasikan setiap kelas pada perangkat lunak yang dibuat.

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

**Kata-kata kunci:** Skripsi, *Javadoc*, *Doclet*,  $\text{\LaTeX}$



## ABSTRACT

Student Thesis The Informatics Engineering and Science Faculty (FTIS) Informatics Engineering Study Program at Parahyangan Catholic University (Unpar) is making software. Software made generally uses the programming language Java. As explained, the programming language Java has Javadoc as information from the class, interface, method and also custom tag created, so that the information can be used as an explanation of the software in the software documentation. To document the software created, all students are required to use  $\text{\LaTeX}$  in making a Thesis documentation.  $\text{\LaTeX}$  is the language markup to compile a documentation.  $\text{\LaTeX}$  makes what is displayed the same as what was written. Generally the final form of a document created by  $\text{\LaTeX}$  is usually a file PDF.

In one of the Thesis documentation chapters, there is an explanation of each class in the software made. This explanation can actually be taken from Javadoc which was created in the Java class, but currently based on these observations it is still manually typed from Javadoc into the  $\text{\LaTeX}$  format, so that it takes more time to document each class in the software created.

Therefore, it is necessary to develop a software that can extract information on Javadoc to  $\text{\LaTeX}$  format automatically. This software implements a Application Programming Interface (API) which is used to retrieve information in the form of a class name, interface, method and also custom tag contained in a Java file.

**Keywords:** Thesis, Javadoc, Doclet,  $\text{\LaTeX}$





# DAFTAR ISI

DAFTAR ISI	ix
DAFTAR GAMBAR	xi
DAFTAR TABEL	xiii
<b>1 PENDAHULUAN</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	2
1.3 Tujuan . . . . .	2
1.4 Batasan Masalah . . . . .	2
1.5 Metodologi . . . . .	2
1.6 Sistematika Pembahasan . . . . .	3
<b>2 LANDASAN TEORI</b>	<b>5</b>
2.1 Javadoc . . . . .	5
2.1.1 <i>Processing of source files</i> . . . . .	5
2.1.2 Terminologi . . . . .	6
2.1.3 <i>Source Files</i> . . . . .	6
2.1.4 <i>Generated Files</i> . . . . .	8
2.2 Doclet . . . . .	10
2.2.1 <i>Interface-interface</i> pada Doclet . . . . .	10
2.2.2 Penggunaan Doclet . . . . .	11
2.3 L <sup>A</sup> T <sub>E</sub> X . . . . .	12
<b>3 ANALISIS</b>	<b>15</b>
3.1 Analisis Kebutuhan Perangkat Lunak . . . . .	15
3.2 Analisis Program Sejenis TeXDoclet . . . . .	18
<b>4 PERANCANGAN</b>	<b>19</b>
4.1 Rancangan Kelas Lengkap . . . . .	19
4.2 Rancangan Antarmuka . . . . .	22
<b>5 IMPLEMENTASI DAN PENGUJIAN</b>	<b>25</b>
5.1 Implementasi Perangkat Lunak . . . . .	25
5.2 Pengujian Perangkat Lunak . . . . .	26
5.2.1 Lingkungan Pengujian . . . . .	26
5.2.2 Pengujian Fungsional . . . . .	26
5.2.3 Pengujian Eksperimental . . . . .	27
<b>6 KESIMPULAN DAN SARAN</b>	<b>29</b>
6.1 Kesimpulan . . . . .	29
6.2 Saran . . . . .	29

<b>DAFTAR REFERENSI</b>	<b>31</b>
<b>A PENGUJIAN TERHADAP KODE PROGRAM SEDERHANA</b>	<b>33</b>
A.1 Kode Program . . . . .	33
A.2 Hasil Latex . . . . .	34
A.3 Hasil PDF . . . . .	36
<b>B PENGUJIAN TERHADAP KODE PROGRAM PERANGKAT LUNAK</b>	<b>39</b>
B.1 Kode Program . . . . .	39
B.2 Hasil Latex . . . . .	43
B.3 Hasil PDF . . . . .	45
<b>C PENGUJIAN TERHADAP KODE PROGRAM SIAMODELS</b>	<b>51</b>
C.1 Kode Program . . . . .	51
C.2 Hasil Latex . . . . .	90
C.3 Hasil PDF . . . . .	117
<b>D HASIL PDF TEXDOCLET</b>	<b>159</b>

## DAFTAR GAMBAR

4.1	Kelas Diagram . . . . .	19
A.1	operasimatematika.pdf . . . . .	38
B.1	javadocolatex.pdf . . . . .	49
C.1	javadocolatex.pdf . . . . .	158
D.1	TeXDoclet_report.pdf . . . . .	184



## 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 mengambil informasi dari sebuah *file Java* menjadi sebuah dokumentasi. Umumnya digunakan untuk mendokumentasikan sebuah kelas, *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 kelas 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 kelas, *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  $\text{\LaTeX}$  dalam pembuatan sebuah dokumentasi Skripsi.  $\text{\LaTeX}$  merupakan bahasa *markup* untuk menyusun sebuah dokumentasi.  $\text{\LaTeX}$  membuat apa yang ditampilkan sama seperti apa yang ditulis. Umumnya bentuk akhir dari dokumen yang dibuat oleh  $\text{\LaTeX}$  biasanya berupa sebuah *file PDF*.

Pada salah satu bab dokumentasi Skripsi, terdapat penjelasan dari setiap kelas 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  $\text{\LaTeX}$ , sehingga membutuhkan lebih banyak waktu untuk mendokumentasikan setiap kelas pada perangkat lunak yang dibuat.

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

## 1.2 Rumusan Masalah

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

- Bagaimana membuat perangkat lunak yang dapat mengambil informasi dari sekumpulan *file Java* yang terdapat pada sebuah *package* dan mengonversikan format *Javadoc* ke dalam format  $\text{\LaTeX}$  secara otomatis?

## 1.3 Tujuan

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

- Membuat perangkat lunak yang dapat mengambil informasi dari sekumpulan *file Java* yang terdapat pada sebuah *package* dan mengonversikan format *Javadoc* ke dalam format  $\text{\LaTeX}$  secara otomatis.

## 1.4 Batasan Masalah

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

1. Perangkat lunak memanfaatkan *library Doclet* yang sudah disediakan.
2. Perangkat lunak menghasilkan *file \LaTeX* tanpa mengonversikan menjadi *file PDF*.

## 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  $\text{\LaTeX}$  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.
3. Mengimplementasikan langkah-langkah untuk mengonversi *Javadoc* ke format  $\text{\LaTeX}$ .
4. Melakukan pengujian terhadap perangkat lunak yang telah diimplementasi.
5. Menarik kesimpulan berdasarkan hasil pengujian.



## 1.6 Sistematika Pembahasan

### 1. Bab 1 Pendahuluan

Bab ini akan membahas mengenai latar belakang, rumusan masalah, tujuan, batas masalah, metodologi penelitian dan sistematika penulisan.

### 2. Bab 2 Landasan Teori

Bab ini akan membahas mengenai pengertian *Javadoc*, Doclet dan  $\text{\LaTeX}$ .

### 3. Bab 3 Analisis

Bab ini akan membahas mengenai analisis struktur  $\text{\LaTeX}$  dan analisis program sejenis TeXDoclet.

### 4. Bab 4 Perancangan

Bab ini akan membahas mengenai tahap-tahap perancangan dan penjelasan perangkat lunak.

### 5. Bab 5 Implementasi dan Pengujian

Bab ini akan membahas mengenai implementasi kode program dan pengujian perangkat lunak.

### 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*, *Doclet* dan  $\text{\LaTeX}$ .

#### 2.1 Javadoc

*Javadoc* adalah sebuah *tools* yang dimiliki oleh *Java* yang berguna untuk mengambil informasi dari sekumpulan *source file Java* menjadi sebuah dokumentasi. Umumnya *Javadoc* menghasilkan sekumpulan *file HTML* yang mendeskripsikan sebuah kelas, *interface*, *method* dan *custom tag*. *Javadoc* dapat mengambil 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"*. *Javadoc* dapat mengambil informasi dari lebih dari 1 *file Java* atau sebuah *package*.

*Javadoc* secara otomatis akan menambahkan sebuah tautan yang mengarahkan ke sebuah *package*, kelas dan anggota dari kelas tersebut yang akan didokumentasikan pada saat *Javadoc* memprosesnya. Tautan tersebut berada pada beberapa posisi seperti:

1. *Declaration (return types, argument types, field types)*.
2. *"See Also"* yang dihasilkan oleh tag *@see*.
3. *In-line text* yang dihasilkan oleh tag *@link*.
4. *Exception* yang dihasilkan oleh tag *@throws*.
5. *Link "Specified by"* untuk *member* dari sebuah *interface*.
6. *Link "Override"* untuk *member* dari sebuah kelas.
7. Ringkasan daftar tabel *package*, kelas dan seluruh anggota dari kelas.
8. Turunan dari setiap *package* dan kelas.
9. Indeks.

Dalam mengambil 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 diinginkan, dapat menggunakan sebuah *doclet* yang disediakan oleh *Javadoc*.

### 2.1.2 Terminologi

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 standar *doclet*.

- *Name*

Nama dari elemen program yang dideklarasikan atau dilakukan pemanggilan maka dapat berupa informasi lengkapnya seperti `java.lang.String` dan `java.lang.String.equals(java.lang.Object)` atau informasi pendeknya seperti `String` dan `equals(Object)`. Nama-nama tersebut dapat berupa nama *package*, kelas, *interface*, *field*, *constructor* atau *method*.

- *Documented Classes*

Detail dari sebuah kelas 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*

Kelas dan *Interface* akan didokumentasikan pada saat *Javadoc* berjalan, hal ini sama seperti *Documented Classes*.

- *Excluded Classes*

Kelas dan *Interface* tidak akan didokumentasikan pada saat *Javadoc* berjalan.

- *Referenced Classes*

Kelas dan *Interface* yang secara eksplisit disebutkan oleh kelas dan *interface* lainnya, seperti *return type*, *parameter type*, *cast type*, *extended class*, *implemented interface*, *imported class*, kelas yang digunakan pada *method body*, `@see`, `@link`, `@linkplain` dan `@inheritDoc` tag.

### 2.1.3 Source Files

*Javadoc* akan menghasilkan *output* yang berasal dari 4 tipe *file*, yaitu sebagai berikut:

- *Class Source Code Files*

Setiap kelas atau *interface* memiliki dokumentasinya masing-masing.

- *Package Comment Files*

Setiap *package* dapat memiliki dokumentasinya masing-masing yang terdapat pada *root folder package*. Contohnya `java/applet/package.html`, kemudian *Javadoc* akan menggabungkan dokumentasi tersebut menjadi sebuah ringkasan dari *package* yang ada. Terdapat 2 cara

penulisan yaitu dengan sebuah *file* `package.html` pada listing 2.1 atau sebuah *file* `package-info.java` pada listing 2.2.

```
<html>
<body>
  Provides the classes necessary to create an applet and the classes
  an applet uses to communicate with its applet context.

  @since 1.0
  @see java.awt
</body>
</html>
```

Listing 2.1: *File package.html*

```
/**
 * Provides the classes necessary to create an applet
 * and the classes an applet uses to communicate
 * with its applet context.
 *
 * @since 1.0
 * @see java.awt
 */
package java.lang.applet;
```

Listing 2.2: *File package-info.java*

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* yang ada.
3. Memasukkan teks yang sudah diproses tersebut pada bagian bawah halaman dokumentasi yang dihasilkan.
4. Menyalin 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* yang diletakkan pada bagian paling atas sebuah *source code* yang dibuat. *Javadoc* akan melakukan beberapa langkah yaitu sebagai berikut:

1. Menyalin informasi untuk diproses.
2. Memproses semua *tag* yang ada.
3. Memasukan teks yang sudah diproses tersebut pada bagian bawah halaman dokumentasi yang dihasilkan.

4. Menyalin kalimat pertama pada *package* tersebut pada bagian atas halaman dokumentasi.

- *Miscellaneous Unprocessed Files*

*Javadoc* dapat memproses berbagai macam format *file*. *File* tersebut dapat berupa sebuah *graphic files*, *Java source (.java)* dan *class (.class)* dan sebuah *file* HTML. Untuk memproses *file* tersebut, perlu membuat sebuah folder bernama *doc-files* yang diletakkan pada *package* tertentu lalu *file-file* tersebut dapat diakses dengan menggunakan lokasi direktori yang menyimpan *file* tersebut.

#### 2.1.4 *Generated Files*

*Javadoc* memiliki sebuah *doclet* bawaan yang disebut dengan standar *doclet*. Standar *doclet* akan menghasilkan dokumentasi dengan format HTML. Terdapat 3 grup yang masing-masing memiliki kriterianya sendiri, ketiga grup tersebut adalah sebagai berikut:

- *Basic Content Pages*

- Sebuah halaman kelas atau *interface* (*classname.html*) untuk masing-masing kelas 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 untuk keseluruhan *package* (*overview-tree.html*).
- Sebuah halaman hirarki untuk setiap *package* (*package-tree.html*).
- Sebuah halaman *"use"* (*package-use.html*) yang berisikan *package*, *classes*, *methods*, *constructors* atau *interface*. Jika diberikan sebuah kelas bernama A, maka 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 kelas atau *interface* 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*.
- Seberapa *frame file* (*\*-frame.html*) yang berisi sekumpulan *packages*, kelas 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 halaman 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 kelas pada *package* tersebut. Jika terdapat lebih dari 2 *package*, maka *Javadoc* akan membuat 3 *frame* dari sekumpulan *package*. Jika kelas 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 sentences 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

---



---

1	Applet.html	Page for Applet source code
2	AppletContext.html	Page for AppletContext source code
3	AppletStub.html	Page for AppletStub source code
4	AudioClip.html	Page for AudioClip source code

Listing 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 mengambil informasi secara spesifik [2].

### 2.2.1 *Interface-interface* 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 diambil. *Method-method* yang digunakan adalah sebagai berikut:

- **classes()**

*Method* ini akan mengembalikan sejumlah kelas dan *interface* pada *package*.

- **ClassDoc**

sebuah *interface* yang menyatakan informasi dari sebuah kelas. Informasi tersebut dapat berupa nama kelas, nama *method* dan *tag*. *Method-method* yang digunakan adalah sebagai berikut:

- **name()**

*Method* ini akan mengembalikan sebuah nama kelas atau *interface* pada *package*.

- **commentText()**

*Method* ini akan mengembalikan sebuah informasi dari deskripsi kelas.

- **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*.



1       – `returnType()`

2       *Method* ini akan mengembalikan sebuah *return type* dari sebuah *method*.

3       – `flatSignature()`

4       *Method* ini akan mengembalikan *signature* dari sebuah *method*. Jika terdapat *Method*  
5       dengan parameter (String x, int y), maka akan mengembalikan (String, int).

6       • **ParamTag**

7       sebuah *interface* yang menyatakan informasi dari sebuah *Tag* parameter. *Method-method* yang  
8       digunakan adalah sebagai berikut:

9       – `name()`

10       *Method* ini akan mengembalikan sebuah *tag @param*.

11       – `parameterName()`

12       *Method* ini akan mengembalikan sebuah nama parameter dari sebuah *method*.

13       – `parameterComment()`

14       *Method* ini akan mengembalikan sebuah deskripsi dari parameter yang terdapat pada  
15       *method*.

16   **2.2.2 Penggunaan Doclet**

17   Doclet dapat menghasilkan sebuah *output Javadoc* yang dapat disesuaikan. Penggunaan *Doclet* API  
18   dapat mengambil bermacam-macam informasi seperti nama kelas, nama *method*, deskripsi singkat  
19   untuk sebuah parameter dari sebuah *method* hingga *return type* dari *method*.

20   Berikut adalah langkah-langkah dasar untuk membuat dan menggunakan *doclet*:

- 21   1. Membuat sebuah kelas *Java* yang akan menjadi *doclet*. Kelas *Java* tersebut harus ditambahkan  
22    *package com.sun.javadoc.\** untuk menggunakan *doclet* API.
- 23   2. Kelas tersebut harus diawali dengan sebuah *method* `public static boolean start` yang  
24    memiliki parameter `RootDoc`.
- 25   3. *Compile* doclet tersebut dengan menggunakan *compiler Java* yaitu `javac` yang dapat dilakukan  
26    melalui *Terminal* pada Linux/Mac atau *Command Prompt* pada Windows.
- 27   4. Jalankan perintah *Javadoc* menggunakan *option -doclet <class>* untuk menghasilkan *output*  
28    yang telah disesuaikan, dimana `<class>` adalah kelas *doclet* yang sudah di-*compile* pada  
29    langkah ketiga. Jika tidak menggunakan *option -doclet* maka *Javadoc* akan menghasilkan  
30    *output* standar yaitu berupa *file HTML*.

31   Kelas-kelas yang dimiliki oleh *doclet* API terdapat pada direktori `lib\tools.jar`. *Doclet* yang  
32   sudah dibuat harus di-*compile* menggunakan *file tools.jar* dan menambahkan *option -classpath*  
33   setelah *command javac*.

34   *Package com.sun.javadoc.\** terdiri dari *interface* yang mendefinisikan *doclet* API. `tools.jar`  
35   berisikan *interface* tersebut.

```

1  import com.sun.javadoc.*;
2
3  public class ListClass {
4      public static boolean start(RootDoc doc) {
5          ClassDoc[] classes = doc.classes();
6          for(int i=0, i < classes.length; i++) {
7              System.out.println(classes[i]);
8          }
9          return true;
10     }
11 }

```

Listing 2.4: kelas ListClass.java

Potongan *program* pada Listing 2.4 adalah sebuah *doclet* sederhana untuk menampilkan nama dari kelas yang dijadikan sebagai masukan. *Method* `public static boolean start` memiliki parameter `RootDoc doc` yang akan menampung masukan dari sekumpulan *file Java* yang akan diproses lalu sekumpulan *file* tersebut akan ditampung pada `ClassDoc[] classes` sebagai *array* kemudian *array of classes* akan dijalankan sebanyak panjang *array* tersebut.

## 2.3 L<sup>A</sup>T<sub>E</sub>X

L<sup>A</sup>T<sub>E</sub>X adalah sebuah bahasa *markup* untuk sistem penulisan dokumen yang dikembangkan oleh Leslie B. Lamport dan dirilis pada tahun 1985 [3]. L<sup>A</sup>T<sub>E</sub>X Memiliki filosofi WYMIWYG (*What you Mean Is What You Get*) yang berarti sesuatu yang ditulis akan berdasarkan arti dari hal tersebut. Oleh karena itu, untuk menambahkan suatu perintah pada dokumen yang sedang ditulis perlu menambahkan suatu *command*. *Command* adalah kata spesial yang menentukan suatu sifat pada L<sup>A</sup>T<sub>E</sub>X. Hampir semua *command* pada L<sup>A</sup>T<sub>E</sub>X selalu diawali dengan tanda '\' dan beberapa *command* memiliki *parameter*. *Parameter* diawali dengan tanda kurung kurawal buka dan diakhiri dengan kurung kurawal tutup ({...}). File L<sup>A</sup>T<sub>E</sub>X memiliki ekstensi .tex. Pada saat membuat sebuah *project* L<sup>A</sup>T<sub>E</sub>X hanya perlu menuliskan *command* `\documentclass[option]{class}` 1 kali.

Untuk menulis dokumen pada L<sup>A</sup>T<sub>E</sub>X dibutuhkan beberapa *command* yang wajib ada dalam sebuah dokumen, yaitu:

1. `\documentclass[option]{class}`  
Digunakan untuk menentukan jenis dokumen yang *layout* dokumen. Bagian *option* dapat dikosongkan atau dapat digunakan untuk menyimpan pilihan pengaturan *layouting*. Pada Bagian kelas digunakan untuk menentukan tipe dokumen yang akan dibuat. *Command* ini hanya perlu ditulis 1 kali dalam sebuah dokumen.
2. `\maketitle`  
Digunakan untuk menampilkan halaman judul. Biasanya halaman judul akan memuat judul dokumen, nama pengarang dan tanggal pembuatan dokumen. Judul dokumen, nama pengarang dan tanggal pembuatan dapat ditampilkan dengan menambahkan perintah `\title{judul}`, `\author{nama}` dan `\date{tanggal}`.
3. `\begin{document}...\end{document}`  
Digunakan untuk mengawali dan mengakhiri sebuah dokumen.

- 
- 1    4. `\section{section}`  
2        Digunakan untuk menampilkan subbab sebuah dokumen.
  - 3    5. `\texttt{text}`  
4        Digunakan untuk menampilkan tulisan *monospaced*.
  - 5    6. `\begin{enumerate}...\end{enumerate}`  
6        Digunakan untuk menampilkan *ordered list*. *List* ini akan menampilkan angka yang terurut.  
7        Di dalam *list* ini terdapat *command* `\item` untuk menambahkan isi dari *list* tersebut.
  - 8    7. `\begin{itemize}...\end{itemize}`  
9        Digunakan untuk menampilkan *unordered list*. *List* ini akan menampilkan simbol spesial. Di  
10       dalam *list* ini terdapat *command* `\item` untuk menambahkan isi dari *list* tersebut.



## BAB 3

## ANALISIS

Bab ini membahas mengenai analisis kebutuhan perangkat lunak dan analisis program sejenis.

### 3.1 Analisis Kebutuhan Perangkat Lunak

Struktur  $\text{\LaTeX}$  yang digunakan memiliki format sebagai berikut.

```
\begin{enumerate}
\item \texttt{namaKelas}\\
{penjelasan kelas}

Atribut yang dimiliki kelas ini adalah sebagai berikut.
\begin{itemize}
\item \texttt{atribut} –
{penjelasan tentang atribut}.
\end{itemize}

\textit{Method} yang terdapat pada kelas namaKelas adalah sebagai berikut.
\begin{itemize}
\item \texttt{method}\\
{penjelasan method}

\textbf{Parameter:}
\begin{itemize}
\item \texttt{parameter} –
{penjelasan dari parameter}.
\end{itemize}

\textbf{Return Value:} {penjelasan return-type method}\\
\textbf{Exception:} {penjelasan exception jika terdapat exception}
\textbf{See Also:} {penjelasan tag @see jika terdapat tag tersebut}
\textbf{Override:} {penjelasan apabila jika terdapat {\it override method} }
\end{itemize}
\end{enumerate}
```

Listing 3.1: Potongan kode  $\text{\LaTeX}$

Potongan kode yang terdapat pada Listing 3.1 adalah bagian isi dari sebuah dokumen  $\text{\LaTeX}$ . Bagian ini nantinya akan dipindahkan ke salah satu bab pada skripsi yang akan menjelaskan perangkat lunak secara rinci. Bagian ini akan dijelaskan sebagai berikut.

### 1. *List* indentasi tingkat pertama

Pada *list* tingkat pertama ini menampilkan sebuah nama kelas dan penjelasan terkait dengan kelas tersebut. *List* yang dibuat menggunakan *ordered list* dengan *command* `\begin{enumerate}...`  
`\end{enumerate}` dan *command* `\texttt{namaKelas}` akan digunakan untuk menampilkan nama kelas.

### 2. *List* indentasi tingkat kedua

Pada *list* tingkat kedua ini terdapat dua *list* yang masing-masing menampilkan atribut dan *method* yang dimiliki oleh kelas tersebut. *List* pertama yang dibuat menggunakan *unordered list* dengan *command* `\begin{itemize}...\end{itemize}` untuk mengisi atribut-atribut yang terdapat pada kelas ini jika kelas ini tidak memiliki atribut maka menampilkan tulisan tidak memiliki atribut. *Command* `\texttt{atribut}` digunakan untuk menampilkan atribut. Atribut ini menampilkan tipe atribut dan nama atribut. *List* kedua menggunakan *unordered list* dengan *command* `\begin{itemize}...\end{itemize}` untuk mengisi *method-method* yang terdapat pada kelas ini dan penjelasan terkait dengan *method* tersebut. *Command* `\texttt{method}` digunakan untuk menampilkan *method*. *Method* ini menampilkan *access modifier* dari *method*, tipe kembalian *method*, nama *method* dan daftar nama parameter.

### 3. *List* indentasi tingkat ketiga

Pada *list* tingkat ketiga ini menampilkan parameter yang digunakan pada *method* dan penjelasan terkait dengan parameter tersebut. *List* yang dibuat menggunakan *unordered list* dengan *command* `\begin{itemize}...\end{itemize}` jika *method* tidak memiliki parameter maka menampilkan tulisan tidak memiliki parameter dan *command* `\texttt{parameter}` akan digunakan untuk menampilkan parameter. Parameter ini menampilkan tipe parameter dan nama parameter.

### 4. *Return Value & Exception*

*Return value* yang terdapat dalam *method* tersebut akan ditampilkan setelah *list level* ketiga jika tipe *return value* adalah *void* maka akan menampilkan tulisan tidak memiliki *return value*. *Exception* maka ditampilkan setelah *Return value* jika *method* tidak terdapat *exception* maka akan menampilkan tulisan tidak memiliki *exception*.

### 5. *Optional Tags*

*Optional tags* akan menampilkan informasi dari sebuah *tag @see* ataupun *tag {@link}*. Jika tidak ada informasi dari *tag - tag* tersebut akan menampilkan tulisan tidak ada.

### 6. *Override*

*Override* akan menampilkan informasi apakah *method* dari sebuah *superclass* ditulis kembali di sebuah *subclass*. jika tidak ada informasi tersebut maka bagian penjelasan akan dihilangkan.

Perangkat lunak yang dibuat akan menerima sebuah masukan berupa sekumpulan *file Java* yang berada di dalam sebuah *package*. Struktur kode *Java* yang digunakan dapat dilihat pada Lampiran A. Struktur kode akan dijelaskan sebagai berikut.

#### 1. Setiap *file Java* harus terletak di dalam sebuah *package* yang sama.

2. Setiap deklarasi kelas harus diawali dengan huruf kapital serta memiliki *Javadoc* untuk penjelasan tentang kelas tersebut dan secara opsional dapat menambahkan *tag* - *tag javadoc* seperti *tag @see* sebagai penunjuk ke sebuah referensi dan *tag {@link}* sebagai penunjuk ke dokumentasi sebuah *package*, *class* ataupun *method* yang dimiliki oleh kelas lain.
3. Setiap deklarasi atribut harus memiliki *access modifier*, tipe atribut dan nama atribut serta memiliki *Javadoc* untuk penjelasan tentang atribut tersebut.
4. Setiap deklarasi *method* harus memiliki *access modifier*, tipe kembalian, nama *method*, tipe dan variabel parameter serta memiliki *Javadoc* untuk penjelasan *method*, parameter yang digunakan dan hasil kembalian sebuah *method*.

Hasil dari sebuah perangkat lunak yang dibuat adalah sebuah *file* berformat  $\text{\LaTeX}$ . Perangkat lunak akan membaca satu persatu *file Java* dan informasi yang terdapat pada setiap *file Java* tersebut dimasukan ke dalam *file \LaTeX*.

```

\begin{enumerate}
  \item \texttt{Pertambahan}\\
  Kelas ini merupakan Kelas Pertambahan.

  Atribut yang dimiliki kelas ini adalah sebagai berikut.
  \begin{itemize}
    \item \texttt{int a} -
      Atribut A.
    \item \texttt{int b} -
      Atribut B.
  \end{itemize}

  \textit{Method} yang terdapat pada kelas Pertambahan adalah sebagai berikut.
  \begin{itemize}
    \item \texttt{public int pertambahan(int a, int b)}\\
    Method Pertambahan.

    \textbf{Parameter:}
    \begin{itemize}
      \item \texttt{int a} -
        Bilangan Pertama.
      \item \texttt{int b} -
        Bilangan Kedua.
    \end{itemize}

    \textbf{Return Value:} hasil penjumlahan 2 buah bilangan.\\
    \textbf{Exception:} tidak memiliki \textit{exception}.
    \textbf{Override:} \texttt{pertambahan} dari kelas \texttt{operasiMatematikaInterface}
  \end{itemize}
\end{enumerate}

```

Listing 3.2: Contoh hasil konversi *Javadoc* ke  $\text{\LaTeX}$

Hasil konversi pada Listing 3.2 akan menampilkan nama kelas serta penjelasan kelas tersebut, atribut yang digunakan serta penjelasan untuk setiap atributnya, *method* yang digunakan serta

- 1 penjelasan *method*, parameter yang digunakan serta penjelasan setiap parameternya, *return value*
- 2 dan *exception*.

## 3.2 Analisis Program Sejenis TeXDoclet

TeXDoclet merupakan sebuah program yang mengimplementasi *Doclet* yang dimiliki oleh *Java*. Program ini akan mengonversi sekumpulan *file Java* yang terletak di dalam satu *package* yang sama. TeXDoclet dapat menghasilkan dokumen berupa *file L<sup>A</sup>T<sub>E</sub>X* atau *file PDF*. Untuk dapat menghasilkan *file PDF*, TeXDoclet mengintegrasikan *LuaL<sup>A</sup>T<sub>E</sub>X* untuk menghasilkan dokumen PDF dari sebuah *file L<sup>A</sup>T<sub>E</sub>X*. Hasil PDF yang dihasilkan oleh TeXDoclet dapat dilihat pada Lampiran D.

TeXDoclet memiliki beberapa *option* yang dapat digunakan, akan dijelaskan sebagai berikut.

### 1. -sectionlevel <level>

Untuk menentukan *level* teratas dari *section* sebuah dokumen. *Section* tersebut bisa berupa *chapter*, *section* atau *subsection*.

### 2. -createPdf

Untuk menghasilkan *file PDF* dari sebuah hasil *file L<sup>A</sup>T<sub>E</sub>X* dengan menggunakan *LuaL<sup>A</sup>T<sub>E</sub>X*.

### 3. -twosided

Untuk menghasilkan dokumen 2 sisi. Jika dokumen tersebut menggunakan *option* ini maka dokumen tersebut pada saat dicetak akan memiliki 2 sisi yaitu depan dan belakang.

### 4. -texinit <file>

Untuk menambahkan *command-command* yang lain sebelum *command \begin{document}*.

### 5. -docclass <class>

Untuk menentukan tipe dokumen yang akan dibuat. *Default* untuk *option* adalah tipe dokumen *report*.

### 6. -title <title>

Untuk menentukan judul dari dokumentasi yang dibuat.

### 7. -output <outfile>

Menentukan nama *file* yang akan dihasilkan. Jika perintah ini tidak digunakan maka secara otomatis akan menghasilkan file *doc.tex*.

### 8. -date <date string>

Menentukan tanggal dari dokumen.

### 9. -author <author>

Menentukan penulis dari dokumen.



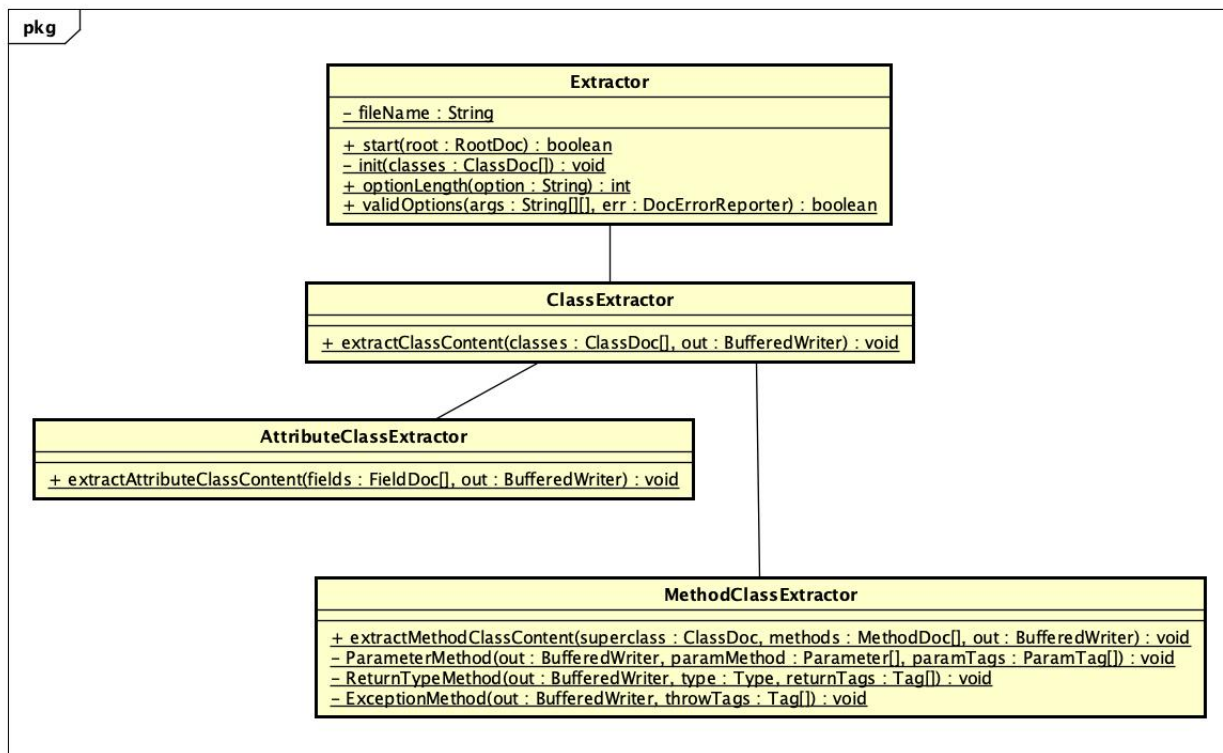
## BAB 4

## PERANCANGAN

Bab ini membahas mengenai perancangan aplikasi yang akan dibangun meliputi diagram kelas rinci beserta deskripsi dan fungsinya.

### 4.1 Rancangan Kelas Lengkap

Rancangan kelas dibawah ini akan menampilkan keseluruhan kelas yang akan digunakan. Deskripsi kelas beserta fungsi dari diagram kelas tersebut adalah sebagai berikut:



Gambar 4.1: Kelas Diagram

#### 1. Extractor

Kelas ini merupakan kelas untuk menjalankan *custom doclet*

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- **String fileName** - atribut untuk nama *file*

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static boolean start(com.sun.javadoc.RootDoc root)`

*Method* ini berperan sebagai *method* untuk menjalankan *custom doclet*

**Parameter:**

- `RootDoc root` - berperan sebagai mengambil seluruh informasi spesifik dari *option* yang terdapat pada *command-line* sebuah *terminal*. Selain itu berperan juga untuk mengambil informasi dari sekumpulan *file java* yang akan di proses.

**Return Value:** kondisi true

**Exception:** Tidak memiliki *exception*

- `private static void init(com.sun.javadoc.ClassDoc[] classes)`

*Method* ini berperan untuk menulis kedalam sebuah *file* saat *javadoc* berjalan.

**Parameter:**

- `ClassDoc[] classes` - sebuah array yang berisikan sekumpulan *file java* yang akan di proses.

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public static int optionLength(java.lang.String option)`

*Method* untuk menghitung banyak *option* yang digunakan pada *command-line*

**Parameter:**

- `String option` - sebuah *option*

**Return Value:** panjang setiap *option*

**Exception:** Tidak memiliki *exception*

- `public static boolean validOptions(java.lang.String[][] args, com.sun.javadoc.DocErrorReporter err)`

Pengecekan *option* valid

**Parameter:**

- `String[][] args` - *String* array 2 dimensi dari *option*
- `DocErrorReporter err` - sebuah error jika tidak terdapat *option* tersebut.

**Return Value:** bernilai true jika *option* tersebut dikenali, false jika *option* tersebut tidak dikenali

**Exception:** Tidak memiliki *exception*

## 2. ClassExtractor

Kelas ini merupakan kelas untuk mengambil informasi dari sebuah kelas

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static void extractClassContent(com.sun.javadoc.ClassDoc[] classes, java.io.BufferedWriter out)`

*Method* ini akan menampilkan nama kelas beserta penjelasan dari sebuah kelas

**Parameter:**

- `ClassDoc[] classes` - sebuah array berisikan sejumlah kelas
- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

### 3. MethodClassExtractor

Kelas ini merupakan kelas untuk mengambil informasi sebuah *method* terdapat pada kelas

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static void extractMethodClassContent(com.sun.javadoc.ClassDoc superclass, com.sun.javadoc.MethodDoc[] methods, java.io.BufferedWriter out)`

*Method* ini akan menampilkan *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `ClassDoc superclass` - sebuah objek `ClassDoc`
- `MethodDoc[] methods` - sebuah array berisikan sejumlah *method* dari kelas
- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `private static void ParameterMethod(java.io.BufferedWriter out, com.sun.javadoc.ParameterDoc paramMethod, com.sun.javadoc.ParamTag[] paramTags)`

*Method* ini akan menampilkan parameter *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text
- `Parameter[] paramMethod` - sebuah array berisikan sejumlah *method* dari kelas
- `ParamTag[] paramTags` - sebuah array berisikan sejumlah parameter *method* dari kelas

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `private static void ReturnTypeMethod(java.io.BufferedWriter out, com.sun.javadoc.Type type, com.sun.javadoc.Tag[] returnTags)`

*Method* ini akan menampilkan *return type* dari *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text
- `Type type` - sebuah objek `Type`

– `Tag[] returnTags` - sebuah array berisikan sejumlah *return type* dari *method* dari kelas

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `private static void ExceptionMethod(java.io.BufferedWriter out, com.sun.javadoc.Tag[] throwTags)`

*Method* ini akan menampilkan *return type* dari *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text
- `Tag[] throwTags` - sebuah array berisikan sejumlah *exception* dari *method* dari kelas

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

#### 4. AttributeClassExtractor

Kelas ini merupakan kelas untuk mengambil informasi sebuah atribut yang terdapat pada kelas

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static void extractAttributeClassContent(com.sun.javadoc.FieldDoc[] fields, java.io.BufferedWriter out)`

*Method* ini akan menampilkan atribut-atribut yang dimiliki oleh sebuah kelas

**Parameter:**

- `FieldDoc[] fields` - sebuah array berisikan sejumlah atribut dari kelas
- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

## 4.2 Rancangan Antarmuka

Rancangan antarmuka perangkat lunak adalah melalui sebuah *Terminal* pada *Linux/Mac* atau *Command Prompt* pada *Windows*. Berikut adalah antarmuka jika menggunakan *Terminal* pada *Mac*:

```
Last login: Mon Nov 26 18:49:17 on ttys002
[roth:~ abathz$ cd Documents/KULIAH/Skripsi/
```

Gambar 4.2: Mengarahkan kedalam folder dari perangkat lunak

1 Langkah pertama adalah berpindah dari direktori awal ke direktori perangkat lunak yang dibuat.  
 2 Untuk berpindah direktori perlukan perintah `cd` atau kepanjangan dari *change directory* lalu diikuti  
 3 dengan lokasi direktori yang diinginkan. Pada Gambar 4.2 direktori perangkat lunak terdapat di  
 4 dalam folder Document lalu folder KULIAH lalu folder Skripsi kemudian tekan tombol *enter* lalu  
 5 direktori akan langsung berpindah ke direktori yang dituju.

```
root:Skripsi abathz$ javadoc -filename file -classpath tools.jar -doclet extractor.Extractor -docletpath
GenerateJavadocToLatex.jar -sourcepath GenerateJavadocToLatex/src extractor
```

Gambar 4.3: Memasukkan perintah yang akan digunakan

6 Langkah kedua adalah menjalankan perangkat lunak yang dibuat. Diawali dengan perintah perintah  
 7 `javadoc` lalu diikuti 5 buah argumen. Argumen pertama yaitu `-filename` adalah *option* untuk  
 8 menamai *file* sesuai dengan yang ditentukan. Sebagai contoh pada Gambar 4.3, *file* akan bernama  
 9 "file", jika argumen pertama tidak dimasukkan pada *command-line* maka nama dari *file* tersebut  
 10 secara otomatis menjadi "doc". Argumen kedua yaitu `-classpath` berperan sebagai penunjuk  
 11 kelas-kelas yang digunakan. Argumen kedua ini bersifat *optional*, jika kode program yang akan  
 12 didokumentasikan menggunakan *external library* maka argumen ini dibutuhkan. Argumen ketiga  
 13 yaitu `-doclet` adalah sebuah kelas untuk menjalankan *custom doclet* dari perangkat lunak yang  
 14 dibuat. Argumen ketiga tersebut akan menjalankan kelas bernama `Extractor` yang terdapat  
 15 didalam *package extractor*. Kemudian argumen keempat yaitu `-docletpath` adalah *custom doclet*  
 16 yang berperan untuk mengambil informasi kelas, atribut, *method* dari sekumpulan *file java*. Argumen  
 17 kelima yaitu `-sourcepath` adalah lokasi sekumpulan *file java* yang akan diproses. Pada gambar 4.3,  
 18 lokasi *file-file* tersebut terdapat pada folder GenerateJavadocToLatex lalu folder *src* lalu argumen  
 19 keenam yaitu adalah sebuah *package* yang berisikan sekumpulan *file Java*, pada kasus ini adalah  
 20 *package extractor*.

```
root:Skripsi abathz$ javadoc -filename file -classpath tools.jar -doclet extractor.Extractor -docletpath
GenerateJavadocToLatex.jar -sourcepath GenerateJavadocToLatex/src extractor
Loading source files for package extractor...
Constructing Javadoc information...
root:Skripsi abathz$
```

Gambar 4.4: Hasil tampilan jika proses konversi selesai

21 Perangkat lunak yang dibuat akan membaca seluruh isi *package*. Pada contoh Gambar 4.4,  
 22 perangkat lunak akan melakukan pengambilan informasi terhadap *package* tersebut. Jika proses  
 23 pengambilan selesai maka proses berhenti.



## BAB 5

### IMPLEMENTASI DAN PENGUJIAN

Bab ini terdiri atas dua bagian, yaitu Implementasi Perangkat Lunak dan Pengujian Perangkat Lunak. Bagian implementasi berisi perintah menggunakan perangkat lunak dan langkah-langkah dalam penggunaan perangkat lunak. Sedangkan bagian pengujian berisi hasil pengujian fungsional dan pengujian eksperimental.

#### 5.1 Implementasi Perangkat Lunak

Perangkat lunak dibuat menggunakan bahasa *java* dan dimasukkan ke dalam sebuah *jar* sehingga dapat digunakan dengan cara mengeksekusi perintah. Penggunaan *file jar* tersebut dapat dilakukan melalui *Terminal* di Linux/Mac atau *Command Prompt* di Windows. Berikut perintah yang digunakan untuk menjalankan perangkat lunak.

```
javadoc -filename <file-name>
        -classpath <pathlist>
        -doclet extractor.Extractor
        -docletpath GenerateJavadocToLatex.jar
        -sourcepath <pathlist>
        -subpackages <subpkglist>
        <packagenames>
```

Pada potongan perintah diatas memiliki 2 parameter yaitu *option* dan *packagenames*. Parameter *packagename* adalah parameter untuk *package* yang akan sebagai masukan dari perangkat lunak. Parameter *option* adalah beberapa perintah pendukung. Berikut beberapa perintah *option* yang digunakan untuk mendukung berjalannya perangkat lunak.

- **-filename <file-name>** - Menghasilkan *output file* dengan nama **file-name.tex**. Jika *option* ini tidak digunakan maka nama *file* yang dihasilkan akan bernama **doc.tex**.
- **-classpath <pathlist>** - Letak *file jar* yang dipergunakan. Perintah ini bersifat opsional jika kelas tersebut membutuhkan kelas yang tidak ada di dalam standar *library Java*.
- **-doclet <class>** - Kelas yang dibuat untuk menghasilkan *output*. Pada kasus ini adalah **extractor.Extractor**.
- **-docletpath <path>** - Letak *doclet* yang sudah di-*package* menjadi *file jar*.
- **-sourcepath <pathlist>** - Letak *source file* sebagai masukan.

- `-subpackages <subpkglist>` - Letak *subpackage* yang akan dimuat secara rekursif. Perintah ini dapat digunakan secara opsional jika terdapat sebuah *subpackage* didalam *package* yang menjadi masukan.
- `<packagenames>` - Nama *package* yang menjadi masukan. Lokasi *package* bergantung pada `-sourcepath` yang dituju.

Untuk penggunaan perintah diatas, Langkah pertama membuka aplikasi *Terminal* atau *Command Prompt*. Langkah kedua mengetik perintah `javadoc` lalu diikuti dengan perintah pendukungnya seperti yang sudah dijelaskan diatas. Berikut contoh perintah lengkap yang digunakan.

```
javadoc -filename siamodels
        -doclet extractor.Extractor
        -docletpath GenerateJavadocToLatex.jar
        -sourcepath SIAModels/src/main/java
        -subpackages id.ac.unpar.siamodels
        id.ac.unpar.siamodels
```

## 5.2 Pengujian Perangkat Lunak

Pada sub bab ini akan menjelaskan Lingkungan Pengujian dan Pengujian Fungsional. Pengujian Fungsional akan menguji perangkat lunak terhadap kode program sederhana serta menguji kode program perangkat lunak yang dibuat.

### 5.2.1 Lingkungan Pengujian

Dalam proses pengujian perangkat lunak ini digunakan spesifikasi perangkat sebagai berikut.

1. Processor: Intel Core i7 2.5-3.7GHz
2. RAM: 16.00 GB DDR3
3. Harddisk : 512MB SSD
4. VGA : Intel Iris Pro dan AMD Radeon R9 M370X
5. Sistem Operasi: macOS High Sierra
6. Versi Java: 1.8.0\_121
7. Code Editor: Netbeans 8.2

### 5.2.2 Pengujian Fungsional

Pada pengujian fungsional dilakukan terhadap 2 kasus yaitu pengujian kode program sederhana dan kode program perangkat lunak yang dibuat.

Pada kasus pertama dijalankan perintah sebagai berikut.



```

1 javadoc -filename operasimatematika
2       -doclet extractor.Extractor
3       -docletpath GenerateJavadocToLatex.jar
4       -sourcepath OperasiMatematika/src
5       operasi

```

Perintah tersebut akan mengambil informasi dari kode program yang dapat dilihat pada Lampiran A.1. Setelah perintah tersebut dieksekusi akan menghasilkan *file*  $\text{\LaTeX}$  yang dapat dilihat pada Lampiran A.2. Jika *file*  $\text{\LaTeX}$  tersebut dieksekusi menggunakan perintah `pdflatex` akan menghasilkan sebuah *file* PDF yang dapat dilihat pada Lampiran A.3.

Dapat dilihat dari hasil yang terdapat Lampiran A.3, didapatkan 5 kelas yang dikonversi menjadi 5 buah *item list enumerate* yang masing berisikan nama kelas dari 5 file tersebut. *Method-method* yang terdapat kelas tersebut akan dibuat dengan menggunakan *list itemize* sebanyak  $n$  buah *method* dan parameter dari *method* tersebut dibuat menggunakan *list itemize* sebanyak  $n$  buah parameter.

