Dokumentasi dan Gaya Penulisan Program
Khusus Kode Sumber dengan Bahasa Pemrograman Java
Versi 2.1 (21 September 2010)

Ricky Suryadharma, Denvil Prasetya
Fakultas Ilmu Komputer Universitas Indonesia

Persentase nilai-nilai di bawah berdasarkan porsi/bobot nilai yang diberikan untuk dokumentasi dan gaya penulisan program (**PORSI_DOKUMENTASI** ≠ 0). Perhatikan bahwa tulisan berwarna merah, hijau, dan biru memiliki arti khusus. Secara umum (dan untuk selanjutnya), tulisan-tulisan yang memiliki arti khusus adalah sebagai berikut.

- Sebuah karakter # berarti sebuah karakter spasi yang ditulis demikian untuk kejelasan.
- *text* (tulisan berwarna hijau) berarti *text* dapat disesuaikan sendiri isinya dengan keterangan yang ada atau format sebelumnya.
- Tanda [*text*] berarti *text* bersifat pilihan (dapat ditulis atau dapat juga tidak ditulis).
- Tanda [$text_1$|$text_2$] berarti terdapat pilihan antara $text_1$ dan $text_2$ yang harus dipilih tepat satu.

kode pada tulisan di bawah disesuaikan dengan nama kode untuk kuliah, jenis pekerjaan (tutorial, tugas, atau kuis), dan nomor pekerjaan. Kriteria penilaian di bawah dapat diambil semua, atau disesuaikan dengan keputusan koordinator asisten dosen dan/atau dosen mata kuliah terkait. Jika persentase tertinggi untuk setiap kriteria yang dipilih dijumlahkan belum mencapai 100%, maka kriteria lain dapat ditambahkan atau cukup dengan memberikan bonus persentase yang tersisa hingga mencapai 100%. Jika persentase tertinggi untuk setiap kriteria yang dipilih dijumlahkan melebihi 100%, maka nilai apa adanya dapat diambil atau dilakukan pemotongan hingga menjadi 100% saja.

Istilah *javadoc-compliant* memiliki pengertian bahwa dokumentasi yang dibuat sesuai dengan aturan dokumentasi Java yang benar. Cara menguji bahwa dokumentasi-dokumentasi pada sebuah berkas kode sumber sudah *javadoc-compliant* atau belum adalah dengan melihat apakah ada peringatan dan/atau kesalahan saat menjalankan perintah javadoc pada *prompt*. Jika masih ada peringatan dan/atau kesalahan, berarti dokumentasi tertentu di dalam berkas kode sumber belum *javadoc-compliant*; sebaliknya berarti sudah *javadoc-compliant*. Contoh penggunaan perintah *javadoc* pada *prompt* adalah sebagai berikut (contoh dijalankan pada DOS-*prompt*).

```
javadoc -private -author -d doc MyClass.java
```

1. Berkas Tambahan

kode.2.1_1.1

Terdapat sebuah berkas bernama “README” (tanda kutip hanya untuk kejelasan) pada direktori terluar program atau kode sumber, nilainya 2%.

Tidak terdapat berkas dengan nama persis (*case-sensitive*) "README" pada direktori terluar program atau kode sumber, nilainya 0.

Pseudocode:

```
value[kode][2.1_1.1] := PORSI_DOKUMENTASI / 50;
change current_directory to the outmost program's directory;
if current_directory does not contain a file with the name "README" then
    value[kode][2.1_1.1] := 0;
end;
```

kode.2.1_1.2

Jika terdapat berkas "README" pada direktori terluar program atau kode sumber:

- jika isinya kosong atau hanya berupa kumpulan karakter spasi putih, nilainya 0.
- jika isinya tidak/kurang mencerminkan isi berkas yang dideskripsikan pada soal atau hal lain yang dimaksud, nilainya 4%.
- jika isinya mencerminkan isi berkas yang dideskripsikan pada soal atau hal lain yang dimaksud, nilainya 10%.

Jika tidak terdapat berkas "README" pada direktori terluar program atau kode sumber, nilainya 0.

