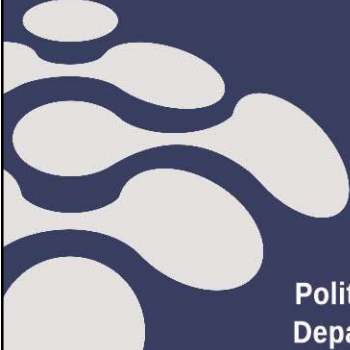


Pemrograman Berbasis Obyek

Pengenalan Konsep OOP

Oleh Politeknik Elektronika Negeri Surabaya
2017



Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika dan Komputer

1

Konten

- Pengantar OOP
 - Paradigma Pemrograman
 - Perbedaan OOP dan terstruktur
 - Class VS Object
- Starting OOP Programming
 - Deklarasi class, attribute, method
 - Pembuatan Object
 - Pengaksesan Anggota Object
 - Constructor
 - Reference type
- Dasar Class Diagram
- Java Naming Convention



Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika & Komputer

2

Pengantar OOP



Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika & Komputer

3

Politeknik Elektronika Negeri Surabaya

Pengantar OOP

WHAT IS
OBJECT ORIENTED
PROGRAMMING ?



Departemen Teknik Informatika & Komputer

4

Paradigma Pemrograman

- Suatu cara berpikir dalam membuat program komputer yang direpresentasikan dalam sejumlah konsep dan teknik pemrograman
- Terdapat banyak paradigma pemrograman
- Suatu bahasa pemrograman bisa mendukung lebih dari satu paradigma pemrograman



Contoh Bahasa Pemrograman

- Procedural
 - Pascal, C, COBOL, Fortran, ALGOL, Basic, PHP, dll.
- Object Oriented
 - Java, C#, C++, Objective C, PHP, Visual Basic.Net, Object Pascal, dll.
- Pada C++, bisa procedural, bisa juga Object Oriented !



OOP

- Paradigma pemrograman yg menggunakan pendekatan berorientasi pada obyek
- Jadi permasalahan yang ada dipandang sebagai obyek
- Obyek => suatu bentuk nyata yang dapat dibayangkan, memiliki segala sesuatu yang memang melekat padanya, dan dapat melakukan tindakan tertentu