Pada kasus kedua dijalankan perintah sebagai berikut.

```

15 javadoc -filename javadoctolatex
16       -classpath tools.jar
17       -doclet extractor.Extractor
18       -docletpath GenerateJavadocToLatex.jar
19       -sourcepath GenerateJavadocToLatex/src
20       extractor

```

Perintah tersebut akan mengambil informasi dari kode program yang dapat dilihat pada Lampiran B.1. Setelah perintah tersebut dieksekusi akan menghasilkan *file*  $\text{\LaTeX}$  yang dapat dilihat pada Lampiran B.2. Jika *file*  $\text{\LaTeX}$  tersebut dieksekusi menggunakan perintah `pdflatex` akan menghasilkan sebuah *file* PDF yang dapat dilihat pada Lampiran B.3.

Dapat dilihat dari hasil yang terdapat Lampiran B.3, didapatkan 4 kelas yang dikonversi menjadi 4 buah *item list enumerate* yang masing berisikan nama kelas dari 4 file tersebut. Atribut kelas akan dibuat dengan menggunakan *list itemize* sebanyak  $n$  buah atribut. *Method-method* yang terdapat kelas tersebut akan dibuat dengan menggunakan *list itemize* sebanyak  $n$  buah *method* dan parameter dari *method* tersebut dibuat menggunakan *list itemize* sebanyak  $n$  buah parameter.

### 5.2.3 Pengujian Eksperimental

Pengujian eksperimental dilakukan terhadap kode program SIAModels. SIAModels adalah sekumpulan kelas *java* yang mewakili objek-objek Sistem Informasi Akademik Universitas Katolik Parahyangan. SIAModels ini berisikan objek matakuliah pada Fakultas Teknologi Informasi dan Sains.

Pada pengujian menggunakan SIAModels dilakukan perintah sebagai berikut.

```

36 javadoc -filename siamodels
37       -doclet extractor.Extractor
38       -docletpath GenerateJavadocToLatex.jar
39       -sourcepath SIAModels/src/main/java

```

```

1      -subpackages id.ac.unpar.siamodels
2      id.ac.unpar.siamodels

```

Perintah tersebut akan mengambil informasi dari kode program yang dapat dilihat pada Lampiran C.1. Setelah perintah tersebut dieksekusi akan menghasilkan *file*  $\text{\LaTeX}$  yang dapat dilihat pada Lampiran C.2. Jika *file*  $\text{\LaTeX}$  tersebut dieksekusi menggunakan perintah `pdflatex` akan menghasilkan sebuah *file* PDF yang dapat dilihat pada Lampiran C.3.

Dapat dilihat dari hasil yang terdapat Lampiran C.3, didapatkan  $n$  buah kelas dan  $n$  buah *innerclass* yang dikonversi menjadi  $n$  buah *item list enumerate* yang masing berisikan nama kelas dan *innerclass* dari  $n$  *file* tersebut. Atribut kelas akan dibuat dengan menggunakan *list itemize* sebanyak  $n$  buah atribut. *Method-method* yang terdapat kelas tersebut akan dibuat dengan menggunakan *list itemize* sebanyak  $n$  buah *method* dan parameter dari *method* tersebut dibuat menggunakan *list itemize* sebanyak  $n$  buah parameter. Terdapat beberapa ketidaklaziman yang ditemukan setelah mengeksekusi perintah `pdflatex`.

- Terdapat karakter yang tidak lazim yaitu tanda `;` yang merepresentasikan tanda lebih kecil "<" dan tanda `,` yang merepresentasikan tanda lebih besar ">". Karakter tersebut muncul pada kelas:

1. TahunSemester - pada atribut kelas
2. Mahasiswa - pada *method* `calculateIPKLulus()`, `calculateIPLulus()`, `calculateIPTempuh()`, `calculateIPKumulatif()`, `calculateIPKTempuh()` dan `calculateIPS()`

Karakter yang tidak lazim tersebut muncul karena secara otomatis  $\text{\LaTeX}$  menggunakan OT1 *font encoding*. Pada *Ascii Table* tanda < memiliki kode 3C dan tanda > memiliki kode 3E dan secara otomatis  $\text{\LaTeX}$  akan memetakan tanda < dan tanda > menjadi tanda `;` dan `,` [4].

- Beberapa deskripsi tidak muncul pada hasil PDF, contohnya seperti pada kelas:

1. Semester - pada deskripsi kelas, atribut kelas dan *method*
2. TahunSemester - pada deskripsi *method* seperti pada *method* `getTahun()`
3. Mahasiswa - pada deskripsi kelas, atribut kelas dan beberapa *method*
4. Mahasiswa.Nilai - pada beberapa deskripsi *method*
5. JadwalKuliah - pada deskripsi kelas, atribut kelas dan *method*
6. MataKuliah - pada deskripsi kelas, atribut kelas, dan *method*
7. Dosen - pada deskripsi kelas, atribut kelas dan *method*

Setelah dilakukan pemeriksaan pada hal tersebut, kesalahan yang terjadi bukan dari perangkat lunak melainkan kode program yang menjadi masukan memiliki dokumentasi yang kurang lengkap.

- Parameter setiap *method* memiliki tipe variabel yang lengkap. Setelah dilakukan pemeriksaan, tipe variabel yang bersifat *Object* akan menghasilkan tipe variabel yang *fully-qualified* seperti contohnya pada tipe variabel `String` yang akan menghasilkan `java.lang.String`.

## BAB 6

### KESIMPULAN DAN SARAN

#### 6.1 Kesimpulan

Dari hasil pembangunan perangkat lunak Konversi Javadoc ke  $\text{\LaTeX}$ , didapatkanlah kesimpulan-kesimpulan sebagai berikut:

- Telah berhasil mengimplementasi *Javadoc Doclet* API untuk mengambil informasi dari sekumpulan *file Java* yang terdapat pada sebuah *package* dan mengonversikannya dari format *Javadoc* ke format  $\text{\LaTeX}$ .

#### 6.2 Saran

Dari hasil penelitian termasuk kesimpulan yang didapat, berikut adalah beberapa saran untuk pengembangan.

1. Pada saat ini, perangkat lunak tidak mengatasi *tag-tag html*. Pada pengujian menggunakan SIAModels terdapat beberapa kelas yang memiliki *tag-tag html* pada *javadoc* seperti contoh pada kelas Mahasiswa pada fungsi `calculateIPKLulus()`. Terdapat kode yang tidak lazim yaitu tanda `;` yang merepresentasikan tanda lebih kecil "<" dan tanda `;` yang merepresentasikan tanda lebih besar ">". Sebaiknya perangkat lunak dapat mengatasi *tag-tag html* jika javadoc memiliki *tag-tag* tersebut.
2. Pada saat ini, perangkat lunak menampilkan tulisan "tidak memiliki *exception*" pada bagian **Exception**. Sebaiknya mengikuti standar dari *javadoc* bahwa jika *method* yang tidak memiliki *exception* maka bagian **Exception** tidak akan muncul pada dokumentasi.



## DAFTAR REFERENSI

- [1] Oracle (1993) javadoc - the java api documentation generator. <https://docs.oracle.com/javase/8/docs/technotes/tools/windows/javadoc.html>. 27 September 2017.
- [2] Oracle (1993) Javadoc doclet api. <https://docs.oracle.com/javase/8/docs/jdk/api/javadoc/doclet/index.html>. 5 oktober 2017.
- [3] Lamport, L. (1994) *LaTeX: A Document Preparation System*, 2 edition.
- [4] Frank Mittelbach, W. L., Robin Fairbairns (2016). *LaTeX font encodings*, **1**, 19–30.



# LAMPIRAN A

## PENGUJIAN TERHADAP KODE PROGRAM SEDERHANA

### A.1 Kode Program

Listing A.1: OperasiMatematikaInterface.java

```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6  package javadoc;
7
8  /**
9   * Kelas Abstract OperasiMatematika. Kelas ini memiliki method {@link #calculate(int, int)}
10   *
11   * @author abathz
12   */
13  public abstract class OperasiMatematikaInterface {
14
15      /**
16       * Method untuk menghasilkan perhitungan 2 buah bilangan
17       * @param a Bilangan pertama
18       * @param b Bilangan kedua
19       * @return hasil perhitungan 2 buah bilangan {@link {@link Integer#NaN}}
20       */
21      public int calculate(int a, int b){return 0;}
22  }
```

Listing A.2: Pembagian.java

```
1  package javadoc;
2
3  /**
4   * Kelas ini merupakan Kelas Pembagian
5   *
6   * @author Adli Fariz Bonaputra
7   * @see "Pembagian"
8   */
9  public class Pembagian extends OperasiMatematikaInterface {
10
11      /**
12       * Atribut A
13       */
14      private int a;
15      /**
16       * Atribut B
17       */
18      private int b;
19
20      @Override
21      public int calculate(int a, int b) {
22          int hasil = a / b;
23          return hasil;
24      }
25
26  }
```

Listing A.3: Pengurangan.java

```
1  package javadoc;
2
3  /**
4   * Kelas ini merupakan Kelas Pengurangan
5   *
6   * @author Adli Fariz Bonaputra
7   * @see "Pengurangan"
8   */
9  public class Pengurangan extends OperasiMatematikaInterface {
10
11      /**
12       * Atribut A
13       */
```

```

14 private int a;
15 /**
16  * Atribut B
17  */
18 private int b;
19
20 @Override
21 public int calculate(int a, int b) {
22     int hasil = a - b;
23     return hasil;
24 }
25 }

```

Listing A.4: Perkalian.java

```

1 package javadoc;
2
3 /**
4  * Kelas ini merupakan Kelas Perkalian
5  *
6  * @author Adli Fariz Bonaputra
7  * @see "Perkalian"
8  *
9  */
10 public class Perkalian extends OperasiMatematikaInterface {
11
12     /**
13     * Atribut A
14     */
15     private int a;
16     /**
17     * Atribut B
18     */
19     private int b;
20
21     @Override
22     public int calculate(int a, int b) {
23         int hasil = a * b;
24         return hasil;
25     }
26 }

```

Listing A.5: Pertambahan.java

```

1 package javadoc;
2
3 /**
4  * Kelas ini merupakan Kelas Pertambahan
5  *
6  * @author Adli Fariz Bonaputra
7  * @see "Pertambahan"
8  *
9  */
10 public class Pertambahan extends OperasiMatematikaInterface {
11
12     /**
13     * Atribut A
14     */
15     private int a;
16     /**
17     * Atribut B
18     */
19     private int b;
20
21     @Override
22     public int calculate(int a, int b) {
23         int hasil = a + b;
24         return hasil;
25     }
26 }

```

## A.2 Hasil Latex

Listing A.6: operasimatematika.tex

```

1 \documentclass{article}
2 \begin{document}
3 \begin{enumerate}
4 \item \texttt{OperasiMatematikaInterface}
5
6 Kelas Abstract OperasiMatematika. Kelas ini memiliki method \texttt{calculate(int, int)}
7
8 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
9 \begin{itemize}
10 \item \texttt{public int calculate(int a, int b)}
11
12 Method untuk menghasilkan perhitungan 2 buah bilangan
13
14 \textbf{Parameter:}
15 \begin{itemize}
16 \item \texttt{int a} -
17 Bilangan pertama

```



```

18 \item \texttt{int b} -
19 Bilangan kedua
20 \end{itemize}
21 \textbf{Return Value}: hasil perhitungan 2 buah bilangan DoubleNaN
22
23 \textbf{Exception}: Tidak memiliki \textit{exception}
24
25 \end{itemize}
26 \item \texttt{Pembagian}
27
28 Kelas ini merupakan Kelas Pembagian
29
30 Atribut yang dimiliki kelas ini adalah sebagai berikut.
31 \begin{itemize}
32 \item \texttt{int a} - Atribut A
33 \item \texttt{int b} - Atribut B
34 \end{itemize}
35 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
36 \begin{itemize}
37 \item \texttt{public int calculate(int a, int b)}
38
39 Method untuk menghasilkan perhitungan 2 buah bilangan
40
41 \textbf{Parameter}:
42 \begin{itemize}
43 \item \texttt{int a} -
44 Bilangan pertama
45 \item \texttt{int b} -
46 Bilangan kedua
47 \end{itemize}
48 \textbf{Return Value}: hasil perhitungan 2 buah bilangan DoubleNaN
49
50 \textbf{Exception}: Tidak memiliki \textit{exception}
51
52 \end{itemize}
53 \item \texttt{Pengurangan}
54
55 Kelas ini merupakan Kelas Pengurangan
56
57 Atribut yang dimiliki kelas ini adalah sebagai berikut.
58 \begin{itemize}
59 \item \texttt{int a} - Atribut A
60 \item \texttt{int b} - Atribut B
61 \end{itemize}
62 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
63 \begin{itemize}
64 \item \texttt{public int calculate(int a, int b)}
65
66 Method untuk menghasilkan perhitungan 2 buah bilangan
67
68 \textbf{Parameter}:
69 \begin{itemize}
70 \item \texttt{int a} -
71 Bilangan pertama
72 \item \texttt{int b} -
73 Bilangan kedua
74 \end{itemize}
75 \textbf{Return Value}: hasil perhitungan 2 buah bilangan DoubleNaN
76
77 \textbf{Exception}: Tidak memiliki \textit{exception}
78
79 \end{itemize}
80 \item \texttt{Perkalian}
81
82 Kelas ini merupakan Kelas Perkalian
83
84 Atribut yang dimiliki kelas ini adalah sebagai berikut.
85 \begin{itemize}
86 \item \texttt{int a} - Atribut A
87 \item \texttt{int b} - Atribut B
88 \end{itemize}
89 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
90 \begin{itemize}
91 \item \texttt{public int calculate(int a, int b)}
92
93 Method untuk menghasilkan perhitungan 2 buah bilangan
94
95 \textbf{Parameter}:
96 \begin{itemize}
97 \item \texttt{int a} -
98 Bilangan pertama
99 \item \texttt{int b} -
100 Bilangan kedua
101 \end{itemize}
102 \textbf{Return Value}: hasil perhitungan 2 buah bilangan DoubleNaN
103
104 \textbf{Exception}: Tidak memiliki \textit{exception}
105
106 \end{itemize}
107 \item \texttt{Pertambahan}
108
109 Kelas ini merupakan Kelas Pertambahan
110
111 Atribut yang dimiliki kelas ini adalah sebagai berikut.
112 \begin{itemize}
113 \item \texttt{int a} - Atribut A
114 \item \texttt{int b} - Atribut B
115 \end{itemize}
116 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.

```

```
117 \begin{itemize}
118 \item \texttt{public int calculate(int a, int b)}
119
120 Method untuk menghasilkan perhitungan 2 buah bilangan
121
122 \textbf{Parameter:}
123 \begin{itemize}
124 \item \texttt{int a} -
125 Bilangan pertama
126 \item \texttt{int b} -
127 Bilangan kedua
128 \end{itemize}
129 \textbf{Return Value:} hasil perhitungan 2 buah bilangan DoubleNaN
130
131 \textbf{Exception:} Tidak memiliki \textit{exception}
132
133 \end{itemize}
134 \end{enumerate}
135 \end{document}
```

### A.3 Hasil PDF

### 1. OperasiMatematikaInterface

Kelas Abstract OperasiMatematika. Kelas ini memiliki method `calculate(int, int)`

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public int calculate(int a, int b)`  
Method untuk menghasilkan perhitungan 2 buah bilangan  
**Parameter:**
  - `int a` - Bilangan pertama
  - `int b` - Bilangan kedua**Return Value:** hasil perhitungan 2 buah bilangan DoubleNaN  
**Exception:** Tidak memiliki *exception*

### 2. Pembagian

Kelas ini merupakan Kelas Pembagian

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `int a` - Atribut A
- `int b` - Atribut B

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public int calculate(int a, int b)`  
Method untuk menghasilkan perhitungan 2 buah bilangan  
**Parameter:**
  - `int a` - Bilangan pertama
  - `int b` - Bilangan kedua**Return Value:** hasil perhitungan 2 buah bilangan DoubleNaN  
**Exception:** Tidak memiliki *exception*

### 3. Pengurangan

Kelas ini merupakan Kelas Pengurangan

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `int a` - Atribut A
- `int b` - Atribut B

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public int calculate(int a, int b)`  
Method untuk menghasilkan perhitungan 2 buah bilangan  
**Parameter:**

- `int a` - Bilangan pertama
- `int b` - Bilangan kedua

**Return Value:** hasil perhitungan 2 buah bilangan DoubleNaN

**Exception:** Tidak memiliki *exception*

#### 4. Perkalian

Kelas ini merupakan Kelas Perkalian

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `int a` - Atribut A
- `int b` - Atribut B

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public int calculate(int a, int b)`

Method untuk menghasilkan perhitungan 2 buah bilangan

**Parameter:**

- `int a` - Bilangan pertama
- `int b` - Bilangan kedua

**Return Value:** hasil perhitungan 2 buah bilangan DoubleNaN

**Exception:** Tidak memiliki *exception*

#### 5. Pertambahan

Kelas ini merupakan Kelas Pertambahan

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `int a` - Atribut A
- `int b` - Atribut B

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public int calculate(int a, int b)`

Method untuk menghasilkan perhitungan 2 buah bilangan

**Parameter:**

- `int a` - Bilangan pertama
- `int b` - Bilangan kedua

**Return Value:** hasil perhitungan 2 buah bilangan DoubleNaN

**Exception:** Tidak memiliki *exception*

# LAMPIRAN B

## PENGUJIAN TERHADAP KODE PROGRAM PERANGKAT LUNAK

### B.1 Kode Program

Listing B.1: Extractor.java

```
1 package extractor;
2
3 import com.sun.javadoc.*;
4 import java.io.*;
5
6 /**
7  * Kelas ini merupakan kelas untuk menjalankan \textit{custom doclet}
8  *
9  * @author Adli Fariz Bonaputra
10 */
11 public class Extractor {
12
13     /**
14      * atribut untuk nama \textit{file}
15      */
16     private static String fileName;
17
18     /**
19      * \textit{Method} ini berperan sebagai \textit{method} untuk menjalankan
20      * \textit{custom doclet}
21      *
22      * @param root berperan sebagai mengambil seluruh informasi spesifik dari
23      * \textit{option} yang terdapat pada \textit{command-line} sebuah
24      * \textit{terminal}. Selain itu berperan juga untuk mengambil informasi dari
25      * sekumpulan \textit{file java} yang akan di proses.
26      * @return kondisi true
27      */
28     public static boolean start(RootDoc root) {
29         init(root.classes());
30         return true;
31     }
32
33     /**
34      * \textit{Method} ini berperan untuk menulis kedalam sebuah \textit{file}
35      * saat \textit{javadoc} berjalan.
36      *
37      * @param classes sebuah array yang berisikan sekumpulan \textit{file java}
38      * yang akan di proses.
39      */
40     private static void init(ClassDoc[] classes) {
41         FileWriter file;
42         new File("output").mkdirs();
43         try {
44             if (fileName == null) {
45                 file = new FileWriter("output/doc.tex");
46             } else {
47                 file = new FileWriter("output/" + fileName + ".tex");
48             }
49             BufferedWriter out = new BufferedWriter(file);
50             out.write("\\documentclass{article}\n");
51             out.write("\\begin{document}\n");
52             out.write("\\begin{enumerate}\n");
53
54             ClassExtractor.extractClassContent(classes, out);
55
56             out.write("\\end{enumerate}\n");
57             out.write("\\end{document}\n");
58             out.close();
59         } catch (IOException e) { }
60     }
61
62     /**
63      * Method untuk menghitung banyak option yang digunakan pada
64      * \textit{command-line}
65      *
66      * @param option sebuah option
67      * @return panjang setiap option
68      */
69 }
```

```

69 public static int optionLength(String option) {
70     if (option.equals("-filename")) {
71         return 2;
72     }
73     return Doclet.optionLength(option);
74 }
75 }
76
77 /**
78  * Pengecekan option valid
79  *
80  * @param args String array 2 dimensi dari option
81  * @param err sebuah error jika tidak terdapat option tersebut.
82  * @return bernilai true jika option tersebut dikenali, false jika option
83  * tersebut tidak dikenali
84  */
85 public static boolean validOptions(String[][] args, DocErrorReporter err) {
86     for (int i = 0; i < args.length; ++i) {
87         if (args[i][0].equals("-filename")) {
88             fileName = args[i][1];
89         }
90     }
91     return Doclet.validOptions(args, err);
92 }
93 }

```

Listing B.2: ClassExtractor.java

```

1 package extractor;
2
3 import com.sun.javadoc.*;
4 import java.io.*;
5
6 /**
7  * Kelas ini merupakan kelas untuk mengambil informasi dari sebuah kelas
8  *
9  * @author Adli Fariz Bonaputra
10  */
11 public class ClassExtractor {
12
13     /**
14      * \textit{Method} ini akan menampilkan nama kelas berserta penjelasan dari
15      * sebuah kelas
16      *
17      * @param classes sebuah array berisikan sejumlah kelas
18      * @param out turunan dari kelas \texttt{Writer} yang digunakan untuk menulis
19      * file text
20      */
21     public static void extractClassContent(ClassDoc[] classes, BufferedWriter out) {
22         try {
23             for (ClassDoc classDoc : classes) {
24                 Type[] interfaces = classDoc.interfaceTypes();
25                 if (interfaces.length > 0) {
26                     if (!classDoc.isInterface()) {
27                         for (int i = 0; i < interfaces.length; i++) {
28                             out.write("\\item_\\texttt{" + classDoc.name() + "_implements_" + interfaces[i].typeName() + "}" + "\\n\\n");
29                         }
30                     } else {
31                         out.write("\\item_\\texttt{" + classDoc.name() + "}" + "\\n\\n");
32                     }
33                 } else {
34                     out.write("\\item_\\texttt{" + classDoc.name() + "}" + "\\n\\n");
35                 }
36
37                 Tag[] inlineTags = classDoc.inlineTags();
38                 for (int i = 0; i < inlineTags.length; i++) {
39                     if (i == 1) {
40                         out.write("\\texttt{" + inlineTags[i].text().replace("#", "") + "}");
41                     } else {
42                         out.write(inlineTags[i].text().replace("#", "").replace("_", "\\_").replace("&", "\\&"));
43                     }
44                 }
45                 out.write("\\n\\n");
46
47                 FieldDoc[] fields = classDoc.fields(false);
48                 AttributeClassExtractor.extractAttributeClassContent(fields, out);
49
50                 MethodDoc[] methods = classDoc.methods(false);
51                 MethodClassExtractor.extractMethodClassContent(classDoc, methods, out);
52             }
53         } catch (IOException e) { }
54     }
55 }

```

Listing B.3: AttributeClassExtractor.java

```

1 package extractor;
2
3 import com.sun.javadoc.*;
4 import java.io.*;
5
6 /**
7  * Kelas ini merupakan kelas untuk mengambil informasi sebuah atribut yang
8  * terdapat pada kelas
9  *
10  * @author Adli Fariz Bonaputra

```

```

11  */
12  public class AttributeClassExtractor {
13
14      /**
15       * \textit{Method} ini akan menampilkan atribut-atribut yang dimiliki oleh
16       * sebuah kelas
17       *
18       * @param fields sebuah array berisikan sejumlah atribut dari kelas
19       * @param out turunan dari kelas \texttt{Writer} yang digunakan untuk menulis
20       * file text
21       */
22  public static void extractAttributeClassContent(FieldDoc[] fields, BufferedWriter out) {
23      try {
24          if (fields.length == 0) {
25              out.write("Kelas_in_i_tidak_memiliki_atribut.");
26          } else {
27              out.write("Atribut_yang_dimiliki_kelas_in_i_adalah_sebagai_berikut.\n");
28              out.write("\begin{itemize}\n");
29              for (int j = 0; j < fields.length; j++) {
30                  Type type = fields[j].type();
31                  out.write("\item_\texttt{" + type.typeName() + "_" + fields[j].name().replace("_", "\\_") + "}_-" + fields[j].
32                      commentText() + "\n");
33              }
34              out.write("\end{itemize}\n");
35          } catch (IOException e) { }
36      }
37  }

```

Listing B.4: MethodClassExtractor.java

```

1  package extractor;
2
3  import com.sun.javadoc.*;
4  import java.io.*;
5
6  /**
7   * Kelas ini merupakan kelas untuk mengambil informasi sebuah \textit{method}
8   * terdapat pada kelas
9   *
10  * @author Adli Fariz Bonaputra
11  */
12  public class MethodClassExtractor {
13
14      /**
15       * \textit{Method} ini akan menampilkan \textit{method-method} yang dimiliki
16       * oleh sebuah kelas
17       *
18       * @param superclass sebuah objek ClassDoc
19       * @param methods sebuah array berisikan sejumlah \textit{method} dari kelas
20       * @param out turunan dari kelas \texttt{Writer} yang digunakan untuk menulis
21       * file text
22       */
23  public static void extractMethodClassContent(ClassDoc superclass, MethodDoc[] methods, BufferedWriter out) {
24      try {
25          if (methods.length == 0) {
26              out.write("Kelas_in_i_tidak_memiliki_method.");
27          } else {
28              out.write("\textit{Method-method}_yang_dimiliki_kelas_in_i_adalah_sebagai_berikut.\n");
29              out.write("\begin{itemize}\n");
30              for (MethodDoc method : methods) {
31                  if (method.overriddenMethod() != null) {
32                      MethodDoc overrideMethod = method.overriddenMethod();
33                      Type type = overrideMethod.returnType();
34                      Parameter[] paramOverrideMethod = overrideMethod.parameters();
35                      out.write("\item_\texttt{" + " " + type.typeName() + "_" + overrideMethod.name() + "("
36                          ;
37                      for (int k = 0; k < paramOverrideMethod.length; k++) {
38                          Type typeParam = paramOverrideMethod[k].type();
39                          out.write(typeParam.toString() + "_" + paramOverrideMethod[k].name());
40                          if (k < paramOverrideMethod.length - 1) {
41                              out.write(",");
42                          }
43                      }
44                      out.write(")}\n\n");
45
46                      Tag[] inlineTags = overrideMethod.inlineTags();
47                      for (int i = 0; i < inlineTags.length; i++) {
48                          if (i == 1) {
49                              out.write("\texttt{" + inlineTags[i].text().replace("#", " ") + "}");
50                          } else {
51                              out.write(inlineTags[i].text().replace("#", " "));
52                          }
53                      }
54
55                      out.write("\n\n");
56
57                      ParamTag[] paramTags = overrideMethod.paramTags();
58                      ParameterMethod(out, paramOverrideMethod, paramTags);
59
60                      Tag[] returnTags = overrideMethod.tags("return");
61                      Tag[] throwTags = overrideMethod.tags("throws");
62                      AnnotationDesc[] override = overrideMethod.annotations();
63                      ReturnMethod(out, type, returnTags);
64                      ExceptionMethod(out, throwTags);
65                      if (override.length != 0) {
66                          out.write("\textbf{Override}:\texttt{" + overrideMethod.name() + "}_dari_kelas_\texttt{" + superclass.superclass
67                              ().name() + "}");

```

```

66         out.write("\n\n");
67     }
68 } else {
69     Type type = method.returnType();
70     Parameter[] paramMethod = method.parameters();
71     out.write("\\item\\texttt{" + method.modifiers() + " " + type.typeName() + " " + method.name() + "(");
72     for (int k = 0; k < paramMethod.length; k++) {
73         Type typeParam = paramMethod[k].type();
74         out.write(typeParam.toString() + " " + paramMethod[k].name());
75         if (k < paramMethod.length - 1) {
76             out.write(", ");
77         }
78     }
79     out.write(")\\n\n");
80
81     Tag[] inlineTags = method.inlineTags();
82     for (int i = 0; i < inlineTags.length; i++) {
83         if (i == 1) {
84             out.write("\\texttt{" + inlineTags[i].text().replace("#", " ") + "}");
85         } else {
86             out.write(inlineTags[i].text().replace("#", " "));
87         }
88     }
89
90     out.write("\n\n");
91
92     ParamTag[] paramTags = method.paramTags();
93     ParameterMethod(out, paramMethod, paramTags);
94
95     Tag[] returnTags = method.tags("return");
96     Tag[] throwTags = method.tags("throws");
97     AnnotationDesc[] override = method.annotations();
98     ReturnTypeInfo(out, type, returnTags);
99     ExceptionMethod(out, throwTags);
100     if (override.length != 0) {
101         out.write("\\textbf{Override}:\\texttt{" + method.name() + "}_dari_kelas\\texttt{" + superclass.superclass().name
102             + "}" + ")");
103         out.write("\n\n");
104     }
105 }
106 out.write("\\end{itemize}\\n");
107 }
108 } catch (IOException e) { }
109 }
110
111 /**
112  * \\textit{Method} ini akan menampilkan parameter \\textit{method-method} yang dimiliki
113  * oleh sebuah kelas
114  *
115  * @param out turunan dari kelas \\texttt{Writer} yang digunakan untuk menulis file text
116  * @param paramMethod sebuah array berisikan sejumlah \\textit{method} dari kelas
117  * @param paramTags sebuah array berisikan sejumlah parameter \\textit{method} dari kelas
118  */
119 private static void ParameterMethod(BufferedWriter out, Parameter[] paramMethod, ParamTag[] paramTags){
120     try {
121         if (paramTags.length == 0) {
122             out.write("\\textbf{Parameter:}\\n");
123             out.write("\\begin{itemize}\\n");
124             if (paramMethod.length != 0) {
125                 for (int i = 0; i < paramMethod.length; i++) {
126                     out.write("\\item\\texttt{" + paramMethod[i].toString() + "}_\\n");
127                 }
128             } else {
129                 out.write("\\item_Tidak_miliki_parameter_\\textit{method}\\n");
130             }
131             out.write("\\end{itemize}\\n");
132         } else {
133             out.write("\\textbf{Parameter:}\\n");
134             out.write("\\begin{itemize}\\n");
135             for (int k = 0; k < paramTags.length; k++) {
136                 Type type = paramMethod[k].type();
137                 String[] typeParam = type.toString().split("\\.");
138                 out.write("\\item\\texttt{" + typeParam[typeParam.length - 1] + " " + paramTags[k].parameterName() + "}_\\n");
139                 Tag[] inlineTagsInParameter = paramTags[k].inlineTags();
140                 for (int i = 0; i < inlineTagsInParameter.length; i++) {
141                     if (i == 1) {
142                         out.write("\\texttt{" + inlineTagsInParameter[i].text().replace("#", " ") + "}");
143                     } else {
144                         out.write(inlineTagsInParameter[i].text());
145                     }
146                 }
147                 out.write("\n");
148             }
149             out.write("\\end{itemize}\\n");
150         }
151     } catch (IOException e) { }
152 }
153
154 /**
155  * \\textit{Method} ini akan menampilkan \\textit{return type} dari \\textit{method-method} yang dimiliki
156  * oleh sebuah kelas
157  *
158  * @param out turunan dari kelas \\texttt{Writer} yang digunakan untuk menulis file text
159  * @param type sebuah objek Type
160  * @param returnTags sebuah array berisikan sejumlah \\textit{return type} dari \\textit{method} dari kelas
161  */
162 private static void ReturnTypeInfo(BufferedWriter out, Type type, Tag[] returnTags){
163     try {

```



```

164     if (type.typeName().equals("void") || returnTags.length == 0) {
165         out.write("\\textbf{Return Value}:_Tidak_miliki_\\textit{return_value}\\n\\n");
166     } else {
167         out.write("\\textbf{Return Value}:_");
168         Tag[] inlineTagsInReturnValue = returnTags[0].inlineTags();
169         for (int i = 0; i < inlineTagsInReturnValue.length; i++) {
170             if (i == 1) {
171                 out.write(inlineTagsInReturnValue[i].text().replace("#", ""));
172             } else {
173                 out.write(inlineTagsInReturnValue[i].text().replace("#{link", "").replace("}", "").replace("#", ""));
174             }
175         }
176         out.write("\\n\\n");
177     }
178 } catch (IOException e) { }
179 }
180
181 /**
182  * \\textit{Method} ini akan menampilkan \\textit{return type} dari \\textit{method-method} yang dimiliki
183  * oleh sebuah kelas
184  *
185  * @param out turunan dari kelas \\texttt{Writer} yang digunakan untuk menulis file text
186  * @param throwTags sebuah array berisikan sejumlah \\textit{exception} dari \\textit{method} dari kelas
187  */
188 private static void ExceptionMethod(BufferedWriter out, Tag[] throwTags) throws IOException {
189     try {
190         if (throwTags.length == 0) {
191             out.write("\\textbf{Exception}:_Tidak_miliki_\\textit{exception}");
192             out.write("\\n\\n");
193         } else {
194             out.write("\\textbf{Exception}:_" + throwTags[0].text());
195             out.write("\\n\\n");
196         }
197     } catch (IOException e) { }
198 }
199 }

```

## B.2 Hasil Latex

Listing B.5: javadoctolatex.tex

```

1 \documentclass{article}
2 \begin{document}
3 \begin{enumerate}
4 \item \texttt{Extractor}
5
6 Kelas ini merupakan kelas untuk menjalankan \\textit{custom doclet}
7
8 Atribut yang dimiliki kelas ini adalah sebagai berikut.
9 \begin{itemize}
10 \item \texttt{String fileName} - atribut untuk nama \\textit{file}
11 \end{itemize}
12 \\textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
13 \begin{itemize}
14 \item \texttt{public static boolean start(com.sun.javadoc.RootDoc root)}
15
16 \\textit{Method} ini berperan sebagai \\textit{method} untuk menjalankan
17 \\textit{custom doclet}
18
19 \textbf{Parameter:}
20 \begin{itemize}
21 \item \texttt{RootDoc root} -
22 berperan sebagai mengambil seluruh informasi spesifik dari
23 \\textit{option} yang terdapat pada \\textit{command-line} sebuah
24 \\textit{terminal}. Selain itu berperan juga untuk mengambil informasi dari
25 sekumpulan \\textit{file java} yang akan di proses.
26 \end{itemize}
27 \textbf{Return Value:} kondisi true
28
29 \textbf{Exception:} Tidak memiliki \\textit{exception}
30
31 \item \texttt{private static void init(com.sun.javadoc.ClassDoc[] classes)}
32
33 \\textit{Method} ini berperan untuk menulis kedalam sebuah \\textit{file}
34 saat \\textit{javadoc} berjalan.
35
36 \textbf{Parameter:}
37 \begin{itemize}
38 \item \texttt{ClassDoc[] classes} -
39 sebuah array yang berisikan sekumpulan \\textit{file java}
40 yang akan di proses.
41 \end{itemize}
42 \textbf{Return Value:} Tidak memiliki \\textit{return value}
43
44 \textbf{Exception:} Tidak memiliki \\textit{exception}
45
46 \item \texttt{public static int optionLength(java.lang.String option)}
47
48 Method untuk menghitung banyak option yang digunakan pada
49 \\textit{command-line}
50
51 \textbf{Parameter:}
52 \begin{itemize}
53 \item \texttt{String option} -
54 sebuah option

```

```

55 \end{itemize}
56 \textbf{Return Value}: panjang setiap option
57
58 \textbf{Exception}: Tidak memiliki \textit{exception}
59
60 \item \texttt{public static boolean validOptions(java.lang.String[][] args, com.sun.javadoc.DocErrorReporter err)}
61
62 Pengecekan option valid
63
64 \textbf{Parameter}:
65 \begin{itemize}
66 \item \texttt{String[][] args} -
67 String array 2 dimensi dari option
68 \item \texttt{DocErrorReporter err} -
69 sebuah error jika tidak terdapat option tersebut.
70 \end{itemize}
71 \textbf{Return Value}: bernilai true jika option tersebut dikenali, false jika option
72 tersebut tidak dikenali
73
74 \textbf{Exception}: Tidak memiliki \textit{exception}
75
76 \end{itemize}
77 \item \texttt{ClassExtractor}
78
79 Kelas ini merupakan kelas untuk mengambil informasi dari sebuah kelas
80
81 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
82 \begin{itemize}
83 \item \texttt{public static void extractClassContent(com.sun.javadoc.ClassDoc[] classes, java.io.BufferedWriter out)}
84
85 \textit{Method} ini akan menampilkan nama kelas berserta penjelasan dari
86 sebuah kelas
87
88 \textbf{Parameter}:
89 \begin{itemize}
90 \item \texttt{ClassDoc[] classes} -
91 sebuah array berisikan sejumlah kelas
92 \item \texttt{BufferedWriter out} -
93 turunan dari kelas \texttt{Writer} yang digunakan untuk menulis
94 file text
95 \end{itemize}
96 \textbf{Return Value}: Tidak memiliki \textit{return value}
97
98 \textbf{Exception}: Tidak memiliki \textit{exception}
99
100 \end{itemize}
101 \item \texttt{MethodClassExtractor}
102
103 Kelas ini merupakan kelas untuk mengambil informasi sebuah \textit{method}
104 terdapat pada kelas
105
106 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
107 \begin{itemize}
108 \item \texttt{public static void extractMethodClassContent(com.sun.javadoc.ClassDoc superclass, com.sun.javadoc.MethodDoc[]
109 methods, java.io.BufferedWriter out)}
110
111 \textit{Method} ini akan menampilkan \textit{method-method} yang dimiliki
112 oleh sebuah kelas
113
114 \textbf{Parameter}:
115 \begin{itemize}
116 \item \texttt{ClassDoc superclass} -
117 sebuah objek ClassDoc
118 \item \texttt{MethodDoc[] methods} -
119 sebuah array berisikan sejumlah \textit{method} dari kelas
120 \item \texttt{BufferedWriter out} -
121 turunan dari kelas \texttt{Writer} yang digunakan untuk menulis
122 file text
123 \end{itemize}
124 \textbf{Return Value}: Tidak memiliki \textit{return value}
125
126 \textbf{Exception}: Tidak memiliki \textit{exception}
127
128 \item \texttt{private static void ParameterMethod(java.io.BufferedWriter out, com.sun.javadoc.Parameter[] paramMethod, com.sun.
129 javadoc.ParamTag[] paramTags)}
130
131 \textit{Method} ini akan menampilkan parameter \textit{method-method} yang dimiliki
132 oleh sebuah kelas
133
134 \textbf{Parameter}:
135 \begin{itemize}
136 \item \texttt{BufferedWriter out} -
137 turunan dari kelas \texttt{Writer} yang digunakan untuk menulis file text
138 \item \texttt{Parameter[] paramMethod} -
139 sebuah array berisikan sejumlah \textit{method} dari kelas
140 \item \texttt{ParamTag[] paramTags} -
141 sebuah array berisikan sejumlah parameter \textit{method} dari kelas
142 \end{itemize}
143 \textbf{Return Value}: Tidak memiliki \textit{return value}
144
145 \textbf{Exception}: Tidak memiliki \textit{exception}
146
147 \item \texttt{private static void ReturnTypeInfoMethod(java.io.BufferedWriter out, com.sun.javadoc.Type type, com.sun.javadoc.Tag[]
148 returnTags)}
149
150 \textit{Method} ini akan menampilkan \textit{return type} dari \textit{method-method} yang dimiliki
151 oleh sebuah kelas
152
153 \textbf{Parameter}:

```

```

151 \begin{itemize}
152 \item \texttt{BufferedWriter out} -
153 turunan dari kelas \texttt{Writer} yang digunakan untuk menulis file text
154 \item \texttt{Type type} -
155 sebuah objek Type
156 \item \texttt{Tag[] returnTags} -
157 sebuah array berisikan sejumlah \textit{return type} dari \textit{method} dari kelas
158 \end{itemize}
159 \textbf{Return Value}: Tidak memiliki \textit{return value}
160
161 \textbf{Exception}: Tidak memiliki \textit{exception}
162
163 \item \texttt{private static void ExceptionMethod(java.io.BufferedWriter out, com.sun.javadoc.Tag[] throwTags)}
164
165 \textit{Method} ini akan menampilkan \textit{return type} dari \textit{method-method} yang dimiliki
166 oleh sebuah kelas
167
168 \textbf{Parameter:}
169 \begin{itemize}
170 \item \texttt{BufferedWriter out} -
171 turunan dari kelas \texttt{Writer} yang digunakan untuk menulis file text
172 \item \texttt{Tag[] throwTags} -
173 sebuah array berisikan sejumlah \textit{exception} dari \textit{method} dari kelas
174 \end{itemize}
175 \textbf{Return Value}: Tidak memiliki \textit{return value}
176
177 \textbf{Exception}: Tidak memiliki \textit{exception}
178
179 \end{itemize}
180 \item \texttt{AttributeClassExtractor}
181
182 Kelas ini merupakan kelas untuk mengambil informasi sebuah atribut yang
183 terdapat pada kelas
184
185 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
186 \begin{itemize}
187 \item \texttt{public static void extractAttributeClassContent(com.sun.javadoc.FieldDoc[] fields, java.io.BufferedWriter out)}
188
189 \textit{Method} ini akan menampilkan atribut-atribut yang dimiliki oleh
190 sebuah kelas
191
192 \textbf{Parameter:}
193 \begin{itemize}
194 \item \texttt{FieldDoc[] fields} -
195 sebuah array berisikan sejumlah atribut dari kelas
196 \item \texttt{BufferedWriter out} -
197 turunan dari kelas \texttt{Writer} yang digunakan untuk menulis
198 file text
199 \end{itemize}
200 \textbf{Return Value}: Tidak memiliki \textit{return value}
201
202 \textbf{Exception}: Tidak memiliki \textit{exception}
203
204 \end{itemize}
205 \end{enumerate}
206 \end{document}

```

## B.3 Hasil PDF

## 1. Extractor

Kelas ini merupakan kelas untuk menjalankan *custom doclet*

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- **String fileName** - atribut untuk nama *file*

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- **public static boolean start(com.sun.javadoc.RootDoc root)**  
*Method* ini berperan sebagai *method* untuk menjalankan *custom doclet*

**Parameter:**

- **RootDoc root** - berperan sebagai mengambil seluruh informasi spesifik dari *option* yang terdapat pada *command-line* sebuah *terminal*. Selain itu berperan juga untuk mengambil informasi dari sekumpulan *file java* yang akan di proses.

**Return Value:** kondisi true

**Exception:** Tidak memiliki *exception*

- **private static void init(com.sun.javadoc.ClassDoc[] classes)**  
*Method* ini berperan untuk menulis kedalam sebuah *file* saat *javadoc* berjalan.

**Parameter:**

- **ClassDoc[] classes** - sebuah array yang berisikan sekumpulan *file java* yang akan di proses.

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- **public static int optionLength(java.lang.String option)**  
*Method* untuk menghitung banyak *option* yang digunakan pada *command-line*

**Parameter:**

- **String option** - sebuah *option*

**Return Value:** panjang setiap *option*

**Exception:** Tidak memiliki *exception*

- **public static boolean validOptions(java.lang.String[][] args, com.sun.javadoc.DocErrorReporter err)**  
Pengecekan *option* valid

**Parameter:**

- **String[][] args** - String array 2 dimensi dari *option*
- **DocErrorReporter err** - sebuah error jika tidak terdapat *option* tersebut.