Pseudocode:

```
► assume kode.2.1_1.1 has been executed in the same scope before
if value[kode][2.1_1.1] ≠ 0 then
    open f as "README";
    read s as text in f;
    close f;
    if trim(s) = "" then
        value[kode][2.1_1.2] := 0;
    else if s is not defined well as the description said then
        value[kode][2.1_1.2] := PORSI_DOKUMENTASI / 25;
    else
        value[kode][2.1_1.2] := PORSI_DOKUMENTASI / 10;
    end;
else
    value[kode][2.1_1.2] := 0;
end;
```

2. Komentar Awal Berkas Kode Sumber

kode.2.1_2.1

Untuk setiap berkas kode sumber terdapat komentar awal dengan semua informasi yang diperlukan dan format sama persis (urutan nama-nama interface atau nama-nama kelas sesuai dengan urutan kemunculannya dalam berkas dari awal sampai akhir; nama-nama

interface (jika ada) harus mendahului nama-nama kelas; nomor versi harus terurut dari kecil ke besar; dd, mm, dan yyyy merupakan tanggal 2-digit, bulan 2-digit, dan tahun 4-digit; Name(s) merupakan nama-nama pengembang (*developer*) berkas kode sumber yang sesuai dengan nama di SIAK NG dan masing-masing dipisahkan oleh "**,#**" (tanda kutip hanya untuk kejelasan))

```
/*
#*#interface1,#interface2,#...,#class1,#class2,#...
#*
#*#Version:
#*#-#mayor.minor[.build[.revision]]#(dd-mm-yyyy):#explanation of
#*#######this version or the change(s).
#*#-#...
#*
#*#Copyright#yyyy#Name(s).
[#*#license]
#*/
```

yang terletak paling awal, nilainya 6%.
Untuk setiap berkas kode sumber terdapat komentar awal dengan semua informasi pada format di atas tertulis, tetapi tidak tertulis sama persis dengan format atau tidak terletak paling awal, nilainya 4%.
Untuk setiap berkas kode sumber terdapat komentar awal dengan ada (tetapi tidak semua) informasi pada format di atas tertulis, nilainya 2%.
Untuk setiap berkas kode sumber tidak terdapat komentar awal atau terdapat komentar awal dengan tidak ada informasi pada format di atas tertulis, nilainya 0.

Contoh komentar awal berkas kode sumber:

```
/*
 * MyInterface, MyClass1, MyClass2
 *
 * Version:
 * - 1.0 (10-02-2010): develop MyInterface, MyClass1, and MyClass2.
 * - 1.0.1.1 (12-02-2010): add MyMethod method to MyClass1.
 *
 * Copyright 2010 Ricky Suryadharma, Denvil Prasetya.
 */
```

3. package dan import

kode.2.1_3.1
Untuk setiap berkas kode sumber, letak pengenal package dan penggunaan import harus sama persis dengan

```
[package#name-of-package;
]
[import#name1;
[import#name2;
[.
.
```

```
.
import#name_n;]]]
```

nilainya 2%.
Terdapat satu berkas kode sumber saja yang letak pengenal package dan penggunaan import tidak sama persis, nilainya 0.

kode.2.1_3.2
Untuk setiap penggunaan import pada setiap berkas kode sumber, semua kelas dan/atau interface digunakan pada berkas kode sumber itu, nilainya 2%.
Terdapat satu kelas atau interface saja pada penggunaan import yang tidak digunakan dalam berkas yang sama, nilainya 0.