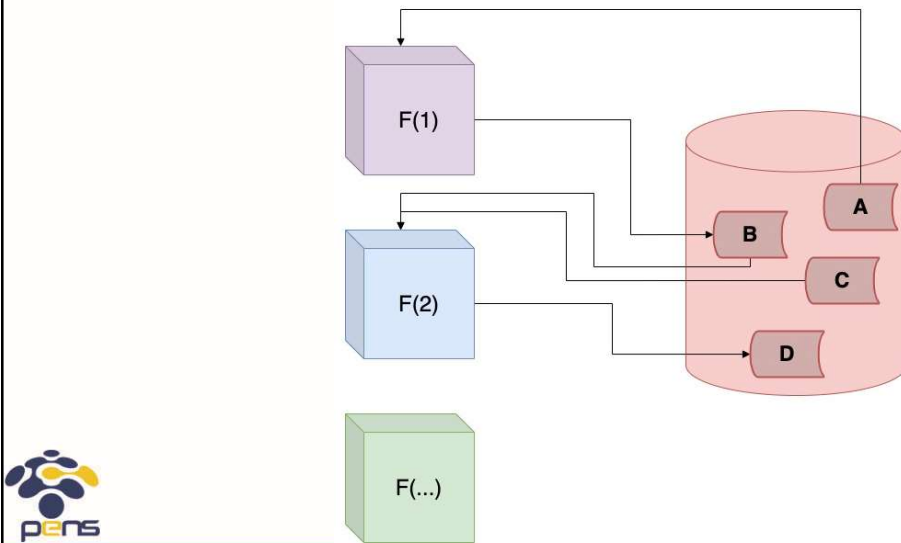


Pemrograman Berorientasi Obyek

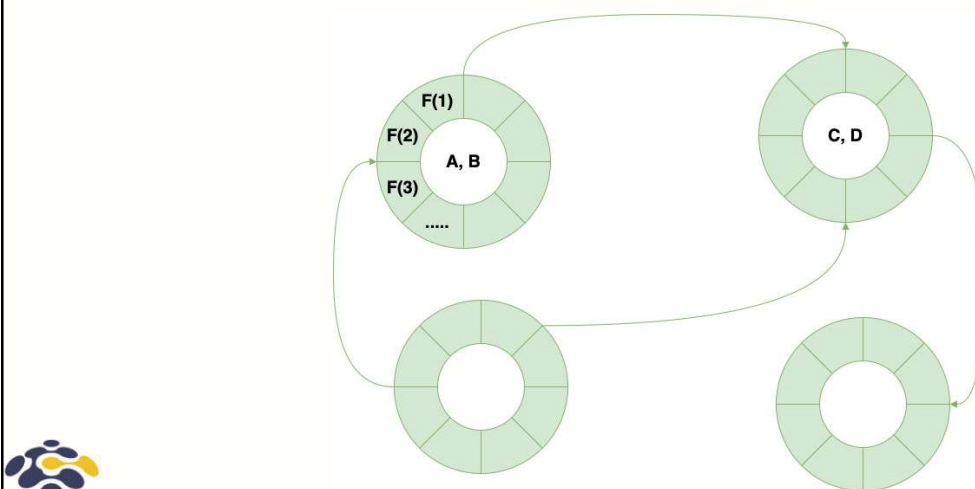
- Fungsi dan data bukan menjadi dua hal yang terpisah.
- Fungsi dan data menjadi satu kesatuan yang disebut sebagai obyek aktif.
- Cara pandang → program adalah serangkaian obyek yang bekerjasama untuk menyelesaikan suatu problem.



Pemrograman Prosedural



Pemrograman berorientasi obyek



Procedural / Struktural vs OOP

- Procedural

- Menyusun langkah-langkah untuk menyelesaikan suatu masalah
- Misal: menghitung luas bangun segi empat "X"
- Langkahnya:
 - Input panjang dan lebar segi empat "X"
 - Cari luas segi empat "X" dengan cara kalikan panjang dan lebar
 - Tampilkan luas segi empat "X"



Procedural / Struktural vs OOP

- Object Oriented

- Menyusun / merancang obyek yang akan dioperasikan
- Segi empat "X" memiliki panjang dan lebar
- Segi empat "X" bisa dihitung luasnya dengan panjang * lebar
- Langkahnya:
 - Buat Obyek segi empat "X", isikan data panjang dan lebar
 - Meminta obyek segi empat "X" menghitung luasnya



Obyek segi empat "X" dibuat dari class Segi Empat

- Class: Segi Empat
- Atribut: sifat atau identitas yg melekat
 - Panjang
 - Lebar
- Behaviour: tingkah laku / kegiatan
 - Hitung Luas
 - Hitung Keliling



13

Contoh lain



14

Procedural / Struktural vs OOP

- Procedural
 - Lebih cepat untuk memecahkan masalah-masalah berskala kecil
 - Mudah membuatnya
- Object Oriented
 - Scalable, cocok untuk masalah-masalah berskala besar
 - Pengembangannya mudah



Apa yang dimaksud Class dan Obyek?

- Class: merupakan template untuk membuat obyek.
- Class: merupakan prototipe / blue prints yang mendefinisikan variable-variabel dan method-method secara umum.
- Obyek merupakan hasil instansiasi dari suatu kelas.
- Proses pembentukan obyek dari suatu class disebut dengan **instantiation**.
- Obyek disebut juga **instances**.
- Setiap obyek memiliki state sebagai status (atribut).
- Setiap obyek memiliki tingkah laku (method).



Contoh Class dan Obyek

- Perguruan Tinggi
 - PENS, Harvard, MIT
- Presiden
 - Joko Widodo, Kim Jong Un, Donald Trump
- Girlband
 - Blackpink, Red Velvet, Twice



Karakteristik Obyek

- Penggambaran pemrograman berorientasi obyek = penggambaran pada dunia nyata.
- Pada pemrograman berorientasi obyek:
 - State disimpan pada → variabel
 - Tingkah laku disimpan pada → method



Atribut

- Definisi atribut : adalah **data** yang membedakan antara obyek satu dengan yang lain.
- Contoh: Class **Negara**
 - Object 1 : **Indonesia**
 - Attribute : Bahasa (Indonesia), Ibu Kota (Jakarta)
 - Object 2 : **Jepang**
 - Attribute : Bahasa (Jepang), Ibu Kota (Tokyo)
 - Object 3: **Thailand**
 - Attribute : Bahasa (Thai), Ibu Kota (Bangkok)
- Di dalam class, atribut disebut juga dengan **variabel**.



Tingkah Laku

- Tingkah laku adalah hal – hal yang bisa dilakukan oleh obyek dari suatu class.
- Tingkah laku dapat digunakan untuk mengubah nilai atribut suatu obyek, menerima informasi dari obyek lain, dan mengirim informasi ke obyek lain untuk melakukan suatu task.
- Dalam class, tingkah laku disebut juga sebagai **method**.
- Method: adalah serangkaian statements dalam suatu class yang handle suatu task tertentu.
- Cara obyek **berkomunikasi** dengan obyek lain adalah dengan menggunakan **method**.