**Return Value:** bernilai true jika option tersebut dikenali, false jika option tersebut tidak dikenali

**Exception:** Tidak memiliki *exception*

## 2. ClassExtractor

Kelas ini merupakan kelas untuk mengambil informasi dari sebuah kelas

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static void extractClassContent(com.sun.javadoc.ClassDoc[] classes, java.io.BufferedWriter out)`

*Method* ini akan menampilkan nama kelas berserta penjelasan dari sebuah kelas

**Parameter:**

- `ClassDoc[] classes` - sebuah array berisikan sejumlah kelas
- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

## 3. MethodClassExtractor

Kelas ini merupakan kelas untuk mengambil informasi sebuah *method* terdapat pada kelas

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static void extractMethodClassContent(com.sun.javadoc.ClassDoc superclass, com.sun.javadoc.MethodDoc[] methods, java.io.BufferedWriter out)`

*Method* ini akan menampilkan *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `ClassDoc superclass` - sebuah objek `ClassDoc`
- `MethodDoc[] methods` - sebuah array berisikan sejumlah *method* dari kelas
- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `private static void ParameterMethod(java.io.BufferedWriter out, com.sun.javadoc.Parameter[] paramMethod, com.sun.javadoc.ParamTag[] paramTags)`

*Method* ini akan menampilkan parameter *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text
- `Parameter[] paramMethod` - sebuah array berisikan sejumlah *method* dari kelas
- `ParamTag[] paramTags` - sebuah array berisikan sejumlah parameter *method* dari kelas

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `private static void ReturnTypeInfoMethod(java.io.BufferedWriter out, com.sun.javadoc.Type type, com.sun.javadoc.Tag[] returnTags)`

*Method* ini akan menampilkan *return type* dari *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text
- `Type type` - sebuah objek `Type`
- `Tag[] returnTags` - sebuah array berisikan sejumlah *return type* dari *method* dari kelas

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `private static void ExceptionMethod(java.io.BufferedWriter out, com.sun.javadoc.Tag[] throwTags)`

*Method* ini akan menampilkan *return type* dari *method-method* yang dimiliki oleh sebuah kelas

**Parameter:**

- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text
- `Tag[] throwTags` - sebuah array berisikan sejumlah *exception* dari *method* dari kelas

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

#### 4. `AttributeClassExtractor`

Kelas ini merupakan kelas untuk mengambil informasi sebuah atribut yang terdapat pada kelas

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static void extractAttributeClassContent(com.sun.javadoc.FieldDoc[] fields, java.io.BufferedWriter out)`

*Method* ini akan menampilkan atribut-atribut yang dimiliki oleh sebuah kelas

**Parameter:**

- `FieldDoc[] fields` - sebuah array berisikan sejumlah atribut dari kelas
- `BufferedWriter out` - turunan dari kelas `Writer` yang digunakan untuk menulis file text

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*





# LAMPIRAN C

## PENGUJIAN TERHADAP KODE PROGRAM SIAMODELS

### C.1 Kode Program

Listing C.1: Dosen.java

```
1 package id.ac.unpar.siamodels;
2
3 public class Dosen {
4     private String nik;
5     private String nama;
6
7     public Dosen(String nik, String nama) throws IllegalArgumentException {
8         super();
9         if (nik == null && nama == null) {
10             throw new IllegalArgumentException("Salah_satu_dari_NIK_atau_nama_harus_diisi!");
11         }
12         this.nik = nik;
13         this.nama = nama;
14     }
15
16     public String getNik() {
17         return nik;
18     }
19
20     public void setNik(String nik) {
21         this.nik = nik;
22     }
23
24     public String getNama() {
25         return nama;
26     }
27
28     public void setNama(String nama) {
29         this.nama = nama;
30     }
31
32     /**
33      * Equality check for dosen. First check NIK if both available.
34      * Otherwise, check for name.
35      * @return true if equal, false otherwise
36      */
37     @Override
38     public boolean equals(Object obj) {
39         if (this == obj)
40             return true;
41         if (obj == null)
42             return false;
43         if (getClass() != obj.getClass())
44             return false;
45         Dosen other = (Dosen) obj;
46         if (nik != null && other.nik != null) {
47             return nik.equals(other.nik);
48         }
49         if (nama != null) {
50             return nama.equals(other.nama);
51         }
52         return false;
53     }
54
55 }
56 }
```

Listing C.2: InfoMataKuliah.java

```
1 package id.ac.unpar.siamodels;
2
3 import java.lang.annotation.Target;
4 import java.lang.annotation.ElementType;
5 import java.lang.annotation.Retention;
6 import java.lang.annotation.RetentionPolicy;
7
8 @Retention(RetentionPolicy.RUNTIME)
9 @Target(ElementType.TYPE)
10 public @interface InfoMataKuliah {
```

```

11  /**
12   * Jumlah bobot sks dari mata kuliah ini
13   *
14   * @return jumlah bobot sks
15   */
16  public int sks();
17
18  /**
19   * Nama mata kuliah ini
20   *
21   * @return nama mata kuliah
22   */
23  public String nama();
24 }

```

Listing C.3: JadwalKuliah.java

```

1  package id.ac.unpar.siamodels;
2
3  import java.time.DayOfWeek;
4  import java.time.LocalDateTime;
5
6  public class JadwalKuliah {
7      protected MataKuliah mataKuliah;
8      protected Character kelas;
9      protected DayOfWeek hari;
10     protected LocalDateTime waktuMulai;
11     protected LocalDateTime waktuSelesai;
12     protected String lokasi;
13     protected Dosen pengajar;
14
15     /**
16      * Membuat jadwal kuliah baru
17      * @param mataKuliah mata kuliah yang dibuat jadwalnya
18      * @param kelas kelas kuliah atau null jika tidak tersedia
19      * @param pengajar nama pengajar
20      * @param hariString hari dalam Bahasa Indonesia (Senin, Selasa, ...)
21      * @param waktuString rentang waktu kuliah (08.00-09.00 atau 08:00-09:00)
22      * @param lokasi kode ruangan
23      */
24     public JadwalKuliah(MataKuliah mataKuliah, Character kelas, Dosen pengajar, String hariString, String waktuString,
25         String lokasi) {
26         this.mataKuliah = mataKuliah;
27         this.kelas = kelas;
28         this.waktuMulai = LocalDateTime.parse(waktuString.substring(0, 5).replace('.', ':'));
29         this.waktuSelesai = LocalDateTime.parse(waktuString.substring(6, 11).replace('.', ':'));
30         this.lokasi = lokasi;
31         this.pengajar = pengajar;
32         this.hari = indonesianToDayOfWeek(hariString);
33     }
34
35     public JadwalKuliah() {
36         // void
37     }
38
39     public MataKuliah getMataKuliah() {
40         return mataKuliah;
41     }
42
43     public void setMataKuliah(MataKuliah mataKuliah) {
44         this.mataKuliah = mataKuliah;
45     }
46
47     public Character getKelas() {
48         return kelas;
49     }
50
51     public void setKelas(Character kelas) {
52         this.kelas = kelas;
53     }
54
55     public DayOfWeek getHari() {
56         return hari;
57     }
58
59     public void setHari(DayOfWeek hari) {
60         this.hari = hari;
61     }
62
63     public LocalDateTime getWaktuMulai() {
64         return waktuMulai;
65     }
66
67     public void setWaktuMulai(LocalTime waktuMulai) {
68         this.waktuMulai = waktuMulai;
69     }
70
71     public LocalDateTime getWaktuSelesai() {
72         return waktuSelesai;
73     }
74
75     public void setWaktuSelesai(LocalTime waktuSelesai) {
76         this.waktuSelesai = waktuSelesai;
77     }
78
79     public String getLokasi() {
80         return lokasi;
81     }

```

```

82 |
83 |     public void setLokasi(String lokasi) {
84 |         this.lokasi = lokasi;
85 |     }
86 |
87 |     public Dosen getPengajar() {
88 |         return pengajar;
89 |     }
90 |
91 |     public void setPengajar(Dosen pengajar) {
92 |         this.pengajar = pengajar;
93 |     }
94 |
95 |     public String getWaktuString() {
96 |         return waktuMulai + "-" + waktuSelesai;
97 |     }
98 |
99 |     /**
100 |      * Converts Indonesian day names to {@link DayOfWeek} enumeration.
101 |      * @param indonesian the day name in Indonesian
102 |      * @return {@link DayOfWeek} object or null if not found.
103 |      */
104 |     public static DayOfWeek indonesianToDayOfWeek(String indonesian) {
105 |         switch (indonesian.toLowerCase()) {
106 |             case "senin":
107 |                 return DayOfWeek.MONDAY;
108 |             case "selasa":
109 |                 return DayOfWeek.TUESDAY;
110 |             case "rabu":
111 |                 return DayOfWeek.WEDNESDAY;
112 |             case "kamis":
113 |                 return DayOfWeek.THURSDAY;
114 |             case "jumat":
115 |                 return DayOfWeek.FRIDAY;
116 |             case "sabtu":
117 |                 return DayOfWeek.SATURDAY;
118 |             case "minggu":
119 |                 return DayOfWeek.SUNDAY;
120 |             default:
121 |                 return null;
122 |         }
123 |     }
124 | }

```

Listing C.4: Mahasiswa.java

```

1 | package id.ac.unpar.siamodels;
2 |
3 | import java.net.URL;
4 | import java.time.LocalDate;
5 | import java.util.ArrayList;
6 | import java.util.Comparator;
7 | import java.util.HashMap;
8 | import java.util.HashSet;
9 | import java.util.List;
10 | import java.util.Map;
11 | import java.util.Set;
12 | import java.util.SortedMap;
13 | import java.util.TreeSet;
14 |
15 | public class Mahasiswa {
16 |     protected final String npm;
17 |     protected String nama;
18 |     protected final List<Nilai> riwayatNilai;
19 |     protected URL photoURL;
20 |     protected List<JadwalKuliah> jadwalKuliahList;
21 |     protected SortedMap<LocalDate, Integer> nilaiTOEFL;
22 |
23 |     public Mahasiswa(String npm) throws NumberFormatException {
24 |         super();
25 |         if (!npm.matches("[0-9]{10}")) {
26 |             throw new NumberFormatException("NPM_tidak_valid:_" + npm);
27 |         }
28 |         this.npm = npm;
29 |         this.riwayatNilai = new ArrayList<Nilai>();
30 |     }
31 |
32 |     public String getNama() {
33 |         return nama;
34 |     }
35 |
36 |     public void setNama(String nama) {
37 |         this.nama = nama;
38 |     }
39 |
40 |     public String getNpm() {
41 |         return npm;
42 |     }
43 |
44 |     public URL getPhotoURL() {
45 |         return photoURL;
46 |     }
47 |
48 |     public void setPhotoURL(URL photoURL) {
49 |         this.photoURL = photoURL;
50 |     }
51 |
52 |     public List<JadwalKuliah> getJadwalKuliahList() {

```

```

53     return jadwalKuliahList;
54 }
55
56 public void setJadwalKuliahList(List<JadwalKuliah> jadwalKuliahList) {
57     this.jadwalKuliahList = jadwalKuliahList;
58 }
59
60 public String getEmailAddress() {
61     return npm.substring(4, 6) + npm.substring(2, 4) + npm.substring(7, 10) + "@student.unpar.ac.id";
62 }
63
64 public List<Nilai> getRiwayatNilai() {
65     return riwayatNilai;
66 }
67
68 public SortedMap<LocalDate, Integer> getNilaiTOEFL(){
69     return nilaiTOEFL;
70 }
71
72 public void setNilaiTOEFL(SortedMap<LocalDate, Integer> nilaiTOEFL){
73     this.nilaiTOEFL = nilaiTOEFL;
74 }
75
76 /**
77  * Menghitung IPK mahasiswa sampai saat ini, dengan aturan:
78  * <ul>
79  * <li>Kuliah yang tidak lulus tidak dihitung
80  * <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
81  * </ul>
82  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
83  * @return IPK Lulus, atau {@link Double#NaN} jika belum mengambil satu kuliah pun.
84  * @deprecated Gunakan {@link #calculateIPLulus()}, nama lebih konsisten dengan DPS.
85  */
86 public double calculateIPKLulus() throws ArrayIndexOutOfBoundsException {
87     return calculateIPTempuh(true);
88 }
89
90 /**
91  * Menghitung IP mahasiswa sampai saat ini, dengan aturan:
92  * <ul>
93  * <li>Kuliah yang tidak lulus tidak dihitung
94  * <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
95  * </ul>
96  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
97  * @return IPK Lulus, atau {@link Double#NaN} jika belum mengambil satu kuliah pun.
98  */
99 public double calculateIPLulus() throws ArrayIndexOutOfBoundsException {
100     return calculateIPTempuh(true);
101 }
102
103 /**
104  * Menghitung IP mahasiswa sampai saat ini, dengan aturan:
105  * <ul>
106  * <li>Perhitungan kuliah yang tidak lulus ditentukan parameter
107  * <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
108  * </ul>
109  * @param lulusSaja set true jika ingin membuang mata kuliah tidak lulus, false jika ingin semua (sama dengan "IP N. Terbaik"
110  * di DPS)
111  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
112  * @return IPK Lulus, atau {@link Double#NaN} jika belum mengambil satu kuliah pun.
113  */
114 public double calculateIPTempuh(boolean lulusSaja) throws ArrayIndexOutOfBoundsException {
115     List<Nilai> riwayatNilai = getRiwayatNilai();
116     if (riwayatNilai.size() == 0) {
117         return Double.NaN;
118     }
119     Map<String, Double> nilaiTerbaik = new HashMap<String, Double>();
120     int totalSKS = 0;
121     // Cari nilai lulus terbaik setiap kuliah
122     for (Nilai nilai: riwayatNilai) {
123         if (nilai.getNilaiAkhir() == null) {
124             continue;
125         }
126         if (lulusSaja && nilai.getNilaiAkhir().equals('E')) {
127             continue;
128         }
129         String kodeMK = nilai.getMataKuliah().getKode();
130         Double angkaAkhir = nilai.getAngkaAkhir();
131         int sks = nilai.getMataKuliah().getSKS();
132         if (!nilaiTerbaik.containsKey(kodeMK)) {
133             totalSKS += sks;
134             nilaiTerbaik.put(kodeMK, sks * angkaAkhir);
135         } else if (sks * angkaAkhir > nilaiTerbaik.get(kodeMK)) {
136             nilaiTerbaik.put(kodeMK, sks * angkaAkhir);
137         }
138     }
139     // Hitung IPK dari nilai-nilai terbaik
140     double totalNilai = 0.0;
141     for (Double nilaiAkhir: nilaiTerbaik.values()) {
142         totalNilai += nilaiAkhir;
143     }
144     return totalNilai / totalSKS;
145 }
146
147 /**
148  * Menghitung IP Kumulatif mahasiswa sampai saat ini, dengan aturan:
149  * <ul>
150  * <li>Jika pengambilan beberapa kali, diambil semua.
151  * </ul>

```

```

151 * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
152 * @return IPK Lulus, atau {@link Double#NaN} jika belum mengambil satu kuliah pun.
153 */
154 public double calculateIPKumulatif() throws ArrayIndexOutOfBoundsException {
155     List<Nilai> riwayatNilai = getRiwayatNilai();
156     if (riwayatNilai.size() == 0) {
157         return Double.NaN;
158     }
159     double totalNilai = 0.0;
160     int totalSKS = 0;
161     // Cari nilai setiap kuliah
162     for (Nilai nilai: riwayatNilai) {
163         if (nilai.getNilaiAkhir() == null) {
164             continue;
165         }
166         Double angkaAkhir = nilai.getAngkaAkhir();
167         int sks = nilai.getMataKuliah().getSks();
168         totalSKS += sks;
169         totalNilai += sks * angkaAkhir;
170     }
171     return totalNilai / totalSKS;
172 }
173
174 /**
175  * Menghitung IPK mahasiswa sampai saat ini, dengan aturan:
176  * <ul>
177  * <li>Perhitungan kuliah yang tidak lulus ditentukan parameter
178  * <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
179  * </ul>
180  * @param lulusSaja set true jika ingin membuang mata kuliah tidak lulus
181  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
182  * @return IPK Lulus, atau {@link Double#NaN} jika belum mengambil satu kuliah pun.
183  * @deprecated Gunakan {@link #calculateIPKTempuh(boolean)}, nama lebih konsisten dengan DPS.
184  */
185 public double calculateIPKTempuh(boolean lulusSaja) throws ArrayIndexOutOfBoundsException {
186     return calculateIPKTempuh(lulusSaja);
187 }
188
189
190 /**
191  * Menghitung IPS semester terakhir sampai saat ini, dengan aturan:
192  * <ul>
193  * <li>Kuliah yang tidak lulus <em>dihitung</em>.
194  * </ul>
195  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
196  * @return nilai IPS sampai saat ini
197  * @throws ArrayIndexOutOfBoundsException jika belum ada nilai satupun
198  */
199 public double calculateIPS() throws ArrayIndexOutOfBoundsException {
200     List<Nilai> riwayatNilai = getRiwayatNilai();
201     if (riwayatNilai.size() == 0) {
202         throw new ArrayIndexOutOfBoundsException("Minimal_harus_ada_satu_nilai_untuk_menghitung_IPS");
203     }
204     int lastIndex = riwayatNilai.size() - 1;
205     TahunSemester tahunSemester = riwayatNilai.get(lastIndex).getTahunSemester();
206     double totalNilai = 0;
207     double totalSKS = 0;
208     for (int i = lastIndex; i >= 0; i--) {
209         Nilai nilai = riwayatNilai.get(i);
210         if (nilai.tahunSemester.equals(tahunSemester)) {
211             if (nilai.getAngkaAkhir() != null) {
212                 totalNilai += nilai.getMataKuliah().getSks() * nilai.getAngkaAkhir();
213                 totalSKS += nilai.getMataKuliah().getSks();
214             }
215         } else {
216             break;
217         }
218     }
219     return totalNilai / totalSKS;
220 }
221
222 /**
223  * Menghitung jumlah SKS lulus mahasiswa saat ini.
224  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
225  * @return SKS Lulus
226  */
227 public int calculateSKSLulus() throws ArrayIndexOutOfBoundsException {
228     return calculateSKSTempuh(true);
229 }
230
231 /**
232  * Menghitung jumlah SKS tempuh mahasiswa saat ini.
233  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
234  * @param lulusSaja set true jika ingin membuang SKS tidak lulus
235  * @return SKS tempuh
236  */
237 public int calculateSKSTempuh(boolean lulusSaja) throws ArrayIndexOutOfBoundsException {
238     List<Nilai> riwayatNilai = getRiwayatNilai();
239     Set<String> sksTerhitung = new HashSet<String>();
240     int totalSKS = 0;
241     // Tambahkan SKS setiap kuliah
242     for (Nilai nilai: riwayatNilai) {
243         if (nilai.getNilaiAkhir() == null) {
244             continue;
245         }
246         if (lulusSaja && nilai.getNilaiAkhir().equals('E')) {
247             continue;
248         }
249         String kodeMK = nilai.getMataKuliah().getKode();

```

```

250         if (!sksTerhitung.contains(kodeMK)) {
251             totalSKS += nilai.getMataKuliah().getSks();
252             sksTerhitung.add(kodeMK);
253         }
254     }
255     return totalSKS;
256 }
257
258 /**
259  * Mendapatkan seluruh tahun semester di mana mahasiswa ini tercatat
260  * sebagai mahasiswa aktif, dengan strategi memeriksa riwayat nilainya.
261  * Jika ada satu nilai saja pada sebuah tahun semester, maka dianggap
262  * aktif pada semester tersebut.
263  * @return kumpulan tahun semester di mana mahasiswa ini aktif
264  */
265 public Set<TahunSemester> calculateTahunSemesterAktif() {
266     Set<TahunSemester> tahunSemesterAktif = new TreeSet<TahunSemester>();
267     List<Nilai> riwayatNilai = getRiwayatNilai();
268     for (Nilai nilai: riwayatNilai) {
269         tahunSemesterAktif.add(nilai.getTahunSemester());
270     }
271     return tahunSemesterAktif;
272 }
273
274 /**
275  * Periksa apakah mahasiswa ini sudah lulus mata kuliah tertentu. Kompleksitas O(n).
276  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
277  * Note: jika yang dimiliki adalah {@link MataKuliah}, gunakanlah {@link MataKuliah#getKode()}.
278  * @param kodeMataKuliah kode mata kuliah yang ingin diperiksa kelulusannya.
279  * @return true jika sudah pernah mengambil dan lulus, false jika belum
280  */
281 public boolean hasLulusKuliah(String kodeMataKuliah) {
282     for (Nilai nilai: riwayatNilai) {
283         if (nilai.getMataKuliah().getKode().equals(kodeMataKuliah)) {
284             Character na = nilai.getNilaiAkhir();
285             if (na != null && na >= 'A' && na <= 'D') {
286                 return true;
287             }
288         }
289     }
290     return false;
291 }
292
293 /**
294  * Periksa apakah mahasiswa ini sudah pernah menempuh mata kuliah tertentu. Kompleksitas O(n).
295  * Sebelum memanggil method ini, {@link #getRiwayatNilai()} harus sudah ada isinya!
296  * Note: jika yang dimiliki adalah {@link MataKuliah}, gunakanlah {@link MataKuliah#getKode()}.
297  * @param kodeMataKuliah kode mata kuliah yang ingin diperiksa.
298  * @return true jika sudah pernah mengambil, false jika belum
299  */
300 public boolean hasTempuhKuliah(String kodeMataKuliah) {
301     for (Nilai nilai: riwayatNilai) {
302         if (nilai.getMataKuliah().getKode().equals(kodeMataKuliah)) {
303             return true;
304         }
305     }
306     return false;
307 }
308
309 /**
310  * Mendapatkan tahun angkatan mahasiswa ini, berdasarkan NPM nya
311  * @return tahun angkatan
312  */
313 public int getTahunAngkatan() {
314     return Integer.parseInt(getNpm().substring(0, 4));
315 }
316
317 @Override
318 public String toString() {
319     return "Mahasiswa_[npm=" + npm + ",nama=" + nama + "]";
320 }
321
322 /**
323  * Merepresentasikan nilai yang ada di riwayat nilai mahasiswa
324  * @author pascal
325  */
326
327 public static class Nilai {
328     /** Tahun dan Semester kuliah ini diambil */
329     protected final TahunSemester tahunSemester;
330     /** Mata kuliah yang diambil */
331     protected final MataKuliah mataKuliah;
332     /** Kelas kuliah */
333     protected final Character kelas;
334     /** Nilai ART */
335     protected final Double nilaiART;
336     /** Nilai UTS */
337     protected final Double nilaiUTS;
338     /** Nilai UAS */
339     protected final Double nilaiUAS;
340     /** Nilai Akhir */
341     protected final Character nilaiAkhir;
342
343     public Nilai(TahunSemester tahunSemester, MataKuliah mataKuliah,
344                 Character kelas, Double nilaiART, Double nilaiUTS, Double nilaiUAS,
345                 Character nilaiAkhir) {
346         super();
347         this.tahunSemester = tahunSemester;
348         this.mataKuliah = mataKuliah;

```

```

349     this.kelas = kelas;
350     this.nilaiART = nilaiART;
351     this.nilaiUTS = nilaiUTS;
352     this.nilaiUAS = nilaiUAS;
353     this.nilaiAkhir = nilaiAkhir;
354 }
355
356 public MataKuliah getMataKuliah() {
357     return mataKuliah;
358 }
359
360 public Character getKelas() {
361     return kelas;
362 }
363
364 public Double getNilaiART() {
365     return nilaiART;
366 }
367
368 public Double getNilaiUTS() {
369     return nilaiUTS;
370 }
371
372 public Double getNilaiUAS() {
373     return nilaiUAS;
374 }
375
376 /**
377  * Mengembalikan nilai akhir dalam bentuk huruf (A, B, C, D, ..., atau K)
378  * @return nilai akhir dalam huruf, atau null jika tidak ada.
379  */
380 public Character getNilaiAkhir() {
381     return nilaiAkhir;
382 }
383
384 /**
385  * Mendapatkan nilai akhir dalam bentuk angka
386  * @return nilai akhir dalam angka, atau null jika {@link #getNilaiAkhir()} mengembalikan 'K' atau null}
387  */
388 public Double getAngkaAkhir() {
389     if (nilaiAkhir == null) {
390         return null;
391     }
392     switch (nilaiAkhir) {
393         case 'A':
394             return 4.0;
395         case 'B':
396             return 3.0;
397         case 'C':
398             return 2.0;
399         case 'D':
400             return 1.0;
401         case 'E':
402             return 0.0;
403         case 'K':
404             return null;
405     }
406     return null;
407 }
408
409 public TahunSemester getTahunSemester() {
410     return tahunSemester;
411 }
412
413 public int getTahunAjaran() {
414     return tahunSemester.getTahun();
415 }
416
417 public Semester getSemester() {
418     return tahunSemester.getSemester();
419 }
420
421 @Override
422 public String toString() {
423     return "Nilai_["tahunSemester=" + tahunSemester + ",mataKuliah=" + mataKuliah + ",kelas=" + kelas
424         + ",nilaiART=" + nilaiART + ",nilaiUTS=" + nilaiUTS + ",nilaiUAS=" + nilaiUAS + ",nilaiAkhir="
425         + nilaiAkhir + "]\n";
426 }
427
428 /**
429  * Pembanding antara satu nilai dengan nilai lainnya, secara
430  * kronologis waktu pengambilan.
431  * @author pascal
432  */
433
434 public static class ChronologicalComparator implements Comparator<Nilai> {
435
436     @Override
437     public int compare(Nilai o1, Nilai o2) {
438         return o1.getTahunSemester().compareTo(o2.getTahunSemester());
439     }
440 }
441 }
442 }

```

Listing C.5: MataKuliah.java

1 | `package id.ac.unpar.siamodels;`

```

2
3 public abstract class MataKuliah {
4     private final String kode;
5     private final String nama;
6     private final Integer sks;
7
8     public MataKuliah() {
9         this.kode = this.getClass().getSimpleName();
10        if (this.getClass().isAnnotationPresent(InfoMataKuliah.class)) {
11            InfoMataKuliah infoMK = (InfoMataKuliah) this.getClass().getAnnotation(InfoMataKuliah.class);
12            this.nama = infoMK.nama();
13            this.sks = infoMK.sks();
14        } else {
15            this.nama = null;
16            this.sks = null;
17        }
18    }
19
20    public MataKuliah(String kode, String nama, int sks) {
21        this.kode = kode;
22        this.nama = nama;
23        this.sks = sks;
24    }
25
26    public String getKode() {
27        return kode;
28    }
29
30    public String getNama() {
31        return nama;
32    }
33
34    public Integer getSks() {
35        return sks;
36    }
37 }

```

Listing C.6: MataKuliahFactory.java

```

1 package id.ac.unpar.siamodels;
2
3 import java.lang.annotation.Annotation;
4 import java.util.SortedMap;
5 import java.util.TreeMap;
6 import java.util.logging.Logger;
7
8 /**
9  * Kelas yang bertugas membuat kelas mata kuliah, dan menyimpannya untuk bisa
10  * digunakan kemudian (untuk hemat memori).
11  *
12  * @author pascal
13  */
14
15 public class MataKuliahFactory {
16
17     /**
18      * Lokasi package untuk daftar mata kuliah
19      */
20     public static String DEFAULT_MATAKULIAH_PACKAGE = "id.ac.unpar.siamodels.matakuliah";
21
22     /**
23      * Singleton instance to factory.
24      */
25     private static MataKuliahFactory instance = null;
26
27     /**
28      * Singleton instances untuk mata kuliah.
29      */
30     private final SortedMap<String, MataKuliah> mataKuliahCache;
31
32     public static MataKuliahFactory getInstance() {
33         if (instance == null) {
34             instance = new MataKuliahFactory();
35         }
36         return instance;
37     }
38
39     protected MataKuliahFactory() {
40         this.mataKuliahCache = new TreeMap<>();
41     }
42
43     /**
44      * Membuat baru atau mendapatkan mata kuliah, jika memiliki informasi
45      * nama dan jumlah SKS.
46      *
47      * @param kode
48      *         kode mata kuliah
49      * @param sks
50      *         jumlah SKS
51      * @param nama
52      *         nama mata kuliah
53      * @return objek mata kuliah
54      */
55     public MataKuliah createMataKuliah(String kode, int sks, String nama) {
56         MataKuliah mk = this.mataKuliahCache.get(kode);
57         // Coba dapatkan mata kuliah dari cache
58         if (mk != null) {
59             // Update jika kita punya info lebih baik

```



```

60         if (mk.getSks() == null || mk.getNama() == null) {
61             mk = new MataKuliah(kode, nama, sks) {};
62             this.mataKuliahCache.put(kode, mk);
63         }
64         return mk;
65     }
66
67     // Coba dapatkan dari kelas statik
68     Class<?> mkClass;
69     try {
70         mkClass = Class.forName(DEFAULT_MATAKULIAH_PACKAGE + "." + kode);
71         mk = (MataKuliah) mkClass.newInstance();
72     } catch (ClassNotFoundException e) {
73         mk = new MataKuliah(kode, nama, sks) {};
74         Logger.getGlobal().warning("Class_is_not_listed:_" + String.format("%s-%d_%s", kode, sks, nama));
75     } catch (InstantiationException e) {
76         Logger.getGlobal().warning("Internal_error:_" + e.getMessage());
77         e.printStackTrace();
78     } catch (IllegalAccessException e) {
79         Logger.getGlobal().warning("Internal_error:_" + e.getMessage());
80         e.printStackTrace();
81     }
82     this.mataKuliahCache.put(kode, mk);
83     return mk;
84 }
85
86 /**
87  * Membuat baru atau mendapatkan mata kuliah, jika tidak memiliki informasi
88  * nama dan jumlah SKS.
89  *
90  * @param kode
91  *      kode mata kuliah
92  * @return objek mata kuliah
93  * @throws IllegalStateException
94  *      jika sks dan tidak sesuai dengan yang ada di kode
95  */
96 public MataKuliah createMataKuliah(final String kode)
97     throws IllegalStateException {
98     MataKuliah mk = this.mataKuliahCache.get(kode);
99     // Coba dapatkan mata kuliah dari cache
100    if (mk != null) {
101        return mk;
102    }
103
104    // Coba dapatkan dari kelas statik
105    Class<?> mkClass;
106    try {
107        mkClass = Class.forName(DEFAULT_MATAKULIAH_PACKAGE + "." + kode);
108        mk = (MataKuliah) mkClass.newInstance();
109    } catch (ClassNotFoundException e) {
110        throw new IllegalStateException("Mata_kuliah_" + kode + "_tidak_ditemukan!");
111    } catch (InstantiationException e) {
112        Logger.getGlobal().warning("Internal_error:_" + e.getMessage());
113        e.printStackTrace();
114    } catch (IllegalAccessException e) {
115        Logger.getGlobal().warning("Internal_error:_" + e.getMessage());
116        e.printStackTrace();
117    }
118    this.mataKuliahCache.put(kode, mk);
119    return mk;
120 }
121
122 }

```

Listing C.7: Semester.java

```

1 package id.ac.unpar.siamodels;
2
3 public enum Semester {
4     UNKNOWN5(5), TRANSFER(6), PENDEK(10), GANJIL(20), GENAP(30);
5
6     public static Semester fromString(String text) {
7         return Semester.valueOf(text.toUpperCase());
8     }
9
10    private int order;
11
12    private Semester(int order) {
13        this.order = order;
14    }
15
16    int getOrder() {
17        return order;
18    }
19
20 }

```

Listing C.8: TahunSemester.java

```

1 package id.ac.unpar.siamodels;
2
3 /**
4  * Menyimpan konstanta untuk semester beserta tahunnya di UNPAR.
5  * @author pascal
6  *
7  */

```

```

8 public final class TahunSemester implements Comparable<TahunSemester> {
9
10     /**
11      * Kode semester 3 digit, sesuai DPS:
12      * <ul>
13      * <li>2 digit pertama berupa tahun, 2 digit terakhir</li>
14      * <li>digit terakhir: 1 untuk ganjil, 2 untuk genap, 4 untuk pendek.
15      * </ul>
16      */
17     private String kodeTahunSemester;
18
19     public TahunSemester(String kodeTahunSemester) throws IllegalArgumentException {
20         validateKodeSemester(kodeTahunSemester);
21         this.kodeTahunSemester = kodeTahunSemester;
22     }
23
24     public TahunSemester(int tahun, Semester semester) throws IllegalArgumentException {
25         char kodeSemester = '\0';
26         switch (semester) {
27             case GANJIL:
28                 kodeSemester = '1';
29                 break;
30             case GENAP:
31                 kodeSemester = '2';
32                 break;
33             case PENDEK:
34                 kodeSemester = '4';
35                 break;
36             case UNKNOWN5:
37                 kodeSemester = '5';
38                 break;
39             case TRANSFER:
40                 kodeSemester = '6';
41                 break;
42             default:
43                 throw new InternalError("Incomplete_TahunSemester_switch_case");
44         }
45         String kodeTahunSemester = (" " + tahun).substring(2, 4) + kodeSemester;
46         validateKodeSemester(kodeTahunSemester);
47         this.kodeTahunSemester = kodeTahunSemester;
48     }
49
50     public Semester getSemester() {
51         switch (kodeTahunSemester.charAt(2)) {
52             case '1': return Semester.GANJIL;
53             case '2': return Semester.GENAP;
54             case '4': return Semester.PENDEK;
55             case '5': return Semester.UNKNOWN5;
56             case '6': return Semester.TRANSFER;
57         }
58         return null;
59     }
60
61     public int getTahun() {
62         return 2000 + Integer.parseInt(kodeTahunSemester.substring(0, 2));
63     }
64
65     private static void validateKodeSemester(String kodeTahunSemester) throws IllegalArgumentException {
66         if (!kodeTahunSemester.matches("[0-9][0-9][12456]")) {
67             throw new IllegalArgumentException("Kode_semester_tidak_valid:_" + kodeTahunSemester);
68         }
69     }
70
71     /**
72      * Mendapatkan kode tahun/semester sesuai aturan di DPS.
73      * @return kode tahun/semester sesuai aturan di DPS.
74      */
75     public String getKodeDPS() {
76         return kodeTahunSemester;
77     }
78
79     @Override
80     public int compareTo(TahunSemester o) {
81         if (this.getTahun() < o.getTahun()) {
82             return -1;
83         }
84         if (this.getTahun() > o.getTahun()) {
85             return +1;
86         }
87         if (this.getSemester().getOrder() < o.getSemester().getOrder()) {
88             return -1;
89         }
90         if (this.getSemester().getOrder() > o.getSemester().getOrder()) {
91             return +1;
92         }
93         return 0;
94     }
95
96     @Override
97     public boolean equals(Object obj) {
98         if (!(obj instanceof TahunSemester)) {
99             return false;
100         }
101         return this.compareTo((TahunSemester)obj) == 0;
102     }
103
104     @Override
105     public String toString() {
106         return "TahunSemester_" + getTahun() + "/" + getSemester() + ";";

```

```

107 |     }
108 |
109 |
110 | }

```

Listing C.9: HasPraktikum.java

```

1 | /*
2 |  * To change this license header, choose License Headers in Project Properties.
3 |  * To change this template file, choose Tools | Templates
4 |  * and open the template in the editor.
5 |  */
6 | package id.ac.unpar.siamodels.matakuliah.interfaces;
7 |
8 | /**
9 |  *
10 |  * @author FTIS\i13037
11 |  */
12 | public interface HasPraktikum {
13 |
14 | }

```

Listing C.10: HasPrasyarat.java

```

1 | package id.ac.unpar.siamodels.matakuliah.interfaces;
2 |
3 | import java.util.List;
4 |
5 | import id.ac.unpar.siamodels.Mahasiswa;
6 |
7 | /**
8 |  * Mendefinisikan kelas-kelas yang memiliki prasyarat, terkustomisasi
9 |  * untuk seorang {@link Mahasiswa}. Jika ada tambahan, jangan lupa untuk
10 |  * mendaftarkannya di {@link #DEFAULT_HASPRASYARAT_CLASSES}. Jika berubah package,
11 |  * jangan lupa update {@link #DEFAULT_PRASYARAT_PACKAGE}.
12 |  * @author pascal
13 |  */
14 |
15 | public interface HasPrasyarat {
16 |
17 |     /**
18 |      * Daftar dari nama kelas default seluruh turunan interface ini. Perlu didaftarkan
19 |      * manual, karena Java reflection tidak dapat mendeteksi otomatis.
20 |      */
21 |     public String[] DEFAULT_HASPRASYARAT_CLASSES = { "AIF102", "AIF202",
22 | "AIF204", "AIF206", "AIF208", "AIF302", "AIF304", "AIF306",
23 | "AIF312", "AIF314", "AIF316", "AIF318", "AIF332", "AIF339",
24 | "AIF342", "AIF344", "AIF360", "AIF401", "AIF402", "AIF445",
25 | "AIF458", "AIF461", "APS402" };
26 |
27 |     /**
28 |      * Package tempat menyimpan seluruh turunan standar interface ini. Perlu didefinisikan
29 |      * manual, karena Java reflection tidak dapat mendeteksi otomatis.
30 |      */
31 |     public String DEFAULT_PRASYARAT_PACKAGE = "id.ac.unpar.siamodels.matakuliah";
32 |
33 |     /**
34 |      * Memeriksa prasyarat-prasyarat dari kuliah, spesifik untuk mahasiswa
35 |      * yang dituju. Jika ada pesan-pesan khusus, akan ditambahkan pada parameter
36 |      * reasonsContainer.
37 |      * @param mahasiswa prasyarat kuliah akan diperiksa spesifik pada mahasiswa ini
38 |      * @param reasonsContainer pesan-pesan terkait prasyarat akan ditambahkan di sini, jika ada.
39 |      * @return true jika seluruh prasyarat dipenuhi, false jika tidak.
40 |      */
41 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer);
42 | }

```

Listing C.11: HasResponsi.java

```

1 | /*
2 |  * To change this license header, choose License Headers in Project Properties.
3 |  * To change this template file, choose Tools | Templates
4 |  * and open the template in the editor.
5 |  */
6 | package id.ac.unpar.siamodels.matakuliah.interfaces;
7 |
8 | /**
9 |  *
10 |  * @author FTIS\i13037
11 |  */
12 | public interface HasResponsi {
13 |
14 | }

```

Listing C.12: Kelulusan.java

```

1 | package id.ac.unpar.siamodels.prodi.teknikinformatika;
2 |
3 | import java.time.LocalDate;
4 | import java.util.Arrays;
5 | import java.util.Iterator;
6 | import java.util.List;

```

```

7 import java.util.Map;
8 import java.util.Set;
9 import java.util.SortedMap;
10
11 import id.ac.unpar.siamodels.Mahasiswa;
12 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
13 import java.util.Collection;
14
15 public class Kelulusan implements HasPrasyarat {
16
17     public static final String[] PILIHAN_WAJIB = {"AIF311", "AIF312", "AIF313", "AIF314", "AIF315", "AIF316",
18         "AIF317"};
19     public static final String[][] WAJIB = {{ "AIF101", "AIF103", "AIF105", "MKU001", "MKU008", "MKU010"},
20         {"AIF102", "AIF104", "AIF106", "AMS100", "MKU009", "MKU011"},
21         {"AIF210", "AIF203", "AIF205", "AMS200", "MKU012", {"AIF202", "AIF204", "AIF206", "AIF208", "AIF210"}},
22         {"AIF301", "AIF303", "AIF305", "MKU002", {"AIF302"},
23         {"AIF401", "AIF403"}, {"AIF402", "APS402"}}};
24     public static final String[] AGAMA = {"MKU003", "MKU004"};
25
26     public static final int MIN_SKS_LULUS = 144;
27
28     public static final int MIN_PILIHAN_WAJIB = 4;
29
30     @Override
31     /**
32      * Melakukan pengecekan syarat kelulusan
33      *
34      * @param mahasiswa mahasiswa yang dicek
35      * @param reasonsContainer alasan2 yang ada jika tidak lulus
36      * @return boolean true jika memenuhi syarat, false jika sebaliknya
37      */
38     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
39         // cek sks
40         boolean bisaLulus = true;
41         int sks = mahasiswa.calculateSKSLulus();
42         if (sks < MIN_SKS_LULUS) {
43             reasonsContainer.add(String.format("Anda_baru_lulus_%d_SKS,_masih_kurang_%d_SKS_lagi_untuk_mencapai_%d.", sks,
44                 MIN_SKS_LULUS - sks, MIN_SKS_LULUS));
45             bisaLulus = false;
46         }
47         // cek agama
48         boolean lulusSalahSatuMKAgama = false;
49         for (int i = 0; i < AGAMA.length; i++) {
50             if (mahasiswa.hasLulusKuliah(AGAMA[i])) {
51                 lulusSalahSatuMKAgama = true;
52                 break;
53             }
54         }
55         if (!lulusSalahSatuMKAgama) {
56             reasonsContainer.add("Anda_belum_lulus_salah_satu_dari_MK_Agama_" + Arrays.toString(AGAMA));
57             bisaLulus = false;
58         }
59         // cek kuliah wajib
60         for (String[] mkWajibSemesterI : WAJIB) {
61             for (String mkWajib : mkWajibSemesterI) {
62                 if (!mahasiswa.hasLulusKuliah(mkWajib)) {
63                     reasonsContainer.add("Anda_belum_lulus_MK_wajib_" + mkWajib);
64                     bisaLulus = false;
65                 }
66             }
67         }
68         // cek pilihan wajib
69         int lulusPilihanWajib = 0;
70         for (String mkPilihanWajib : PILIHAN_WAJIB) {
71             if (mahasiswa.hasLulusKuliah(mkPilihanWajib)) {
72                 lulusPilihanWajib++;
73             }
74         }
75         if (lulusPilihanWajib < MIN_PILIHAN_WAJIB) {
76             reasonsContainer.add(String.format("Anda_baru_lulus_%d_MK_pilihan_wajib,_sedangkan_Andaperlu_lulus_%d",
77                 lulusPilihanWajib, MIN_PILIHAN_WAJIB));
78             bisaLulus = false;
79         }
80         // cek proyek
81         if (!mahasiswa.hasLulusKuliah("AIF306") && !mahasiswa.hasLulusKuliah("AIF405")) {
82             reasonsContainer.add("Anda_belum_lulus_salah_satu_dari_MK_Projek_AIF306_atau_AIF304_&_AIF405");
83             bisaLulus = false;
84         }
85         // cek nilai TOEFL
86         SortedMap<LocalDate, Integer> toeflScore = mahasiswa.getNilaiTOEFL();
87         if (toeflScore == null) {
88             reasonsContainer.add("Belum_ada_skor_TOEFL.");
89             bisaLulus = false;
90         }
91         Collection nilai = toeflScore.values();
92         int maxToefl = 0;
93         Iterator i = nilai.iterator();
94         while (i.hasNext()) {
95             int val = (int) i.next();
96             if (maxToefl < val) {
97                 maxToefl = val;
98             }
99         }
100         if (!(maxToefl >= 500)) {
101             if (toeflScore.size() <= 8) {
102                 reasonsContainer.add("Belum_mencapai_nilai_TOEFL_sebesar_500.");
103                 bisaLulus = false;
104             } else {
105                 if (maxToefl < 450) {

```

```

104         reasonsContainer.add("Belum_mencapai_nilai_TOEFL_sebesar_450.");
105         bisaLulus = false;
106     } else {
107         reasonsContainer.add("Sudah_lulus_TOEFL_dengan_nilai_" + maxToefl + "_dan_memerlukan_dispensasi_dari_rektor_
108             karena_sudah_mengambil_tes_TOEFL_sebanyak_" + toeflScore.size() + "_kali.");
109     }
110 }
111 return bisaLulus;
112 }
113 }
114 }

```

Listing C.13: AIF101.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
6
7
8 /**
9  * Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar pemrograman
10 * seperti pengulangan dan percabangan, konsep dasar penyimpanan data kontigu
11 * menggunakan array, konsep dasar pemrograman berorientasi objek seperti kelas
12 * & objek, method, dll, termasuk di dalamnya 4 prinsip dasar pemrograman
13 * berorientasi objek : data abstraction, encapsulation, inheritance dan
14 * polymorphism. Selain itu diberikan masalah-masalah komputasi sederhana
15 * yang harus diselesaikan menggunakan konsep-konsep yang sudah diperkenalkan
16 * dan mengimplementasikannya menggunakan bahasa pemrograman Java
17 * @author Lionov, M.Sc. (lionov@unpar.ac.id)
18 */
19 @InfoMataKuliah(nama = "Pemrograman_Berorientasi_Objek", sks = 6)
20 public class AIF101 extends MataKuliah implements HasPraktikum {
21
22 }

```

Listing C.14: AIF102.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
4 import id.ac.unpar.siamodels.Mahasiswa;
5 import id.ac.unpar.siamodels.MataKuliah;
6 import id.ac.unpar.siamodels.InfoMataKuliah;
7 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
8
9 import java.util.List;
10
11 /**
12 * Mata kuliah ini memperkenalkan berbagai algoritma dan teknik-teknik
13 * penyelesaian masalah komputasi seperti rekursif, sorting, teknik divide dan
14 * conquer, serta exhaustive search. Selain itu, pada kuliah ini juga
15 * dikenalkan berbagai struktur data yang dapat digunakan untuk mendukung
16 * penyelesaian masalah komputasi seperti ADT List, Stack dan Queue. Baik
17 * algoritma maupun struktur data yang dikenalkan harus dapat diimplementasikan
18 * dan digunakan oleh mahasiswa untuk menyelesaikan masalah dengan menggunakan
19 * suatu bahasa pemrograman berorientasi objek.
20 * @author husnulhakim@unpar.ac.id
21 */
22 @InfoMataKuliah(nama = "Algoritma_dan_Struktur_Data", sks = 4)
23 public class AIF102 extends MataKuliah implements HasPrasyarat, HasPraktikum {
24
25     @Override
26     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
27         boolean ok = true;
28         if (!mahasiswa.hasLulusKuliah("AIF101") && !mahasiswa.hasLulusKuliah("AIF191")) {
29             reasonsContainer.add("Tidak_memenuhi_prasyarat_lulus_AIF101_atau_AIF191");
30             ok = false;
31         }
32         if (!mahasiswa.hasTempuhKuliah("AIF103")) {
33             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF103");
34             ok = false;
35         }
36         return ok;
37     }
38 }
39 }

```

Listing C.15: AIF103.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Mata kuliah ini merupakan salah satu cara untuk mencapai kompetensi dasar
8  * tentang matematika diskrit yang prinsipnya banyak digunakan dalam bidang
9  * ilmu komputer. Selain itu, kuliah ini juga merupakan cara untuk membentuk
10 * pola pikir logis yang dibutuhkan untuk menempuh kuliah-kuliah di tingkat
11 * yang lebih tinggi.
12 * @author Mariskha Tri Adithia (mariskha@unpar.ac.id)

```

```

13  */
14  @InfoMataKuliah(nama = "Matematika_Diskrit", sks = 3)
15  public class AIF103 extends MataKuliah{
16
17  }

```

Listing C.16: AIF104.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5
6  /**
7   * Mata kuliah ini memberikan pengetahuan tentang logika yang digunakan di
8   * dalam ilmu komputer. Dalam kuliah ini, mahasiswa belajar untuk bisa
9   * memodelkan suatu kalimat dalam kehidupan sehari-hari, ke dalam kalimat
10  * dengan sintaks tertentu, yang hanya memiliki satu arti. Lalu, diperkenalkan
11  * juga, bagaimana mengartikan suatu kalimat (benar atau salah) dan bagaimana
12  * menentukan sifat dari kalimat tersebut.
13  * @author Mariskha Tri Adithia (mariskha@unpar.ac.id)
14  */
15  @InfoMataKuliah(nama = "Logika_Informatika", sks = 3)
16  public class AIF104 extends MataKuliah{
17
18
19  }

```

Listing C.17: AIF105.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5
6  /**
7   * Mata kuliah ini memperkenalkan kepada mahasiswa terminologi dan konsep dasar
8   * yang akan banyak dipakai sepanjang kuliah di Teknik Informatika. Selain itu
9   * mata kuliah ini juga mempersiapkan dan membiasakan mahasiswa dengan suasana
10  * akademik yang khas perguruan tinggi seperti kedisiplinan, kerja sama,
11  * kemampuan menggunakan teknologi informasi dalam pembuatan tugas, kemampuan
12  * komunikasi, dsb.
13  * @author Thomas Anung Basuki (anung@unpar.ac.id)
14  * @author Mariskha Tri Adithia (mariskha@unpar.ac.id)
15  */
16  @InfoMataKuliah(nama = "Pengantar_Informatika", sks = 3)
17  public class AIF105 extends MataKuliah{
18
19
20  }

```

Listing C.18: AIF106.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5
6  /**
7   * Mata kuliah ini memberikan pengetahuan tentang cara kerja komputer, dimulai
8   * dari representasi data dan berbagai macam operasinya. Selanjutnya, juga
9   * diperkenalkan bagaimana merepresentasikan suatu fungsi dalam rangkaian
10  * gerbang logika, dan bagaimana menyederhanakannya. Berbagai rangkaian dasar
11  * yang digunakan di dalam komputer juga dipekenalkan. Mahasiswa juga akan
12  * mempelajari komponen komputer, misalnya register dan memori.
13  * @author Mariskha Tri Adithia (mariskha@unpar.ac.id)
14  */
15  @InfoMataKuliah(nama = "Sistem_Dijital", sks = 3)
16  public class AIF106 extends MataKuliah{
17
18  }

```

Listing C.19: AIF181.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5
6  @InfoMataKuliah(nama = "Dasar-dasar_Pemrograman", sks = 3)
7  public class AIF181 extends MataKuliah {
8
9  }

```

Listing C.20: AIF182.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;

```

```

5 |
6 | @InfoMataKuliah(nama = "Pengantar_Basis_Data", sks = 2)
7 | public class AIF182 extends MataKuliah {
8 |
9 | }