Contoh penggunaan package dan import:

```
package pack1.Example;

import java.io.BufferedReader;
import java.io.File;
```

Contoh lain penggunaan import:

```
import java.io.*;
// all classes and interfaces from java.io must be used in this file
```

4. **class**

kode.2.1_4.1
Untuk setiap kelas, dokumentasi, penjelasan, pengenal (jika ada), konstruktor (jika ada), dan metode (jika ada) mempunyai format sama persis ($Nama_i$ dan NPM_i sesuai dengan yang terdaftar di SIAK NG)

```
/**
##*#The documentation of class.
##*
##*#@author#Name1#(NPM1)
[##*#@author#Name2#(NPM2)
[.
.
.
##*#@author#Namen#(NPMn)]]
[##*
##*#Futhermore documentation(s)...]
##*/
class-specifier#class#Class-name#{
####/*#The explanation of class implementation.
[#####*#Futhermore explanation(s)...]
#####*/
[
####/**#The explanation of declaration. */
```

```
####public#static#type#a1;
[...
####/**#The explanation of declaration. */
####public#static#type#an;]]
[
####/**#The explanation of declaration. */
####protected#static#type#b1;
[...
####/**#The explanation of declaration. */
####protected#static#type#bn;]]
[
####/**#The explanation of declaration. */
####static#type#c1;
[...
####/**#The explanation of declaration. */
####static#type#cn;]]
[
####/**#The explanation of declaration. */
####private#static#type#d1;
[...
####/**#The explanation of declaration. */
####private#static#type#dn;]]
[
####/**#The explanation of declaration. */
####public#type#e1;
[...
####/**#The explanation of declaration. */
####public#type#en;]]
[
####/**#The explanation of declaration. */
####protected#type#f1;
[...
####/**#The explanation of declaration. */
####protected#type#fn;]]
[
####/**#The explanation of declaration. */
####type#g1;
[...
####/**#The explanation of declaration. */
####type#gn;]]
[
####/**#The explanation of declaration. */
####private#type#h1;
[...
####/**#The explanation of declaration. */
####private#type#hn;]]
[
####/**#The explanation of declaration. */
####public#static#final#type#A1;
[...
####/**#The explanation of declaration. */
```

```
####public#static#final#type#An;]]
[
####/**#The explanation of declaration. */
####protected#static#final#type#B1;
[...
####/**#The explanation of declaration. */
####protected#static#final#type#Bn;]]
[
####/**#The explanation of declaration. */
####static#final#type#C1;
[...
####/**#The explanation of declaration. */
####static#final#type#Cn;]]
[
####/**#The explanation of declaration. */
####private#static#final#type#D1;
[...
####/**#The explanation of declaration. */
####private#static#final#type#Dn;]]
[
####/**#The explanation of declaration. */
####public#final#type#E1;
[...
####/**#The explanation of declaration. */
####public#final#type#En;]]
[
####/**#The explanation of declaration. */
####protected#final#type#F1;
[...
####/**#The explanation of declaration. */
####protected#final#type#Fn;]]
[
####/**#The explanation of declaration. */
####final#type#G1;
[...
####/**#The explanation of declaration. */
####final#type#Gn;]]
[
####/**#The explanation of declaration. */
####private#final#type#H1;
[...
####/**#The explanation of declaration. */
####private#final#type#Hn;]]
[
####//Constructor(s)

####/**#The documentation of constructor.
[######*#@param#par1#explanation of par1
[.
.
.
```

```
######*#@param#par_n#explanation of par_n]]
######*/
####public#constructor-name([type#par_1,#[...,#type#par_n]])#{

####}]
[
####//Method(s)

####/**#The documentation of method.
[######*#@param#par_1#explanation of par_1
[.
.
.
######*#@param#par_n#explanation of par_n]]
[######*#@return#explanation]
######*/
####method-specifier#return-type#method-name([par(s)])#{

####}
[
####/**#The documentation of abstract method.
[######*#@param#par_1#explanation of par_1
[.
.
.
######*#@param#par_n#explanation of par_n]]
[######*#@return#explanation]
######*/
####method-specifier#abstract#return-type#method-name([par(s)])#{}]]
}
```

nilainya 4%.
Terdapat satu kelas saja yang formatnya tidak sama persis, nilainya 0.

kode.2.1_4.2
Untuk setiap kelas, namanya tepat/deskriptif, nilainya 2%.
Terdapat satu kelas saja yang namanya tidak/kurang tepat/deskriptif, nilainya 0.
Pengecualian diberikan untuk nama kelas yang memang diminta dalam deskripsi soal.