```

Listing C.21: AIF183.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | @InfoMataKuliah(nama = "Pemrograman_Prosedural", sks = 4)
7 | public class AIF183 extends MataKuliah {
8 |
9 | }

```

Listing C.22: AIF191.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | /**
7 |  * @deprecated Mata kuliah tidak dibuka lagi.
8 |  */
9 | @InfoMataKuliah(nama = "Pemrograman_Berorientasi_Objek", sks = 3)
10 | public class AIF191 extends MataKuliah {
11 |
12 | }

```

Listing C.23: AIF192.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | /**
7 |  * @deprecated Mata kuliah tidak dibuka lagi.
8 |  */
9 | @InfoMataKuliah(nama = "Algoritma_&Struktur_Data", sks = 3)
10 | public class AIF192 extends MataKuliah {
11 |
12 | }

```

Listing C.24: AIF201.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
4 | import id.ac.unpar.siamodels.Mahasiswa;
5 | import id.ac.unpar.siamodels.MataKuliah;
6 | import id.ac.unpar.siamodels.InfoMataKuliah;
7 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
8 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasResponsi;
9 |
10 | import java.util.List;
11 |
12 | /**
13 |  * Mata kuliah ini memperkenalkan prinsip-prinsip yang digunakan dalam
14 |  * melakukan analisa serta desain program berorientasi objek. Di samping itu,
15 |  * mahasiswa juga belajar menggunakan kaskas berupa diagram UML (Unified
16 |  * Modelling Language) sehingga dapat mengkomunikasikan desain secara visual.
17 |  * Mahasiswa juga akan mengenal beberapa software design pattern dari Gang of
18 |  * Four. Terakhir, mahasiswa akan belajar mengenai konsep MVC
19 |  * (Model-View-Controller) yang menjadi dasar dari banyak framework masa kini.
20 |  * Bahasa yang digunakan adalah bahasa Java, namun diusahakan tetap umum
21 |  * sehingga dapat diaplikasikan pada bahasa yang lain.
22 |  * @author Pascal (pascal@unpar.ac.id)
23 |  */
24 | @InfoMataKuliah(nama = "Analisis_&Desain_Berorientasi_Objek", sks = 4)
25 | public class AIF201 extends MataKuliah implements HasPrasyarat, HasPraktikum, HasResponsi {
26 |
27 |     @Override
28 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
29 |         boolean ok = true;
30 |         if (!mahasiswa.hasLulusKuliah("AIF101") && !mahasiswa.hasLulusKuliah("AIF191")) {
31 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_lulus_AIF101_atau_AIF191");
32 |             ok = false;
33 |         }
34 |         return ok;
35 |     }
36 |
37 | }

```

Listing C.25: AIF202.java

```

1 | package id.ac.unpar.siamodels.matakuliah;

```

```

2
3 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
4 import id.ac.unpar.siamodels.Mahasiswa;
5 import id.ac.unpar.siamodels.MataKuliah;
6 import id.ac.unpar.siamodels.InfoMataKuliah;
7 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
8 import id.ac.unpar.siamodels.matakuliah.interfaces.HasResponsi;
9
10
11 import java.util.List;
12
13 /**
14  * Mata kuliah ini memperkenalkan kepada mahasiswa beberapa algoritma dan
15  * struktur data, alternatif cara implementasinya, dan analisis kompleksitas
16  * waktunya. Mahasiswa diberikan beberapa masalah komputasi yang harus
17  * diselesaikan dengan menggunakan algoritma atau struktur data yang sudah
18  * diperkenalkan dan mengimplementasikannya dalam bahasa pemrograman Java.
19  * @author Joanna Helga, M.Sc. (joanna@unpar.ac.id)
20  */
21 @InfoMataKuliah(nama = "Desain_& Analisis_Algoritma", sks = 4)
22 public class AIF202 extends MataKuliah implements HasPrasyarat, HasResponsi, HasPraktikum {
23
24     @Override
25     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
26         boolean ok = true;
27         if (!mahasiswa.hasLulusKuliah("AIF102") && !mahasiswa.hasLulusKuliah("AIF192")) {
28             reasonsContainer.add("Tidak_menuhi_prasyarat_lulus_AIF102_atau_AIF192");
29             ok = false;
30         }
31         int angkatan = mahasiswa.getTahunAngkatan();
32         if (angkatan >= 2012 && angkatan <= 2014) {
33             if (!mahasiswa.hasTempuhKuliah("AIF203")) {
34                 reasonsContainer.add("Angkatan_" + angkatan + "_tetapi_tidak_menuhi_prasyarat_tempuh_AIF103");
35                 ok = false;
36             }
37         }
38         return ok;
39     }
40 }
41

```

Listing C.26: AIF203.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 /**
11  * Mata kuliah ini memperkenalkan kepada mahasiswa konsep struktur diskret yang
12  * digunakan pada bidang informatika diantaranya graph, pohon dan finite state
13  * machine
14  * @author Dr.rer.nat Cecilia Esti Nugraheni (cheni@unpar.ac.id)
15  */
16 @InfoMataKuliah(nama = "Struktur_Diskrit", sks = 4)
17 public class AIF203 extends MataKuliah implements HasPrasyarat {
18
19     @Override
20     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
21         boolean ok = true;
22         if (!mahasiswa.hasTempuhKuliah("AIF103")) {
23             reasonsContainer.add("Tidak_menuhi_prasyarat_tempuh_AIF103");
24             ok = false;
25         }
26         return ok;
27     }
28 }
29

```

Listing C.27: AIF204.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
4 import id.ac.unpar.siamodels.Mahasiswa;
5 import id.ac.unpar.siamodels.MataKuliah;
6 import id.ac.unpar.siamodels.InfoMataKuliah;
7 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
8
9 import java.util.List;
10
11 /**
12  * Mata kuliah ini memperkenalkan konsep dan arsitektur DBMS, mengajarkan
13  * aljabar relasional dan SQL serta pemanfaatannya pada pemrograman kueri
14  * sederhana s/d relatif kompleks. Selain itu, mata kuliah ini juga mengajarkan
15  * dan mempraktekkan perancangan basisdata untuk masalah sederhana
16  * (lingkup kecil) termasuk pengembangan program aplikasinya;
17  * @author Veronica S. Moertini (moertini@unpar.ac.id)
18  */
19 @InfoMataKuliah(nama = "Manajemen_Informasi_dan_Basis_Data", sks = 4)
20 public class AIF204 extends MataKuliah implements HasPrasyarat, HasPraktikum {
21

```



```

22 |     @Override
23 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
24 |         boolean ok = true;
25 |         int angkatan = mahasiswa.getTahunAngkatan();
26 |         if (angkatan >= 2012 && angkatan <= 2014) {
27 |             if (!mahasiswa.hasTempuhKuliah("AIF203")) {
28 |                 reasonsContainer.add("Angkatan_" + angkatan + "_tetapi_tidak_memenuhi_prasyarat_tempuh_AIF103");
29 |                 ok = false;
30 |             }
31 |         }
32 |         return ok;
33 |     }
34 | }
35 | }

```

Listing C.28: AIF205.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | /**
11 |  * Mata kuliah ini memperkenalkan kepada mahasiswa arsitektur komputer
12 |  * sederhana, modern, dan Advance. Perbedaan, kelebihan dan kekurangan untuk
13 |  * masing-masing arsitektur. Selain itu mahasiswa juga mempelajari cara kerja
14 |  * dari komponen-komponen komputer, terutama memory, cache, system BUS dan
15 |  * input/output.
16 |  * @author Chandra Wijaya (chandraw@unpar.ac.id)
17 |  */
18 | @InfoMataKuliah(nama = "Arsitektur_&_Organisasi_Komputer", sks = 3)
19 | public class AIF205 extends MataKuliah implements HasPrasyarat {
20 |
21 |     @Override
22 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
23 |         boolean ok = true;
24 |         if (!mahasiswa.hasTempuhKuliah("AIF106")) {
25 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF106");
26 |             ok = false;
27 |         }
28 |         return ok;
29 |     }
30 | }
31 | }

```

Listing C.29: AIF206.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | /**
11 |  * Mata kuliah ini memperkenalkan kepada mahasiswa mengenai konsep sistem
12 |  * operasi, jenis-jenis sistem operasi yang digunakan dalam kehidupan
13 |  * sehari-hari dan beberapa perangkat keras yang dibutuhkan pada komputer.
14 |  * Selain itu juga mempelajari mengenai teknik dan algoritma yang digunakan
15 |  * dalam pengelolaan sistem operasi.
16 |  * @author Chandra Wijaya (chandraw@unpar.ac.id)
17 |  */
18 | @InfoMataKuliah(nama = "Sistem_Operasi", sks = 4)
19 | public class AIF206 extends MataKuliah implements HasPrasyarat {
20 |
21 |     @Override
22 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
23 |         boolean ok = true;
24 |         if (!mahasiswa.hasTempuhKuliah("AIF102") && !mahasiswa.hasTempuhKuliah("AIF192")) {
25 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF102_atau_AIF192");
26 |             ok = false;
27 |         }
28 |         if (!mahasiswa.hasTempuhKuliah("AIF205")) {
29 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF205");
30 |             ok = false;
31 |         }
32 |         return ok;
33 |     }
34 | }
35 | }

```

Listing C.30: AIF208.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;

```

```

6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | /**
11 |  * Mata kuliah ini memperkenalkan kepada mahasiswa tahapan rekayasa perangkat
12 |  * lunak, terutama dengan paradigma berorientasi objek, dilengkapi dengan
13 |  * pengenalan tentang manajemen proyek perangkat lunak.
14 |  * Selain, itu diberikan deskripsi proyek berskala kecil yang harus dikerjakan
15 |  * oleh mahasiswa dalam kelompok dengan menerapkan teori yang telah
16 |  * dipelajarinya.
17 |  * @author Thomas Anung Basuki (anung@unpar.ac.id)
18 |  */
19 | @InfoMataKuliah(nama = "Rekayasa_Perangkat_Lunak", sks = 4)
20 | public class AIF208 extends MataKuliah implements HasPrasyarat{
21 |
22 |     @Override
23 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
24 |         if (!mahasiswa.hasTempuhKuliah("AIF201")) {
25 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF201");
26 |             return false;
27 |         }
28 |         return true;
29 |     }
30 | }
31 |

```

Listing C.31: AIF210.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | @InfoMataKuliah(nama = "Interaksi_Manusia_Komputer", sks = 2)
7 | public class AIF210 extends MataKuliah {
8 |
9 |
10 | }

```

Listing C.32: AIF280.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | @InfoMataKuliah(nama = "Praktika_Interaksi_Manusia_Komputer", sks = 1)
7 | public class AIF280 extends MataKuliah {
8 |
9 | }

```

Listing C.33: AIF281.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | @InfoMataKuliah(nama = "Pengenalan_Bidang_Ilmu_Teknologi_Informasi_&_Komunikasi", sks = 2)
7 | public class AIF281 extends MataKuliah {
8 |
9 | }

```

Listing C.34: AIF282.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | @InfoMataKuliah(nama = "Algoritma_&_Struktur_Data_Lanjut", sks = 3)
7 | public class AIF282 extends MataKuliah {
8 |
9 | }

```

Listing C.35: AIF292.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | /**
7 |  * @deprecated Mata kuliah tidak dibuka lagi.
8 |  */
9 | @InfoMataKuliah(nama = "Desain_&_Analisis_Algoritma", sks = 3)
10 | public class AIF292 extends MataKuliah {
11 |
12 | }

```

Listing C.36: AIF294.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * @deprecated Mata kuliah tidak dibuka lagi.
8  */
9 @InfoMataKuliah(nama = "Manajemen_Informasi_&_Basis_Data", sks = 3)
10 public class AIF294 extends MataKuliah {
11
12 }

```

Listing C.37: AIF301.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 /**
11  * Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar tentang *sistem
12  * cerdas dan komponen-komponennya. " Terdapat 4 topik utama yang dibahas yaitu
13  * teknik pencarian untuk *penyelesaian masalah, representasi pengetahuan dalam
14  * sistem *cerdas, pemodelan ketidakpastian dalam masalah dan teknik
15  * pembelajaran mesin.
16  *
17  * @author Thomas Anung Basuki
18  */
19
20
21 @InfoMataKuliah(nama = "Pengantar_Sistem_Cerdas_", sks = 3)
22 public class AIF301 extends MataKuliah implements HasPrasyarat {
23
24     @Override
25     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
26         boolean ok = true;
27         if (!mahasiswa.hasTempuhKuliah("AIF204") && !mahasiswa.hasTempuhKuliah("AIF294")) {
28             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF204_atau_AIF294");
29             ok = false;
30         }
31         return ok;
32     }
33 }
34 }

```

Listing C.38: AIF302.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 /**
11  * Mata kuliah ini melatih mahasiswa dalam menulis ilmiah serta memperkenalkan
12  * metodologi penelitian serta kakas untuk menulis ilmiah.
13  *
14  * @author Thomas Anung Basuki (anung@unpar.ac.id)
15  */
16
17 @InfoMataKuliah(nama = "Penulisan_Ilmiiah", sks = 2)
18 public class AIF302 extends MataKuliah implements HasPrasyarat {
19
20     @Override
21     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
22         int sksLulus = mahasiswa.calculateSKSLulus();
23         if (sksLulus < 84) {
24             reasonsContainer.add("SKS_Lulus_" + sksLulus + ",_belum_mencapai_syarat_minimal_84");
25             return false;
26         }
27         return true;
28     }
29 }
30 }

```

Listing C.39: AIF303.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7

```

```

8 import java.util.List;
9
10 /**
11  * Mempelajari Konsep Data, Informasi, Pengetahuan, Sistem Informasi, proses dan
12  * pemodelan bisnis, jenis-jenis sistem informasi, untuk mendukung pengambilan
13  * keputusan. Mempelajari trend Teknologi Informasi, tahap-tahap pembangunan
14  * sistem informasi. Mempelajari pengantar : EIS, e-bisnis/e-commerce, Business
15  * Intelligence, Cloud Computing dan Mobile Applications
16  *
17  * @author Rosa de Lima E. Padmowati (rosad5@unpar.ac.id)
18  */
19
20 @InfoMataKuliah(nama = "Pengantar_Sistem_Informasi", sks = 3)
21 public class AIF303 extends MataKuliah implements HasPrasyarat {
22
23     @Override
24     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25         boolean ok = true;
26         if (!mahasiswa.hasTempuhKuliah("AIF204") && !mahasiswa.hasTempuhKuliah("AIF294")) {
27             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF204_atau_AIF294");
28             ok = false;
29         }
30         return ok;
31     }
32 }
33 }

```

Listing C.40: AIF304.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8 import id.ac.unpar.siamodels.matakuliah.interfaces.HasResponsi;
9
10 import java.util.List;
11
12 /**
13  * Mata kuliah ini memberikan kesempatan bagi mahasiswa untuk memperdalam konsep
14  * tentang pengembangan sistem informasi dan mempraktekkan analisis kebutuhan,
15  * analisis sistem dan perancangan sitem pada organisasi studi kasus;
16  *
17  * @author Veronica S. Moertini (moertini@unpar.ac.id)
18  */
19
20 @InfoMataKuliah(nama = "Proyek_Sistem_Informasi_1", sks = 3)
21 public class AIF304 extends MataKuliah implements HasPrasyarat, HasPraktikum, HasResponsi {
22
23     @Override
24     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25         if (!mahasiswa.hasTempuhKuliah("AIF303")) {
26             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF303");
27             return false;
28         }
29         return true;
30     }
31 }
32 }

```

Listing C.41: AIF305.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 /**
11  * Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar jaringan
12  * komputer dengan menggunakan top-down. Selain itu mengajarkan juga kepada
13  * mahasiswa mengenai aplikasi-aplikasi berbasis jaringan sehingga diharapkan
14  * mahasiswa dapat membuat aplikasi berbasis jaringan dengan menggunakan socket.
15  * Pada akhirnya, mahasiswa akan ditugaskan untuk membangun jaringan komputer
16  * LAN, baik menggunakan kabel maupun nirkabel.
17  *
18  * @author Chandra Wijaya (chandraw@unpar.ac.id)
19  */
20
21 @InfoMataKuliah(nama = "Jaringan_Komputer", sks = 4)
22 public class AIF305 extends MataKuliah implements HasPrasyarat {
23
24     @Override
25     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
26         boolean ok = true;
27         if (!mahasiswa.hasTempuhKuliah("AIF206")) {
28             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF206");
29             ok = false;
30         }
31         return ok;
32     }
33 }

```

```
33|
34| }
```

Listing C.42: AIF306.java

```
1| package id.ac.unpar.siamodels.matakuliah;
2|
3| import id.ac.unpar.siamodels.Mahasiswa;
4| import id.ac.unpar.siamodels.MataKuliah;
5| import id.ac.unpar.siamodels.InfoMataKuliah;
6| import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7| import java.util.List;
8|
9| /**
10|  * Mata kuliah ini bertujuan untuk memberikan pengalaman bagi mahasiswa dalam
11|  * mengerjakan proyek dengan teknologi-teknologi terkini, secara berkelompok.
12|  * Teknologi-teknologi yang digunakan pada kuliah ini tidak spesifik dan dapat
13|  * berubah seiring perkembangan teknologi maupun disesuaikan dengan kompetensi
14|  * dosen pengajar. Beberapa teknologi yang dapat dimanfaatkan antara lain: DVCS
15|  * tool menggunakan Git + Github, Mobile native app (Android, iOS, dll), dan
16|  * responsive web design.
17|  *
18|  * @author Pascal (pascal@unpar.ac.id)
19|  */
20|
21| @InfoMataKuliah(nama = "Proyek_Informatika", sks = 6)
22| public class AIF306 extends MataKuliah implements HasPrasyarat {
23|
24|     @Override
25|     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
26|         if (!mahasiswa.hasTempuhKuliah("AIF208")) {
27|             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF208");
28|             return false;
29|         }
30|         return true;
31|     }
32| }
33| }
```

Listing C.43: AIF311.java

```
1| package id.ac.unpar.siamodels.matakuliah;
2|
3| import id.ac.unpar.siamodels.Mahasiswa;
4| import id.ac.unpar.siamodels.MataKuliah;
5| import id.ac.unpar.siamodels.InfoMataKuliah;
6| import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7| import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8|
9| import java.util.List;
10|
11| /**
12|  * Kuliah Pemrograman Fungsional bertujuan untuk: 1. memperkenalkan paradigma
13|  * pemrograman fungsional, yaitu sebuah pemrograman yang didasarkan pada konsep
14|  * pemetaan dan fungsi matematika. Penyelesaian suatu masalah didasari atas
15|  * aplikasi dari fungsi-fungsi tersebut. 2. memberikan dasar-dasar pemrograman
16|  * fungsional dengan menggunakan bahasa fungsional Haskell.
17|  *
18|  * @author Cecilia E. Nugraheni (cheni@unpar.ac.id)
19|  */
20|
21| @InfoMataKuliah(nama = "Pemrograman_Fungsional", sks = 2)
22| public class AIF311 extends MataKuliah implements HasPrasyarat, HasPraktikum {
23|
24|     @Override
25|     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
26|         boolean ok = true;
27|         if (!mahasiswa.hasTempuhKuliah("AIF103")) {
28|             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF103");
29|             ok = false;
30|         }
31|         return ok;
32|     }
33| }
34| }
```

Listing C.44: AIF312.java

```
1| package id.ac.unpar.siamodels.matakuliah;
2|
3| import id.ac.unpar.siamodels.Mahasiswa;
4| import id.ac.unpar.siamodels.MataKuliah;
5| import id.ac.unpar.siamodels.InfoMataKuliah;
6| import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7| import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8|
9| import java.util.List;
10|
11| /**
12|  * Mata kuliah ini memberikan pengetahuan awal tentang keamanan informasi. Pada
13|  * beberapa pertemuan awal, dibahas keamanan informasi secara matematis, yaitu
14|  * di materi-materi seputar kriptografi dan serangannya. Lalu, dibahas pula
15|  * konsep keamanan informasi pada jaringan komputer dan pada software.
16|  *
17|  */
```

```

17  * @author Mariskha Tri Adithia (mariskha@unpar.ac.id)
18  */
19
20  @InfoMataKuliah(nama = "Keamanan_Informasi", sks = 2)
21  public class AIF312 extends MataKuliah implements HasPrasyarat, HasPraktikum {
22
23      @Override
24      public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25          if (!mahasiswa.hasTempuhKuliah("AIF305")) {
26              reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF305");
27              return false;
28          }
29          return true;
30      }
31  }
32 }

```

Listing C.45: AIF313.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5  import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
6
7  /**
8   * Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar pembuatan grafik
9   * dengan menggunakan komputer seperti mengenal berbagai algoritma pembuatan
10  * primitif 2 dimensi seperti titik, garis, lingkaran, elips, berbagai macam
11  * bentuk kurva, fraktal, konsep warna (RGB), dasar-dasar grafika 3 dimensi
12  * seperti pewarnaan, pencahayaan, pemberian tekstur pada objek, transformasi,
13  * animasi, dan sebagainya. Selain itu diberikan masalah-masalah komputasi
14  * sederhana yang harus diselesaikan menggunakan konsep-konsep yang sudah
15  * diperkenalkan dan mengimplementasikannya menggunakan bahasa pemrograman Java.
16  */
17  * @author Luciana (luciana@unpar.ac.id)
18  */
19  @InfoMataKuliah(nama = "Grafika_Komputer", sks = 2)
20  public class AIF313 extends MataKuliah implements HasPraktikum {
21
22  }

```

Listing C.46: AIF314.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.Mahasiswa;
4  import id.ac.unpar.siamodels.MataKuliah;
5  import id.ac.unpar.siamodels.InfoMataKuliah;
6  import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7  import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8
9  import java.util.List;
10
11  /**
12   * Kuliah ini merupakan kelanjutan dari kuliah Manajemen Informasi Basisdata.
13   * Pada perkuliahan ini, mahasiswa akan mempelajari teknik-teknik pengelolaan
14   * basis data dan membuat program dengan basis data yang optimal/efisien.
15  */
16  * @author Falahah . S.
17  */
18  @InfoMataKuliah(nama = "Pemrograman_Basis_Data", sks = 2)
19  public class AIF314 extends MataKuliah implements HasPrasyarat, HasPraktikum {
20
21      @Override
22      public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
23          if (!mahasiswa.hasTempuhKuliah("AIF204") && !mahasiswa.hasTempuhKuliah("AIF294")) {
24              reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF204_atau_AIF294");
25              return false;
26          }
27          return true;
28      }
29  }
30
31 }

```

Listing C.47: AIF315.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.Mahasiswa;
4  import id.ac.unpar.siamodels.MataKuliah;
5  import id.ac.unpar.siamodels.InfoMataKuliah;
6  import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7  import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8
9  import java.util.List;
10
11  /**
12   * Mata kuliah ini memperkenalkan konsep dan lingkungan pemrograman berbasis web,
13   * kemudian belajar membuat aplikasi berbasis web menggunakan HTML5, CSS, Java Script
14   * dan PHP. Untuk meningkatkan keterampilan pemrograman dilengkapi dengan praktikum.
15   * Sedangkan untuk mendapatkan pengalaman penerapan konsep diberikan tugas besar membuat
16   * program berbasis web dengan kasus yang ditentukan oleh mahasiswa.

```

```

17 | * @author Gede Karya(gkarya@unpar.ac.id)
18 | * @author Husnul Hakim (husnulhakim@unpar.ac.id)
19 | */
20 | @InfoMataKuliah(nama = "Pemrograman_Berbasis_Web", sks = 2)
21 | public class AIF315 extends MataKuliah implements HasPrasyarat, HasPraktikum {
22 |
23 |     @Override
24 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25 |         if (!mahasiswa.hasTempuhKuliah("AIF204") && !mahasiswa.hasTempuhKuliah("AIF294")) {
26 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF204_atau_AIF294");
27 |             return false;
28 |         }
29 |         return true;
30 |     }
31 | }

```

Listing C.48: AIF316.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8 |
9 | import java.util.List;
10 | /**
11 |  * Mata kuliah ini memperkenalkan konsep-konsep dasar komputasi paralel, dimana sebuah
12 |  * program yang berjalan secara paralel harus memiliki safety property dan liveness property.
13 |  * Mahasiswa dikenalkan dengan beberapa teknik pemrograman multi-thread
14 |  * seperti lock, monitor, barrier, thread pool, dan sebagainya, yang diimplementasikan
15 |  * dalam bahasa pemrograman Java. Mahasiswa juga dikenalkan dengan beberapa metode untuk
16 |  * menganalisis kebenaran program baik secara matematis maupun secara praktis dengan bantuan
17 |  * model checker.
18 |  * @author Joanna Helga, M.Sc. (joanna@unpar.ac.id)
19 |  */
20 | @InfoMataKuliah(nama = "Komputasi_Paralel", sks = 2)
21 | public class AIF316 extends MataKuliah implements HasPrasyarat, HasPraktikum {
22 |
23 |     @Override
24 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25 |         if (!mahasiswa.hasTempuhKuliah("AIF102") && !mahasiswa.hasTempuhKuliah("AIF192")) {
26 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF102_atau_AIF192");
27 |             return false;
28 |         }
29 |         return true;
30 |     }
31 | }

```

Listing C.49: AIF317.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | @InfoMataKuliah(nama = "Desain_Antarmuka_Grafis", sks = 2)
11 | public class AIF317 extends MataKuliah implements HasPrasyarat {
12 |
13 |     @Override
14 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15 |         if (!mahasiswa.hasTempuhKuliah("AIF210")) {
16 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF210");
17 |             return false;
18 |         }
19 |         return true;
20 |     }
21 | }
22 | }

```

Listing C.50: AIF318.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8 |
9 | import java.util.List;
10 | /**
11 |  * Mata kuliah ini memperkenalkan konsep perangkat mobile dan pemrograman pada perangkat
12 |  * mobile. Pemrograman dikhususkan pada lingkungan J2ME dan Android.
13 |  * Untuk meningkatkan keterampilan pemrograman dilengkapi dengan praktikum.
14 |  * Sedangkan untuk mendapatkan pengalaman penerapan konsep diberikan tugas implementasi suatu
15 |  * kasus pada lingkungan mobile-cloud dengan kasus yang sudah ditentukan.
16 |  *
17 |  * @author Gede Karya (gkarya@unpar.ac.id)

```

```

18  */
19  @InfoMataKuliah(nama = "Pemrograman_Aplikasi_Bergerak", sks = 2)
20  public class AIF318 extends MataKuliah implements HasPrasyarat, HasPraktikum {
21
22      @Override
23      public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
24          boolean ok = true;
25          if (!mahasiswa.hasTempuhKuliah("AIF102") && !mahasiswa.hasTempuhKuliah("AIF192")) {
26              reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF102_atau_AIF192");
27              ok = false;
28          }
29          if (!mahasiswa.hasTempuhKuliah("AIF201")) {
30              reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF201");
31              ok = false;
32          }
33          return ok;
34      }
35  }

```

Listing C.51: AIF330.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5
6  @InfoMataKuliah(nama = "Kerja_Praktek_1", sks = 2)
7  public class AIF330 extends MataKuliah {
8
9  }

```

Listing C.52: AIF332.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.Mahasiswa;
4  import id.ac.unpar.siamodels.MataKuliah;
5  import id.ac.unpar.siamodels.InfoMataKuliah;
6  import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8  import java.util.List;
9
10 @InfoMataKuliah(nama = "Topik_Khusus_Informatika_2", sks = 3)
11 public class AIF332 extends MataKuliah implements HasPrasyarat {
12
13     @Override
14     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15         // Note: Topik Khusus Informatika 2: Ruby on Rails
16         if (!mahasiswa.hasTempuhKuliah("AIF315")) {
17             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF315");
18             return false;
19         }
20         return true;
21     }
22 }
23

```

Listing C.53: AIF334.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5
6
7  @InfoMataKuliah(nama = "Topik_Khusus_Sistem_Informasi_2", sks = 3)
8  public class AIF334 extends MataKuliah {
9
10 }

```

Listing C.54: AIF335.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2  import id.ac.unpar.siamodels.InfoMataKuliah;
3  import id.ac.unpar.siamodels.MataKuliah;
4
5  @InfoMataKuliah(nama = "Pembelajaran_Mesin", sks = 3)
6  public class AIF335 extends MataKuliah {
7
8  }

```

Listing C.55: AIF336.java

```

1  package id.ac.unpar.siamodels.matakuliah;
2
3  import id.ac.unpar.siamodels.InfoMataKuliah;
4  import id.ac.unpar.siamodels.MataKuliah;
5
6  /**
7   * Mata kuliah ini merupakan mata kuliah lanjutan dari mata kuliah Keamanan

```



```

8 | * Informasi, dengan titik berat pada materi kriptografi. Mata kuliah ini
9 | * memperkenalkan tambahan konsep kriptografi, misalnya tentang otentikasi
10 | * yaitu otentikasi entitas, manajemen kunci, dan bentuk lain dari metode
11 | * merahasiakan pesan, yaitu dengan menggunakan secret sharing. Selanjutnya,
12 | * diperkenalkan juga penggunaan kriptografi pada protokol-protokol yang
13 | * sebenarnya banyak digunakan sehari-hari, misalnya pada e-cash, auction,
14 | * dan electronic voting.
15 | * @author Mariskha Tri Adithia (mariskha@unpar.ac.id)
16 | *
17 | */
18 | @InfoMataKuliah(nama = "Algoritma_Kriptografi", sks = 3)
19 | public class AIF336 extends MataKuliah {
20 |
21 | }

```

Listing C.56: AIF337.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 | import id.ac.unpar.siamodels.InfoMataKuliah;
3 | import id.ac.unpar.siamodels.MataKuliah;
4 |
5 | @InfoMataKuliah(nama = "Matematika_Teknik", sks = 3)
6 | public class AIF337 extends MataKuliah {
7 |
8 | }

```

Listing C.57: AIF339.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | @InfoMataKuliah(nama = "Pemodelan_Formal", sks = 3)
11 | public class AIF339 extends MataKuliah implements HasPrasyarat {
12 |
13 |     @Override
14 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15 |         boolean ok = true;
16 |         if (!mahasiswa.hasTempuhKuliah("AIF104")) {
17 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF104");
18 |             ok = false;
19 |         }
20 |         if (!mahasiswa.hasTempuhKuliah("AIF208")) {
21 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF208");
22 |             ok = false;
23 |         }
24 |         return ok;
25 |     }
26 |
27 | }

```

Listing C.58: AIF341.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
6 |
7 | /**
8 | * Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar
9 | * jaringan dan aplikasinya di kehidupan sehari-hari. Mahasiswa
10 | * dikenalkan dengan teknologi-teknologi terbaru di bidang jaringan,
11 | * sehingga mahasiswa memiliki pengetahuan yang dapat digunakan
12 | * dalam kehidupan sehari-hari. Selain itu mahasiswa juga
13 | * diperkenalkan dengan NetAcad, sebuah layanan dari Cisco yang
14 | * dapat digunakan untuk memenuhi segala macam kebutuhan terkait
15 | * dengan Cisco Academy.
16 | * @author Chandra Wijaya, ST., MT. (chandraw@unpar.ac.id)
17 | *
18 | */
19 | @InfoMataKuliah(nama = "Administrasi_Jaringan_Komputer_1", sks = 3)
20 | public class AIF341 extends MataKuliah implements HasPraktikum {
21 |
22 | }

```

Listing C.59: AIF342.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import java.util.List;
4 |
5 | import id.ac.unpar.siamodels.Mahasiswa;
6 | import id.ac.unpar.siamodels.MataKuliah;
7 | import id.ac.unpar.siamodels.InfoMataKuliah;
8 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
9 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;

```

```

10
11 @InfoMataKuliah(nama = "Administrasi_Jaringan_Komputer_2", sks = 3)
12 public class AIF342 extends MataKuliah implements HasPraktikum, HasPrasyarat {
13
14     @Override
15     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
16         if (!mahasiswa.hasLulusKuliah("AIF341")) {
17             reasonsContainer.add("Tidak_memenuhi_prasyarat_lulus_AIF341");
18             return false;
19         }
20         return true;
21     }
22 }
23

```

Listing C.60: AIF343.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 @InfoMataKuliah(nama = "Pemrograman_Kompetitif", sks = 3)
7 public class AIF343 extends MataKuliah {
8
9 }

```

Listing C.61: AIF344.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 @InfoMataKuliah(nama = "Pemodelan_&_Simulasi", sks = 3)
11 public class AIF344 extends MataKuliah implements HasPrasyarat {
12
13     @Override
14     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15         boolean ok = true;
16         if (!mahasiswa.hasTempuhKuliah("AIF102") && !mahasiswa.hasTempuhKuliah("AIF192")) {
17             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF102_atau_AIF192");
18             ok = false;
19         }
20         if (!mahasiswa.hasTempuhKuliah("AMS200")) {
21             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AMS200");
22             ok = false;
23         }
24         return ok;
25     }
26 }
27

```

Listing C.62: AIF347.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.MataKuliah;
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5
6 @InfoMataKuliah(nama = "E-Commerce", sks = 2)
7 public class AIF347 extends MataKuliah {
8
9 }

```

Listing C.63: AIF352.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.MataKuliah;
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5
6 @InfoMataKuliah(nama = "Jaringan_Syaraf_Tiruan", sks = 2)
7 public class AIF352 extends MataKuliah {
8
9 }

```

Listing C.64: AIF358.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Jaringan_Komputer_Lanjut", sks = 3)
6 public class AIF358 extends MataKuliah {
7
8 }

```

Listing C.65: AIF360.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | @InfoMataKuliah(nama = "Pemrograman_Berbasis_Web_Lanjut", sks = 3)
11 | public class AIF360 extends MataKuliah implements HasPrasyarat {
12 |
13 |     @Override
14 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15 |         if (!mahasiswa.hasTempuhKuliah("AIF315")) {
16 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF315");
17 |             return false;
18 |         }
19 |         return true;
20 |     }
21 | }
22 |
23 |
24 | }

```

Listing C.66: AIF362.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | @InfoMataKuliah(nama = "Sistem_&_Aplikasi_Telematika", sks = 3)
11 | public class AIF362 extends MataKuliah implements HasPrasyarat {
12 |
13 |     @Override
14 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15 |         if (!mahasiswa.hasTempuhKuliah("AIF305")) {
16 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF305");
17 |             return false;
18 |         }
19 |         return true;
20 |     }
21 | }
22 |

```

Listing C.67: AIF380.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 |
7 | @InfoMataKuliah(nama = "Teori_Bahasa_&_Otomata", sks = 3)
8 | public class AIF380 extends MataKuliah {
9 |
10 | }

```

Listing C.68: AIF381.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 |
4 | import id.ac.unpar.siamodels.InfoMataKuliah;
5 | import id.ac.unpar.siamodels.MataKuliah;
6 |
7 | @InfoMataKuliah(nama = "Analisis_Sistem_Informasi", sks = 2)
8 | public class AIF381 extends MataKuliah {
9 |
10 | }

```

Listing C.69: AIF382.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 |
4 | import id.ac.unpar.siamodels.InfoMataKuliah;
5 | import id.ac.unpar.siamodels.MataKuliah;
6 |
7 | @InfoMataKuliah(nama = "Gudang_Data_&_Penambangan_Data", sks = 3)
8 | public class AIF382 extends MataKuliah {
9 |
10 | }

```

Listing C.70: AIF383.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Praktika_Grafika_Komputer", sks = 1)
6 public class AIF383 extends MataKuliah {
7
8 }

```

Listing C.71: AIF385.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Praktika_Pemrograman_Berbasis_Web", sks = 1)
6 public class AIF385 extends MataKuliah {
7
8 }

```

Listing C.72: AIF386.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Manajemen_Proyek_Teknologi_Informasi", sks = 2)
6 public class AIF386 extends MataKuliah {
7
8 }

```

Listing C.73: AIF387.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Pengantar_Telekomunikasi", sks = 3)
6 public class AIF387 extends MataKuliah {
7
8 }

```

Listing C.74: AIF388.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Praktika_Pemrograman_Aplikasi_Bergerak", sks = 1)
6 public class AIF388 extends MataKuliah {
7
8 }

```

Listing C.75: AIF389.java

```

1
2 package id.ac.unpar.siamodels.matakuliah;
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 @InfoMataKuliah(nama = "Kriptografi", sks = 2)
7 public class AIF389 extends MataKuliah {
8
9 }

```

Listing C.76: AIF401.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 @InfoMataKuliah(nama = "Skripsi_1", sks = 4)
11 public class AIF401 extends MataKuliah implements HasPrasyarat {
12
13     @Override
14     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15         boolean ok = true;
16         if (!mahasiswa.hasLulusKuliah("AIF302")) {
17             reasonsContainer.add("Tidak_memenuhi_prasyarat_lulus_AIF302");
18             ok = false;
19         }
20         int sksLulus = mahasiswa.calculateSKSLulus();

```

```

21 |         if (sksLulus < 108) {
22 |             reasonsContainer.add("SKS_Lulus_" + sksLulus + ",_belum_mencapai_syarat_minimal_108");
23 |             return false;
24 |         }
25 |         return ok;
26 |     }
27 | }
28 | }

```

Listing C.77: AIF402.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 | import java.util.List;
8 |
9 | @InfoMataKuliah(nama = "Skripsi_2", sks = 6)
10 | public class AIF402 extends MataKuliah implements HasPrasyarat {
11 |
12 |     @Override
13 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
14 |         if (mahasiswa.hasLulusKuliah("AIF401")) {
15 |             return true;
16 |         } else if (mahasiswa.calculateSKSLulus() >= 124) {
17 |             reasonsContainer.add("CATATAN:_Mahasiswa_harus_mengambil_juga_AIF401_(tempuh_bersama)");
18 |             return true;
19 |         } else {
20 |             reasonsContainer.add("Harus_sudah_mengambil_AIF401_atau_lulus_124_SKS");
21 |             return false;
22 |         }
23 |     }
24 | }
25 | }

```

Listing C.78: AIF403.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import java.util.List;
4 |
5 | import id.ac.unpar.siamodels.Mahasiswa;
6 | import id.ac.unpar.siamodels.MataKuliah;
7 | import id.ac.unpar.siamodels.InfoMataKuliah;
8 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
9 | /**
10 |  * 1. Memberikan wawasan kepada mahasiswa tentang kemunculan dan pemanfaatan teknologi baru,
11 |  * khususnya yang berkaitan dengan komputer, dan dampaknya terhadap masyarakat luas.
12 |  * 2. Memberikan kesadaran dan panduan bersikap kepada mahasiswa dalam menghadapi gejala yang
13 |  * disebabkan oleh munculnya teknologi baru, khususnya yang berkaitan dengan komputer.
14 |  * @author Oerip S. Santosa (oerip@unpar.ac.id)
15 |  */
16 | @InfoMataKuliah(nama = "Komputer_&_Masyarakat", sks = 2)
17 | public class AIF403 extends MataKuliah implements HasPrasyarat {
18 |
19 |     @Override
20 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
21 |         int sksLulus = mahasiswa.calculateSKSLulus();
22 |         if (sksLulus < 70) {
23 |             reasonsContainer.add("SKS_Lulus_" + sksLulus + ",_belum_mencapai_syarat_minimal_70");
24 |             return false;
25 |         }
26 |         return true;
27 |     }
28 | }
29 | }

```

Listing C.79: AIF405.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
7 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
8 |
9 | import java.util.List;
10 |
11 | /**
12 |  * Mata kuliah ini merupakan lanjutan dari Projek Sistem Informasi 1 dan
13 |  * memberikan kesempatan bagi mahasiswa untuk melanjutkan/mengembangkan
14 |  * perancangan sitem pada organisasi studi kasus, mengimplementasikan rancangan
15 |  * dan melakukan pengujian perangkat lunak;
16 |  *
17 |  * @author Veronica S. Moertini (moertini@unpar.ac.id)
18 |  */
19 | @InfoMataKuliah(nama = "Proyek_Sistem_Informasi_2", sks = 3)
20 | public class AIF405 extends MataKuliah implements HasPrasyarat, HasPraktikum {
21 |
22 |     @Override
23 |

```

```

24 public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25     boolean ok = true;
26     if (!mahasiswa.hasTempuhKuliah("AIF304")) {
27         reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF304");
28         ok = false;
29     }
30     return ok;
31 }
32
33 }

```

Listing C.80: AIF438.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 /**
11  * Mata kuliah ini: Memperkenalkan karakteristik dan teknik visualisasi dari
12  * berbagai jenis data yang dapat dianalisis dengan teknik-teknik data mining;
13  * mempelajari teknik-teknik penyiapan data untuk berbagai jenis data dan teknik
14  * data mining; mempraktekkan teknik-teknik penyiapan data untuk menganalisis
15  * data nyata/simulasi dengan memanfaatkan perangkat lunak aplikasi.
16  *
17  * @author Veronica S. Moertini (moertini@unpar.ac.id)
18  */
19
20 @InfoMataKuliah(nama = "Penambangan_Data", sks = 3)
21 public class AIF438 extends MataKuliah implements HasPrasyarat {
22
23     @Override
24     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25         boolean ok = true;
26         if (!mahasiswa.hasLulusKuliah("AIF102") && !mahasiswa.hasLulusKuliah("AIF192")) {
27             reasonsContainer.add("Tidak_memenuhi_prasyarat_lulus_AIF102_atau_AIF192");
28             ok = false;
29         }
30         return ok;
31     }
32
33 }

```

Listing C.81: AIF441.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import java.util.List;
4
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.Mahasiswa;
7 import id.ac.unpar.siamodels.MataKuliah;
8 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
9 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
10
11 /**
12  * Mata kuliah ini memperkenalkan kepada mahasiswa konsep jaringan lanjut
13  * terutama di layer data link dan layer network. Materi utama dari mata kuliah
14  * ini adalah pengembangan jaringan dan pengenalan fungsi-fungsi yang terdapat
15  * pada alat jaringan Cisco yang berkaitan dengan layer 2 dan layer 3.
16  *
17  * @author Chandra Wijaya, ST., MT. (chandraw@unpar.ac.id)
18  */
19
20 @InfoMataKuliah(nama = "Administrasi_Jaringan_Komputer_3", sks = 3)
21 public class AIF441 extends MataKuliah implements HasPraktikum, HasPrasyarat {
22
23     @Override
24     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25         if (!mahasiswa.hasLulusKuliah("AIF342")) {
26             reasonsContainer.add("Tidak_memenuhi_prasyarat_lulus_AIF342");
27             return false;
28         }
29         return true;
30     }
31
32 }

```

Listing C.82: AIF442.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import java.util.List;
4
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.Mahasiswa;
7 import id.ac.unpar.siamodels.MataKuliah;
8 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPraktikum;
9 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
10

```

```

11 | @InfoMataKuliah(nama = "Administrasi_Jaringan_Komputer_4", sks = 3)
12 | public class AIF442 extends MataKuliah implements HasPraktikum, HasPrasyarat {
13 |
14 |     @Override
15 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
16 |         if (!mahasiswa.hasLulusKuliah("AIF441")) {
17 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_lulus_AIF441");
18 |             return false;
19 |         }
20 |         return true;
21 |     }
22 | }
23 |

```

Listing C.83: AIF443.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 | import id.ac.unpar.siamodels.InfoMataKuliah;
3 | import id.ac.unpar.siamodels.MataKuliah;
4 |
5 | @InfoMataKuliah(nama = "Matematika_Kombinatorial", sks = 3)
6 | public class AIF443 extends MataKuliah {
7 |
8 | }

```

Listing C.84: AIF445.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.Mahasiswa;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 | import id.ac.unpar.siamodels.InfoMataKuliah;
6 | import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 |
8 | import java.util.List;
9 |
10 | @InfoMataKuliah(nama = "Metode_Numerik", sks = 3)
11 | public class AIF445 extends MataKuliah implements HasPrasyarat {
12 |
13 |     @Override
14 |     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15 |         boolean ok = true;
16 |         if (!mahasiswa.hasTempuhKuliah("AIF102") && !mahasiswa.hasTempuhKuliah("AIF192")) {
17 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF102_atau_AIF192");
18 |             ok = false;
19 |         }
20 |         if (!mahasiswa.hasTempuhKuliah("AMS100")) {
21 |             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AMS100");
22 |             ok = false;
23 |         }
24 |         return ok;
25 |     }
26 | }
27 |

```

Listing C.85: AIF446.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.InfoMataKuliah;
4 | import id.ac.unpar.siamodels.MataKuliah;
5 |
6 | @InfoMataKuliah(nama = "Kompresi_Data", sks = 3)
7 | public class AIF446 extends MataKuliah {
8 |
9 | }

```

Listing C.86: AIF450.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 | import id.ac.unpar.siamodels.MataKuliah;
4 | import id.ac.unpar.siamodels.InfoMataKuliah;
5 |
6 | @InfoMataKuliah(nama = "Pengolahan_Citra", sks = 3)
7 | public class AIF450 extends MataKuliah {
8 |
9 | }

```

Listing C.87: AIF451.java

```

1 | package id.ac.unpar.siamodels.matakuliah;
2 |
3 |
4 | import id.ac.unpar.siamodels.InfoMataKuliah;
5 | import id.ac.unpar.siamodels.MataKuliah;
6 |
7 | @InfoMataKuliah(nama = "Audit_Sistem_Informasi", sks = 3)
8 | public class AIF451 extends MataKuliah {
9 |
10 | }

```

Listing C.88: AIF453.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 import java.util.List;
8
9 /**
10  * Mata kuliah ini memperkenalkan kebutuhan organisasi terhadap sistem business
11  * intelligent (BI) dan pemanfaatan BI untuk organisasi; memperkenalkan konsep
12  * sistem business intelligent dan komponennya; Mempelajari teknik-teknik
13  * analisis data bisnis dan visualisasi hasil analisis; Mempelajari konsep data
14  * warehouse dan perancangannya dan fungsi OLAP; Mempraktekkan teknik-teknik
15  * analisis data dan visualisasi hasil analisis.
16  *
17  * @author Veronica S. Moertini (moertini@unpar.ac.id)
18  */
19
20 @InfoMataKuliah(nama = "Kecerdasan_Bisnis", sks = 3)
21 public class AIF453 extends MataKuliah implements HasPrasyarat {
22
23     @Override
24     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25         boolean ok = false;
26         if (mahasiswa.hasTempuhKuliah("AIF204") || mahasiswa.hasTempuhKuliah("AIF294")) {
27             ok = true;
28         }
29         if ((mahasiswa.hasTempuhKuliah("AIF102") || !mahasiswa.hasTempuhKuliah("AIF192")) && mahasiswa.calculateIPLulus() > 2.75)
30             ok = true;
31         if (!ok) {
32             reasonsContainer.add("Tidak_memenuhi_prasyarat_((tempuh_AIF204/294)_atau_(tempuh_AIF102/AIF192_&_IPK_Lulus_>_2.75))");
33         }
34         return ok;
35     }
36 }
37
38 }

```

Listing C.89: AIF455.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Sistem_Pendukung_Keputusan", sks = 3)
6 public class AIF455 extends MataKuliah {
7
8 }