Kode.2.1_4.3
Untuk setiap kelas, aturan penamaan yang digunakan adalah huruf pertama untuk setiap kata merupakan huruf kapital dan sisanya huruf kecil, dan karakter yang digunakan adalah huruf latin dan angka, nilainya 2%.
Terdapat satu kelas saja yang namanya tidak memenuhi aturan tersebut, nilainya 0.

kode.2.1_4.4
Untuk setiap kelas terdapat dokumentasi dan penjelasan implementasi di awal kelas yang mencerminkan isinya, nilainya 4%.

Untuk setiap kelas terdapat dokumentasi dan penjelasan implementasi di awal kelas, tetapi ada satu kelas saja yang dokumentasi dan/atau penjelasannya tidak/kurang mencerminkan isinya, nilainya 2%.
Terdapat satu kelas saja yang tidak memiliki dokumentasi atau penjelasan implementasi di awal kelas, nilainya 0.

kode.2.1_4.5
Jika untuk setiap kelas terdapat dokumentasi di awal, maka:

- untuk setiap kelas terdapat dokumentasi di awal kelas yang *javadoc-compliant*, nilainya 2%.
- terdapat satu kelas saja dengan dokumentasi di awal kelas tidak *javadoc-compliant*, nilainya 0.

Terdapat satu kelas saja yang tidak memiliki dokumentasi di awal kelas, nilainya 0.

kode.2.1_4.6
Kelas yang berisi metode `main` terletak mendahului kelas-kelas lainnya (jika ada) pada berkas yang sama, nilainya 2%;
selain itu, nilainya 0.

5. Konstruktor, Metode dan Parameter

kode.2.1_5.1
Untuk setiap konstruktor dan metode, namanya deskriptif (termasuk menggunakan singkatan yang lazim – singkatan baku di dalam Kamus Besar Bahasa Indonesia, Oxford Dictionary, dan/atau kamus/ensiklopedia komputer yang diakui dan dapat dipercaya), nilainya 2%.
Terdapat satu konstruktor dan/atau satu metode saja yang namanya tidak/kurang deskriptif, nilainya 0.

kode.2.1_5.2
Untuk setiap metode dan parameter, aturan penamaan yang digunakan adalah

- huruf-huruf yang merupakan singkatan merupakan huruf kapital
- huruf-huruf yang bukan merupakan singkatan:
  - jika terletak pada kata pertama, semuanya huruf kecil
  - jika terletak pada kata berikutnya, huruf pertama setiap kata merupakan huruf kapital dan sisanya huruf kecil
- karakter yang digunakan adalah huruf latin dan angka

nilainya 2%.

Terdapat satu metode dan/atau satu parameter saja yang namanya tidak memenuhi aturan tersebut, nilainya 0.

`kode`.2.1_5.3
Untuk setiap konstruktor dan metode terdapat dokumentasi di awal yang mencerminkan isinya, nilainya 4%.
Untuk setiap konstruktor dan metode terdapat dokumentasi di awal, tetapi ada satu konstruktor dan/atau satu metode saja yang dokumentasinya di awal tidak/kurang mencerminkan isinya, nilainya 2%.
Terdapat satu konstruktor dan/atau satu metode saja yang tidak memiliki dokumentasi di awal, nilainya 0.

`kode`.2.1_5.4
Untuk setiap konstruktor dan metode dengan parameter terdapat penjelasan semua parameternya yang mencerminkan, nilainya 4%.
Untuk setiap konstruktor dan metode dengan parameter terdapat penjelasan semua parameternya, tetapi satu konstruktor dengan parameter dan/atau satu metode dengan parameter saja yang penjelasan sebuah parameternya saja tidak/kurang mencerminkan, nilainya 2%.
Terdapat satu kelas dengan parameter dan/atau satu metode dengan parameter saja yang tidak memiliki penjelasan sebuah parameternya saja, nilainya 0.