```

Listing C.90: AIF456.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 @InfoMataKuliah(nama = "Strategi_Sistem_Informasi_&_Arsitektur_Perusahaan_Berskala_Besar", sks = 3)
7 public class AIF456 extends MataKuliah {
8
9 }

```

Listing C.91: AIF457.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7 import java.util.List;
8
9 /**
10  * Mata kuliah ini memperkenalkan konsep kewirausahaan dengan memanfaatkan teknologi, khususnya
11  * teknologi informasi, sebagai basis usaha dan inovasi produk/jasa; Mempelajari
12  * teknik mencari peluang dan merumuskan bidang usaha spesifik yang akan
13  * diterjuni; Mempelajari konsep manajemen pemasaran, keuangan dan SDM dalam
14  * kaitannya dengan berwira-usaha di bidang TI; Menyusun proposal bisnis untuk
15  * berwira-usaha di bidang TI dan mempresentasikannya.
16  *
17  * @author Veronica S. Moertini (moertini@unpar.ac.id)
18  */
19
20 @InfoMataKuliah(nama = "Kewirausahaan_Berbasis_Teknologi", sks = 3)
21 public class AIF457 extends MataKuliah implements HasPrasyarat {
22
23     @Override
24     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
25         boolean ok = true;
26         int sksLulus = mahasiswa.calculateSKSLulus();
27         if (sksLulus < 70) {

```



```

28|         reasonsContainer.add("SKS_Lulus_" + sksLulus + ",_belum_mencapai_syarat_minimal_70");
29|         return false;
30|     }
31|     return ok;
32| }
33|
34| }

```

Listing C.92: AIF458.java

```

1| package id.ac.unpar.siamodels.matakuliah;
2|
3| import id.ac.unpar.siamodels.Mahasiswa;
4| import id.ac.unpar.siamodels.MataKuliah;
5| import id.ac.unpar.siamodels.InfoMataKuliah;
6| import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7|
8| import java.util.List;
9|
10| @InfoMataKuliah(nama = "Pengendalian_&_Audit_Teknologi_Informasi", sks = 3)
11| public class AIF458 extends MataKuliah implements HasPrasyarat {
12|
13|     @Override
14|     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15|         if (!mahasiswa.hasTempuhKuliah("AIF315")) {
16|             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF315");
17|             return false;
18|         }
19|         return true;
20|     }
21| }
22| }

```

Listing C.93: AIF459.java

```

1| package id.ac.unpar.siamodels.matakuliah;
2|
3|
4| import id.ac.unpar.siamodels.InfoMataKuliah;
5| import id.ac.unpar.siamodels.MataKuliah;
6|
7| @InfoMataKuliah(nama = "Administrasi_Basis_Data", sks = 3)
8| public class AIF459 extends MataKuliah {
9|
10| }

```

Listing C.94: AIF460.java

```

1| package id.ac.unpar.siamodels.matakuliah;
2|
3| import id.ac.unpar.siamodels.MataKuliah;
4| import id.ac.unpar.siamodels.InfoMataKuliah;
5|
6| @InfoMataKuliah(nama = "Manajemen_Pengetahuan", sks = 2)
7| public class AIF460 extends MataKuliah {
8|
9| }

```

Listing C.95: AIF461.java

```

1| package id.ac.unpar.siamodels.matakuliah;
2|
3| import id.ac.unpar.siamodels.InfoMataKuliah;
4| import id.ac.unpar.siamodels.MataKuliah;
5|
6| @InfoMataKuliah(nama = "Pencarian_&_Temu_Kembali_Informasi", sks = 2)
7| public class AIF461 extends MataKuliah {
8|
9| }

```

Listing C.96: AIF462.java

```

1| package id.ac.unpar.siamodels.matakuliah;
2|
3| import id.ac.unpar.siamodels.InfoMataKuliah;
4| import id.ac.unpar.siamodels.MataKuliah;
5|
6| @InfoMataKuliah(nama = "Manajemen_Proses_Bisnis", sks = 3)
7| public class AIF462 extends MataKuliah {
8|
9| }

```

Listing C.97: AIF463.java

```

1| package id.ac.unpar.siamodels.matakuliah;
2| import id.ac.unpar.siamodels.InfoMataKuliah;
3| import id.ac.unpar.siamodels.MataKuliah;
4|

```

```

5 @InfoMataKuliah(nama = "Jaringan_Nirkabel", sks = 3)
6 public class AIF463 extends MataKuliah {
7
8 }

```

Listing C.98: AIF465.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Teknologi_Middleware", sks = 3)
6 public class AIF465 extends MataKuliah {
7
8 }

```

Listing C.99: AIF468.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Teknologi_Multimedia", sks = 3)
6 public class AIF468 extends MataKuliah {
7
8 }

```

Listing C.100: AIF469.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 /**
11  * Mata kuliah ini mengajarkan kepada mahasiswa teknik-teknik untuk membuat
12  * layanan berbasis web. Mahasiswa diperkenalkan dengan standar-standar seperti
13  * HTTP, XML, JSON dan diajarkan untuk memanfaatkannya dalam membuat maupun
14  * menggunakan layanan pihak ketiga. Dalam kuliah ini, juga akan diperkenalkan
15  * minimal satu layanan pihak ketiga yang dapat dimanfaatkan mahasiswa, seperti
16  * Google Places Web Service.
17  * @author Pascal (pascal@unpar.ac.id)
18  */
19 @InfoMataKuliah(nama = "Layanan_Berbasis_Web", sks = 3)
20 public class AIF469 extends MataKuliah implements HasPrasyarat {
21
22     @Override
23     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
24         boolean ok = true;
25         if (!mahasiswa.hasTempuhKuliah("AIF305")) {
26             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF305");
27             ok = false;
28         }
29         if (!mahasiswa.hasTempuhKuliah("AIF315")) {
30             reasonsContainer.add("Tidak_memenuhi_prasyarat_tempuh_AIF315");
31             ok = false;
32         }
33         return ok;
34     }
35 }
36 }

```

Listing C.101: AIF480.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Pemrograman_Sistem", sks = 2)
6 public class AIF480 extends MataKuliah {
7
8 }

```

Listing C.102: AIF483.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Teknik_Kompilasi", sks = 2)
6 public class AIF483 extends MataKuliah {
7
8 }

```

Listing C.103: AIF484.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Kewirausahaan", sks = 3)
6 public class AIF484 extends MataKuliah {
7
8 }

```

Listing C.104: AIF486.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 @InfoMataKuliah(nama = "Keamanan_Jaringan", sks = 3)
6 public class AIF486 extends MataKuliah {
7
8 }

```

Listing C.105: AKS122.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 @InfoMataKuliah(nama = "Logika_Formal_Dasar", sks = 2)
7 public class AKS122 extends MataKuliah {
8
9 }

```

Listing C.106: AKS124.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 @InfoMataKuliah(nama = "Logika_Informatika", sks = 3)
7 public class AKS124 extends MataKuliah {
8
9 }

```

Listing C.107: AMS100.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Sistem Bilangan, Fungsi, Limit dan Kekontinuan Fungsi, Turunan, Integral,
8  * Penggunaan Integral, Sistem Persamaan Linear, Determinan, Vektor, Nilai dan
9  * Vektor Eigen.
10  * @author Taufik Limansyah, S.Si, MT.
11 */
12 @InfoMataKuliah(nama = "Matematika_Informatika", sks = 4)
13 public class AMS100 extends MataKuliah {
14 }

```

Listing C.108: AMS190.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 /**
6  * @deprecated Mata kuliah tidak dibuka lagi.
7  */
8 @InfoMataKuliah(nama = "Matematika_Informatika", sks = 3)
9 public class AMS190 extends MataKuliah {
10
11 }

```

Listing C.109: AMS191.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2 import id.ac.unpar.siamodels.InfoMataKuliah;
3 import id.ac.unpar.siamodels.MataKuliah;
4
5 /**
6  * @deprecated Mata kuliah tidak dibuka lagi.
7  */
8 @InfoMataKuliah(nama = "Kalkulus", sks = 4)
9 public class AMS191 extends MataKuliah {
10
11 }

```

Listing C.110: AMS200.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 @InfoMataKuliah(nama = "Probabilitas_dan_Statistika", sks = 3)
7 public class AMS200 extends MataKuliah {
8
9 }

```

Listing C.111: AMS290.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6
7 @InfoMataKuliah(nama = "Aljabar_&_Linear_Matriks", sks = 3)
8 public class AMS290 extends MataKuliah {
9
10 }

```

Listing C.112: APS182.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 @InfoMataKuliah(nama = "Fisika_Dasar", sks = 3)
8 public class APS182 extends MataKuliah {
9
10 }

```

Listing C.113: APS302.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 @InfoMataKuliah(nama = "Dunia_Digital_Dan_Sains", sks = 2)
7 public class APS302 extends MataKuliah {
8
9 }

```

Listing C.114: APS309.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  *
8  * APS302 atau APS309 ?
9  */
10
11 @InfoMataKuliah(nama = "Dunia_Digital_Dan_Sains", sks = 2)
12 public class APS309 extends MataKuliah {
13
14 }

```

Listing C.115: APS402.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.InfoMataKuliah;
6 import id.ac.unpar.siamodels.matakuliah.interfaces.HasPrasyarat;
7
8 import java.util.List;
9
10 @InfoMataKuliah(nama = "Etika_Profesi", sks = 2)
11 public class APS402 extends MataKuliah implements HasPrasyarat {
12
13     @Override
14     public boolean checkPrasyarat(Mahasiswa mahasiswa, List<String> reasonsContainer) {
15         int sksLulus = mahasiswa.calculateSKSLulus();
16         if (sksLulus < 90) {
17             reasonsContainer.add("SKS_Lulus_" + sksLulus + ",_belum_mencapai_syarat_minimal_90");
18             return false;
19         }
20         return true;
21     }
22
23 }

```

Listing C.116: EAA101.java

```
1 package id.ac.unpar.siamodels.matakuliah;  
2  
3  
4 import id.ac.unpar.siamodels.InfoMataKuliah;  
5 import id.ac.unpar.siamodels.MataKuliah;  
6  
7 @InfoMataKuliah(nama = "Akuntansi_Keuangan_Dasar_I", sks = 2)  
8 public class EAA101 extends MataKuliah {  
9  
10 }
```

Listing C.117: EAA102.java

```
1 package id.ac.unpar.siamodels.matakuliah;  
2  
3 import id.ac.unpar.siamodels.InfoMataKuliah;  
4 import id.ac.unpar.siamodels.MataKuliah;  
5  
6 @InfoMataKuliah(nama = "Akuntansi_Keuangan_Dasar_II", sks = 2)  
7 public class EAA102 extends MataKuliah {  
8  
9 }
```

Listing C.118: ESA101.java

```
1 package id.ac.unpar.siamodels.matakuliah;  
2  
3 import id.ac.unpar.siamodels.InfoMataKuliah;  
4 import id.ac.unpar.siamodels.MataKuliah;  
5  
6 @InfoMataKuliah(nama = "Akuntansi_Keuangan_Dasar", sks = 4)  
7 public class ESA101 extends MataKuliah {  
8  
9 }
```

Listing C.119: ESM101.java

```
1 package id.ac.unpar.siamodels.matakuliah;  
2  
3  
4 import id.ac.unpar.siamodels.InfoMataKuliah;  
5 import id.ac.unpar.siamodels.MataKuliah;  
6  
7 @InfoMataKuliah(nama = "Pengantar_Bisnis", sks = 3)  
8 public class ESM101 extends MataKuliah {  
9  
10 }
```

Listing C.120: ESM105.java

```
1 package id.ac.unpar.siamodels.matakuliah;  
2  
3  
4 import id.ac.unpar.siamodels.InfoMataKuliah;  
5 import id.ac.unpar.siamodels.MataKuliah;  
6  
7 @InfoMataKuliah(nama = "Manajemen", sks = 3)  
8 public class ESM105 extends MataKuliah {  
9  
10 }
```

Listing C.121: ESM201.java

```
1 package id.ac.unpar.siamodels.matakuliah;  
2 import id.ac.unpar.siamodels.InfoMataKuliah;  
3 import id.ac.unpar.siamodels.MataKuliah;  
4  
5 @InfoMataKuliah(nama = "Perilaku_Keorganisasian", sks = 3)  
6 public class ESM201 extends MataKuliah {  
7  
8 }
```

Listing C.122: ESM203.java

```
1 package id.ac.unpar.siamodels.matakuliah;  
2 import id.ac.unpar.siamodels.InfoMataKuliah;  
3 import id.ac.unpar.siamodels.MataKuliah;  
4  
5 @InfoMataKuliah(nama = "Pengantar_Manajemen_Pemasaran", sks = 3)  
6 public class ESM203 extends MataKuliah {  
7  
8 }
```

Listing C.123: ESM204.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 @InfoMataKuliah(nama = "Manajemen_Kuangan_I", sks = 3)
8 public class ESM204 extends MataKuliah {
9
10 }

```

Listing C.124: IIE103.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 @InfoMataKuliah(nama = "Statistika_Desriptif", sks = 3)
8 public class IIE103 extends MataKuliah {
9
10 }

```

Listing C.125: IIE207.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 @InfoMataKuliah(nama = "Proses_Produksi_Pembentukan", sks = 2)
8 public class IIE207 extends MataKuliah {
9
10 }

```

Listing C.126: IIE210.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 @InfoMataKuliah(nama = "Perancangan_Sistem_Kerja_Dan_Ergonomi", sks = 2)
8 public class IIE210 extends MataKuliah {
9
10 }

```

Listing C.127: IIE214.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 @InfoMataKuliah(nama = "Ekologi_Industri", sks = 2)
8 public class IIE214 extends MataKuliah {
9
10 }

```

Listing C.128: MKU001.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 /**
8  * Mata Kuliah Pendidikan Pancasila berupaya menelaah/mengkaji berbagai fenomena kehidupan
9  * bangsa dan Negara Indonesia sebagai sebuah ruang publik dengan menggunakan pendekatan
10  * hermeneutika (filsafat) dan pendidikan nilai (pedagogik). Dengan bantuan hermeneutika
11  * mahasiswa diajak berpikir kritis terhadap segala bentuk ideologisme Pancasila dan melalui
12  * pendidikan nilai mahasiswa dilatih untuk memiliki nilai Pancasila. Nilai pengembangan diri
13  * intra-personal dan relasi inter-personal dapat tertanam melalui pendidikan Pancasila yang
14  * tujuannya adalah membangun kepribadian (character building) manusia Indonesia yang utuh,
15  * baik menyangkut aspek kognitif, afektif, maupun psikomotor. Dengan demikian, Pendidikan
16  * Pancasila mengajak mahasiswa menilai realitas ruang publik sehari-hari secara mandiri
17  * dengan panduan nilai-nilai etis Pancasila.
18  */
19 @InfoMataKuliah(nama = "Pendidikan_Pancasila", sks = 2)
20 public class MKU001 extends MataKuliah {
21
22 }

```

Listing C.129: MKU002.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Pendidikan Kewarganegaraan menjelaskan pentingnya pemahaman tentang identitas nasional
8  * Indonesia, hak dan kewajiban warga negara Indonesia serta hubungannya dengan hak dan
9  * kewajiban asasi manusia. Materi kuliah mencakup juga wawasan nusantara, ketahanan nasional,
10 * politik dan strategi nasional, serta implementasinya dalam kehidupan bermasyarakat, berbangsa
11 * dan bernegara kesatuan Republik Indonesia.
12 */
13
14 @InfoMataKuliah(nama = "Pendidikan_Kewarganegaraan", sks = 2)
15 public class MKU002 extends MataKuliah {
16
17 }

```

Listing C.130: MKU003.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Mata kuliah ini membentuk karakteristik mahasiswa sebagai manusia yang memiliki religiusitas
8  * melalui pendalaman akan makna agama dan beragama, mendeteksi dinamika Wahyu Tuhan dan iman
9  * mereka, memahami relasi dengan Tuhan dan sesama, mengenal makna keselamatan dalam konteks
10 * Kerajaan Allah, dan mampu menyatakan ajaran Gereja dalam pelayanan terhadap orang miskin dan
11 * terlantar.
12 */
13
14 @InfoMataKuliah(nama = "Pendidikan_Agama_(Katolik)", sks = 2)
15 public class MKU003 extends MataKuliah {
16
17 }

```

Listing C.131: MKU004.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Fenomenologi Agama merupakan bagian yang tak terpisahkan dari kajian filosofis, kritis,
8  * rasional, dan obyektif mengenai substansi ajaran agama. Fenomenologi merupakan sebuah
9  * disiplin ilmu yang secara kritis-rasional mengkaji fenomena dan dinamika kehidupan manusia
10 * beragama, dari upaya menjadikan Tuhan sebagai tujuan sesembahan sampai menempatkan Tuhan
11 * sebagai instrumen legitimasi untuk melakukan tindakan yang justru bertolak belakang dengan
12 * kehendak Tuhan yang disembah. Sehubungan dengan itu, kritik konstruktif terhadap perilaku
13 * manusia beragama menjadi salah satu poin utama dalam mata kuliah ini. Kesiediaan untuk
14 * melakukan otoritik terhadap agama sendiri erat terkait dengan upaya menemukan kembali nilai
15 * sejati agama atau otentisitas hidup beragama.
16 */
17
18 @InfoMataKuliah(nama = "Pendidikan_Agama_(Fenomenologi)", sks = 2)
19 public class MKU004 extends MataKuliah {
20
21 }

```

Listing C.132: MKU008.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Mendalami perilaku sehari-hari yang baik dalam bermasyarakat.
8  */
9
10 @InfoMataKuliah(nama = "Etika", sks = 2)
11 public class MKU008 extends MataKuliah{
12
13 }

```

Listing C.133: MKU009.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Tujuan dari mata kuliah ini adalah untuk mendalami keterampilan berbahasa Indonesia, agar
8  * mampu mengkomunikasikan hasil pemikiran serta meningkatkan keterampilan dalam menyusun karya
9  * ilmiah. Mata kuliah Bahasa Indonesia ini dimulai dengan mempelajari penulisan kata baku dan
10 * non baku serta pengungkapan pikiran dengan puntuasi yang benar. Selanjutnya dipelajari

```

```

11 * penyusunan kalimat yang baku serta menghubungkan kalimat-kalimat yang padu dalam menuangkan
12 * gagasan dalam sebuah paragraf. Selain itu, dalam matakuliah ini dipelajari cara menyusun
13 * surat dinas yang jelas dan komunikatif. Di akhir kuliah ini, mahasiswa diberi tugas
14 * penyusunan makalah dengan benar.
15 *
16 */
17 @InfoMataKuliah(nama = "Bahasa_Indonesia", sks = 2)
18 public class MKU009 extends MataKuliah{
19
20 }

```

Listing C.134: MKU010.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Mata kuliah ini difokuskan pada pemahaman sumber referensi dalam Bahasa Inggris dan
8  * pengembangan kosakata Bahasa Inggris (vocabularies). Hampir keseluruhan waktu perkuliahan
9  * didedikasikan untuk menjelaskan metode mengekstraksi isi bacaan secara tepat dan melatih
10 * mahasiswa untuk menerapkan metode tersebut seraya menambah kosakata-kosakata baru.
11 * Mahasiswa juga dilatih untuk mempresentasikan hasil pemahamannya akan isi bahan bacaan.
12 *
13 */
14 @InfoMataKuliah(nama = "Bahasa_Ingggris", sks = 2)
15 public class MKU010 extends MataKuliah{
16
17 }

```

Listing C.135: MKU011.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Mata kuliah estetika memberi pemahaman konseptual filosofis "seni" dalam khasanah keilmuan,
8  * pembentukan kesadaran ekologis juga dalam proses pembudayaan dan peradaban. Mata kuliah ini
9  * akan menjadi fondasi bagi mahasiswa untuk memahami dan mempraktekkan seni dari sudut pandang
10 * filsafat, sejarah, kultural, dan global. Melalui mata kuliah ini, mahasiswa mempelajari
11 * mengenai dunia manusia (manusia dan pikirannya), pluralitas dan relativitas seni, serta
12 * aliran-aliran seni rupa Barat.
13 *
14 */
15 @InfoMataKuliah(nama = "Estetika", sks = 2)
16 public class MKU011 extends MataKuliah{
17
18 }

```

Listing C.136: MKU012.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3 import id.ac.unpar.siamodels.InfoMataKuliah;
4 import id.ac.unpar.siamodels.MataKuliah;
5
6 /**
7  * Perkuliahan logika ditujukan untuk memberikan dasar-dasar ketrampilan berpikir rasional dan
8  * sistematis. Isinya mencakup ketrampilan berpikir deduktif dan induktif, seperti silogisme,
9  * argumen analogikal dan generalisasi induktif. Pembahasan teoretis disertai pula dengan
10 * pelatihan praktis yang diarahkan pada proses berpikir. Untuk menajamkan kemampuan berpikir
11 * tersebut, mahasiswa dilatih pula mengidentifikasi kerancuan-kerancuan (fallacies) yang sering
12 * dijumpai baik dalam kehidupan sehari-hari maupun dalam konteks akademik.
13 *
14 */
15 @InfoMataKuliah(nama = "Logika", sks = 2)
16 public class MKU012 extends MataKuliah{
17
18 }

```

Listing C.137: SIR104.java

```

1 package id.ac.unpar.siamodels.matakuliah;
2
3
4 import id.ac.unpar.siamodels.InfoMataKuliah;
5 import id.ac.unpar.siamodels.MataKuliah;
6
7 @InfoMataKuliah(nama = "Bahasa_Jepang", sks = 3)
8 public class SIR104 extends MataKuliah {
9
10 }

```

## C.2 Hasil Latex



Listing C.138: siamodels.tex

```

1 \documentclass{article}
2 \begin{document}
3 \begin{enumerate}
4 \item \texttt{InfoMataKuliah}
5
6
7
8 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
9 \begin{itemize}
10 \item \texttt{public int sks()}
11
12 Jumlah bobot sks dari mata kuliah ini
13
14 \textbf{Parameter:}
15 \begin{itemize}
16 \item Tidak memiliki parameter \textit{method}
17 \end{itemize}
18 \textbf{Return Value:} jumlah bobot sks
19
20 \textbf{Exception:} Tidak memiliki \textit{exception}
21
22 \item \texttt{public String nama()}
23
24 Nama mata kuliah ini
25
26 \textbf{Parameter:}
27 \begin{itemize}
28 \item Tidak memiliki parameter \textit{method}
29 \end{itemize}
30 \textbf{Return Value:} nama mata kuliah
31
32 \textbf{Exception:} Tidak memiliki \textit{exception}
33
34 \end{itemize}
35 \item \texttt{MataKuliahFactory}
36
37 Kelas yang bertugas membuat kelas mata kuliah, dan menyimpannya untuk bisa
38 digunakan kemudian (untuk hemat memori).
39
40 Atribut yang dimiliki kelas ini adalah sebagai berikut.
41 \begin{itemize}
42 \item \texttt{String DEFAULT\_MATAKULIAH\_PACKAGE} - Lokasi package untuk daftar mata kuliah
43 \item \texttt{MataKuliahFactory instance} - Singleton instance to factory.
44 \item \texttt{SortedMap mataKuliahCache} - Singleton instances untuk mata kuliah.
45 \end{itemize}
46 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
47 \begin{itemize}
48 \item \texttt{public static MataKuliahFactory getInstance()}
49
50
51
52 \textbf{Parameter:}
53 \begin{itemize}
54 \item Tidak memiliki parameter \textit{method}
55 \end{itemize}
56 \textbf{Return Value:} Tidak memiliki \textit{return value}
57
58 \textbf{Exception:} Tidak memiliki \textit{exception}
59
60 \item \texttt{public MataKuliah createMataKuliah(java.lang.String kode, int sks, java.lang.String nama)}
61
62 Membuat baru atau mendapatkan mata kuliah, jika memiliki informasi
63 nama dan jumlah SKS.
64
65 \textbf{Parameter:}
66 \begin{itemize}
67 \item \texttt{String kode} -
68 kode mata kuliah
69 \item \texttt{int sks} -
70 jumlah SKS
71 \item \texttt{String nama} -
72 nama mata kuliah
73 \end{itemize}
74 \textbf{Return Value:} objek mata kuliah
75
76 \textbf{Exception:} Tidak memiliki \textit{exception}
77
78 \item \texttt{public MataKuliah createMataKuliah(java.lang.String kode)}
79
80 Membuat baru atau mendapatkan mata kuliah, jika tidak memiliki informasi
81 nama dan jumlah SKS.
82
83 \textbf{Parameter:}
84 \begin{itemize}
85 \item \texttt{String kode} -
86 kode mata kuliah
87 \end{itemize}
88 \textbf{Return Value:} objek mata kuliah
89
90 \textbf{Exception:} IllegalStateException
91 jika sks dan tidak sesuai dengan yang ada di kode
92
93 \end{itemize}
94 \item \texttt{Semester}
95
96
97

```

```

98 Atribut yang dimiliki kelas ini adalah sebagai berikut.
99 \begin{itemize}
100 \item \texttt{Semester UNKNOWN5} -
101 \item \texttt{Semester TRANSFER} -
102 \item \texttt{Semester PENDEK} -
103 \item \texttt{Semester GANJIL} -
104 \item \texttt{Semester GENAP} -
105 \item \texttt{int order} -
106 \end{itemize}
107 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
108 \begin{itemize}
109 \item \texttt{public static Semester values()}
110
111
112
113 \textbf{Parameter:}
114 \begin{itemize}
115 \item Tidak memiliki parameter \textit{method}
116 \end{itemize}
117 \textbf{Return Value:} Tidak memiliki \textit{return value}
118
119 \textbf{Exception:} Tidak memiliki \textit{exception}
120
121 \item \texttt{public static Semester valueOf(java.lang.String name)}
122
123
124
125 \textbf{Parameter:}
126 \begin{itemize}
127 \item \texttt{String name} -
128 \end{itemize}
129 \textbf{Return Value:} Tidak memiliki \textit{return value}
130
131 \textbf{Exception:} Tidak memiliki \textit{exception}
132
133 \item \texttt{public static Semester fromString(java.lang.String text)}
134
135
136
137 \textbf{Parameter:}
138 \begin{itemize}
139 \item \texttt{String text} -
140 \end{itemize}
141 \textbf{Return Value:} Tidak memiliki \textit{return value}
142
143 \textbf{Exception:} Tidak memiliki \textit{exception}
144
145 \item \texttt{int getOrder()}
146
147
148
149 \textbf{Parameter:}
150 \begin{itemize}
151 \item Tidak memiliki parameter \textit{method}
152 \end{itemize}
153 \textbf{Return Value:} Tidak memiliki \textit{return value}
154
155 \textbf{Exception:} Tidak memiliki \textit{exception}
156
157 \end{itemize}
158 \item \texttt{TahunSemester implements Comparable}
159
160 Menyimpan konstanta untuk semester beserta tahunnya di UNPAR.
161
162 Atribut yang dimiliki kelas ini adalah sebagai berikut.
163 \begin{itemize}
164 \item \texttt{String kodeTahunSemester} - Kode semester 3 digit, sesuai DPS:
165 <ul>
166 <li>2 digit pertama berupa tahun, 2 digit terakhir</li>
167 <li>digit terakhir: 1 untuk ganjil, 2 untuk genap, 4 untuk pendek.
168 </ul>
169 \end{itemize}
170 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
171 \begin{itemize}
172 \item \texttt{public Semester getSemester()}
173
174
175
176 \textbf{Parameter:}
177 \begin{itemize}
178 \item Tidak memiliki parameter \textit{method}
179 \end{itemize}
180 \textbf{Return Value:} Tidak memiliki \textit{return value}
181
182 \textbf{Exception:} Tidak memiliki \textit{exception}
183
184 \item \texttt{public int getTahun()}
185
186
187
188 \textbf{Parameter:}
189 \begin{itemize}
190 \item Tidak memiliki parameter \textit{method}
191 \end{itemize}
192 \textbf{Return Value:} Tidak memiliki \textit{return value}
193
194 \textbf{Exception:} Tidak memiliki \textit{exception}
195
196 \item \texttt{private static void validateKodeSemester(java.lang.String kodeTahunSemester)}

```

```

197 |
198 |
199 |
200 | \textbf{Parameter:}
201 | \begin{itemize}
202 | \item \texttt{String kodeTahunSemester} -
203 | \end{itemize}
204 | \textbf{Return Value:} Tidak memiliki \textit{return value}
205 |
206 | \textbf{Exception:} Tidak memiliki \textit{exception}
207 |
208 | \item \texttt{public String getKodeDPS()}
209 |
210 | Mendapatkan kode tahun/semester sesuai aturan di DPS.
211 |
212 | \textbf{Parameter:}
213 | \begin{itemize}
214 | \item Tidak memiliki parameter \textit{method}
215 | \end{itemize}
216 | \textbf{Return Value:} kode tahun/semester sesuai aturan di DPS.
217 |
218 | \textbf{Exception:} Tidak memiliki \textit{exception}
219 |
220 | \item \texttt{public int compareTo(id.ac.unpar.siamodels.TahunSemester o)}
221 |
222 |
223 |
224 | \textbf{Parameter:}
225 | \begin{itemize}
226 | \item \texttt{TahunSemester o} -
227 | \end{itemize}
228 | \textbf{Return Value:} Tidak memiliki \textit{return value}
229 |
230 | \textbf{Exception:} Tidak memiliki \textit{exception}
231 |
232 | \textbf{Override:} \texttt{compareTo} dari kelas \texttt{Object}
233 |
234 | \item \texttt{public boolean equals(java.lang.Object arg0)}
235 |
236 |
237 |
238 | \textbf{Parameter:}
239 | \begin{itemize}
240 | \item \texttt{Object arg0} -
241 | \end{itemize}
242 | \textbf{Return Value:} Tidak memiliki \textit{return value}
243 |
244 | \textbf{Exception:} Tidak memiliki \textit{exception}
245 |
246 | \item \texttt{public String toString()}
247 |
248 |
249 |
250 | \textbf{Parameter:}
251 | \begin{itemize}
252 | \item Tidak memiliki parameter \textit{method}
253 | \end{itemize}
254 | \textbf{Return Value:} Tidak memiliki \textit{return value}
255 |
256 | \textbf{Exception:} Tidak memiliki \textit{exception}
257 |
258 | \end{itemize}
259 | \item \texttt{Mahasiswa}
260 |
261 |
262 |
263 | Atribut yang dimiliki kelas ini adalah sebagai berikut.
264 | \begin{itemize}
265 | \item \texttt{String npm} -
266 | \item \texttt{String nama} -
267 | \item \texttt{List riwayatNilai} -
268 | \item \texttt{URL photoURL} -
269 | \item \texttt{List jadwalKuliahList} -
270 | \item \texttt{SortedMap nilaiTOEFL} -
271 | \end{itemize}
272 | \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
273 | \begin{itemize}
274 | \item \texttt{public String getName()}
275 |
276 |
277 |
278 | \textbf{Parameter:}
279 | \begin{itemize}
280 | \item Tidak memiliki parameter \textit{method}
281 | \end{itemize}
282 | \textbf{Return Value:} Tidak memiliki \textit{return value}
283 |
284 | \textbf{Exception:} Tidak memiliki \textit{exception}
285 |
286 | \item \texttt{public void setName(java.lang.String nama)}
287 |
288 |
289 |
290 | \textbf{Parameter:}
291 | \begin{itemize}
292 | \item \texttt{String nama} -
293 | \end{itemize}
294 | \textbf{Return Value:} Tidak memiliki \textit{return value}
295 |

```

```

296 \textbf{Exception}: Tidak memiliki \textit{exception}
297
298 \item \texttt{public String getNpm()}
299
300
301
302 \textbf{Parameter:}
303 \begin{itemize}
304 \item Tidak memiliki parameter \textit{method}
305 \end{itemize}
306 \textbf{Return Value}: Tidak memiliki \textit{return value}
307
308 \textbf{Exception}: Tidak memiliki \textit{exception}
309
310 \item \texttt{public URL getPhotoURL()}
311
312
313
314 \textbf{Parameter:}
315 \begin{itemize}
316 \item Tidak memiliki parameter \textit{method}
317 \end{itemize}
318 \textbf{Return Value}: Tidak memiliki \textit{return value}
319
320 \textbf{Exception}: Tidak memiliki \textit{exception}
321
322 \item \texttt{public void setPhotoURL(java.net.URL photoURL)}
323
324
325
326 \textbf{Parameter:}
327 \begin{itemize}
328 \item \texttt{URL photoURL} -
329 \end{itemize}
330 \textbf{Return Value}: Tidak memiliki \textit{return value}
331
332 \textbf{Exception}: Tidak memiliki \textit{exception}
333
334 \item \texttt{public List getJadwalKuliahList()}
335
336
337
338 \textbf{Parameter:}
339 \begin{itemize}
340 \item Tidak memiliki parameter \textit{method}
341 \end{itemize}
342 \textbf{Return Value}: Tidak memiliki \textit{return value}
343
344 \textbf{Exception}: Tidak memiliki \textit{exception}
345
346 \item \texttt{public void setJadwalKuliahList(java.util.List jadwalKuliahList)}
347
348
349
350 \textbf{Parameter:}
351 \begin{itemize}
352 \item \texttt{java.util.List jadwalKuliahList} -
353 \end{itemize}
354 \textbf{Return Value}: Tidak memiliki \textit{return value}
355
356 \textbf{Exception}: Tidak memiliki \textit{exception}
357
358 \item \texttt{public String getEmailAddress()}
359
360
361
362 \textbf{Parameter:}
363 \begin{itemize}
364 \item Tidak memiliki parameter \textit{method}
365 \end{itemize}
366 \textbf{Return Value}: Tidak memiliki \textit{return value}
367
368 \textbf{Exception}: Tidak memiliki \textit{exception}
369
370 \item \texttt{public List getRiwayatNilai()}
371
372
373
374 \textbf{Parameter:}
375 \begin{itemize}
376 \item Tidak memiliki parameter \textit{method}
377 \end{itemize}
378 \textbf{Return Value}: Tidak memiliki \textit{return value}
379
380 \textbf{Exception}: Tidak memiliki \textit{exception}
381
382 \item \texttt{public SortedMap getNilaiTOEFL()}
383
384
385
386 \textbf{Parameter:}
387 \begin{itemize}
388 \item Tidak memiliki parameter \textit{method}
389 \end{itemize}
390 \textbf{Return Value}: Tidak memiliki \textit{return value}
391
392 \textbf{Exception}: Tidak memiliki \textit{exception}
393
394 \item \texttt{public void setNilaiTOEFL(java.util.SortedMap nilaiTOEFL)}

```

```

395 |
396 |
397 |
398 | \textbf{Parameter:}
399 | \begin{itemize}
400 | \item \texttt{java.util.SortedMap nilaiTOEFL} -
401 | \end{itemize}
402 | \textbf{Return Value:} Tidak memiliki \textit{return value}
403 |
404 | \textbf{Exception:} Tidak memiliki \textit{exception}
405 |
406 | \item \texttt{public double calculateIPKLulus()}
407 |
408 | Menghitung IPK mahasiswa sampai saat ini, dengan aturan:
409 | <ul>
410 | <li>Kuliah yang tidak lulus tidak dihitung
411 | <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
412 | </ul>
413 | Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
414 |
415 | \textbf{Parameter:}
416 | \begin{itemize}
417 | \item Tidak memiliki parameter \textit{method}
418 | \end{itemize}
419 | \textbf{Return Value:} IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.
420 |
421 | \textbf{Exception:} Tidak memiliki \textit{exception}
422 |
423 | \item \texttt{public double calculateIPLulus()}
424 |
425 | Menghitung IP mahasiswa sampai saat ini, dengan aturan:
426 | <ul>
427 | <li>Kuliah yang tidak lulus tidak dihitung
428 | <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
429 | </ul>
430 | Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
431 |
432 | \textbf{Parameter:}
433 | \begin{itemize}
434 | \item Tidak memiliki parameter \textit{method}
435 | \end{itemize}
436 | \textbf{Return Value:} IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.
437 |
438 | \textbf{Exception:} Tidak memiliki \textit{exception}
439 |
440 | \item \texttt{public double calculateIPTempuh(boolean lulusSaja)}
441 |
442 | Menghitung IP mahasiswa sampai saat ini, dengan aturan:
443 | <ul>
444 | <li>Perhitungan kuliah yang tidak lulus ditentukan parameter
445 | <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
446 | </ul>
447 |
448 | \textbf{Parameter:}
449 | \begin{itemize}
450 | \item \texttt{boolean lulusSaja} -
451 | set true jika ingin membuang mata kuliah tidak lulus, false jika ingin semua (sama dengan "IP N. Terbaik" di DPS)
452 | Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
453 | \end{itemize}
454 | \textbf{Return Value:} IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.
455 |
456 | \textbf{Exception:} Tidak memiliki \textit{exception}
457 |
458 | \item \texttt{public double calculateIPKumulatif()}
459 |
460 | Menghitung IP Kumulatif mahasiswa sampai saat ini, dengan aturan:
461 | <ul>
462 | <li>Jika pengambilan beberapa kali, diambil semua.
463 | </ul>
464 | Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
465 |
466 | \textbf{Parameter:}
467 | \begin{itemize}
468 | \item Tidak memiliki parameter \textit{method}
469 | \end{itemize}
470 | \textbf{Return Value:} IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.
471 |
472 | \textbf{Exception:} Tidak memiliki \textit{exception}
473 |
474 | \item \texttt{public double calculateIPKTempuh(boolean lulusSaja)}
475 |
476 | Menghitung IPK mahasiswa sampai saat ini, dengan aturan:
477 | <ul>
478 | <li>Perhitungan kuliah yang tidak lulus ditentukan parameter
479 | <li>Jika pengambilan beberapa kali, diambil <em>nilai terbaik</em>.
480 | </ul>
481 |
482 | \textbf{Parameter:}
483 | \begin{itemize}
484 | \item \texttt{boolean lulusSaja} -
485 | set true jika ingin membuang mata kuliah tidak lulus
486 | Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
487 | \end{itemize}
488 | \textbf{Return Value:} IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.
489 |
490 | \textbf{Exception:} Tidak memiliki \textit{exception}
491 |
492 | \item \texttt{public double calculateIPS()}
493 |

```

```

494 Menghitung IPS semester terakhir sampai saat ini, dengan aturan:
495 <ul>
496 <li>Kuliah yang tidak lulus <em>dihitung</em>.
497 </ul>
498 Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
499
500 \textbf{Parameter:}
501 \begin{itemize}
502 \item Tidak memiliki parameter \textit{method}
503 \end{itemize}
504 \textbf{Return Value:} nilai IPS sampai saat ini
505
506 \textbf{Exception:} ArrayIndexOutOfBoundsException jika belum ada nilai satupun
507
508 \item \texttt{public int calculateSKSLulus()}
509
510 Menghitung jumlah SKS lulus mahasiswa saat ini.
511 Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
512
513 \textbf{Parameter:}
514 \begin{itemize}
515 \item Tidak memiliki parameter \textit{method}
516 \end{itemize}
517 \textbf{Return Value:} SKS Lulus
518
519 \textbf{Exception:} Tidak memiliki \textit{exception}
520
521 \item \texttt{public int calculateSKSTempuh(boolean lulusSaja)}
522
523 Menghitung jumlah SKS tempuh mahasiswa saat ini.
524 Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
525
526 \textbf{Parameter:}
527 \begin{itemize}
528 \item \texttt{boolean lulusSaja} -
529 set true jika ingin membuang SKS tidak lulus
530 \end{itemize}
531 \textbf{Return Value:} SKS tempuh
532
533 \textbf{Exception:} Tidak memiliki \textit{exception}
534
535 \item \texttt{public Set calculateTahunSemesterAktif()}
536
537 Mendapatkan seluruh tahun semester di mana mahasiswa ini tercatat
538 sebagai mahasiswa aktif, dengan strategi memeriksa riwayat nilainya.
539 Jika ada satu nilai saja pada sebuah tahun semester, maka dianggap
540 aktif pada semester tersebut.
541
542 \textbf{Parameter:}
543 \begin{itemize}
544 \item Tidak memiliki parameter \textit{method}
545 \end{itemize}
546 \textbf{Return Value:} kumpulan tahun semester di mana mahasiswa ini aktif
547
548 \textbf{Exception:} Tidak memiliki \textit{exception}
549
550 \item \texttt{public boolean hasLulusKuliah(java.lang.String kodeMataKuliah)}
551
552 Memeriksa apakah mahasiswa ini sudah lulus mata kuliah tertentu. Kompleksitas  $O(n)$ .
553 Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah mengandung nilai per mata kuliah!
554 Note: jika yang dimiliki adalah MataKuliah, gunakanlah MataKuliahgetKode().
555
556 \textbf{Parameter:}
557 \begin{itemize}
558 \item \texttt{String kodeMataKuliah} -
559 kode mata kuliah yang ingin diperiksa kelulusannya.
560 \end{itemize}
561 \textbf{Return Value:} true jika sudah pernah mengambil dan lulus, false jika belum
562
563 \textbf{Exception:} Tidak memiliki \textit{exception}
564
565 \item \texttt{public boolean hasTempuhKuliah(java.lang.String kodeMataKuliah)}
566
567 Memeriksa apakah mahasiswa ini sudah pernah menempuh mata kuliah tertentu. Kompleksitas  $O(n)$ .
568 Sebelum memanggil method ini, \texttt{getRiwayatNilai()} harus sudah ada isinya!
569 Note: jika yang dimiliki adalah MataKuliah, gunakanlah MataKuliahgetKode().
570
571 \textbf{Parameter:}
572 \begin{itemize}
573 \item \texttt{String kodeMataKuliah} -
574 kode mata kuliah yang ingin diperiksa.
575 \end{itemize}
576 \textbf{Return Value:} true jika sudah pernah mengambil, false jika belum
577
578 \textbf{Exception:} Tidak memiliki \textit{exception}
579
580 \item \texttt{public int getTahunAngkatan()}
581
582 Mendapatkan tahun angkatan mahasiswa ini, berdasarkan NPM nya
583
584 \textbf{Parameter:}
585 \begin{itemize}
586 \item Tidak memiliki parameter \textit{method}
587 \end{itemize}
588 \textbf{Return Value:} tahun angkatan
589
590 \textbf{Exception:} Tidak memiliki \textit{exception}
591
592 \item \texttt{public String toString()}

```

```

593 |
594 |
595 |
596 | \textbf{Parameter:}
597 | \begin{itemize}
598 | \item Tidak memiliki parameter \textit{method}
599 | \end{itemize}
600 | \textbf{Return Value:} Tidak memiliki \textit{return value}
601 |
602 | \textbf{Exception:} Tidak memiliki \textit{exception}
603 |
604 | \end{itemize}
605 | \item \texttt{Mahasiswa.Nilai}
606 |
607 | Merepresentasikan nilai yang ada di riwayat nilai mahasiswa
608 |
609 | Atribut yang dimiliki kelas ini adalah sebagai berikut.
610 | \begin{itemize}
611 | \item \texttt{TahunSemester tahunSemester} - Tahun dan Semester kuliah ini diambil
612 | \item \texttt{MataKuliah mataKuliah} - Mata kuliah yang diambil
613 | \item \texttt{Character kelas} - Kelas kuliah
614 | \item \texttt{Double nilaiART} - Nilai ART
615 | \item \texttt{Double nilaiUTS} - Nilai UTS
616 | \item \texttt{Double nilaiUAS} - Nilai UAS
617 | \item \texttt{Character nilaiAkhir} - Nilai Akhir
618 | \end{itemize}
619 | \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
620 | \begin{itemize}
621 | \item \texttt{public MataKuliah getMataKuliah()}
622 |
623 |
624 |
625 | \textbf{Parameter:}
626 | \begin{itemize}
627 | \item Tidak memiliki parameter \textit{method}
628 | \end{itemize}
629 | \textbf{Return Value:} Tidak memiliki \textit{return value}
630 |
631 | \textbf{Exception:} Tidak memiliki \textit{exception}
632 |
633 | \item \texttt{public Character getKelas()}
634 |
635 |
636 |
637 | \textbf{Parameter:}
638 | \begin{itemize}
639 | \item Tidak memiliki parameter \textit{method}
640 | \end{itemize}
641 | \textbf{Return Value:} Tidak memiliki \textit{return value}
642 |
643 | \textbf{Exception:} Tidak memiliki \textit{exception}
644 |
645 | \item \texttt{public Double getNilaiART()}
646 |
647 |
648 |
649 | \textbf{Parameter:}
650 | \begin{itemize}
651 | \item Tidak memiliki parameter \textit{method}
652 | \end{itemize}
653 | \textbf{Return Value:} Tidak memiliki \textit{return value}
654 |
655 | \textbf{Exception:} Tidak memiliki \textit{exception}
656 |
657 | \item \texttt{public Double getNilaiUTS()}
658 |
659 |
660 |
661 | \textbf{Parameter:}
662 | \begin{itemize}
663 | \item Tidak memiliki parameter \textit{method}
664 | \end{itemize}
665 | \textbf{Return Value:} Tidak memiliki \textit{return value}
666 |
667 | \textbf{Exception:} Tidak memiliki \textit{exception}
668 |
669 | \item \texttt{public Double getNilaiUAS()}
670 |
671 |
672 |
673 | \textbf{Parameter:}
674 | \begin{itemize}
675 | \item Tidak memiliki parameter \textit{method}
676 | \end{itemize}
677 | \textbf{Return Value:} Tidak memiliki \textit{return value}
678 |
679 | \textbf{Exception:} Tidak memiliki \textit{exception}
680 |
681 | \item \texttt{public Character getNilaiAkhir()}
682 |
683 | Mengembalikan nilai akhir dalam bentuk huruf (A, B, C, D, ..., atau K)
684 |
685 | \textbf{Parameter:}
686 | \begin{itemize}
687 | \item Tidak memiliki parameter \textit{method}
688 | \end{itemize}
689 | \textbf{Return Value:} nilai akhir dalam huruf, atau null jika tidak ada.
690 |
691 | \textbf{Exception:} Tidak memiliki \textit{exception}

```

```

692 \item \texttt{public Double getAngkaAkhir()}
693 Mendapatkan nilai akhir dalam bentuk angka
694
695 \textbf{Parameter:}
696 \begin{itemize}
697 \item Tidak memiliki parameter \textit{method}
698 \end{itemize}
699 \textbf{Return Value:} nilai akhir dalam angka, atau null jika getNilaiAkhir() mengembalikan 'K' atau null
700
701 \textbf{Exception:} Tidak memiliki \textit{exception}
702
703 \item \texttt{public TahunSemester getTahunSemester()}
704
705 \textbf{Parameter:}
706 \begin{itemize}
707 \item Tidak memiliki parameter \textit{method}
708 \end{itemize}
709 \textbf{Return Value:} Tidak memiliki \textit{return value}
710
711 \textbf{Exception:} Tidak memiliki \textit{exception}
712
713 \item \texttt{public int getTahunAjaran()}
714
715 \textbf{Parameter:}
716 \begin{itemize}
717 \item Tidak memiliki parameter \textit{method}
718 \end{itemize}
719 \textbf{Return Value:} Tidak memiliki \textit{return value}
720
721 \textbf{Exception:} Tidak memiliki \textit{exception}
722
723 \item \texttt{public Semester getSemester()}
724
725 \textbf{Parameter:}
726 \begin{itemize}
727 \item Tidak memiliki parameter \textit{method}
728 \end{itemize}
729 \textbf{Return Value:} Tidak memiliki \textit{return value}
730
731 \textbf{Exception:} Tidak memiliki \textit{exception}
732
733 \item \texttt{public String toString()}
734
735 \textbf{Parameter:}
736 \begin{itemize}
737 \item Tidak memiliki parameter \textit{method}
738 \end{itemize}
739 \textbf{Return Value:} Tidak memiliki \textit{return value}
740
741 \textbf{Exception:} Tidak memiliki \textit{exception}
742
743 \textbf{Parameter:}
744 \begin{itemize}
745 \item Tidak memiliki parameter \textit{method}
746 \end{itemize}
747 \textbf{Return Value:} Tidak memiliki \textit{return value}
748
749 \textbf{Exception:} Tidak memiliki \textit{exception}
750
751 \end{itemize}
752
753 \item \texttt{Mahasiswa.Nilai.ChronologicalComparator implements Comparator}
754
755 Pembanding antara satu nilai dengan nilai lainnya, secara
756 kronologis waktu pengambilan.
757
758 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
759 \begin{itemize}
760 \item \texttt{public int compare(id.ac.unpar.siamodels.Mahasiswa.Nilai o1, id.ac.unpar.siamodels.Mahasiswa.Nilai o2)}
761
762 \textbf{Parameter:}
763 \begin{itemize}
764 \item \texttt{Mahasiswa.Nilai o1} -
765 \item \texttt{Mahasiswa.Nilai o2} -
766 \end{itemize}
767 \textbf{Return Value:} Tidak memiliki \textit{return value}
768
769 \textbf{Exception:} Tidak memiliki \textit{exception}
770
771 \textbf{Override:} \texttt{compare} dari kelas \texttt{Object}
772
773 \end{itemize}
774
775 \item \texttt{JadwalKuliah}
776
777 Atribut yang dimiliki kelas ini adalah sebagai berikut.
778 \begin{itemize}
779 \item \texttt{MataKuliah mataKuliah} -
780 \item \texttt{Character kelas} -
781 \item \texttt{DayOfWeek hari} -
782 \item \texttt{LocalTime waktuMulai} -
783 \item \texttt{LocalTime waktuSelesai} -
784 \item \texttt{String lokasi} -
785 \item \texttt{Dosen pengajar} -
786 \end{itemize}
787
788 \end{itemize}

```



```

791 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
792 \begin{itemize}
793 \item \texttt{public MataKuliah getMataKuliah()}
794
795
796
797 \textbf{Parameter:}
798 \begin{itemize}
799 \item Tidak memiliki parameter \textit{method}
800 \end{itemize}
801 \textbf{Return Value:} Tidak memiliki \textit{return value}
802
803 \textbf{Exception:} Tidak memiliki \textit{exception}
804
805 \item \texttt{public void setMataKuliah(id.ac.unpar.siamodels.MataKuliah mataKuliah)}
806
807
808
809 \textbf{Parameter:}
810 \begin{itemize}
811 \item \texttt{MataKuliah mataKuliah} -
812 \end{itemize}
813 \textbf{Return Value:} Tidak memiliki \textit{return value}
814
815 \textbf{Exception:} Tidak memiliki \textit{exception}
816
817 \item \texttt{public Character getKelas()}
818
819
820
821 \textbf{Parameter:}
822 \begin{itemize}
823 \item Tidak memiliki parameter \textit{method}
824 \end{itemize}
825 \textbf{Return Value:} Tidak memiliki \textit{return value}
826
827 \textbf{Exception:} Tidak memiliki \textit{exception}
828
829 \item \texttt{public void setKelas(java.lang.Character kelas)}
830
831
832
833 \textbf{Parameter:}
834 \begin{itemize}
835 \item \texttt{Character kelas} -
836 \end{itemize}
837 \textbf{Return Value:} Tidak memiliki \textit{return value}
838
839 \textbf{Exception:} Tidak memiliki \textit{exception}
840
841 \item \texttt{public DayOfWeek getHari()}
842
843
844
845 \textbf{Parameter:}
846 \begin{itemize}
847 \item Tidak memiliki parameter \textit{method}
848 \end{itemize}
849 \textbf{Return Value:} Tidak memiliki \textit{return value}
850
851 \textbf{Exception:} Tidak memiliki \textit{exception}
852
853 \item \texttt{public void setHari(java.time.DayOfWeek hari)}
854
855
856
857 \textbf{Parameter:}
858 \begin{itemize}
859 \item \texttt{DayOfWeek hari} -
860 \end{itemize}
861 \textbf{Return Value:} Tidak memiliki \textit{return value}
862
863 \textbf{Exception:} Tidak memiliki \textit{exception}
864
865 \item \texttt{public LocalTime getWaktuMulai()}
866
867
868
869 \textbf{Parameter:}
870 \begin{itemize}
871 \item Tidak memiliki parameter \textit{method}
872 \end{itemize}
873 \textbf{Return Value:} Tidak memiliki \textit{return value}
874
875 \textbf{Exception:} Tidak memiliki \textit{exception}
876
877 \item \texttt{public void setWaktuMulai(java.time.LocalTime waktuMulai)}
878
879
880
881 \textbf{Parameter:}
882 \begin{itemize}
883 \item \texttt{LocalTime waktuMulai} -
884 \end{itemize}
885 \textbf{Return Value:} Tidak memiliki \textit{return value}
886
887 \textbf{Exception:} Tidak memiliki \textit{exception}
888
889 \item \texttt{public LocalTime getWaktuSelesai()}

```

```

890
891
892
893 \textbf{Parameter:}
894 \begin{itemize}
895 \item Tidak memiliki parameter \textit{method}
896 \end{itemize}
897 \textbf{Return Value:} Tidak memiliki \textit{return value}
898
899 \textbf{Exception:} Tidak memiliki \textit{exception}
900
901 \item \texttt{public void setWaktuSelesai(java.time.LocalTime waktuSelesai)}
902
903
904
905 \textbf{Parameter:}
906 \begin{itemize}
907 \item \texttt{LocalTime waktuSelesai} -
908 \end{itemize}
909 \textbf{Return Value:} Tidak memiliki \textit{return value}
910
911 \textbf{Exception:} Tidak memiliki \textit{exception}
912
913 \item \texttt{public String getLocation()}
914
915
916
917 \textbf{Parameter:}
918 \begin{itemize}
919 \item Tidak memiliki parameter \textit{method}
920 \end{itemize}
921 \textbf{Return Value:} Tidak memiliki \textit{return value}
922
923 \textbf{Exception:} Tidak memiliki \textit{exception}
924
925 \item \texttt{public void setLocation(java.lang.String lokasi)}
926
927
928
929 \textbf{Parameter:}
930 \begin{itemize}
931 \item \texttt{String lokasi} -
932 \end{itemize}
933 \textbf{Return Value:} Tidak memiliki \textit{return value}
934
935 \textbf{Exception:} Tidak memiliki \textit{exception}
936
937 \item \texttt{public Dosen getPengajar()}
938
939
940
941 \textbf{Parameter:}
942 \begin{itemize}
943 \item Tidak memiliki parameter \textit{method}
944 \end{itemize}
945 \textbf{Return Value:} Tidak memiliki \textit{return value}
946
947 \textbf{Exception:} Tidak memiliki \textit{exception}
948
949 \item \texttt{public void setPengajar(id.ac.unpar.siamodels.Dosen pengajar)}
950
951
952
953 \textbf{Parameter:}
954 \begin{itemize}
955 \item \texttt{Dosen pengajar} -
956 \end{itemize}
957 \textbf{Return Value:} Tidak memiliki \textit{return value}
958
959 \textbf{Exception:} Tidak memiliki \textit{exception}
960
961 \item \texttt{public String getWaktuString()}
962
963
964
965 \textbf{Parameter:}
966 \begin{itemize}
967 \item Tidak memiliki parameter \textit{method}
968 \end{itemize}
969 \textbf{Return Value:} Tidak memiliki \textit{return value}
970
971 \textbf{Exception:} Tidak memiliki \textit{exception}
972
973 \item \texttt{public static DayOfWeek indonesianToDayOfWeek(java.lang.String indonesian)}
974
975 Converts Indonesian day names to \texttt{DayOfWeek} enumeration.
976
977 \textbf{Parameter:}
978 \begin{itemize}
979 \item \texttt{String indonesian} -
980 the day name in Indonesian
981 \end{itemize}
982 \textbf{Return Value:} DayOfWeek object or null if not found.
983
984 \textbf{Exception:} Tidak memiliki \textit{exception}
985
986 \end{itemize}
987 \item \texttt{MataKuliah}
988

```

```

989|
990|
991| Atribut yang dimiliki kelas ini adalah sebagai berikut.
992| \begin{itemize}
993| \item \texttt{String kode} -
994| \item \texttt{String nama} -
995| \item \texttt{Integer sks} -
996| \end{itemize}
997| \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
998| \begin{itemize}
999| \item \texttt{public String getCode()}
1000|
1001|
1002|
1003| \textbf{Parameter:}
1004| \begin{itemize}
1005| \item Tidak memiliki parameter \textit{method}
1006| \end{itemize}
1007| \textbf{Return Value:} Tidak memiliki \textit{return value}
1008|
1009| \textbf{Exception:} Tidak memiliki \textit{exception}
1010|
1011| \item \texttt{public String getName()}
1012|
1013|
1014|
1015| \textbf{Parameter:}
1016| \begin{itemize}
1017| \item Tidak memiliki parameter \textit{method}
1018| \end{itemize}
1019| \textbf{Return Value:} Tidak memiliki \textit{return value}
1020|
1021| \textbf{Exception:} Tidak memiliki \textit{exception}
1022|
1023| \item \texttt{public Integer getSks()}
1024|
1025|
1026|
1027| \textbf{Parameter:}
1028| \begin{itemize}
1029| \item Tidak memiliki parameter \textit{method}
1030| \end{itemize}
1031| \textbf{Return Value:} Tidak memiliki \textit{return value}
1032|
1033| \textbf{Exception:} Tidak memiliki \textit{exception}
1034|
1035| \end{itemize}
1036| \item \texttt{Dosen}
1037|
1038|
1039|
1040| Atribut yang dimiliki kelas ini adalah sebagai berikut.
1041| \begin{itemize}
1042| \item \texttt{String nik} -
1043| \item \texttt{String nama} -
1044| \end{itemize}
1045| \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1046| \begin{itemize}
1047| \item \texttt{public String getNik()}
1048|
1049|
1050|
1051| \textbf{Parameter:}
1052| \begin{itemize}
1053| \item Tidak memiliki parameter \textit{method}
1054| \end{itemize}
1055| \textbf{Return Value:} Tidak memiliki \textit{return value}
1056|
1057| \textbf{Exception:} Tidak memiliki \textit{exception}
1058|
1059| \item \texttt{public void setNik(java.lang.String nik)}
1060|
1061|
1062|
1063| \textbf{Parameter:}
1064| \begin{itemize}
1065| \item \texttt{String nik} -
1066| \end{itemize}
1067| \textbf{Return Value:} Tidak memiliki \textit{return value}
1068|
1069| \textbf{Exception:} Tidak memiliki \textit{exception}
1070|
1071| \item \texttt{public String getName()}
1072|
1073|
1074|
1075| \textbf{Parameter:}
1076| \begin{itemize}
1077| \item Tidak memiliki parameter \textit{method}
1078| \end{itemize}
1079| \textbf{Return Value:} Tidak memiliki \textit{return value}
1080|
1081| \textbf{Exception:} Tidak memiliki \textit{exception}
1082|
1083| \item \texttt{public void setName(java.lang.String nama)}
1084|
1085|
1086|
1087| \textbf{Parameter:}

```

```

1088 \begin{itemize}
1089 \item \texttt{String nama} -
1090 \end{itemize}
1091 \textbf{Return Value}: Tidak memiliki \textit{return value}
1092
1093 \textbf{Exception}: Tidak memiliki \textit{exception}
1094
1095 \item \texttt{public boolean equals(java.lang.Object arg0)}
1096
1097
1098
1099 \textbf{Parameter:}
1100 \begin{itemize}
1101 \item \texttt{Object arg0} -
1102 \end{itemize}
1103 \textbf{Return Value}: Tidak memiliki \textit{return value}
1104
1105 \textbf{Exception}: Tidak memiliki \textit{exception}
1106
1107 \end{itemize}
1108 \item \texttt{MKU008}
1109
1110 Mendalami perilaku sehari-hari yang baik dalam bermasyarakat.
1111
1112 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{IIE210}
1113
1114
1115
1116 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF203 implements HasPrasyarat}
1117
1118 Mata kuliah ini memperkenalkan kepada mahasiswa konsep struktur diskret yang
1119 digunakan pada bidang informatika diantaranya graph, pohon dan finite state
1120 machine
1121
1122 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1123 \begin{itemize}
1124 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1125
1126
1127
1128 \textbf{Parameter:}
1129 \begin{itemize}
1130 \item \texttt{Mahasiswa mahasiswa} -
1131 \item \texttt{java.util.List reasonsContainer} -
1132 \end{itemize}
1133 \textbf{Return Value}: Tidak memiliki \textit{return value}
1134
1135 \textbf{Exception}: Tidak memiliki \textit{exception}
1136
1137 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1138
1139 \end{itemize}
1140 \item \texttt{AIF311 implements HasPrasyarat}
1141
1142 \item \texttt{AIF311 implements HasPraktikum}
1143
1144 Kuliah Pemrograman Fungsional bertujuan untuk: 1. memperkenalkan paradigma
1145 pemrograman fungsional, yaitu sebuah pemrograman yang didasarkan pada konsep
1146 pemetaan dan fungsi matematika. Penyelesaian suatu masalah didasari atas
1147 aplikasi dari fungsi-fungsi tersebut. 2. memberikan dasar-dasar pemrograman
1148 fungsional dengan menggunakan bahasa fungsional Haskell.
1149
1150 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1151 \begin{itemize}
1152 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1153
1154
1155
1156 \textbf{Parameter:}
1157 \begin{itemize}
1158 \item \texttt{Mahasiswa mahasiswa} -
1159 \item \texttt{java.util.List reasonsContainer} -
1160 \end{itemize}
1161 \textbf{Return Value}: Tidak memiliki \textit{return value}
1162
1163 \textbf{Exception}: Tidak memiliki \textit{exception}
1164
1165 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1166
1167 \end{itemize}
1168 \item \texttt{AIF192}
1169
1170
1171
1172 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF468}
1173
1174
1175
1176 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{IIE103}
1177
1178
1179
1180 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF385}
1181
1182
1183
1184 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF106}
1185
1186 Mata kuliah ini memberikan pengetahuan tentang cara kerja komputer, dimulai

```

---

```

1187 | dari representasi data dan berbagai macam operasinya. Selanjutnya, juga
1188 | diperkenalkan bagaimana merepresentasikan suatu fungsi dalam rangkaian
1189 | gerbang logika, dan bagaimana menyederhanakannya. Berbagai rangkaian dasar
1190 | yang digunakan di dalam komputer juga dipekenalkan. Mahasiswa juga akan
1191 | mempelajari komponen komputer, misalnya register dan memori.
1192 |
1193 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF281}
1194 |
1195 |
1196 |
1197 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{EAA102}
1198 |
1199 |
1200 |
1201 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF405 implements HasPrasyarat}
1202 |
1203 | \item \texttt{AIF405 implements HasPraktikum}
1204 |
1205 | Mata kuliah ini merupakan lanjutan dari Projek Sistem Informasi 1 dan
1206 | memberikan kesempatan bagi mahasiswa untuk melanjutkan/mengembangkan
1207 | perancangan sitem pada organisasi studi kasus, mengimplementasikan rancangan
1208 | dan melakukan pengujian perangkat lunak;
1209 |
1210 | Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1211 | \begin{itemize}
1212 | \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1213 |
1214 |
1215 |
1216 | \textbf{Parameter:}
1217 | \begin{itemize}
1218 | \item \texttt{Mahasiswa mahasiswa} -
1219 | \item \texttt{java.util.List reasonsContainer} -
1220 | \end{itemize}
1221 | \textbf{Return Value:} Tidak memiliki \textit{return value}
1222 |
1223 | \textbf{Exception:} Tidak memiliki \textit{exception}
1224 |
1225 | \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1226 |
1227 | \end{itemize}
1228 | \item \texttt{APS182}
1229 |
1230 |
1231 |
1232 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{MKU004}
1233 |
1234 | Fenomenologi Agama merupakan bagian yang tak terpisahkan dari kajian filosofis, kritis,
1235 | rasional, dan obyektif mengenai substansi ajaran agama. Fenomenologi merupakan sebuah
1236 | disiplin ilmu yang secara kritis-rasional mengkaji fenomena dan dinamika kehidupan manusia
1237 | beragama, dari upaya menjadikan Tuhan sebagai tujuan sesembahan sampai menempatkan Tuhan
1238 | sebagai instrumen legitimasi untuk melakukan tindakan yang justru bertolak belakang dengan
1239 | kehendak Tuhan yang disembah. Sehubungan dengan itu, kritik konstruktif terhadap perilaku
1240 | manusia beragama menjadi salah satu poin utama dalam mata kuliah ini. Kesiediaan untuk
1241 | melakukan otoritik terhadap agama sendiri erat terkait dengan upaya menemukan kembali nilai
1242 | sejati agama atau otentisitas hidup beragama.
1243 |
1244 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{MKU012}
1245 |
1246 | Perkuliahan logika ditujukan untuk memberikan dasar-dasar ketrampilan berpikir rasional dan
1247 | sistematis. Isinya mencakup ketrampilan berpikir deduktif dan induktif, seperti silogisme,
1248 | argumen analogikal dan generalisasi induktif. Pembahasan teoretis disertai pula dengan
1249 | pelatihan praktis yang diarahkan pada proses berpikir. Untuk menajamkan kemampuan berpikir
1250 | tersebut, mahasiswa dilatih pula mengidentifikasi kerancuan-kerancuan (fallacies) yang sering
1251 | dijumpai baik dalam kehidupan sehari-hari maupun dalam konteks akademik.
1252 |
1253 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF389}
1254 |
1255 |
1256 |
1257 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AMS190}
1258 |
1259 |
1260 |
1261 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AMS191}
1262 |
1263 |
1264 |
1265 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AMS200}
1266 |
1267 |
1268 |
1269 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF330}
1270 |
1271 |
1272 |
1273 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF388}
1274 |
1275 |
1276 |
1277 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF465}
1278 |
1279 |
1280 |
1281 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF453 implements HasPrasyarat}
1282 |
1283 | Mata kuliah ini memperkenalkan kebutuhan organisasi terhadap sistem business
1284 | intelligent (BI) dan pemanfaatan BI untuk organisasi; memperkenalkan konsep
1285 | sistem business intelligent dan komponennya; Mempelajari teknik-teknik

```

```

1286 analisis data bisnis dan visualisasi hasil analisis; Mempelajari konsep data
1287 warehouse dan perancangannya dan fungsi OLAP; Mempraktekkan teknik-teknik
1288 analisis data dan visualisasi hasil analisis.
1289
1290 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1291 \begin{itemize}
1292 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1293
1294
1295
1296 \textbf{Parameter:}
1297 \begin{itemize}
1298 \item \texttt{Mahasiswa mahasiswa} -
1299 \item \texttt{java.util.List reasonsContainer} -
1300 \end{itemize}
1301 \textbf{Return Value:} Tidak memiliki \textit{return value}
1302
1303 \textbf{Exception:} Tidak memiliki \textit{exception}
1304
1305 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1306
1307 \end{itemize}
1308 \item \texttt{AIF280}
1309
1310
1311
1312 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF445 implements HasPrasyarat}
1313
1314
1315
1316 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1317 \begin{itemize}
1318 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1319
1320
1321
1322 \textbf{Parameter:}
1323 \begin{itemize}
1324 \item \texttt{Mahasiswa mahasiswa} -
1325 \item \texttt{java.util.List reasonsContainer} -
1326 \end{itemize}
1327 \textbf{Return Value:} Tidak memiliki \textit{return value}
1328
1329 \textbf{Exception:} Tidak memiliki \textit{exception}
1330
1331 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1332
1333 \end{itemize}
1334 \item \texttt{AIF469 implements HasPrasyarat}
1335
1336 Mata kuliah ini mengajarkan kepada mahasiswa teknik-teknik untuk membuat
1337 layanan berbasis web. Mahasiswa diperkenalkan dengan standar-standar seperti
1338 HTTP, XML, JSON dan diajarkan untuk memanfaatkannya dalam membuat maupun
1339 menggunakan layanan pihak ketiga. Dalam kuliah ini, juga akan diperkenalkan
1340 minimal satu layanan pihak ketiga yang dapat dimanfaatkan mahasiswa, seperti
1341 Google Places Web Service.
1342
1343 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1344 \begin{itemize}
1345 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1346
1347
1348
1349 \textbf{Parameter:}
1350 \begin{itemize}
1351 \item \texttt{Mahasiswa mahasiswa} -
1352 \item \texttt{java.util.List reasonsContainer} -
1353 \end{itemize}
1354 \textbf{Return Value:} Tidak memiliki \textit{return value}
1355
1356 \textbf{Exception:} Tidak memiliki \textit{exception}
1357
1358 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1359
1360 \end{itemize}
1361 \item \texttt{AIF486}
1362
1363
1364
1365 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{IIE207}
1366
1367
1368
1369 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF202 implements HasPrasyarat}
1370
1371 \item \texttt{AIF202 implements HasResponsi}
1372
1373 \item \texttt{AIF202 implements HasPraktikum}
1374
1375 Mata kuliah ini memperkenalkan kepada mahasiswa beberapa algoritma dan
1376 struktur data, alternatif cara implementasinya, dan analisis kompleksitas
1377 waktunya. Mahasiswa diberikan beberapa masalah komputasi yang harus
1378 diselesaikan dengan menggunakan algoritma atau struktur data yang sudah
1379 diperkenalkan dan mengimplementasikannya dalam bahasa pemrograman Java.
1380
1381 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1382 \begin{itemize}
1383 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1384

```

```

1385 |
1386 |
1387 | \textbf{Parameter:}
1388 | \begin{itemize}
1389 | \item \texttt{Mahasiswa mahasiswa} -
1390 | \item \texttt{java.util.List reasonsContainer} -
1391 | \end{itemize}
1392 | \textbf{Return Value:} Tidak memiliki \textit{return value}
1393 |
1394 | \textbf{Exception:} Tidak memiliki \textit{exception}
1395 |
1396 | \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1397 |
1398 | \end{itemize}
1399 | \item \texttt{AIF347}
1400 |
1401 |
1402 |
1403 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{MKU009}
1404 |
1405 | Tujuan dari mata kuliah ini adalah untuk mendalami keterampilan berbahasa Indonesia, agar
1406 | mampu mengkomunikasikan hasil pemikiran serta meningkatkan keterampilan dalam menyusun karya
1407 | ilmiah. Mata kuliah Bahasa Indonesia ini dimulai dengan mempelajari penulisan kata baku dan
1408 | non baku serta pengungkapan pikiran dengan puntuasi yang benar. Selanjutnya dipelajari
1409 | penyusunan kalimat yang baku serta menghubungkan kalimat-kalimat yang padu dalam menuangkan
1410 | gagasan dalam sebuah paragraf. Selain itu, dalam matakuliah ini dipelajari cara menyusun
1411 | surat dinas yang jelas dan komunikatif. Di akhir kuliah ini, mahasiswa diberi tugas
1412 | penyusunan makalah dengan benar.
1413 |
1414 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{ESM203}
1415 |
1416 |
1417 |
1418 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{APS402 implements HasPrasyarat}
1419 |
1420 |
1421 |
1422 | Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1423 | \begin{itemize}
1424 | \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1425 |
1426 |
1427 |
1428 | \textbf{Parameter:}
1429 | \begin{itemize}
1430 | \item \texttt{Mahasiswa mahasiswa} -
1431 | \item \texttt{java.util.List reasonsContainer} -
1432 | \end{itemize}
1433 | \textbf{Return Value:} Tidak memiliki \textit{return value}
1434 |
1435 | \textbf{Exception:} Tidak memiliki \textit{exception}
1436 |
1437 | \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1438 |
1439 | \end{itemize}
1440 | \item \texttt{AIF306 implements HasPrasyarat}
1441 |
1442 | Mata kuliah ini bertujuan untuk memberikan pengalaman bagi mahasiswa dalam
1443 | mengerjakan proyek dengan teknologi-teknologi terkini, secara berkelompok.
1444 | Teknologi-teknologi yang digunakan pada kuliah ini tidak spesifik dan dapat
1445 | berubah seiring perkembangan teknologi maupun disesuaikan dengan kompetensi
1446 | dosen pengajar. Beberapa teknologi yang dapat dimanfaatkan antara lain: DVCS
1447 | tool menggunakan Git + Github, Mobile native app (Android, iOS, dll), dan
1448 | responsive web design.
1449 |
1450 | Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1451 | \begin{itemize}
1452 | \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1453 |
1454 |
1455 |
1456 | \textbf{Parameter:}
1457 | \begin{itemize}
1458 | \item \texttt{Mahasiswa mahasiswa} -
1459 | \item \texttt{java.util.List reasonsContainer} -
1460 | \end{itemize}
1461 | \textbf{Return Value:} Tidak memiliki \textit{return value}
1462 |
1463 | \textbf{Exception:} Tidak memiliki \textit{exception}
1464 |
1465 | \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1466 |
1467 | \end{itemize}
1468 | \item \texttt{AKS122}
1469 |
1470 |
1471 |
1472 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{MKU002}
1473 |
1474 | Pendidikan Kewarganegaraan menjelaskan pentingnya pemahaman tentang identitas nasional
1475 | Indonesia, hak dan kewajiban warga negara Indonesia serta hubungannya dengan hak dan
1476 | kewajiban asasi manusia. Materi kuliah mencakup juga wawasan nusantara, ketahanan nasional,
1477 | politik dan strategi nasional, serta implementasinya dalam kehidupan bermasyarakat, berbangsa
1478 | dan bernegara kesatuan Republik Indonesia.
1479 |
1480 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF462}
1481 |
1482 |
1483 |

```

```

1484 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF360 implements HasPrasyarat}
1485
1486
1487
1488 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1489 \begin{itemize}
1490 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1491
1492
1493
1494 \textbf{Parameter:}
1495 \begin{itemize}
1496 \item \texttt{Mahasiswa mahasiswa} -
1497 \item \texttt{java.util.List reasonsContainer} -
1498 \end{itemize}
1499 \textbf{Return Value:} Tidak memiliki \textit{return value}
1500
1501 \textbf{Exception:} Tidak memiliki \textit{exception}
1502
1503 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1504
1505 \end{itemize}
1506 \item \texttt{AIF337}
1507
1508
1509
1510 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF458 implements HasPrasyarat}
1511
1512
1513
1514 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1515 \begin{itemize}
1516 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1517
1518
1519
1520 \textbf{Parameter:}
1521 \begin{itemize}
1522 \item \texttt{Mahasiswa mahasiswa} -
1523 \item \texttt{java.util.List reasonsContainer} -
1524 \end{itemize}
1525 \textbf{Return Value:} Tidak memiliki \textit{return value}
1526
1527 \textbf{Exception:} Tidak memiliki \textit{exception}
1528
1529 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1530
1531 \end{itemize}
1532 \item \texttt{AIF301 implements HasPrasyarat}
1533
1534 Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar tentang *sistem
1535 cerdas dan komponen-komponennya. " Terdapat 4 topik utama yang dibahas yaitu
1536 teknik pencarian untuk *penyelesaian masalah, representasi pengetahuan dalam
1537 sistem *cerdas, pemodelan ketidakpastian dalam masalah dan teknik
1538 pembelajaran mesin.
1539
1540 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1541 \begin{itemize}
1542 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1543
1544
1545
1546 \textbf{Parameter:}
1547 \begin{itemize}
1548 \item \texttt{Mahasiswa mahasiswa} -
1549 \item \texttt{java.util.List reasonsContainer} -
1550 \end{itemize}
1551 \textbf{Return Value:} Tidak memiliki \textit{return value}
1552
1553 \textbf{Exception:} Tidak memiliki \textit{exception}
1554
1555 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1556
1557 \end{itemize}
1558 \item \texttt{AIF182}
1559
1560
1561
1562 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{ESM204}
1563
1564
1565
1566 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF205 implements HasPrasyarat}
1567
1568 Mata kuliah ini memperkenalkan kepada mahasiswa arsitektur komputer
1569 sederhana, modern, dan Advance. Perbedaan, kelebihan dan kekurangan untuk
1570 masing-masing arsitektur. Selain itu mahasiswa juga mempelajari cara kerja
1571 dari komponen-komponen komputer, terutama memory, cache, system BUS dan
1572 input/output.
1573
1574 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1575 \begin{itemize}
1576 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1577
1578
1579
1580 \textbf{Parameter:}
1581 \begin{itemize}
1582 \item \texttt{Mahasiswa mahasiswa} -

```



```

1583 \item \texttt{java.util.List reasonsContainer} -
1584 \end{itemize}
1585 \textbf{Return Value}: Tidak memiliki \textit{return value}
1586
1587 \textbf{Exception}: Tidak memiliki \textit{exception}
1588
1589 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1590
1591 \end{itemize}
1592 \item \texttt{AIF317 implements HasPrasyarat}
1593
1594
1595 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1596 \begin{itemize}
1597 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1598
1599
1600 \textbf{Parameter}:
1601 \begin{itemize}
1602 \item \texttt{Mahasiswa mahasiswa} -
1603 \item \texttt{java.util.List reasonsContainer} -
1604 \end{itemize}
1605 \textbf{Return Value}: Tidak memiliki \textit{return value}
1606
1607 \textbf{Exception}: Tidak memiliki \textit{exception}
1608
1609 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1610
1611 \end{itemize}
1612 \item \texttt{AIF383}
1613
1614
1615 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF442 implements HasPraktikum}
1616
1617 \item \texttt{AIF442 implements HasPrasyarat}
1618
1619
1620 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1621 \begin{itemize}
1622 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1623
1624
1625 \textbf{Parameter}:
1626 \begin{itemize}
1627 \item \texttt{Mahasiswa mahasiswa} -
1628 \item \texttt{java.util.List reasonsContainer} -
1629 \end{itemize}
1630 \textbf{Return Value}: Tidak memiliki \textit{return value}
1631
1632 \textbf{Exception}: Tidak memiliki \textit{exception}
1633
1634 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1635
1636 \end{itemize}
1637 \item \texttt{ESM101}
1638
1639
1640 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF403 implements HasPrasyarat}
1641
1642
1643 1. Memberikan wawasan kepada mahasiswa tentang kemunculan dan pemanfaatan teknologi baru,
1644 khususnya yang berkaitan dengan komputer, dan dampaknya terhadap masyarakat luas.
1645 2. Memberikan kesadaran dan panduan bersikap kepada mahasiswa dalam menghadapi gejala yang
1646 disebabkan oleh munculnya teknologi baru, khususnya yang berkaitan dengan komputer.
1647
1648 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1649 \begin{itemize}
1650 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1651
1652
1653 \textbf{Parameter}:
1654 \begin{itemize}
1655 \item \texttt{Mahasiswa mahasiswa} -
1656 \item \texttt{java.util.List reasonsContainer} -
1657 \end{itemize}
1658 \textbf{Return Value}: Tidak memiliki \textit{return value}
1659
1660 \textbf{Exception}: Tidak memiliki \textit{exception}
1661
1662 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1663
1664 \end{itemize}
1665 \item \texttt{AIF402 implements HasPrasyarat}
1666
1667
1668 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1669 \begin{itemize}
1670 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1671
1672
1673 \textbf{Parameter}:
1674
1675
1676
1677
1678
1679
1680
1681 \textbf{Parameter}:

```

```

1682 \begin{itemize}
1683 \item \texttt{Mahasiswa mahasiswa} -
1684 \item \texttt{java.util.List reasonsContainer} -
1685 \end{itemize}
1686 \textbf{Return Value}: Tidak memiliki \textit{return value}
1687
1688 \textbf{Exception}: Tidak memiliki \textit{exception}
1689
1690 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1691
1692 \end{itemize}
1693 \item \texttt{AIF455}
1694
1695
1696
1697 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF443}
1698
1699
1700
1701 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF101 implements HasPraktikum}
1702
1703 Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar pemrograman
1704 seperti pengulangan dan percabangan, konsep dasar penyimpanan data kontigu
1705 menggunakan array, konsep dasar pemrograman berorientasi objek seperti kelas
1706 & objek, method, dll, termasuk di dalamnya 4 prinsip dasar pemrograman
1707 berorientasi objek : data abstraction, encapsulation, inheritance dan
1708 polymorphism. Selain, itu diberikan masalah-masalah komputasi sederhana
1709 yang harus diselesaikan menggunakan konsep-konsep yang sudah diperkenalkan
1710 dan mengimplementasikannya menggunakan bahasa pemrograman Java
1711
1712 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF382}
1713
1714
1715
1716 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF480}
1717
1718
1719
1720 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF316 implements HasPrasyarat}
1721
1722 \item \texttt{AIF316 implements HasPraktikum}
1723
1724 Mata kuliah ini memperkenalkan konsep-konsep dasar komputasi paralel, dimana sebuah
1725 program yang berjalan secara paralel harus memiliki safety property dan liveness property.
1726 Mahasiswa dikenalkan dengan beberapa teknik pemrograman multi-thread
1727 seperti lock, monitor, barrier, thread pool, dan sebagainya, yang diimplementasikan
1728 dalam bahasa pemrograman Java. Mahasiswa juga dikenalkan dengan beberapa metode untuk
1729 menganalisis kebenaran program baik secara matematis maupun secara praktis dengan bantuan
1730 model checker.
1731
1732 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1733 \begin{itemize}
1734 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1735
1736
1737
1738 \textbf{Parameter}:
1739 \begin{itemize}
1740 \item \texttt{Mahasiswa mahasiswa} -
1741 \item \texttt{java.util.List reasonsContainer} -
1742 \end{itemize}
1743 \textbf{Return Value}: Tidak memiliki \textit{return value}
1744
1745 \textbf{Exception}: Tidak memiliki \textit{exception}
1746
1747 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1748
1749 \end{itemize}
1750 \item \texttt{AIF438 implements HasPrasyarat}
1751
1752 Mata kuliah ini: Memperkenalkan karakteristik dan teknik visualisasi dari
1753 berbagai jenis data yang dapat dianalisis dengan teknik-teknik data mining;
1754 mempelajari teknik-teknik penyiapan data untuk berbagai jenis data dan teknik
1755 data mining; mempraktekkan teknik-teknik penyiapan data untuk menganalisis
1756 data nyata/simulasi dengan memanfaatkan perangkat lunak aplikasi.
1757
1758 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1759 \begin{itemize}
1760 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1761
1762
1763
1764 \textbf{Parameter}:
1765 \begin{itemize}
1766 \item \texttt{Mahasiswa mahasiswa} -
1767 \item \texttt{java.util.List reasonsContainer} -
1768 \end{itemize}
1769 \textbf{Return Value}: Tidak memiliki \textit{return value}
1770
1771 \textbf{Exception}: Tidak memiliki \textit{exception}
1772
1773 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1774
1775 \end{itemize}
1776 \item \texttt{AIF204 implements HasPrasyarat}
1777
1778 \item \texttt{AIF204 implements HasPraktikum}
1779
1780 Mata kuliah ini memperkenalkan konsep dan arsitektur DBMS, mengajarkan

```

```

1781 | aljabar relasional dan SQL serta pemanfaatannya pada pemrograman kueri
1782 | sederhana s/d relatif kompleks. Selain itu, mata kuliah ini juga mengajarkan
1783 | dan mempraktekkan perancangan basisdata untuk masalah sederhana
1784 | (lingkup kecil) termasuk pengembangan program aplikasinya;
1785 |
1786 | Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1787 | \begin{itemize}
1788 | \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1789 |
1790 |
1791 |
1792 | \textbf{Parameter:}
1793 | \begin{itemize}
1794 | \item \texttt{Mahasiswa mahasiswa} -
1795 | \item \texttt{java.util.List reasonsContainer} -
1796 | \end{itemize}
1797 | \textbf{Return Value:} Tidak memiliki \textit{return value}
1798 |
1799 | \textbf{Exception:} Tidak memiliki \textit{exception}
1800 |
1801 | \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1802 |
1803 | \end{itemize}
1804 | \item \texttt{AIF341 implements HasPraktikum}
1805 |
1806 | Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar
1807 | jaringan dan aplikasinya di kehidupan sehari-hari. Mahasiswa
1808 | dikenalkan dengan teknologi-teknologi terbaru di bidang jaringan,
1809 | sehingga mahasiswa memiliki pengetahuan yang dapat digunakan
1810 | dalam kehidupan sehari-hari. Selain itu mahasiswa juga
1811 | diperkenalkan dengan NetAcad, sebuah layanan dari Cisco yang
1812 | dapat digunakan untuk memenuhi segala macam kebutuhan terkait
1813 | dengan Cisco Academy.
1814 |
1815 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF183}
1816 |
1817 |
1818 |
1819 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AKS124}
1820 |
1821 |
1822 |
1823 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF459}
1824 |
1825 |
1826 |
1827 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF336}
1828 |
1829 | Mata kuliah ini merupakan mata kuliah lanjutan dari mata kuliah Keamanan
1830 | Informasi, dengan titik berat pada materi kriptografi. Mata kuliah ini
1831 | memperkenalkan tambahan konsep kriptografi, misalnya tentang otentikasi
1832 | yaitu otentikasi entitas, manajemen kunci, dan bentuk lain dari metode
1833 | merahasiakan pesan, yaitu dengan menggunakan secret sharing. Selanjutnya,
1834 | diperkenalkan juga penggunaan kriptografi pada protokol-protokol yang
1835 | sebenarnya banyak digunakan sehari-hari, misalnya pada e-cash, auction,
1836 | dan electronic voting.
1837 |
1838 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF463}
1839 |
1840 |
1841 |
1842 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF208 implements HasPrasyarat}
1843 |
1844 | Mata kuliah ini memperkenalkan kepada mahasiswa tahapan rekayasa perangkat
1845 | lunak, terutama dengan paradigma berorientasi objek, dilengkapi dengan
1846 | pengenalan tentang manajemen proyek perangkat lunak.
1847 | Selain itu diberikan deskripsi proyek berskala kecil yang harus dikerjakan
1848 | oleh mahasiswa dalam kelompok dengan menerapkan teori yang telah
1849 | dipelajarinya.
1850 |
1851 | Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1852 | \begin{itemize}
1853 | \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1854 |
1855 |
1856 |
1857 | \textbf{Parameter:}
1858 | \begin{itemize}
1859 | \item \texttt{Mahasiswa mahasiswa} -
1860 | \item \texttt{java.util.List reasonsContainer} -
1861 | \end{itemize}
1862 | \textbf{Return Value:} Tidak memiliki \textit{return value}
1863 |
1864 | \textbf{Exception:} Tidak memiliki \textit{exception}
1865 |
1866 | \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1867 |
1868 | \end{itemize}
1869 | \item \texttt{MKU003}
1870 |
1871 | Mata kuliah ini membentuk karakteristik mahasiswa sebagai manusia yang memiliki religiusitas
1872 | melalui pendalaman akan makna agama dan beragama, mendeteksi dinamika Wahyu Tuhan dan iman
1873 | mereka, memahami relasi dengan Tuhan dan sesama, mengenal makna keselamatan dalam konteks
1874 | Kerajaan Allah, dan mampu menyatakan ajaran Gereja dalam pelayanan terhadap orang miskin dan
1875 | terlantar.
1876 |
1877 | Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{APS309}
1878 |
1879 | APS302 atau APS309 ?