`kode`.2.1_5.5
Untuk setiap metode dengan nilai balik (bukan *void method*) terdapat penjelasan dari nilai balik metode yang mencerminkan, nilainya 4%.
Untuk setiap metode dengan nilai balik (bukan *void method*) terdapat penjelasan dari nilai balik, tetapi ada satu metode dengan nilai balik saja yang penjelesan nilai baliknya tidak/kurang mencerminkan, nilainya 2%.
Terdapat satu metode dengan nilai balik saja yang tidak memiliki penjelasan nilai balik, nilainya 0.

`kode`.2.1_5.6
Jika untuk setiap konstruktor dan metode memiliki dokumentasi di awal, maka:

- untuk setiap konstruktor dan metode terdapat dokumentasi di awal yang *javadoc-compliant*, nilainya 2%.
- terdapat satu konstruktor dan/atau satu metode saja dengan dokumentasi di awal tidak *javadoc-compliant*, nilainya 0.

Terdapat satu konstruktor dan/atau satu metode saja yang tidak memiliki dokumentasi di awal, nilainya 0.

`kode`.2.1_5.7
Metode `main` di dalam kelas yang memuatnya terletak mendahului metode-metode lain (jika ada) pada kelas yang sama, nilainya 2%;
selain itu, nilainya 0.

kode.2.1_5.8
Untuk setiap metode `toString` tanpa parameter dan tipe nilai baliknya `String`, nilainya 2%.
Terdapat satu metode `toString` saja yang memiliki parameter dan/atau tipe nilai baliknya bukan `String`, nilainya 0.

kode.2.1_5.9
Untuk setiap konstruktor dan metode dengan parameter (kecuali metode `main`), semua parameternya digunakan, nilainya 2%.
Terdapat satu konstruktor dengan parameter dan/atau satu metode dengan parameter (kecuali metode `main`) saja yang satu parameternya saja tidak digunakan, nilainya 0.

kode.2.1_5.10
Tidak terdapat `throw` dan/atau `throws` pada metode `main`, nilainya 2%; selain itu nilainya 0.

6. Peubah dan Konstanta

Catatan:
Konstanta wajib dideklarasikan dengan kata tercadang `final`.
Jika tidak, akan dianggap sebagai peubah (*variable*).

kode.2.1_6.1
Untuk setiap peubah (kecuali untuk perulangan) dan konstanta, namanya deskriptif (termasuk menggunakan singkatan yang lazim), nilainya 2%.
Terdapat satu peubah (kecuali untuk perulangan) dan/atau satu konstanta saja yang namanya tidak/kurang deskriptif, nilainya 0.

kode.2.1_6.2
Untuk setiap peubah dan konstanta terdapat penjelasan yang mencerminkan pada saat pendeklarasian, nilainya 4%.
Untuk setiap peubah dan konstanta terdapat penjelasan pada saat pendeklarasian, tetapi ada satu peubah dan/atau satu konstanta saja yang penjelasannya tidak/kurang mencerminkan pada saat pendeklarasian, nilainya 2%.
Terdapat satu peubah dan/atau satu konstanta saja yang tidak memiliki penjelasan pada saat pendeklarasian, nilainya 0.

kode.2.1_6.3
Untuk setiap peubah, aturan penamaan yang digunakan adalah
- huruf-huruf yang merupakan singkatan merupakan huruf kapital

- huruf-huruf yang bukan merupakan singkatan:
  - jika terletak pada kata pertama, semuanya huruf kecil
  - jika terletak pada kata berikutnya, huruf pertama setiap kata merupakan huruf kapital dan sisanya huruf kecil
- karakter yang digunakan adalah huruf latin dan angka

nilainya 2%.

Terdapat satu peubah saja yang namanya tidak memenuhi aturan tersebut, nilainya 0.

kode.2.1_6.4
Setiap peubah digunakan, nilainya 2%.
Terdapat satu peubah saja yang tidak digunakan, nilainya 0.

kode.2.1_6.5
Untuk setiap konstanta, aturan penamaan yang digunakan adalah
- semua huruf merupakan huruf kapital
- karakter pemisah antar kata yang bukan singkatan, antar singkatan, atau antar keduanya adalah karakter garis bawah (*underscore*)
- karakter yang digunakan adalah huruf latin, angka, dan karakter garis bawah

nilainya 2%.

Terdapat satu konstanta saja yang namanya tidak memenuhi aturan tersebut, nilainya 0.

Contoh yang tidak boleh:
```
MaximumValue
```

Contoh yang baik:
```
MAXIMUM_VALUE
```

kode.2.1_6.6
Untuk setiap nilai bilangan konstanta yang digunakan secara langsung pada pemberian nilai ke peubah, pengecekan kondisi, perulangan, dan/atau proses perhitungan, selain -1, 0, dan 1 (termasuk nilai-nilai yang sama dengan tipe bilangan berbeda) tidak digunakan, nilainya 2%.
Terdapat satu saja penggunaan nilai bilangan konstanta selain -1, 0, dan 1 pada pemberian nilai ke peubah, pengecekan kondisi, perulangan, dan/atau proses perhitungan, nilainya 0.

Contoh yang tidak boleh:
```
if(numberOfStudent > 90)
```

Contoh yang baik:
```
final int MAXIMUM_NUMBER_OF_STUDENT = 90;
.
```

```
.
if(numberOfStudent > MAXIMUM_NUMBER_OF_STUDENT)
```

7. Indentasi

kode.2.1_7.1
Semua pernyataan pada blok paling luar terletak paling kiri, nilainya 2%.
Terdapat satu pernyataan saja pada blok paling luar yang tidak terletak paling kiri, nilainya 0.

kode.2.1_7.2
Semua pernyataan pada blok satu tingkat lebih dalam harus berjarak tepat 4 (empat) karakter spasi ke kanan dari blok lebih luar, nilainya 2%.
Terdapat satu pernyataan saja pada blok satu tingkat lebih dalam yang tidak berjarak tepat 4 (empat) karakter spasi ke kanan dari blok lebih luar, nilainya 0.
Pengecualian diberikan untuk baris yang berisi perpanjangan pernyataan, yaitu tepat 8 (delapan) karakter spasi dari blok sebelumnya dengan aturan nilai yang sama.

8. Blok

kode.2.1_8.1
Pengkondisian menggunakan *template* (walau hanya berisi sebuah pernyataan):

```
if#(condition-A)#{
####statement-A1;[...
####statement-An;]
}[#else#if#(condition-B)#{
####statement-B1;[...
####statement-Bn;]
}][...#else#{
####statement-X1;[...
####statement-Xn;]
}]
```

atau

```
switch#(condition)#{
case#cases-A:
####statement-A1;[...
####statement-An;]
[####[break;|/*#falls#through#*/]]
[
case#cases-B:
####statement-B1;[...
####statement-Bn;]
[####[break;|/*#falls#through#*/]]]
[...
```

```
default:
####statement-X1;[...
####statement-Xn;]
[####break;]]
}
```
nilainya 2%; selain itu 0.

kode.2.1_8.2
Terdapat maksimal dua buah pengkondisian tersarang (*nested condition*) di dalam sebuah pengkondisian, nilainya 2%; selain itu 0.
Contoh yang masih boleh (terdapat dua buah pengkondisian tersarang di dalam sebuah pengkondisian):

```
if (...) {
    if (...) {
        if (...) {
        }
    }
}
```

kode.2.1_8.3
Perulangan menggunakan *template* (walau hanya berisi 1 pernyataan):

```
for#([...];#[...];#[...])#{
####statement-1;[...
####statement-n;]
}
```
atau
```
for#([...];#[...];#[...]);
```
atau
```
for#(class-type#object-name#:#object-collection)#{
####statement-1;[...
####statement-n;]
}
```
atau
```
while#(condition)#{
####statement-1;[...
####statement-n;]
}
```
atau
```
while#(condition);
```
atau
```
do#{
####statement-1;[...
####statement-n;]
}#while#(condition);
```
nilainya 2%; selain itu 0.

kode.2.1_8.4
Terdapat maksimal tiga buah perulangan tersarang (*nested loop*) di dalam sebuah perulangan, nilainya 2%; selain itu 0.

kode.2.1_8.5
Blok try...catch menggunakan *template*:

```
try#{
####statement-1;[...
####statement-n;]
[}#catch#(Exception-class#object-name-1)#{
####statement-1;[...
####statement-n;]
[}#catch#(Exception-class#object-name-n)#{
####statement-1;[...
####statement-n;]]]
[}#finally#{
####statement-1;[...
####statement-n;]]
}
```

nilainya 2%; selain itu 0.