```

```

1880
1881 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF335}
1882
1883
1884
1885 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF362 implements HasPrasyarat}
1886
1887
1888
1889 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1890 \begin{itemize}
1891 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1892
1893
1894
1895 \textbf{Parameter:}
1896 \begin{itemize}
1897 \item \texttt{Mahasiswa mahasiswa} -
1898 \item \texttt{java.util.List reasonsContainer} -
1899 \end{itemize}
1900 \textbf{Return Value:} Tidak memiliki \textit{return value}
1901
1902 \textbf{Exception:} Tidak memiliki \textit{exception}
1903
1904 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1905
1906 \end{itemize}
1907 \item \texttt{AIF460}
1908
1909
1910
1911 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF358}
1912
1913
1914
1915 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AMS100}
1916
1917 Sistem Bilangan, Fungsi, Limit dan Kekontinuan Fungsi, Turunan, Integral,
1918 Penggunaan Integral, Sistem Persamaan Linear, Determinan, Vektor, Nilai dan
1919 Vektor Eigen.
1920
1921 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF401 implements HasPrasyarat}
1922
1923
1924
1925 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1926 \begin{itemize}
1927 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1928
1929
1930
1931 \textbf{Parameter:}
1932 \begin{itemize}
1933 \item \texttt{Mahasiswa mahasiswa} -
1934 \item \texttt{java.util.List reasonsContainer} -
1935 \end{itemize}
1936 \textbf{Return Value:} Tidak memiliki \textit{return value}
1937
1938 \textbf{Exception:} Tidak memiliki \textit{exception}
1939
1940 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1941
1942 \end{itemize}
1943 \item \texttt{AIF456}
1944
1945
1946
1947 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{SIR104}
1948
1949
1950
1951 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF339 implements HasPrasyarat}
1952
1953
1954
1955 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1956 \begin{itemize}
1957 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1958
1959
1960
1961 \textbf{Parameter:}
1962 \begin{itemize}
1963 \item \texttt{Mahasiswa mahasiswa} -
1964 \item \texttt{java.util.List reasonsContainer} -
1965 \end{itemize}
1966 \textbf{Return Value:} Tidak memiliki \textit{return value}
1967
1968 \textbf{Exception:} Tidak memiliki \textit{exception}
1969
1970 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
1971
1972 \end{itemize}
1973 \item \texttt{AIF102 implements HasPrasyarat}
1974
1975 \item \texttt{AIF102 implements HasPraktikum}
1976
1977 Mata kuliah ini memperkenalkan berbagai algoritma dan teknik-teknik
1978 penyelesaian masalah komputasi seperti rekursif, sorting, teknik divide dan

```

```

1979 conquer, serta exhaustive search. Selain itu, pada kuliah ini juga
1980 dikenalkan berbagai struktur data yang dapat digunakan untuk mendukung
1981 penyelesaian masalah komputasi seperti ADT List, Stack dan Queue. Baik
1982 algoritma maupun struktur data yang dikenalkan harus dapat diimplementasikan
1983 dan digunakan oleh mahasiswa untuk menyelesaikan masalah dengan menggunakan
1984 suatu bahasa pemrograman berorientasi objek.
1985
1986 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
1987 \begin{itemize}
1988 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
1989
1990
1991
1992 \textbf{Parameter:}
1993 \begin{itemize}
1994 \item \texttt{Mahasiswa mahasiswa} -
1995 \item \texttt{java.util.List reasonsContainer} -
1996 \end{itemize}
1997 \textbf{Return Value:} Tidak memiliki \textit{return value}
1998
1999 \textbf{Exception:} Tidak memiliki \textit{exception}
2000
2001 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2002
2003 \end{itemize}
2004 \item \texttt{AIF381}
2005
2006
2007
2008 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF483}
2009
2010
2011
2012 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF315 implements HasPrasyarat}
2013
2014 \item \texttt{AIF315 implements HasPraktikum}
2015
2016 Mata kuliah ini memperkenalkan konsep dan lingkungan pemrograman berbasis web,
2017 kemudian belajar membuat aplikasi berbasis web menggunakan HTML5, CSS, Java Script
2018 dan PHP. Untuk meningkatkan keterampilan pemrograman dilengkapi dengan praktikum.
2019 Sedangkan untuk mendapatkan pengalaman penerapan konsep diberikan tugas besar membuat
2020 program berbasis web dengan kasus yang ditentukan oleh mahasiswa.
2021
2022 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2023 \begin{itemize}
2024 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2025
2026
2027
2028 \textbf{Parameter:}
2029 \begin{itemize}
2030 \item \texttt{Mahasiswa mahasiswa} -
2031 \item \texttt{java.util.List reasonsContainer} -
2032 \end{itemize}
2033 \textbf{Return Value:} Tidak memiliki \textit{return value}
2034
2035 \textbf{Exception:} Tidak memiliki \textit{exception}
2036
2037 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2038
2039 \end{itemize}
2040 \item \texttt{AIF342 implements HasPraktikum}
2041
2042 \item \texttt{AIF342 implements HasPrasyarat}
2043
2044
2045
2046 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2047 \begin{itemize}
2048 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2049
2050
2051
2052 \textbf{Parameter:}
2053 \begin{itemize}
2054 \item \texttt{Mahasiswa mahasiswa} -
2055 \item \texttt{java.util.List reasonsContainer} -
2056 \end{itemize}
2057 \textbf{Return Value:} Tidak memiliki \textit{return value}
2058
2059 \textbf{Exception:} Tidak memiliki \textit{exception}
2060
2061 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2062
2063 \end{itemize}
2064 \item \texttt{IIE214}
2065
2066
2067
2068 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF303 implements HasPrasyarat}
2069
2070 Mempelajari Konsep Data, Informasi, Pengetahuan, Sistem Informasi, proses dan
2071 pemodelan bisnis, jenis-jenis sistem informasi, untuk mendukung pengambilan
2072 keputusan. Mempelajari trend Teknologi Informasi, tahap-tahap pembangunan
2073 sistem informasi. Mempelajari pengantar : EIS, e-bisnis/e-commerce, Business
2074 Intelligence, Cloud Computing dan Mobile Applications
2075
2076 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2077 \begin{itemize}

```

```

2078 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2079
2080
2081
2082 \textbf{Parameter:}
2083 \begin{itemize}
2084 \item \texttt{Mahasiswa mahasiswa} -
2085 \item \texttt{java.util.List reasonsContainer} -
2086 \end{itemize}
2087 \textbf{Return Value:} Tidak memiliki \textit{return value}
2088
2089 \textbf{Exception:} Tidak memiliki \textit{exception}
2090
2091 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2092
2093 \end{itemize}
2094 \item \texttt{AIF302 implements HasPrasyarat}
2095
2096 Mata kuliah ini melatih mahasiswa dalam menulis ilmiah serta memperkenalkan
2097 metodologi penelitian serta kakas untuk menulis ilmiah.
2098
2099 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2100 \begin{itemize}
2101 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2102
2103
2104
2105 \textbf{Parameter:}
2106 \begin{itemize}
2107 \item \texttt{Mahasiswa mahasiswa} -
2108 \item \texttt{java.util.List reasonsContainer} -
2109 \end{itemize}
2110 \textbf{Return Value:} Tidak memiliki \textit{return value}
2111
2112 \textbf{Exception:} Tidak memiliki \textit{exception}
2113
2114 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2115
2116 \end{itemize}
2117 \item \texttt{AIF181}
2118
2119
2120
2121 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF210}
2122
2123
2124
2125 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF343}
2126
2127
2128
2129 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF206 implements HasPrasyarat}
2130
2131 Mata kuliah ini memperkenalkan kepada mahasiswa mengenai konsep sistem
2132 operasi, jenis-jenis sistem operasi yang digunakan dalam kehidupan
2133 sehari-hari dan beberapa perangkat keras yang dibutuhkan pada komputer.
2134 Selain itu juga mempelajari mengenai teknik dan algoritma yang digunakan
2135 dalam pengelolaan sistem operasi.
2136
2137 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2138 \begin{itemize}
2139 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2140
2141
2142
2143 \textbf{Parameter:}
2144 \begin{itemize}
2145 \item \texttt{Mahasiswa mahasiswa} -
2146 \item \texttt{java.util.List reasonsContainer} -
2147 \end{itemize}
2148 \textbf{Return Value:} Tidak memiliki \textit{return value}
2149
2150 \textbf{Exception:} Tidak memiliki \textit{exception}
2151
2152 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2153
2154 \end{itemize}
2155 \item \texttt{AIF314 implements HasPrasyarat}
2156
2157 \item \texttt{AIF314 implements HasPraktikum}
2158
2159 Kuliah ini merupakan kelanjutan dari kuliah Manajemen Informasi Basisdata.
2160 Pada perkuliahan ini, mahasiswa akan mempelajari teknik-teknik pengelolaan
2161 basis data dan membuat program dengan basis data yang optimal/efisien.
2162
2163 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2164 \begin{itemize}
2165 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2166
2167
2168
2169 \textbf{Parameter:}
2170 \begin{itemize}
2171 \item \texttt{Mahasiswa mahasiswa} -
2172 \item \texttt{java.util.List reasonsContainer} -
2173 \end{itemize}
2174 \textbf{Return Value:} Tidak memiliki \textit{return value}
2175
2176 \textbf{Exception:} Tidak memiliki \textit{exception}

```

```

2177 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2178
2179 \end{itemize}
2180 \item \texttt{AIF380}
2181
2182
2183
2184
2185 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF103}
2186
2187 Mata kuliah ini merupakan salah satu cara untuk mencapai kompetensi dasar
2188 tentang matematika diskrit yang prinsipnya banyak digunakan dalam bidang
2189 ilmu komputer. Selain itu, kuliah ini juga merupakan cara untuk membentuk
2190 pola pikir logis yang dibutuhkan untuk menempuh kuliah-kuliah di tingkat
2191 yang lebih tinggi.
2192
2193 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF292}
2194
2195
2196
2197 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF441 implements HasPraktikum}
2198
2199 \item \texttt{AIF441 implements HasPrasyarat}
2200
2201 Mata kuliah ini memperkenalkan kepada mahasiswa konsep jaringan lanjut
2202 terutama di layer data link dan layer network. Materi utama dari mata kuliah
2203 ini adalah pengembangan jaringan dan pengenalan fungsi-fungsi yang terdapat
2204 pada alat jaringan Cisco yang berkaitan dengan layer 2 dan layer 3.
2205
2206 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2207 \begin{itemize}
2208 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2209
2210
2211
2212 \textbf{Parameter}:
2213 \begin{itemize}
2214 \item \texttt{Mahasiswa mahasiswa} -
2215 \item \texttt{java.util.List reasonsContainer} -
2216 \end{itemize}
2217 \textbf{Return Value}: Tidak memiliki \textit{return value}
2218
2219 \textbf{Exception}: Tidak memiliki \textit{exception}
2220
2221 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2222
2223 \end{itemize}
2224 \item \texttt{AIF457 implements HasPrasyarat}
2225
2226 Mata kuliah ini memperkenalkan konsep kewirausahaan dengan memanfaatkan teknologi, khususnya
2227 teknologi informasi, sebagai basis usaha dan inovasi produk/jasa; Mempelajari
2228 teknik mencari peluang dan merumuskan bidang usaha spesifik yang akan
2229 diterjuni; Mempelajari konsep manajemen pemasaran, keuangan dan SDM dalam
2230 kaitannya dengan berwira-usaha di bidang TI; Menyusun proposal bisnis untuk
2231 berwira-usaha di bidang TI dan mempresentasikannya.
2232
2233 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2234 \begin{itemize}
2235 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2236
2237
2238
2239 \textbf{Parameter}:
2240 \begin{itemize}
2241 \item \texttt{Mahasiswa mahasiswa} -
2242 \item \texttt{java.util.List reasonsContainer} -
2243 \end{itemize}
2244 \textbf{Return Value}: Tidak memiliki \textit{return value}
2245
2246 \textbf{Exception}: Tidak memiliki \textit{exception}
2247
2248 \textbf{Override}: \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2249
2250 \end{itemize}
2251 \item \texttt{MKU001}
2252
2253 Mata Kuliah Pendidikan Pancasila berupaya menelaah/mengkaji berbagai fenomena kehidupan
2254 bangsa dan Negara Indonesia sebagai sebuah ruang publik dengan menggunakan pendekatan
2255 hermeneutika (filsafat) dan pendidikan nilai (pedagogik). Dengan bantuan hermeneutika
2256 mahasiswa diajak berpikir kritis terhadap segala bentuk ideologisme Pancasila dan melalui
2257 pendidikan nilai mahasiswa dilatih untuk memiliki nilai Pancasila. Nilai pengembangan diri
2258 intra-personal dan relasi inter-personal dapat tertanam melalui pendidikan Pancasila yang
2259 tujuannya adalah membangun kepribadian (character building) manusia Indonesia yang utuh,
2260 baik menyangkut aspek kognitif, afektif, maupun psikomotor. Dengan demikian, Pendidikan
2261 Pancasila mengajak mahasiswa menilai realitas ruang publik sehari-hari secara mandiri
2262 dengan panduan nilai-nilai etis Pancasila.
2263
2264 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{ESA101}
2265
2266
2267
2268 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AMS290}
2269
2270
2271
2272 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF461}
2273
2274
2275

```



```

2276 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF318 implements HasPrasyarat}
2277
2278 \item \texttt{AIF318 implements HasPraktikum}
2279
2280 Mata kuliah ini memperkenalkan konsep perangkat mobile dan pemrograman pada perangkat
2281 mobile. Pemrograman dikhususkan pada lingkungan J2ME dan Android.
2282 Untuk meningkatkan keterampilan pemrograman dilengkapi dengan praktikum.
2283 Sedangkan untuk mendapatkan pengalaman penerapan konsep diberikan tugas implementasi suatu
2284 kasus pada lingkungan mobile-cloud dengan kasus yang sudah ditentukan.
2285
2286 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2287 \begin{itemize}
2288 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2289
2290
2291
2292 \textbf{Parameter:}
2293 \begin{itemize}
2294 \item \texttt{Mahasiswa mahasiswa} -
2295 \item \texttt{java.util.List reasonsContainer} -
2296 \end{itemize}
2297 \textbf{Return Value:} Tidak memiliki \textit{return value}
2298
2299 \textbf{Exception:} Tidak memiliki \textit{exception}
2300
2301 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2302
2303 \end{itemize}
2304 \item \texttt{AIF334}
2305
2306
2307
2308 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{ESM105}
2309
2310
2311
2312 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF450}
2313
2314
2315
2316 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF446}
2317
2318
2319
2320 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF104}
2321
2322 Mata kuliah ini memberikan pengetahuan tentang logika yang digunakan di
2323 dalam ilmu komputer. Dalam kuliah ini, mahasiswa belajar untuk bisa
2324 memodelkan suatu kalimat dalam kehidupan sehari-hari, ke dalam kalimat
2325 dengan sintaks tertentu, yang hanya memiliki satu arti. Lalu, diperkenalkan
2326 juga, bagaimana mengartikan suatu kalimat (benar atau salah) dan bagaimana
2327 menentukan sifat dari kalimat tersebut.
2328
2329 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF387}
2330
2331
2332
2333 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF313 implements HasPraktikum}
2334
2335 Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar pembuatan grafik
2336 dengan menggunakan komputer seperti mengenal berbagai algoritma pembuatan
2337 primitif 2 dimensi seperti titik, garis, lingkaran, elips, berbagai macam
2338 bentuk kurva, fraktal, konsep warna (RGB), dasar-dasar grafika 3 dimensi
2339 seperti pewarnaan, pencahayaan, pemberian tekstur pada objek, transformasi,
2340 animasi, dan sebagainya. Selain, itu diberikan masalah-masalah komputasi
2341 sederhana yang harus diselesaikan menggunakan konsep-konsep yang sudah
2342 diperkenalkan dan mengimplementasikannya menggunakan bahasa pemrograman Java.
2343
2344 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF344 implements HasPrasyarat}
2345
2346
2347
2348 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2349 \begin{itemize}
2350 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2351
2352
2353
2354 \textbf{Parameter:}
2355 \begin{itemize}
2356 \item \texttt{Mahasiswa mahasiswa} -
2357 \item \texttt{java.util.List reasonsContainer} -
2358 \end{itemize}
2359 \textbf{Return Value:} Tidak memiliki \textit{return value}
2360
2361 \textbf{Exception:} Tidak memiliki \textit{exception}
2362
2363 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2364
2365 \end{itemize}
2366 \item \texttt{AIF201 implements HasPrasyarat}
2367
2368 \item \texttt{AIF201 implements HasPraktikum}
2369
2370 \item \texttt{AIF201 implements HasResponsi}
2371
2372 Mata kuliah ini memperkenalkan prinsip-prinsip yang digunakan dalam
2373 melakukan analisa serta desain program berorientasi objek. Di samping itu,
2374 mahasiswa juga belajar menggunakan kakas berupa diagram UML (Unified

```



```

2375 Modelling Language) sehingga dapat mengkomunikasikan desain secara visual.
2376 Mahasiswa juga akan mengenal beberapa software design pattern dari Gang of
2377 Four. Terakhir, mahasiswa akan belajar mengenai konsep MVC
2378 (Model-View-Controller) yang menjadi dasar dari banyak framework masa kini.
2379 Bahasa yang digunakan adalah bahasa Java, namun diusahakan tetap umum
2380 sehingga dapat diaplikasikan pada bahasa yang lain.
2381
2382 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2383 \begin{itemize}
2384 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2385
2386
2387 \textbf{Parameter:}
2388 \begin{itemize}
2389 \item \texttt{Mahasiswa mahasiswa} -
2390 \item \texttt{java.util.List reasonsContainer} -
2391 \end{itemize}
2392 \textbf{Return Value:} Tidak memiliki \textit{return value}
2393
2394 \textbf{Exception:} Tidak memiliki \textit{exception}
2395
2396 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2397
2398 \end{itemize}
2399 \item \texttt{AIF352}
2400
2401
2402
2403
2404 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF305 implements HasPrasyarat}
2405
2406 Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar jaringan
2407 komputer dengan menggunakan top-down. Selain itu mengajarkan juga kepada
2408 mahasiswa mengenai aplikasi-aplikasi berbasis jaringan sehingga diharapkan
2409 mahasiswa dapat membuat aplikasi berbasis jaringan dengan menggunakan socket.
2410 Pada akhirnya, mahasiswa akan ditugaskan untuk membangun jaringan komputer
2411 LAN, baik menggunakan kabel maupun nirkabel.
2412
2413 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2414 \begin{itemize}
2415 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2416
2417
2418 \textbf{Parameter:}
2419 \begin{itemize}
2420 \item \texttt{Mahasiswa mahasiswa} -
2421 \item \texttt{java.util.List reasonsContainer} -
2422 \end{itemize}
2423 \textbf{Return Value:} Tidak memiliki \textit{return value}
2424
2425 \textbf{Exception:} Tidak memiliki \textit{exception}
2426
2427 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2428
2429 \end{itemize}
2430 \item \texttt{MKU010}
2431
2432
2433 Mata kuliah ini difokuskan pada pemahaman sumber referensi dalam Bahasa Inggris dan
2434 pengembangan kosakata Bahasa Inggris (vocabularies). Hampir keseluruhan waktu perkuliahan
2435 didedikasikan untuk menjelaskan metode mengekstraksi isi bacaan secara tepat dan melatih
2436 mahasiswa untuk menerapkan metode tersebut seraya menambah kosakata-kosakata baru.
2437 Mahasiswa juga dilatih untuk mempresentasikan hasil pemahamannya akan isi bahan bacaan.
2438
2439 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{MKU011}
2440
2441 Mata kuliah estetika memberi pemahaman konseptual filosofis "seni" dalam khasanah keilmuan,
2442 pembentukan kesadaran ekologis juga dalam proses pembudayaan dan peradaban. Mata kuliah ini
2443 akan menjadi fondasi bagi mahasiswa untuk memahami dan mempraktekkan seni dari sudut pandang
2444 filsafat, sejarah, kultural, dan global. Melalui mata kuliah ini, mahasiswa mempelajari
2445 mengenai dunia manusia (manusia dan pikirannya), pluralitas dan relativitas seni, serta
2446 aliran-aliran seni rupa Barat.
2447
2448 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF332 implements HasPrasyarat}
2449
2450
2451
2452 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2453 \begin{itemize}
2454 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2455
2456
2457 \textbf{Parameter:}
2458 \begin{itemize}
2459 \item \texttt{Mahasiswa mahasiswa} -
2460 \item \texttt{java.util.List reasonsContainer} -
2461 \end{itemize}
2462 \textbf{Return Value:} Tidak memiliki \textit{return value}
2463
2464 \textbf{Exception:} Tidak memiliki \textit{exception}
2465
2466 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2467
2468 \end{itemize}
2469 \item \texttt{AIF304 implements HasPrasyarat}
2470
2471 \item \texttt{AIF304 implements HasPraktikum}
2472
2473

```

```

2474 \item \texttt{AIF304 implements HasResponsi}
2475
2476 Mata kuliah ini memberikan kesempatan bagi mahasiswa untuk memperdalam konsep
2477 tentang pengembangan sistem informasi dan mempraktekkan analisis kebutuhan,
2478 analisis sistem dan perancangan sitem pada organisasi studi kasus;
2479
2480 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2481 \begin{itemize}
2482 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2483
2484
2485
2486 \textbf{Parameter:}
2487 \begin{itemize}
2488 \item \texttt{Mahasiswa mahasiswa} -
2489 \item \texttt{java.util.List reasonsContainer} -
2490 \end{itemize}
2491 \textbf{Return Value:} Tidak memiliki \textit{return value}
2492
2493 \textbf{Exception:} Tidak memiliki \textit{exception}
2494
2495 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2496
2497 \end{itemize}
2498 \item \texttt{ESM201}
2499
2500
2501 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF312 implements HasPrasyarat}
2502
2503 \item \texttt{AIF312 implements HasPraktikum}
2504
2505 Mata kuliah ini memberikan pengetahuan awal tentang keamanan informasi. Pada
2506 beberapa pertemuan awal, dibahas keamanan informasi secara matematis, yaitu
2507 di materi-materi seputar kriptografi dan serangannya. Lalu, dibahas pula
2508 konsep keamanan informasi pada jaringan komputer dan pada software.
2509
2510 Kelas ini tidak memiliki atribut. \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2511 \begin{itemize}
2512 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2513
2514
2515
2516
2517 \textbf{Parameter:}
2518 \begin{itemize}
2519 \item \texttt{Mahasiswa mahasiswa} -
2520 \item \texttt{java.util.List reasonsContainer} -
2521 \end{itemize}
2522 \textbf{Return Value:} Tidak memiliki \textit{return value}
2523
2524 \textbf{Exception:} Tidak memiliki \textit{exception}
2525
2526 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{MataKuliah}
2527
2528 \end{itemize}
2529 \item \texttt{AIF484}
2530
2531
2532
2533 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF191}
2534
2535
2536 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF386}
2537
2538
2539
2540
2541 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF105}
2542
2543 Mata kuliah ini memperkenalkan kepada mahasiswa terminologi dan konsep dasar
2544 yang akan banyak dipakai sepanjang kuliah di Teknik Informatika. Selain itu
2545 mata kuliah ini juga mempersiapkan dan membiasakan mahasiswa dengan suasana
2546 akademik yang khas perguruan tinggi seperti kedisiplinan, kerja sama,
2547 kemampuan menggunakan teknologi informasi dalam pembuatan tugas, kemampuan
2548 komunikasi, dsb.
2549
2550 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF294}
2551
2552
2553
2554 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF282}
2555
2556
2557
2558 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{AIF451}
2559
2560
2561
2562 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{APS302}
2563
2564
2565
2566 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{EAA101}
2567
2568
2569
2570 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{HasPrasyarat}
2571
2572 Mendefinisikan kelas-kelas yang memiliki prasyarat, terkustomisasi

```

```

2573 untuk seorang \texttt{Mahasiswa}. Jika ada tambahan, jangan lupa untuk
2574 mendaftarkannya di DEFAULT\_HASPRASYARAT\_CLASSES. Jika berubah package,
2575 jangan lupa mengupdate DEFAULT\_PRASYARAT\_PACKAGE.
2576
2577 Atribut yang dimiliki kelas ini adalah sebagai berikut.
2578 \begin{itemize}
2579 \item \texttt{String DEFAULT\_HASPRASYARAT\_CLASSES} - Daftar dari nama kelas default seluruh turunan interface ini. Perlu
    didaftarkan
2580 manual, karena Java reflection tidak dapat mendeteksi otomatis.
2581 \item \texttt{String DEFAULT\_PRASYARAT\_PACKAGE} - Package tempat menyimpan seluruh turunan standar interface ini. Perlu
    didefinisikan
2582 manual, karena Java reflection tidak dapat mendeteksi otomatis.
2583 \end{itemize}
2584 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2585 \begin{itemize}
2586 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2587
2588 Memeriksa prasyarat-prasyarat dari kuliah, spesifik untuk mahasiswa
2589 yang dituju. Jika ada pesan-pesan khusus, akan ditambahkan pada parameter
2590 reasonsContainer.
2591
2592 \textbf{Parameter:}
2593 \begin{itemize}
2594 \item \texttt{Mahasiswa mahasiswa} -
2595 prasyarat kuliah akan diperiksa spesifik pada mahasiswa ini
2596 \item \texttt{List reasonsContainer} -
2597 pesan-pesan terkait prasyarat akan ditambahkan di sini, jika ada.
2598 \end{itemize}
2599 \textbf{Return Value:} true jika seluruh prasyarat dipenuhi, false jika tidak.
2600
2601 \textbf{Exception:} Tidak memiliki \textit{exception}
2602
2603 \end{itemize}
2604 \item \texttt{HasPraktikum}
2605
2606
2607
2608 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{HasResponsi}
2609
2610
2611
2612 Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method. \item \texttt{Kelulusan implements HasPrasyarat}
2613
2614
2615
2616 Atribut yang dimiliki kelas ini adalah sebagai berikut.
2617 \begin{itemize}
2618 \item \texttt{String PILIHAN\_WAJIB} -
2619 \item \texttt{String WAJIB} -
2620 \item \texttt{String AGAMA} -
2621 \item \texttt{int MIN\_SKS\_LULUS} -
2622 \item \texttt{int MIN\_PILIHAN\_WAJIB} -
2623 \end{itemize}
2624 \textit{Method-method} yang dimiliki kelas ini adalah sebagai berikut.
2625 \begin{itemize}
2626 \item \texttt{public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)}
2627
2628
2629
2630 \textbf{Parameter:}
2631 \begin{itemize}
2632 \item \texttt{Mahasiswa mahasiswa} -
2633 \item \texttt{java.util.List reasonsContainer} -
2634 \end{itemize}
2635 \textbf{Return Value:} Tidak memiliki \textit{return value}
2636
2637 \textbf{Exception:} Tidak memiliki \textit{exception}
2638
2639 \textbf{Override:} \texttt{checkPrasyarat} dari kelas \texttt{Object}
2640
2641 \end{itemize}
2642 \end{enumerate}
2643 \end{document}

```

## C.3 Hasil PDF

## 1. InfoMataKuliah

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public int sks()`  
Jumlah bobot sks dari mata kuliah ini  
**Parameter:**
  - Tidak memiliki parameter *method***Return Value:** jumlah bobot sks  
**Exception:** Tidak memiliki *exception*
- `public String nama()`  
Nama mata kuliah ini  
**Parameter:**
  - Tidak memiliki parameter *method***Return Value:** nama mata kuliah  
**Exception:** Tidak memiliki *exception*

## 2. MataKuliahFactory

Kelas yang bertugas membuat kelas mata kuliah, dan menyimpannya untuk bisa digunakan kemudian (untuk hemat memori).

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `String DEFAULT_MATAKULIAH_PACKAGE` - Lokasi package untuk daftar mata kuliah
- `MataKuliahFactory instance` - Singleton instance to factory.
- `SortedMap mataKuliahCache` - Singleton instances untuk mata kuliah.

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static MataKuliahFactory getInstance()`  
**Parameter:**
  - Tidak memiliki parameter *method***Return Value:** Tidak memiliki *return value*  
**Exception:** Tidak memiliki *exception*
- `public MataKuliah createMataKuliah(java.lang.String kode, int sks, java.lang.String nama)`  
Membuat baru atau mendapatkan mata kuliah, jika memiliki informasi nama dan jumlah SKS.  
**Parameter:**
  - `String kode` - kode mata kuliah

- `int sks` - jumlah SKS
- `String nama` - nama mata kuliah

**Return Value:** objek mata kuliah

**Exception:** Tidak memiliki *exception*

- `public MataKuliah createMataKuliah(java.lang.String kode)`  
Membuat baru atau mendapatkan mata kuliah, jika tidak memiliki informasi nama dan jumlah SKS.

**Parameter:**

- `String kode` - kode mata kuliah

**Return Value:** objek mata kuliah

**Exception:** `IllegalStateException` jika `sks` dan tidak sesuai dengan yang ada di kode

### 3. Semester

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `Semester UNKNOWN5` -
- `Semester TRANSFER` -
- `Semester PENDEK` -
- `Semester GANJIL` -
- `Semester GENAP` -
- `int order` -

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public static Semester values()`

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public static Semester valueOf(java.lang.String name)`

**Parameter:**

- `String name` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public static Semester fromString(java.lang.String text)`

**Parameter:**

- `String text` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `int getOrder()`

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

#### 4. `TahunSemester` implements `Comparable`

Menyimpan konstanta untuk semester beserta tahunnya di UNPAR.

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `String kodeTahunSemester` - Kode semester 3 digit, sesuai DPS: `jul`, `li`, 2 digit pertama berupa tahun, 2 digit terakhir `li`, `li`, 2 digit terakhir: 1 untuk ganjil, 2 untuk genap, 4 untuk pendek. `i`, `ul`

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public Semester getSemester()`

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public int getTahun()`

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `private static void validateKodeSemester(java.lang.String kodeTahunSemester)`

**Parameter:**

- `String kodeTahunSemester` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public String getKodeDPS()`

Mendapatkan kode tahun/semester sesuai aturan di DPS.

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** kode tahun/semester sesuai aturan di DPS.

**Exception:** Tidak memiliki *exception*

- `public int compareTo(id.ac.unpar.siamodels.TahunSemester o)`

**Parameter:**

– TahunSemester o -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** compareTo dari kelas Object

- public boolean equals(java.lang.Object arg0)

**Parameter:**

– Object arg0 -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- public String toString()

**Parameter:**

– Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

#### 5. Mahasiswa

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- String npm -
- String nama -
- List riwayatNilai -
- URL photoURL -
- List jadwalKuliahList -
- SortedMap nilaiTOEFL -

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- public String getName()

**Parameter:**

– Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- public void setName(java.lang.String nama)

**Parameter:**

– String nama -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- public String getNpm()

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*
- `public URL getPhotoURL()`

**Parameter:**

  - Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*
- `public void setPhotoURL(java.net.URL photoURL)`

**Parameter:**

  - URL photoURL -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*
- `public List getJadwalkKuliahList()`

**Parameter:**

  - Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*
- `public void setJadwalkKuliahList(java.util.List jadwalKuliahList)`

**Parameter:**

  - java.util.List jadwalKuliahList -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*
- `public String getEmailAddress()`

**Parameter:**

  - Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*
- `public List getRiwayatNilai()`

**Parameter:**

  - Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*
- `public SortedMap getNilaiTOEFL()`

**Parameter:**

  - Tidak memiliki parameter *method*



**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public void setNilaiTOEFL(java.util.SortedMap nilaiTOEFL)`

**Parameter:**

- `java.util.SortedMap nilaiTOEFL` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public double calculateIPKLulus()`

Menghitung IPK mahasiswa sampai saat ini, dengan aturan:  $\sum_{i=1}^n \text{Nilai}_i$  yang tidak lulus tidak dihitung  $\sum_{i=1}^n \text{Nilai}_i$ . Jika pengambilan beberapa kali, diambil  $\max(\text{Nilai}_i)$ . Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.

**Exception:** Tidak memiliki *exception*

- `public double calculateIPLulus()`

Menghitung IP mahasiswa sampai saat ini, dengan aturan:  $\sum_{i=1}^n \text{Nilai}_i$  yang tidak lulus tidak dihitung  $\sum_{i=1}^n \text{Nilai}_i$ . Jika pengambilan beberapa kali, diambil  $\max(\text{Nilai}_i)$ . Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.

**Exception:** Tidak memiliki *exception*

- `public double calculateIPTempuh(boolean lulusSaja)`

Menghitung IP mahasiswa sampai saat ini, dengan aturan:  $\sum_{i=1}^n \text{Perhitungan}_i$  yang tidak lulus ditentukan parameter  $\text{Nilai}_i$ . Jika pengambilan beberapa kali, diambil  $\max(\text{Nilai}_i)$ .

**Parameter:**

- `boolean lulusSaja` - set true jika ingin membuang mata kuliah tidak lulus, false jika ingin semua (sama dengan "IP N. Terbaik" di DPS) Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!

**Return Value:** IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.

**Exception:** Tidak memiliki *exception*

- **public double calculateIPKumulatif()**  
Menghitung IP Kumulatif mahasiswa sampai saat ini, dengan aturan:  $\sum_{i=1}^n \frac{1}{i}$ . Jika pengambilan beberapa kali, diambil semua.  $\frac{1}{i}$  Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!  
**Parameter:**
  - Tidak memiliki parameter *method***Return Value:** IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.  
**Exception:** Tidak memiliki *exception*
- **public double calculateIPKTempuh(boolean lulusSaja)**  
Menghitung IPK mahasiswa sampai saat ini, dengan aturan:  $\sum_{i=1}^n \frac{1}{i}$ . Perhitungan kuliah yang tidak lulus ditentukan parameter  $\frac{1}{i}$ . Jika pengambilan beberapa kali, diambil  $\frac{1}{i}$  nilai terbaik  $\frac{1}{i}$ .  $\frac{1}{i}$  Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!  
**Parameter:**
  - `boolean lulusSaja` - set true jika ingin membuang mata kuliah tidak lulus. Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!**Return Value:** IPK Lulus, atau DoubleNaN jika belum mengambil satu kuliah pun.  
**Exception:** Tidak memiliki *exception*
- **public double calculateIPS()**  
Menghitung IPS semester terakhir sampai saat ini, dengan aturan:  $\sum_{i=1}^n \frac{1}{i}$ . Kuliah yang tidak lulus  $\frac{1}{i}$  dihitung  $\frac{1}{i}$ .  $\frac{1}{i}$  Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!  
**Parameter:**
  - Tidak memiliki parameter *method***Return Value:** nilai IPS sampai saat ini  
**Exception:** `ArrayIndexOutOfBoundsException` jika belum ada nilai satupun
- **public int calculateSKSLulus()**  
Menghitung jumlah SKS lulus mahasiswa saat ini. Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!  
**Parameter:**
  - Tidak memiliki parameter *method***Return Value:** SKS Lulus  
**Exception:** Tidak memiliki *exception*

- **public int calculateSKSTempuh(boolean lulusSaja)**  
Menghitung jumlah SKS tempuh mahasiswa saat ini. Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah!  
**Parameter:**
  - `boolean lulusSaja` - set true jika ingin membuang SKS tidak lulus**Return Value:** SKS tempuh  
**Exception:** Tidak memiliki *exception*
- **public Set calculateTahunSemesterAktif()**  
Mendapatkan seluruh tahun semester di mana mahasiswa ini tercatat sebagai mahasiswa aktif, dengan strategi memeriksa riwayat nilainya. Jika ada satu nilai saja pada sebuah tahun semester, maka dianggap aktif pada semester tersebut.  
**Parameter:**
  - Tidak memiliki parameter *method***Return Value:** kumpulan tahun semester di mana mahasiswa ini aktif  
**Exception:** Tidak memiliki *exception*
- **public boolean hasLulusKuliah(java.lang.String kodeMataKuliah)**  
Memeriksa apakah mahasiswa ini sudah lulus mata kuliah tertentu. Kompleksitas  $O(n)$ . Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah! Note: jika yang dimiliki adalah MataKuliah, gunakanlah `MataKuliahgetKode()`.  
**Parameter:**
  - `String kodeMataKuliah` - kode mata kuliah yang ingin diperiksa kelulusannya.**Return Value:** true jika sudah pernah mengambil dan lulus, false jika belum  
**Exception:** Tidak memiliki *exception*
- **public boolean hasTempuhKuliah(java.lang.String kodeMataKuliah)**  
Memeriksa apakah mahasiswa ini sudah pernah menempuh mata kuliah tertentu. Kompleksitas  $O(n)$ . Sebelum memanggil method ini, `getRiwayatNilai()` harus sudah ada isinya! Note: jika yang dimiliki adalah MataKuliah, gunakanlah `MataKuliahgetKode()`.  
**Parameter:**
  - `String kodeMataKuliah` - kode mata kuliah yang ingin diperiksa.**Return Value:** true jika sudah pernah mengambil, false jika belum  
**Exception:** Tidak memiliki *exception*

- `public int getTahunAngkatan()`  
Mendapatkan tahun angkatan mahasiswa ini, berdasarkan NPM nya  
**Parameter:**  
– Tidak memiliki parameter *method*  
**Return Value:** tahun angkatan  
**Exception:** Tidak memiliki *exception*
- `public String toString()`  
**Parameter:**  
– Tidak memiliki parameter *method*  
**Return Value:** Tidak memiliki *return value*  
**Exception:** Tidak memiliki *exception*

#### 6. Mahasiswa.Nilai

Merepresentasikan nilai yang ada di riwayat nilai mahasiswa  
Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `TahunSemester tahunSemester` - Tahun dan Semester kuliah ini diambil
- `MataKuliah mataKuliah` - Mata kuliah yang diambil
- `Character kelas` - Kelas kuliah
- `Double nilaiART` - Nilai ART
- `Double nilaiUTS` - Nilai UTS
- `Double nilaiUAS` - Nilai UAS
- `Character nilaiAkhir` - Nilai Akhir

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public MataKuliah getMataKuliah()`  
**Parameter:**  
– Tidak memiliki parameter *method*  
**Return Value:** Tidak memiliki *return value*  
**Exception:** Tidak memiliki *exception*
- `public Character getKelas()`  
**Parameter:**  
– Tidak memiliki parameter *method*  
**Return Value:** Tidak memiliki *return value*  
**Exception:** Tidak memiliki *exception*
- `public Double getNilaiART()`  
**Parameter:**

- Tidak memiliki parameter *method*
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public Double getNilaiUTS()`
  - Parameter:**
    - Tidak memiliki parameter *method*
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*
- `public Double getNilaiUAS()`
  - Parameter:**
    - Tidak memiliki parameter *method*
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*
- `public Character getNilaiAkhir()`
  - Mengembalikan nilai akhir dalam bentuk huruf (A, B, C, D, ..., atau K)
  - Parameter:**
    - Tidak memiliki parameter *method*
  - Return Value:** nilai akhir dalam huruf, atau null jika tidak ada.
  - Exception:** Tidak memiliki *exception*
- `public Double getAngkaAkhir()`
  - Mendapatkan nilai akhir dalam bentuk angka
  - Parameter:**
    - Tidak memiliki parameter *method*
  - Return Value:** nilai akhir dalam angka, atau null jika `getNilaiAkhir()` mengembalikan 'K' atau null
  - Exception:** Tidak memiliki *exception*
- `public TahunSemester getTahunSemester()`
  - Parameter:**
    - Tidak memiliki parameter *method*
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*
- `public int getTahunAjaran()`
  - Parameter:**
    - Tidak memiliki parameter *method*
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*

- `public Semester getSemester()`  
**Parameter:**  
– Tidak memiliki parameter *method*  
**Return Value:** Tidak memiliki *return value*  
**Exception:** Tidak memiliki *exception*
- `public String toString()`  
**Parameter:**  
– Tidak memiliki parameter *method*  
**Return Value:** Tidak memiliki *return value*  
**Exception:** Tidak memiliki *exception*

#### 7. Mahasiswa.Nilai.ChronologicalComparator implements Comparator

Pembandingan antara satu nilai dengan nilai lainnya, secara kronologis waktu pengambilan.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public int compare(id.ac.unpar.siamodels.Mahasiswa.Nilai o1, id.ac.unpar.siamodels.Mahasiswa.Nilai o2)`  
**Parameter:**  
– Mahasiswa.Nilai o1 -  
– Mahasiswa.Nilai o2 -  
**Return Value:** Tidak memiliki *return value*  
**Exception:** Tidak memiliki *exception*  
**Override:** `compare` dari kelas `Object`

#### 8. JadwalKuliah

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- MataKuliah mataKuliah -
- Character kelas -
- DayOfWeek hari -
- LocalTime waktuMulai -
- LocalTime waktuSelesai -
- String lokasi -
- Dosen pengajar -

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public MataKuliah getMataKuliah()`  
**Parameter:**

- Tidak memiliki parameter *method*
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public void setMataKuliah(id.ac.unpar.siamodels.MataKuliah mataKuliah)`
- Parameter:**
  - MataKuliah mataKuliah -
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public Character getKelas()`
- Parameter:**
  - Tidak memiliki parameter *method*
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public void setKelas(java.lang.Character kelas)`
- Parameter:**
  - Character kelas -
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public DayOfWeek getHari()`
- Parameter:**
  - Tidak memiliki parameter *method*
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public void setHari(java.time.DayOfWeek hari)`
- Parameter:**
  - DayOfWeek hari -
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public LocalTime getWaktuMulai()`
- Parameter:**
  - Tidak memiliki parameter *method*
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public void setWaktuMulai(java.time.LocalTime waktuMulai)`
- Parameter:**
  - LocalTime waktuMulai -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public LocalTime getWaktuSelesai()`

**Parameter:**

– Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public void setWaktuSelesai(java.time.LocalTime waktuSelesai)`

**Parameter:**

– `LocalTime waktuSelesai` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public String getLocation()`

**Parameter:**

– Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public void setLocation(java.lang.String lokasi)`

**Parameter:**

– `String lokasi` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public Dosen getPengajar()`

**Parameter:**

– Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public void setPengajar(id.ac.unpar.siamodels.Dosen pengajar)`

**Parameter:**

– `Dosen pengajar` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public String getWaktuString()`

**Parameter:**

– Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*



- `public static DayOfWeek indonesianToDayOfWeek(java.lang.String indonesian)`

Converts Indonesian day names to `DayOfWeek` enumeration.

**Parameter:**

- `String indonesian` - the day name in Indonesian

**Return Value:** `DayOfWeek` object or null if not found.

**Exception:** Tidak memiliki *exception*

#### 9. MataKuliah

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `String kode` -
- `String nama` -
- `Integer sks` -

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public String getKode()`

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public String getNama()`

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

- `public Integer getSks()`

**Parameter:**

- Tidak memiliki parameter *method*

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

#### 10. Dosen

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `String nik` -
- `String nama` -

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public String getNik()`

**Parameter:**

- Tidak memiliki parameter *method*
- Return Value:** Tidak memiliki *return value*
- Exception:** Tidak memiliki *exception*
- `public void setNik(java.lang.String nik)`
  - Parameter:**
    - `String nik` -
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*
- `public String getNama()`
  - Parameter:**
    - Tidak memiliki parameter *method*
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*
- `public void setName(java.lang.String nama)`
  - Parameter:**
    - `String nama` -
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*
- `public boolean equals(java.lang.Object arg0)`
  - Parameter:**
    - `Object arg0` -
  - Return Value:** Tidak memiliki *return value*
  - Exception:** Tidak memiliki *exception*

11. MKU008

Mendalami perilaku sehari-hari yang baik dalam bermasyarakat.  
Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

12. IIE210

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

13. AIF203 implements HasPrasyarat

Mata kuliah ini memperkenalkan kepada mahasiswa konsep struktur diskret yang digunakan pada bidang informatika diantaranya graph, pohon dan finite state machine

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`
  - Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

14. AIF311 implements HasPrasyarat

15. AIF311 implements HasPraktikum

Kuliah Pemrograman Fungsional bertujuan untuk: 1. memperkenalkan paradigma pemrograman fungsional, yaitu sebuah pemrograman yang didasarkan pada konsep pemetaan dan fungsi matematika. Penyelesaian suatu masalah didasari atas aplikasi dari fungsi-fungsi tersebut. 2. memberikan dasar-dasar pemrograman fungsional dengan menggunakan bahasa fungsional Haskell.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

16. AIF192

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

17. AIF468

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

18. IIE103

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

19. AIF385

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

20. AIF106

Mata kuliah ini memberikan pengetahuan tentang cara kerja komputer, dimulai dari representasi data dan berbagai macam operasinya. Selanjutnya, juga diperkenalkan bagaimana merepresentasikan suatu fungsi dalam

rangkaian gerbang logika, dan bagaimana menyederhanakannya. Berbagai rangkaian dasar yang digunakan di dalam komputer juga dipekenalkan. Mahasiswa juga akan mempelajari komponen komputer, misalnya register dan memori.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

21. AIF281

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

22. EAA102

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

23. AIF405 implements HasPrasyarat

24. AIF405 implements HasPraktikum

Mata kuliah ini merupakan lanjutan dari Projek Sistem Informasi 1 dan memberikan kesempatan bagi mahasiswa untuk melanjutkan/mengembangkan perancangan sitem pada organisasi studi kasus, mengimplementasikan rancangan dan melakukan pengujian perangkat lunak;

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

25. APS182

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

26. MKU004

Fenomenologi Agama merupakan bagian yang tak terpisahkan dari kajian filosofis, kritis, rasional, dan obyektif mengenai substansi ajaran agama. Fenomenologi merupakan sebuah disiplin ilmu yang secara kritis-rasional mengkaji fenomena dan dinamika kehidupan manusia beragama, dari upaya menjadikan Tuhan sebagai tujuan sesembahan sampai menempatkan Tuhan sebagai instrumen legitimasi untuk melakukan tindakan yang justru bertolak belakang dengan kehendak Tuhan yang disembah. Sehubungan dengan itu, kritik konstruktif terhadap perilaku manusia beragama

menjadi salah satu poin utama dalam mata kuliah ini. Kesiapan untuk melakukan otoritik terhadap agama sendiri erat terkait dengan upaya menemukan kembali nilai sejati agama atau otentisitas hidup beragama.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

27. MKU012

Perkuliahan logika ditujukan untuk memberikan dasar-dasar ketrampilan berpikir rasional dan sistematis. Isinya mencakup ketrampilan berpikir deduktif dan induktif, seperti silogisme, argumen analogikal dan generalisasi induktif. Pembahasan teoretis disertai pula dengan pelatihan praktis yang diarahkan pada proses berpikir. Untuk menajamkan kemampuan berpikir tersebut, mahasiswa dilatih pula mengidentifikasi kerancuan-kerancuan (fallacies) yang sering dijumpai baik dalam kehidupan sehari-hari maupun dalam konteks akademik.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

28. AIF389

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

29. AMS190

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

30. AMS191

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

31. AMS200

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

32. AIF330

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

33. AIF388

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

34. AIF465

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

35. AIF453 implements HasPrasyarat

Mata kuliah ini memperkenalkan kebutuhan organisasi terhadap sistem business intelligent (BI) dan pemanfaatan BI untuk organisasi; memperkenalkan konsep sistem business intelligent dan komponennya; Mempelajari teknik-teknik analisis data bisnis dan visualisasi hasil analisis; Mempelajari konsep data warehouse dan perancangannya dan fungsi OLAP; Mempraktekan teknik-teknik analisis data dan visualisasi hasil analisis.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

#### 36. AIF280

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

#### 37. AIF445 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

#### 38. AIF469 implements HasPrasyarat

Mata kuliah ini mengajarkan kepada mahasiswa teknik-teknik untuk membuat layanan berbasis web. Mahasiswa diperkenalkan dengan standar-standar seperti HTTP, XML, JSON dan diajarkan untuk memanfaatkannya dalam membuat maupun menggunakan layanan pihak ketiga. Dalam kuliah ini, juga akan diperkenalkan minimal satu layanan pihak ketiga yang dapat dimanfaatkan mahasiswa, seperti Google Places Web Service.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

39. AIF486

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

40. IIE207

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

41. AIF202 implements HasPrasyarat

42. AIF202 implements HasResponsi

43. AIF202 implements HasPraktikum

Mata kuliah ini memperkenalkan kepada mahasiswa beberapa algoritma dan struktur data, alternatif cara implementasinya, dan analisis kompleksitas waktunya. Mahasiswa diberikan beberapa masalah komputasi yang harus diselesaikan dengan menggunakan algoritma atau struktur data yang sudah diperkenalkan dan mengimplementasikannya dalam bahasa pemrograman Java.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

44. AIF347

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

45. MKU009

Tujuan dari mata kuliah ini adalah untuk mendalami keterampilan berbahasa Indonesia, agar mampu mengkomunikasikan hasil pemikiran serta meningkatkan keterampilan dalam menyusun karya ilmiah. Mata kuliah Bahasa Indonesia ini dimulai dengan mempelajari penulisan kata baku dan non baku serta pengungkapan pikiran dengan puntuasi yang benar. Selanjutnya dipelajari penyusunan kalimat yang baku serta menghubungkan kalimat-kalimat yang padu dalam menuangkan gagasan dalam sebuah paragraf. Selain itu, dalam matakuliah ini dipelajari cara menyusun surat dinas yang jelas dan komunikatif. Di akhir kuliah ini, mahasiswa diberi tugas penyusunan makalah dengan benar.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

46. ESM203

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

47. APS402 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas MataKuliah

48. AIF306 implements HasPrasyarat

Mata kuliah ini bertujuan untuk memberikan pengalaman bagi mahasiswa dalam mengerjakan proyek dengan teknologi-teknologi terkini, secara berkelompok. Teknologi-teknologi yang digunakan pada kuliah ini tidak spesifik dan dapat berubah seiring perkembangan teknologi maupun disesuaikan dengan kompetensi dosen pengajar. Beberapa teknologi yang dapat dimanfaatkan antara lain: DVCS tool menggunakan Git + Github, Mobile native app (Android, iOS, dll), dan responsive web design.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas MataKuliah

49. AKS122

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

50. MKU002

Pendidikan Kewarganegaraan menjelaskan pentingnya pemahaman tentang identitas nasional Indonesia, hak dan kewajiban warga negara Indonesia serta hubungannya dengan hak dan kewajiban asasi manusia. Materi kuliah mencakup juga wawasan nusantara, ketahanan nasional, politik



dan strategi nasional, serta implementasinya dalam kehidupan bermasyarakat, berbangsa dan bernegara kesatuan Republik Indonesia.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

51. AIF462

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

52. AIF360 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

53. AIF337

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

54. AIF458 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

55. AIF301 implements HasPrasyarat

Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar tentang \*sistem cerdas dan komponen-komponennya. " "Terdapat 4 topik utama yang dibahas yaitu teknik pencarian untuk \*penyelesaian masalah, representasi pengetahuan dalam sistem \*cerdas, pemodelan ketidakpastian dalam masalah dan teknik pembelajaran mesin.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

#### 56. AIF182

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

#### 57. ESM204

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

#### 58. AIF205 implements HasPrasyarat

Mata kuliah ini memperkenalkan kepada mahasiswa arsitektur komputer sederhana, modern, dan Advance. Perbedaan, kelebihan dan kekurangan untuk masing-masing arsitektur. Selain itu mahasiswa juga mempelajari cara kerja dari komponen-komponen komputer, terutama memory, cache, system BUS dan input/output.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

#### 59. AIF317 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

60. AIF383

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

61. AIF442 implements `HasPraktikum`

62. AIF442 implements `HasPrasyarat`

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

63. ESM101

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

64. AIF403 implements `HasPrasyarat`

1. Memberikan wawasan kepada mahasiswa tentang kemunculan dan pemanfaatan teknologi baru, khususnya yang berkaitan dengan komputer, dan dampaknya terhadap masyarakat luas. 2. Memberikan kesadaran dan panduan bersikap kepada mahasiswa dalam menghadapi gejala yang disebabkan oleh munculnya teknologi baru, khususnya yang berkaitan dengan komputer.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

65. AIF402 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

66. AIF455

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

67. AIF443

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

68. AIF101 implements HasPraktikum

Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar pemrograman seperti pengulangan dan percabangan, konsep dasar penyimpanan data kontigu menggunakan array, konsep dasar pemrograman berorientasi objek seperti kelas & objek, method, dll, termasuk di dalamnya 4 prinsip dasar pemrograman berorientasi objek : data abstraction, encapsulation, inheritance dan polymorphism. Selain, itu diberikan masalah-masalah komputasi sederhana yang harus diselesaikan menggunakan konsep-konsep yang sudah diperkenalkan dan mengimplementasikannya menggunakan bahasa pemrograman Java

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

69. AIF382

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

70. AIF480

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

71. AIF316 implements HasPrasyarat

72. AIF316 implements HasPraktikum

Mata kuliah ini memperkenalkan konsep-konsep dasar komputasi paralel, dimana sebuah program yang berjalan secara paralel harus memiliki safety property dan liveness property. Mahasiswa dikenalkan dengan beberapa teknik pemrograman multi-thread seperti lock, monitor, barrier, thread

pool, dan sebagainya, yang diimplementasikan dalam bahasa pemrograman Java. Mahasiswa juga dikenalkan dengan beberapa metode untuk menganalisis kebenaran program baik secara matematis maupun secara praktis dengan bantuan model checker.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

#### 73. AIF438 implements HasPrasyarat

Mata kuliah ini: Memperkenalkan karakteristik dan teknik visualisasi dari berbagai jenis data yang dapat dianalisis dengan teknik-teknik data mining; mempelajari teknik-teknik penyiapan data untuk berbagai jenis data dan teknik data mining; mempraktekkan teknik-teknik penyiapan data untuk menganalisis data nyata/simulasi dengan memanfaatkan perangkat lunak aplikasi.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

#### 74. AIF204 implements HasPrasyarat

#### 75. AIF204 implements HasPraktikum

Mata kuliah ini memperkenalkan konsep dan arsitektur DBMS, mengajarkan aljabar relasional dan SQL serta pemanfaatannya pada pemrograman kueri sederhana s/d relatif kompleks. Selain itu, mata kuliah ini juga mengajarkan dan mempraktekkan perancangan basisdata untuk masalah sederhana (lingkup kecil) termasuk pengembangan program aplikasinya;

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

76. AIF341 implements HasPraktikum

Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar jaringan dan aplikasinya di kehidupan sehari-hari. Mahasiswa dikenalkan dengan teknologi-teknologi terbaru di bidang jaringan, sehingga mahasiswa memiliki pengetahuan yang dapat digunakan dalam kehidupan sehari-hari. Selain itu mahasiswa juga diperkenalkan dengan NetAcad, sebuah layanan dari Cisco yang dapat digunakan untuk memenuhi segala macam kebutuhan terkait dengan Cisco Academy.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

77. AIF183

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

78. AKS124

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

79. AIF459

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

80. AIF336

Mata kuliah ini merupakan mata kuliah lanjutan dari mata kuliah Keamanan Informasi, dengan titik berat pada materi kriptografi. Mata kuliah ini memperkenalkan tambahan konsep kriptografi, misalnya tentang otentikasi yaitu otentikasi entitas, manajemen kunci, dan bentuk lain dari metode merahasiakan pesan, yaitu dengan menggunakan secret sharing. Selanjutnya, diperkenalkan juga penggunaan kriptografi pada protokol-protokol yang sebenarnya banyak digunakan sehari-hari, misalnya pada e-cash, auction, dan electronic voting.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

81. AIF463

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

82. AIF208 implements HasPrasyarat

Mata kuliah ini memperkenalkan kepada mahasiswa tahapan rekayasa perangkat lunak, terutama dengan paradigma berorientasi objek, dilengkapi dengan pengenalan tentang manajemen proyek perangkat lunak. Selain, itu diberikan deskripsi proyek berskala kecil yang harus dikerjakan oleh mahasiswa dalam kelompok dengan menerapkan teori yang telah dipelajarinya.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

83. MKU003

Mata kuliah ini membentuk karakteristik mahasiswa sebagai manusia yang memiliki religiusitas melalui pendalaman akan makna agama dan beragama, mendeteksi dinamika Wahyu Tuhan dan iman mereka, memahami relasi dengan Tuhan dan sesama, mengenal makna keselamatan dalam konteks Kerajaan Allah, dan mampu menyatakan ajaran Gereja dalam pelayanan terhadap orang miskin dan terlantar.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

84. APS309

APS302 atau APS309 ?

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

85. AIF335

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

86. AIF362 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -

– java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

87. AIF460

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

88. AIF358

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

89. AMS100

Sistem Bilangan, Fungsi, Limit dan Kekontinuan Fungsi, Turunan, Integral, Penggunaan Integral, Sistem Persamaan Linear, Determinan, Vektor, Nilai dan Vektor Eigen.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

90. AIF401 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)

**Parameter:**

– Mahasiswa mahasiswa -

– java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

91. AIF456

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

92. SIR104

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

93. AIF339 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)

**Parameter:**

– Mahasiswa mahasiswa -



– `java.util.List reasonsContainer` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

94. AIF102 implements `HasPrasyarat`

95. AIF102 implements `HasPraktikum`

Mata kuliah ini memperkenalkan berbagai algoritma dan teknik-teknik penyelesaian masalah komputasi seperti rekursif, sorting, teknik divide dan conquer, serta exhaustive search. Selain itu, pada kuliah ini juga dikenalkan berbagai struktur data yang dapat digunakan untuk mendukung penyelesaian masalah komputasi seperti ADT List, Stack dan Queue. Baik algoritma maupun struktur data yang dikenalkan harus dapat diimplementasikan dan digunakan oleh mahasiswa untuk menyelesaikan masalah dengan menggunakan suatu bahasa pemrograman berorientasi objek.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

– `Mahasiswa mahasiswa` -

– `java.util.List reasonsContainer` -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

96. AIF381

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

97. AIF483

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

98. AIF315 implements `HasPrasyarat`

99. AIF315 implements `HasPraktikum`

Mata kuliah ini memperkenalkan konsep dan lingkungan pemrograman berbasis web, kemudian belajar membuat aplikasi berbasis web menggunakan HTML5, CSS, Java Script dan PHP. Untuk meningkatkan keterampilan pemrograman dilengkapi dengan praktikum. Sedangkan untuk mendapatkan pengalaman penerapan konsep diberikan tugas besar membuat program berbasis web dengan kasus yang ditentukan oleh mahasiswa.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

100. AIF342 implements HasPraktikum

101. AIF342 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

102. IIE214

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

103. AIF303 implements HasPrasyarat

Mempelajari Konsep Data, Informasi, Pengetahuan, Sistem Informasi, proses dan pemodelan bisnis, jenis-jenis sistem informasi, untuk mendukung pengambilan keputusan. Mempelajari trend Teknologi Informasi, tahap-tahap pembangunan sistem informasi. Mempelajari pengantar : EIS, e-bisnis/e-commerce, Business Intelligence, Cloud Computing dan Mobile Applications

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

104. AIF302 implements `HasPrasyarat`

Mata kuliah ini melatih mahasiswa dalam menulis ilmiah serta memperkenalkan metodologi penelitian serta kakas untuk menulis ilmiah.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

105. AIF181

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

106. AIF210

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

107. AIF343

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

108. AIF206 implements `HasPrasyarat`

Mata kuliah ini memperkenalkan kepada mahasiswa mengenai konsep sistem operasi, jenis-jenis sistem operasi yang digunakan dalam kehidupan sehari-hari dan beberapa perangkat keras yang dibutuhkan pada komputer. Selain itu juga mempelajari mengenai teknik dan algoritma yang digunakan dalam pengelolaan sistem operasi.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

109. AIF314 implements `HasPrasyarat`

110. AIF314 implements `HasPraktikum`

Kuliah ini merupakan kelanjutan dari kuliah Manajemen Informasi Basis-data. Pada perkuliahan ini, mahasiswa akan mempelajari teknik-teknik pengelolaan basis data dan membuat program dengan basis data yang optimal/efisien.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

111. AIF380

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

112. AIF103

Mata kuliah ini merupakan salah satu cara untuk mencapai kompetensi dasar tentang matematika diskrit yang prinsipnya banyak digunakan dalam bidang ilmu komputer. Selain itu, kuliah ini juga merupakan cara untuk membentuk pola pikir logis yang dibutuhkan untuk menempuh kuliah-kuliah di tingkat yang lebih tinggi.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

113. AIF292

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

114. AIF441 implements `HasPraktikum`

115. AIF441 implements `HasPrasyarat`

Mata kuliah ini memperkenalkan kepada mahasiswa konsep jaringan lanjut terutama di layer data link dan layer network. Materi utama dari mata kuliah ini adalah pengembangan jaringan dan pengenalan fungsi-fungsi yang terdapat pada alat jaringan Cisco yang berkaitan dengan layer 2 dan layer 3.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

116. AIF457 implements HasPrasyarat

Mata kuliah ini memperkenalkan konsep kewirausahaan dengan memanfaatkan teknologi, khususnya teknologi informasi, sebagai basis usaha dan inovasi produk/jasa; Mempelajari teknik mencari peluang dan merumuskan bidang usaha spesifik yang akan diterjuni; Mempelajari konsep manajemen pemasaran, keuangan dan SDM dalam kaitannya dengan berwira-usaha di bidang TI; Menyusun proposal bisnis untuk berwira-usaha di bidang TI dan mempresentasikannya.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

117. MKU001

Mata Kuliah Pendidikan Pancasila berupaya menelaah/mengkaji berbagai fenomena kehidupan bangsa dan Negara Indonesia sebagai sebuah ruang publik dengan menggunakan pendekatan hermeneutika (filsafat) dan pendidikan nilai (pedagogik). Dengan bantuan hermeneutika mahasiswa diajak berpikir kritis terhadap segala bentuk ideologisme Pancasila dan melalui pendidikan nilai mahasiswa dilatih untuk memiliki nilai Pancasila. Nilai pengembangan diri intra-personal dan relasi interpersonal dapat tertanam melalui pendidikan Pancasila yang tujuannya adalah membangun kepribadian (character building) manusia Indonesia yang utuh, baik menyangkut aspek kognitif, afektif, maupun psikomotor.

Dengan demikian, Pendidikan Pancasila mengajak mahasiswa menilai realitas ruang publik sehari-hari secara mandiri dengan panduan nilai-nilai etis Pancasila.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

118. ESA101

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

119. AMS290

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

120. AIF461

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

121. AIF318 implements HasPrasyarat

122. AIF318 implements HasPraktikum

Mata kuliah ini memperkenalkan konsep perangkat mobile dan pemrograman pada perangkat mobile. Pemrograman dikhususkan pada lingkungan J2ME dan Android. Untuk meningkatkan keterampilan pemrograman dilengkapi dengan praktikum. Sedangkan untuk mendapatkan pengalaman penerapan konsep diberikan tugas implementasi suatu kasus pada lingkungan mobile-cloud dengan kasus yang sudah ditentukan.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

123. AIF334

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

124. ESM105

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

125. AIF450

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

126. AIF446

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

127. AIF104

Mata kuliah ini memberikan pengetahuan tentang logika yang digunakan di dalam ilmu komputer. Dalam kuliah ini, mahasiswa belajar untuk bisa memodelkan suatu kalimat dalam kehidupan sehari-hari, ke dalam kalimat dengan sintaks tertentu, yang hanya memiliki satu arti. Lalu, diperkenalkan juga, bagaimana mengartikan suatu kalimat (benar atau salah) dan bagaimana menentukan sifat dari kalimat tersebut.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

128. AIF387

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

129. AIF313 implements HasPraktikum

Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar pembuatan grafik dengan menggunakan komputer seperti mengenal berbagai algoritma pembuatan primitif 2 dimensi seperti titik, garis, lingkaran, elips, berbagai macam bentuk kurva, fraktal, konsep warna (RGB), dasar-dasar grafika 3 dimensi seperti pewarnaan, pencahayaan, pemberian tekstur pada objek, transformasi, animasi, dan sebagainya. Selain itu diberikan masalah-masalah komputasi sederhana yang harus diselesaikan menggunakan konsep-konsep yang sudah diperkenalkan dan mengimplementasikan-nya menggunakan bahasa pemrograman Java.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

130. AIF344 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

131. AIF201 implements HasPrasyarat

132. AIF201 implements HasPraktikum

133. AIF201 implements HasResponsi

Mata kuliah ini memperkenalkan prinsip-prinsip yang digunakan dalam melakukan analisa serta desain program berorientasi objek. Di samping itu, mahasiswa juga belajar menggunakan kakas berupa diagram UML (Unified Modelling Language) sehingga dapat mengkomunikasikan desain secara visual. Mahasiswa juga akan mengenal beberapa software design pattern dari Gang of Four. Terakhir, mahasiswa akan belajar mengenai konsep MVC (Model-View-Controller) yang menjadi dasar dari banyak framework masa kini. Bahasa yang digunakan adalah bahasa Java, namun diusahakan tetap umum sehingga dapat diaplikasikan pada bahasa yang lain.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas MataKuliah

134. AIF352

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

135. AIF305 implements HasPrasyarat

Mata kuliah ini memperkenalkan kepada mahasiswa konsep dasar jaringan komputer dengan menggunakan top-down. Selain itu mengajarkan juga kepada mahasiswa mengenai aplikasi-aplikasi berbasis jaringan sehingga diharapkan mahasiswa dapat membuat aplikasi berbasis jaringan dengan menggunakan socket. Pada akhirnya, mahasiswa akan ditugaskan untuk membangun jaringan komputer LAN, baik menggunakan kabel maupun nirkabel.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -



**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

136. MKU010

Mata kuliah ini difokuskan pada pemahaman sumber referensi dalam Bahasa Inggris dan pengembangan kosakata Bahasa Inggris (vocabularies). Hampir keseluruhan waktu perkuliahan didedikasikan untuk menjelaskan metode mengekstraksi isi bacaan secara tepat dan melatih mahasiswa untuk menerapkan metode tersebut seraya menambah kosakata-kosakata baru. Mahasiswa juga dilatih untuk mempresentasikan hasil pemahamannya akan isi bahan bacaan.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

137. MKU011

Mata kuliah estetika memberi pemahaman konseptual filosofis "seni" dalam khasanah keilmuan, pembentukan kesadaran ekologis juga dalam proses pembudayaan dan peradaban. Mata kuliah ini akan menjadi fondasi bagi mahasiswa untuk memahami dan mempraktekkan seni dari sudut pandang filsafat, sejarah, kultural, dan global. Melalui mata kuliah ini, mahasiswa mempelajari mengenai dunia manusia (manusia dan pikirannya), pluralitas dan relativitas seni, serta aliran-aliran seni rupa Barat.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

138. AIF332 implements HasPrasyarat

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

139. AIF304 implements HasPrasyarat

140. AIF304 implements HasPraktikum

141. AIF304 implements HasResponsi

Mata kuliah ini memberikan kesempatan bagi mahasiswa untuk memperdalam konsep tentang pengembangan sistem informasi dan mempraktekkan analisis kebutuhan, analisis sistem dan perancangan sistem pada organisasi studi kasus;

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

142. ESM201

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

143. AIF312 implements HasPrasyarat

144. AIF312 implements HasPraktikum

Mata kuliah ini memberikan pengetahuan awal tentang keamanan informasi. Pada beberapa pertemuan awal, dibahas keamanan informasi secara matematis, yaitu di materi-materi seputar kriptografi dan serangannya. Lalu, dibahas pula konsep keamanan informasi pada jaringan komputer dan pada software.

Kelas ini tidak memiliki atribut. *Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** `checkPrasyarat` dari kelas `MataKuliah`

145. AIF484

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

146. AIF191

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

147. AIF386

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

148. AIF105

Mata kuliah ini memperkenalkan kepada mahasiswa terminologi dan konsep dasar yang akan banyak dipakai sepanjang kuliah di Teknik Informatika. Selain itu mata kuliah ini juga mempersiapkan dan membiasakan mahasiswa dengan suasana akademik yang khas perguruan tinggi seperti kedisiplinan, kerja sama, kemampuan menggunakan teknologi informasi dalam pembuatan tugas, kemampuan komunikasi, dsb.

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

149. AIF294

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

150. AIF282

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

151. AIF451

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

152. APS302

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

153. EAA101

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

154. HasPrasyarat

Mendefinisikan kelas-kelas yang memiliki prasyarat, terkustomisasi untuk seorang **Mahasiswa**. Jika ada tambahan, jangan lupa untuk mendaftarkannya di `DEFAULT_HASPRASYARAT_CLASSES`. Jika berubah package, jangan lupa mengupdate `DEFAULT_PRASYARAT_PACKAGE`.

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- `String DEFAULT_HASPRASYARAT_CLASSES` - Daftar dari nama kelas default seluruh turunan interface ini. Perlu didaftarkan manual, karena Java reflection tidak dapat mendeteksi otomatis.
- `String DEFAULT_PRASYARAT_PACKAGE` - Package tempat menyimpan seluruh turunan standar interface ini. Perlu didefinisikan manual, karena Java reflection tidak dapat mendeteksi otomatis.

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- `public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)`

Memeriksa prasyarat-prasyarat dari kuliah, spesifik untuk mahasiswa yang dituju. Jika ada pesan-pesan khusus, akan ditambahkan pada parameter `reasonsContainer`.

**Parameter:**

- Mahasiswa mahasiswa - prasyarat kuliah akan diperiksa spesifik pada mahasiswa ini
- List reasonsContainer - pesan-pesan terkait prasyarat akan ditambahkan di sini, jika ada.

**Return Value:** true jika seluruh prasyarat dipenuhi, false jika tidak.

**Exception:** Tidak memiliki *exception*

155. HasPraktikum

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

156. HasResponsi

Kelas ini tidak memiliki atribut. Kelas ini tidak memiliki method.

157. Kelulusan implements HasPrasyarat

Atribut yang dimiliki kelas ini adalah sebagai berikut.

- String PILIHAN\_WAJIB -
- String WAJIB -
- String AGAMA -
- int MIN\_SKS\_LULUS -
- int MIN\_PILIHAN\_WAJIB -

*Method-method* yang dimiliki kelas ini adalah sebagai berikut.

- public boolean checkPrasyarat(id.ac.unpar.siamodels.Mahasiswa mahasiswa, java.util.List reasonsContainer)

**Parameter:**

- Mahasiswa mahasiswa -
- java.util.List reasonsContainer -

**Return Value:** Tidak memiliki *return value*

**Exception:** Tidak memiliki *exception*

**Override:** checkPrasyarat dari kelas Object

**LAMPIRAN D**  
**HASIL PDF TEXDOCLET**

# TeXDoclet Java Documentation

Created with Javadoc TeXDoclet Doclet

Greg Wonderly

Søren Caspersen

Stefan Marx

October 3, 2012

# Contents

<b>Class Hierarchy</b>	<b>2</b>
<b>1 Package org.stfm.texdoclet</b>	<b>3</b>
1.1 Interface ClassFilter	4
1.1.1 Declaration	4
1.1.2 All known subinterfaces	4
1.1.3 All classes known to implement interface	4
1.1.4 Method summary	4
1.1.5 Methods	4
1.2 Class ClassHierachy	4
1.2.1 Declaration	4
1.2.2 Field summary	4
1.2.3 Constructor summary	4
1.2.4 Method summary	5
1.2.5 Fields	5
1.2.6 Constructors	5
1.2.7 Methods	5
1.3 Class HelpOutput	5
1.3.1 Declaration	5
1.3.2 Constructor summary	6
1.3.3 Method summary	6
1.3.4 Constructors	6
1.3.5 Methods	6
1.4 Class HTMLtoLaTeXBackEnd	6
1.4.1 See also	6
1.4.2 Declaration	6
1.4.3 Constructor summary	6
1.4.4 Method summary	6
1.4.5 Constructors	7
1.4.6 Methods	7
1.4.7 Members inherited from class HTMLEditorKit.ParserCallback	8
1.5 Class InterfaceHierachy	8
1.5.1 Declaration	8
1.5.2 Field summary	8
1.5.3 Constructor summary	8
1.5.4 Method summary	8

1.5.5	Fields	8
1.5.6	Constructors	8
1.5.7	Methods	8
1.6	Class <code>MarkdownTest</code>	9
1.6.1	Declaration	13
1.6.2	Constructor summary	13
1.6.3	Constructors	13
1.7	Class <code>Package</code>	13
1.7.1	See also	13
1.7.2	Declaration	13
1.7.3	Field summary	14
1.7.4	Constructor summary	14
1.7.5	Method summary	14
1.7.6	Fields	14
1.7.7	Constructors	14
1.7.8	Methods	15
1.8	Class <code>TableInfo</code>	15
1.8.1	Declaration	16
1.8.2	Constructor summary	16
1.8.3	Method summary	16
1.8.4	Constructors	17
1.8.5	Methods	17
1.9	Class <code>TestFilter</code>	18
1.9.1	Declaration	18
1.9.2	Constructor summary	18
1.9.3	Method summary	18
1.9.4	Constructors	19
1.9.5	Methods	19
1.10	Class <code>TeXDoclet</code>	19
1.10.1	See also	21
1.10.2	Declaration	21
1.10.3	Field summary	21
1.10.4	Constructor summary	22
1.10.5	Method summary	22
1.10.6	Fields	22
1.10.7	Constructors	22
1.10.8	Methods	22
1.10.9	Members inherited from class <code>Doclet</code>	23



# Class Hierarchy

## Classes

- `java.lang.Object`
  - `com.sun.javadoc.Doclet`
    - `org.stfm.texdoclet.TeXDoclet` (in [1.10](#), page [19](#))
  - `javax.swing.text.html.HTMLEditorKit.ParserCallback`
    - `org.stfm.texdoclet.HTMLtoLaTeXBackEnd` (in [1.4](#), page [6](#))
  - `org.stfm.texdoclet.ClassHierarchy` (in [1.2](#), page [4](#))
  - `org.stfm.texdoclet.HelpOutput` (in [1.3](#), page [5](#))
  - `org.stfm.texdoclet.InterfaceHierarchy` (in [1.5](#), page [8](#))
  - `org.stfm.texdoclet.MarkdownTest` (in [1.6](#), page [9](#))
  - `org.stfm.texdoclet.Package` (in [1.7](#), page [13](#))
  - `org.stfm.texdoclet.TableInfo` (in [1.8](#), page [15](#))
  - `org.stfm.texdoclet.TestFilter` (in [1.9](#), page [18](#))

## Interfaces

- `org.stfm.texdoclet.ClassFilter` (in [1.1](#), page [4](#))

# Chapter 1

## Package org.stfm.texdoclet

<i>Package Contents</i>	<i>Page</i>
<b>Interfaces</b>	
<b>ClassFilter</b> .....	4
This interface can be implemented and a class name provided to the Doclet to filter which classes are and are not included in the output document.	
<b>Classes</b>	
<b>ClassHierarchy</b> .....	4
Manages and prints a class hierarchy.	
<b>HelpOutput</b> .....	5
<b>HTMLtoLaTeXBackEnd</b> .....	6
This class implements a <code>ParserCallback</code> that translates HTML to the corresponding $\text{\LaTeX}$ .	
<b>InterfaceHierarchy</b> .....	8
Manages and prints a interface hierarchy.	
<b>MarkdownTest</b> .....	9
This class is just for testing the Markdown processing output.	
<b>Package</b> .....	13
This class is used to manage the contents of a Java package.	
<b>TableInfo</b> .....	15
This class provides support for converting HTML tables into $\text{\LaTeX}$ tables.	
<b>TestFilter</b> .....	18
This class filters out classes beginning with "Test" when applied to the Doclet.	
<b>TeXDoclet</b> .....	19
This class provides a Java <code>javadoc</code> Doclet which generates a $\text{\LaTeX}$ 2 $\epsilon$ document out of the java classes that it is used on.	

This doclet is based on the doclet originally created by Greg Wonderly of [C2 technologies Inc.](http://java.net/projects/texdoclet) and its revision by [XO Software](http://java.net/projects/texdoclet). The project of Greg Wonderly is available here : <http://java.net/projects/texdoclet>.

## 1.1 Interface ClassFilter

This interface can be implemented and a class name provided to the Doclet to filter which classes are and are not included in the output document.

### 1.1.1 Declaration

public interface ClassFilter

### 1.1.2 All known subinterfaces

TestFilter (in 1.9, page 18)

### 1.1.3 All classes known to implement interface

TestFilter (in 1.9, page 18)

### 1.1.4 Method summary

[includeClass\(ClassDoc\)](#) Filters the ClassDoc passed.

### 1.1.5 Methods

- **includeClass**  
boolean **includeClass**(com.sun.javadoc.ClassDoc cd)
  - **Description**  
Filters the ClassDoc passed. If true is returned, the passed class will be included into the output. If false is returned, this document will not be included.

## 1.2 Class ClassHierarchy

Manages and prints a class hierarchy. Use **add** to add another class to the hierarchy. Use **printTree** to print the corresponding L<sup>A</sup>T<sub>E</sub>X.

### 1.2.1 Declaration

public class ClassHierarchy  
**extends** java.lang.Object

### 1.2.2 Field summary

[root](#)

### 1.2.3 Constructor summary

[ClassHierarchy\(\)](#) Creates new ClassHierarchy

### 1.2.4 Method summary

**add(ClassDoc)** Adds another class to the hierarchy  
**printBranch(RootDoc, SortedMap, double, double)** Prints a branch of the tree.  
**printTree(RootDoc, double)** Prints the L<sup>A</sup>T<sub>E</sub>X corresponding to the tree.

### 1.2.5 Fields

- public java.util.SortedMap **root**

### 1.2.6 Constructors

- **ClassHierarchy**  
 public **ClassHierarchy()**
  - **Description**  
 Creates new **ClassHierarchy**

### 1.2.7 Methods

- **add**  
 protected java.util.SortedMap **add**(com.sun.javadoc.ClassDoc **cls**)
  - **Description**  
 Adds another class to the hierarchy
- **printBranch**  
 protected void **printBranch**(com.sun.javadoc.RootDoc **rootDoc**, java.util.SortedMap **map**, double **indent**, double **overviewindent**)
  - **Description**  
 Prints a branch of the tree. The branch is printed using **TeXDoclet.os**.
- **printTree**  
 public void **printTree**(com.sun.javadoc.RootDoc **rootDoc**, double **overviewindent**)
  - **Description**  
 Prints the L<sup>A</sup>T<sub>E</sub>X corresponding to the tree. The tree is printed using **TeXDoclet.os**.

## 1.3 Class HelpOutput

### 1.3.1 Declaration

```
public class HelpOutput
extends java.lang.Object
```

### 1.3.2 Constructor summary

[HelpOutput\(\)](#)

### 1.3.3 Method summary

[printHelp\(\)](#)

### 1.3.4 Constructors

- **HelpOutput**  
public **HelpOutput()**

### 1.3.5 Methods

- **printHelp**  
protected static void **printHelp()**

## 1.4 Class HTMLtoLaTeXBackEnd

This class implements a `ParserCallback` that translates HTML to the corresponding  $\text{\LaTeX}$ . Not all tags are processed but the most common are.

HTML links to files located in the doc-files directory (appendix\_a.html, appendix\_b.txt) are transformed to references to the appendix, whereby the referenced files itself are included in the appendix.

### 1.4.1 See also

- [javax.swing.text.html.parser.ParserDelegator](#)

### 1.4.2 Declaration

```
public class HTMLtoLaTeXBackEnd
extends javax.swing.text.html.HTMLEditorKit.ParserCallback
```

### 1.4.3 Constructor summary

[HTMLtoLaTeXBackEnd\(StringBuffer\)](#) Constructs a new instance.

### 1.4.4 Method summary

[fixText\(String\)](#) Converts a HTML string into  $\text{\LaTeX}$  using an instance of `HTMLtoLaTeXBackEnd`.

[handleEndTag\(HTML.Tag, int\)](#) This method handles HTML tags that mark an ending (e.g.

[handleSimpleTag\(HTML.Tag, MutableAttributeSet, int\)](#) This method handles simple HTML tags (e.g.

**handleStartTag(HTML.Tag, MutableAttributeSet, int)** This method handles HTML tags that mark a beginning (e.g. **handleText(char[], int)** This method handles all other text.

#### 1.4.5 Constructors

- **HTMLtoLaTeXBackEnd**  
`public HTMLtoLaTeXBackEnd(java.lang.StringBuffer ret)`
  - **Description**  
Constructs a new instance.
  - **Parameters**  
    - \* `StringBuffer` – The `StringBuffer` where the translated HTML is appended.

#### 1.4.6 Methods

- **fixText**  
`public static java.lang.String fixText(java.lang.String str)`
  - **Description**  
Converts a HTML string into  $\text{\LaTeX}$  using an instance of `HTMLtoLaTeXBackEnd`.
- **handleEndTag**  
`public void handleEndTag(javax.swing.text.html.HTML.Tag tag, int pos)`
  - **Description**  
This method handles HTML tags that mark an ending (e.g. `</P>`-tags). It is called by the parser whenever such a tag is encountered.
- **handleSimpleTag**  
`public void handleSimpleTag(javax.swing.text.html.HTML.Tag tag, javax.swing.text.MutableAttributeSet attrSet, int pos)`
  - **Description**  
This method handles simple HTML tags (e.g. `<HR>`-tags). It is called by the parser whenever such a tag is encountered.
- **handleStartTag**  
`public void handleStartTag(javax.swing.text.html.HTML.Tag tag, javax.swing.text.MutableAttributeSet attrSet, int pos)`
  - **Description**  
This method handles HTML tags that mark a beginning (e.g. `<P>`-tags). It is called by the parser whenever such a tag is encountered.
- **handleText**  
`public void handleText(char[] data, int pos)`
  - **Description**  
This method handles all other text.

### 1.4.7 Members inherited from class `HTMLEditorKit.ParserCallback`

`javax.swing.text.html.HTMLEditorKit.ParserCallback`  
`flush`, `handleComment`, `handleEndOfLineString`, `handleEndTag`, `handleError`, `handleSimpleTag`,  
`handleStartTag`, `handleText`, `IMPLIED`

## 1.5 Class `InterfaceHierarchy`

Manages and prints a interface hierarchy. Use `add` to add another interface to the hierarchy. Use `printTree` to print the corresponding  $\text{\LaTeX}$ .

### 1.5.1 Declaration

```
public class InterfaceHierarchy
extends java.lang.Object
```

### 1.5.2 Field summary

[`root`](#)

### 1.5.3 Constructor summary

[`InterfaceHierarchy\(\)`](#) Creates new `InterfaceHierarchy`

### 1.5.4 Method summary

[`add\(ClassDoc\)`](#) Adds another interface to the hierarchy  
[`printBranch\(RootDoc, SortedMap, double, double\)`](#) Prints a branch of the  
tree.  
[`printTree\(RootDoc, double\)`](#) Prints the  $\text{\LaTeX}$  corresponding to the tree.

### 1.5.5 Fields

- `public java.util.SortedMap` **`root`**

### 1.5.6 Constructors

- **`InterfaceHierarchy`**  
`public` **`InterfaceHierarchy()`**  
  - **Description**  
Creates new `InterfaceHierarchy`

### 1.5.7 Methods

- **`add`**  
`protected` `java.util.SortedMap` **`add(com.sun.javadoc.ClassDoc cls)`**

- **Description**

Adds another interface to the hierarchy

- **printBranch**

```
protected void printBranch(com.sun.javadoc.RootDoc rootDoc,
    java.util.SortedMap map, double indent, double overviewindent)
```

- **Description**

Prints a branch of the tree. The branch is printed using `TeXDoclet.os`.

- **printTree**

```
public void printTree(com.sun.javadoc.RootDoc rootDoc, double
    overviewindent)
```

- **Description**

Prints the  $\text{\LaTeX}$  corresponding to the tree. The tree is printed using `TeXDoclet.os`.

## 1.6 Class MarkdownTest

This class is just for testing the Markdown processing output.

### a) Some text

**Markdown code :**

```
some text some text some text some text some text some text some
text some text some text some text some text some text some text
some text some text some text with 2 ending spaces
```

```
text some text some text some text some text some text some text
some text some text some text some text some text some text some text
```

```
text some text some text some text some text some text some text
some text some text
```

**results in :**

some text some text some text some text some text some text some text some text some text some text some text some text some text some text some text some text some text with 2 ending spaces

text some text some text some text some text some text some text some text some text some text some text some text some text some text some text some text some text

text some text some text some text some text some text some text some text some text some text

### b) Lists

**Markdown code :**

```
unsorted :
```



```
- item1
  - item11
  - item12
- item2
```

or :

```
+ item1
  + item12
    + item13
```

sorted :

```
1. item1
  1. item11
  2. item12
2. item2
  - item21
  - item22
3. item3
```

lists with paragraphs :

```
1. some text some text some text some text some text some text some text some
text some text some text
```

```
    some text some text some text some text some text some text
```

```
2. some text some text some text some text some text some text
```

**results in :**

unsorted :

```
• item1
  - item11
  - item12
• item2
```

or :

```
• item1
  - item12
    * item13
```

sorted :

```
1. item1
```

- (a) item11
- (b) item12

## 2. item2

- item21
- item22

## 3. item3

lists with paragraphs :

1. some text some text some text some text some text some text some text some text some text some text  
some text some text some text some text some text some text
2. some text some text some text some text some text some text

### c) Blockquotes

**Markdown code :**

```
some text some text some text some text some text some text
```

```
> some quoting text
>
> > some nested quoting text
>
> some quoting text
>
> ##### header in blockquote
>
> a list in blockquote :
>
> 1. item1
> 2. item2
>   1. item21
>   2. item22
> 3. item3
>
> some quoting text
>
>   code in blockquote
```

**results in :**

```
some text some text some text some text some text some text
some quoting text
some nested quoting text
some quoting text
```

**header in blockquote**

a list in blockquote :

1. item1
2. item2
  - (a) item21
  - (b) item22
3. item3

some quoting text  
code in blockquote

**d) Preformatted text**

**Markdown code :**

some preformatted :

```
code line 1
code line 2
```

**results in :**

```
some preformatted :
code line 1
code line 2
```

**e) Horizontal rules**

**Markdown code :**

```
***
```

**results in :**

---

**f) Emphasis**

**Markdown code :**

```
*single asterisks* (em)
```

```
_single underscores_ (em)
```

```
**double asterisks** (strong)
```

```
__double underscores__ (strong)
```

results in :

*single asterisks* (em)  
*single underscores* (em)  
**double asterisks** (strong)  
**double underscores** (strong)

## h) Code

Markdown code :

```
some code : 'TeXDoclet extends Doclet' and ''There is a literal backtick (')
here.'''
```

results in :

```
some code : TeXDoclet extends Doclet and There is a literal backtick (') here.
```

### 1.6.1 Declaration

```
public class MarkdownTest
extends java.lang.Object
```

### 1.6.2 Constructor summary

[MarkdownTest\(\)](#)

### 1.6.3 Constructors

- **MarkdownTest**  
 public **MarkdownTest()**

## 1.7 Class Package

This class is used to manage the contents of a Java package. It accepts ClassDoc objects and examines them and groups them according to whether they are classes, interfaces, exceptions or errors. The accumulated Vectors can then be processed to get to all of the elements of the package that fall into each category. If needed the classes, interfaces, exceptions and errors can be sorted using the `sort` method.

### 1.7.1 See also

- [Package.sort\(\)](#) (in 1.7.8, page 15)

### 1.7.2 Declaration

```
public class Package
extends java.lang.Object
```

### 1.7.3 Field summary

**classes** The classes this package has in it  
**errors** The errors this package has in it  
**exceptions** The exceptions this package has in it  
**interfaces** The interfaces this package has in it  
**pkg** The name of the package this object is for  
**pkgDoc**

### 1.7.4 Constructor summary

**Package(String, PackageDoc)** Construct a new object corresponding to the passed package name.

### 1.7.5 Method summary

**addElement(ClassDoc)** Adds a ClassDoc element to this package.  
**sort()** Sorts the vectors of classes, interfaces exceptions and errors.

### 1.7.6 Fields

- protected com.sun.javadoc.PackageDoc **pkgDoc**
- protected java.lang.String **pkg**
  - The name of the package this object is for
- protected java.util.Vector **classes**
  - The classes this package has in it
- protected java.util.Vector **interfaces**
  - The interfaces this package has in it
- protected java.util.Vector **exceptions**
  - The exceptions this package has in it
- protected java.util.Vector **errors**
  - The errors this package has in it

### 1.7.7 Constructors

- **Package**  
 public **Package**(java.lang.String **pkg**, com.sun.javadoc.PackageDoc **doc**)
  - **Description**  
 Construct a new object corresponding to the passed package name.
  - **Parameters**
    - \* **pkg** – the package name to use

### 1.7.8 Methods

- **addElement**

```
public void addElement(com.sun.javadoc.ClassDoc cd)
```

- **Description**

Adds a ClassDoc element to this package.

- **Parameters**

\* `cd` – the object to add to this package

- **sort**

```
public void sort()
```

- **Description**

Sorts the vectors of classes, interfaces exceptions and errors.

## 1.8 Class TableInfo

This class provides support for converting HTML tables into  $\text{\LaTeX}$  tables. Some of the things **NOT** implemented include the following:

- valign attributes are not processed, but align= is.
- rowspan attributes are not processed, but colspan= is.
- the argument to border= in the table tag is not used to control line size

Here is an example table.

Column 1 Heading	Column two heading	Column three heading																								
data	Span two columns																									
more data	right	left																								
<div><table><tr><td colspan="3">A nested table example</td></tr><tr><td>Column one Heading</td><td>Column two heading</td><td>Column three heading</td></tr><tr><td>data</td><td colspan="2">Span two columns</td></tr><tr><td>more data</td><td>right</td><td>left</td></tr><tr><td>1</td><td>first line</td><td></td></tr><tr><td>2</td><td>second line</td><td></td></tr><tr><td>3</td><td>third line</td><td></td></tr><tr><td>4</td><td>fourth line</td><td></td></tr></table></div>			A nested table example			Column one Heading	Column two heading	Column three heading	data	Span two columns		more data	right	left	1	first line		2	second line		3	third line		4	fourth line	
A nested table example																										
Column one Heading	Column two heading	Column three heading																								
data	Span two columns																									
more data	right	left																								
1	first line																									
2	second line																									
3	third line																									
4	fourth line																									

1.8.1 Declaration

```
public class TableInfo
extends java.lang.Object
```

1.8.2 Constructor summary

```
TableInfo()
```

1.8.3 Method summary

- endCol() Ends the current column.
- endRow() Ends the current row.
- endTable() Ends the table, closing the last row as needed
- startCol(MutableAttributeSet) Starts a new column, possibly closing the current column if needed
- startHeadCol(MutableAttributeSet) Starts a new Heading column, possibly closing the current column if needed.
- startRow(MutableAttributeSet) Starts a new row, possibly closing the current row if needed

**startTable(StringBuffer, MutableAttributeSet)** Constructs a new table object and starts processing of the table by scanning the `<table>` passed to count columns.

#### 1.8.4 Constructors

- **TableInfo**  
`public TableInfo()`

#### 1.8.5 Methods

- **endCol**  
`public void endCol()`
  - **Description**  
Ends the current column.
  - **Parameters**  
    - \* **ret** – The output buffer to put  $\text{\LaTeX} 2_{\epsilon}$  into.
- **endRow**  
`public void endRow()`
  - **Description**  
Ends the current row.
  - **Parameters**  
    - \* **ret** – The output buffer to put  $\text{\LaTeX} 2_{\epsilon}$  into.
- **endTable**  
`public java.lang.StringBuffer endTable()`
  - **Description**  
Ends the table, closing the last row as needed
  - **Parameters**  
    - \* **ret** – The output buffer to put  $\text{\LaTeX} 2_{\epsilon}$  into.
- **startCol**  
`public void startCol(javax.swing.text.MutableAttributeSet attrSet)`
  - **Description**  
Starts a new column, possibly closing the current column if needed
  - **Parameters**  
    - \* **ret** – The output buffer to put  $\text{\LaTeX} 2_{\epsilon}$  into.
    - \* **p** – the properties from the `<td>` tag
- **startHeadCol**  
`public void startHeadCol(javax.swing.text.MutableAttributeSet attrSet)`



- **Description**

Starts a new Heading column, possibly closing the current column if needed. A Heading column has a Bold Face font directive around it.

- **Parameters**

- \* **ret** – The output buffer to put  $\text{\LaTeX} 2_{\epsilon}$  into.
- \* **p** – The properties from the `<th>` tag

- **startRow**

```
public void startRow(javax.swing.text.MutableAttributeSet attrSet)
```

- **Description**

Starts a new row, possibly closing the current row if needed

- **Parameters**

- \* **ret** – The output buffer to put  $\text{\LaTeX}$  into.
- \* **p** – The properties from the `<tr>` tag

- **startTable**

```
public java.lang.StringBuffer startTable(java.lang.StringBuffer org,
    javax.swing.text.MutableAttributeSet attrSet)
```

- **Description**

Constructs a new table object and starts processing of the table by scanning the `<table>` passed to count columns.

- **Parameters**

- \* **p** – properties found on the `<table>` tag
- \* **ret** – the result buffer that will contain the output
- \* **table** – the input string that has the entire table definition in it.
- \* **off** – the offset into `<table>` where scanning should start

## 1.9 Class TestFilter

This class filters out classes beginning with "Test" when applied to the Doclet.

### 1.9.1 Declaration

```
public class TestFilter
    extends java.lang.Object
    implements ClassFilter
```

### 1.9.2 Constructor summary

[TestFilter\(\)](#)

### 1.9.3 Method summary

[includeClass\(ClassDoc\)](#) Returns false if class name starts with "Test".

### 1.9.4 Constructors

- **TestFilter**  
`public TestFilter()`

### 1.9.5 Methods

- **includeClass**  
`public boolean includeClass(com.sun.javadoc.ClassDoc cd)`
  - **Description**  
Returns false if class name starts with "Test".

## 1.10 Class TeXDoclet

This class provides a Java javadoc Doclet which generates a  $\text{\LaTeX} 2_{\epsilon}$  document out of the java classes that it is used on. This is convenient for creating printable documentation complete with cross reference information.

### Supported HTML tags

`<a>` including an additional attribut "doprinturl". Since the output of the doclet should be printable, the href attribut of tags is printed in parentheses following the link if attribut "doprinturl" is set. Sometimes this is undesirable, and omitting "doprinturl" attribut will prevent this.

`<dl>` with the associated `<dt><dd></dl>` tags

`<p>` but not `align=center...yet`

`<br>` but not `clear=xxx`

`<table>` including all the associated `<td><th><tr></td></th></tr>`

`<ol>` ordered lists

`<ul>` unordered lists

`<font>` font coloring

`<pre>` preformatted text

`<code>` fixed point fonts

`<i>` italized fonts

`<b>` bold fonts

`<sub>` subscript

`<sup>` superscript

`<center>` center

`<img>` image located in java sources (``)

1. example converted from JPG: (image file not found)
2. example converted from GIF: (image file not found)

`<img>` image located in the www: (see image at <http://upload.wikimedia.org/wikipedia/commons/9/92/LaTeX...>)

### Extra tags

#### `<TEX>`

A new tag is defined: `<TEX>`. This tag is useful for passing  $\text{T}_\text{E}\text{X}$  code directly to the  $\text{T}_\text{E}\text{X}$  compiler. The following code:

```
<TEX txt="\[ F\left( x \right) = \int\limits_{ - \infty }^x {\frac{1}{{\sqrt {2\pi } }}e^{{ - \frac{z^2}{2}} }dz} \]">
<BR><BR><B>This alternative text will appear if the javadoc/HTML is parsed
by any other doclet/browser</B><BR><BR></TEX>
```

will produce the following result:

$$F(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz$$

The "alternative" text is ignored by the `TeXDoclet`, but useful if you want to use both the `TeXDoclet` and a regular HTML based doclet.

#### `<PRE format="markdown">`

Instead of writing your java documentation in often hard to read HTML code you can make use of **Markdown** syntax. The HTML `<PRE>` tag is used therefore to prevent your IDE from automatically reordering your Markdown documentation text. Markdown parsing is based on the **Pegdown** implementation. The following code :

```
<PRE format="markdown">

some text some text some text some text some text some text some text

#### Lists

- item1
    1. item11
    2. item12
- item1

#### Text formatting
```

`_emphasis_` and `--strong--` and some 'code' :

```
code line 1
code line 2
```

some text some text some text some text some text some text some text

<PRE>

will produce the following :

some text some text some text some text some text some text some text

## Lists

- item1
  1. item11
  2. item12
- item1

## Text formatting

*emphasis* and **strong** and some code :

```
code line 1
code line 2
```

some text some text some text some text some text some text some text

### 1.10.1 See also

- [HTMLtoLaTeXBackEnd](#) (in 1.4, page 6)
- [TeXDoclet.start\(RootDoc\)](#) (in 1.10.8, page 23)

### 1.10.2 Declaration

```
public class TeXDoclet
extends com.sun.javadoc.Doclet
```

### 1.10.3 Field summary

```
BOLD
CHAPTER_LEVEL
ITALIC
os Writer for writing to output file
SECTION_LEVEL
SUBSECTION_LEVEL
TRUETYPE
```

#### 1.10.4 Constructor summary

**TeXDoclet()**

#### 1.10.5 Method summary

**finish()**

**init()**

**initSections()**

**main(String[])**

**optionLength(String)** Returns how many arguments would be consumed if **option** is a recognized option.

**start(RootDoc)** Called by the framework to format the entire document

**validOptions(String[], DocErrorReporter)** Checks the passed options and their arguments for validity.

#### 1.10.6 Fields

- public static final java.lang.String **SECTION\_LEVEL**
- public static final java.lang.String **CHAPTER\_LEVEL**
- public static final java.lang.String **SUBSECTION\_LEVEL**
- public static final java.lang.String **BOLD**
- public static final java.lang.String **TRUETYPE**
- public static final java.lang.String **ITALIC**
- public static java.io.PrintWriter **os**
  - Writer for writing to output file

#### 1.10.7 Constructors

- **TeXDoclet**  
public **TeXDoclet()**

#### 1.10.8 Methods

- **finish**  
public static void **finish()**
- **init**  
public static void **init()**
- **initSections**  
public static void **initSections()**
- **main**  
public static void **main**(java.lang.String[] args)

- **optionLength**

```
public static int optionLength(java.lang.String option)
```

- **Description**

Returns how many arguments would be consumed if `option` is a recognized option.

- **Parameters**

- \* `option` – the option to check

- **start**

```
public static boolean start(com.sun.javadoc.RootDoc root)
```

- **Description**

Called by the framework to format the entire document

- **Parameters**

- \* `root` – the root of the starting document

- **validOptions**

```
public static boolean validOptions(java.lang.String[] [] args,  
com.sun.javadoc.DocErrorReporter err)
```

- **Description**

Checks the passed options and their arguments for validity.

- **Parameters**

- \* `args` – the arguments to check

- \* `err` – the interface to use for reporting errors

### 1.10.9 Members inherited from class `Doclet`

`com.sun.javadoc.Doclet`

`languageVersion`, `optionLength`, `start`, `validOptions`