9. Struktur data dan algoritma

kode.2.1_9.1
Untuk setiap struktur data dan algoritma yang digunakan diberikan nama dalam komentar, nilainya 1%.
Terdapat satu saja struktur data atau algoritma yang tidak diberikan nama, nilainya 0.

kode.2.1_9.2
Untuk setiap struktur data dan algoritma yang digunakan diberikan deskripsi dalam komentar, nilainya 2%.
Terdapat satu saja struktur data atau algoritma yang tidak diberikan deskripsi, nilainya 0.

kode.2.1_9.3
Untuk setiap struktur data dan algoritma yang digunakan diberikan kompleksitas waktu dan memori, nilainya 2%.
Terdapat satu saja struktur data atau algoritma yang tidak diberikan kompleksitas waktu dan/atau memori, nilainya 0.

Berikut adalah *template* komentar algoritma yang dapat digunakan.

```
/* Algorithm name: ...
 * Description: ...
 * Time complexity: ...
 * Memory complexity: ...
 */
```

10. Lain-lain

kode.2.1_10.1
Semua dokumentasi dan komentar ditulis dalam bahasa Indonesia atau bahasa Inggris yang baik dan benar, serta konsisten, nilainya 2%; selain itu 0.
Berikut adalah beberapa contoh penggunaan kata dalam kedua bahasa.

Bahasa Indonesia	Bahasa Inggris
masukan	input
keluaran	output
pengguna	user

kode.2.1_10.2
Di sebelah kiri dan kanan operator *binary* (kecuali operator titik) dan *ternary* (operator aritmatika, operator logika, operator pemberian nilai, dsb) ada tepat sebuah karakter spasi, nilainya 2%; selain itu 0.

kode.2.1_10.3
Antara kelas yang satu (tepatnya baris di mana kurung kurawal penutup kelas itu berada) dengan kelas berikutnya (tepatnya baris pertama dokumentasi kelas), atau antara baris pertama dokumentasi kelas yang letaknya teratas dalam berkas dengan baris terakhir import, atau antara interface yang satu (tepatnya baris di mana kurung kurawal penutup interface itu berada) dengan interface berikutnya (tepatnya baris pertama dokumentasi interface), atau antara interface terakhir pada berkas (tepatnya baris di mana kurung kurawal penutup interface itu berada) dengan kelas pertama pada berkas (tepatnya baris pertama dokumentasi kelas) terdapat tepat dua baris kosong, nilainya 2%; selain itu 0.

Contoh 1:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;


/** Class Example ....
  * ...
  */
```

```
public class Example {

}
```

Contoh 2:

```
/** Class Example1 ....
  * ...
  */
class Example1 {

}


/** Class Example2 ....
  * ...
  */
class Example2 {

}
```

kode.2.1_10.4

Antara konstruktor yang satu (tepatnya baris di mana kurung kurawal penutup konstruktor itu berada) dengan konstruktor berikutnya pada sebuah kelas yang sama (tepatnya baris pertama dokumentasi konstruktor), atau antara metode yang satu (tepatnya baris di mana kurung kurawal penutup metode itu berada) dengan metode berikutnya pada sebuah kelas yang sama (tepatnya baris pertama dokumentasi metode) terdapat tepat satu baris kosong, nilainya 2%; selain itu 0.

kode.2.1_10.5

Untuk setiap baris pada setiap berkas terdapat banyak karakter kurang dari atau sama dengan 80, nilainya 2%.

Terdapat satu baris saja pada berkas mana pun yang memiliki banyak karakter lebih dari 80, nilainya 0.

Referensi

Sun Microsystems, Inc. 20 April 1999. *Code Conventions for the Java Programming Language*.