Class VS Object

- **Bus** dapat dinaiki banyak **penumpang**.
- **Budi** adalah salah satu **penumpang** dari **Bus Merah**.
- **Budi** memancing di **kolam tetangga**.
- **Bumi** adalah salah satu **planet** dalam **Galaxy Bima Sakti**.
- **Matahari** adalah **bintang** yang terdekat dengan **bumi**.
- Jawaban:
 - Bus (Class), penumpang (Class)
 - Budi (Object), penumpang (class), Bus Merah (Object)
 - Budi (Object), Kolam Tetangga (object)
 - Bumi (Object), planet (class), galaxy bima sakti (object)
 - Matahari (object), bintang (class), bumi (object)



Class VS Object

- Tentukan class dari gambar-gambar berikut!
- Class:
 - Truk
 - Bus
 - Sedan
 - Hatchback
 - MPV
 - SUV



Start OOP Programming



Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika & Komputer

23

Politeknik Elektronika Negeri Surabaya

Deklarasi Class

- Class merupakan salah satu Source File dalam Java
- Source File Java harus diakhiri dengan ekstensi `.java`.
- Tiga top-level elemen dalam file java:
 - Package Declaration
 - Import Statements
 - Class Definitions

Tiga element tersebut
dituliskan pada
bagian paling awal
sebuah File Java



Departemen Teknik Informatika & Komputer

24

Deklarasi Class

```
<modifier> class <classname> {  
    [deklarasi_atribut]  
    [deklarasi_konstruktor]  
    [deklarasi_method]  
}
```



Contoh Deklarasi Class

```
public class Siswa {  
}
```

modifier

nama class



Deklarasi Atribut

```
<modifier> <tipe> <nama_atribut>;
```



Contoh Deklarasi Atribut

```
public class Siswa {  
    public int nrp;  
    public String nama;  
}
```



Deklarasi method

```
<modifier> <return_type> <nama_method> ([daftar_argumen]){  
    [<statement>]  
}
```



Contoh Deklarasi Method

```
public class Siswa {  
    public int nrp;  
    public String nama;  
    public void Info() {  
        System.out.println("Saya siswa PENS");  
    }  
}
```



Pembuatan Object

- Object adalah instance dari sebuah Class
- Untuk membuat object kita harus menginstansiasi sebuah class

```
public class Siswa{
    public int nrp;
    public String nama;
    public void info(){}
}
```

```
public class IsiData{
    Public static void main(String args[]){
        Siswa budi = New Siswa();
    }
}
```



Pengaksesan Anggota Obyek

- Gunakan notasi titik (.) sebagai berikut:
 <object>.<member>
- Member bisa berupa:
 - atribut
 - method



Pengaksesan Anggota Object

```
public class IsiData {  
    public static void main(String args[ ]) {  
        Siswa budi= new Siswa();  
        budi.nrp = 5;  
        budi.nama = "Andi";  
        budi.info();  
    }  
}
```

budi adalah sebuah Object
budi adalah instance dari Class Siswa



Constructor (Konstruktor)

- Constructor (Konstruktor) adalah kode yang pertama kali dijalankan pada saat pembuatan suatu obyek.
- Ciri-ciri konstruktor:
 - Mempunyai nama yang sama dengan nama kelas
 - Tidak mempunyai return type
 - Memiliki argumen sebanyak 0..n



Contoh Konstruktur

```
public class Siswa{  
    private int nrp;  
    private String nama;  
  
    public Siswa(int n, String m){  
        nrp = n;  
        nama = m;  
    }  
}
```



Default Constructor

- Jika tidak menuliskan kode konstruktur, maka secara otomatis kompiler akan menambahkan default constructor
- Default constructor → no argument and no body.
- The default constructor has the same access modifier as the class itself, either: public, protected, private or package (no modifier)



Contoh Default Constructor

```

modifiers ClassName() {
    super();
}
-----
public class Siswa{
    public Siswa(){
        super(); // menjalankan konstruktor parent
    }
}

```



Ingat!!!

- Sekali saja konstruktor dibuat / ditulis secara eksplisit, maka default konstruktor akan hilang

- Contoh:

```

public class Siswa{
    String nama;
    public Siswa(int n){
        this.mana = n;
    }
}

```



Reference Type

- Tipe selain tipe primitif dinamakan reference type
- Reference type adalah tipe berbentuk suatu class.
- Pembuatan suatu reference type untuk mengalokasikan memori dilakukan dengan menggunakan kata kunci `new XXX()`. Dimana `XXX` adalah **konstruktor** dari reference type



Kejadian bila `new xxx()` dipanggil

- Alokasi memori: ruang untuk obyek baru dibuat di memori dan variabel-variabel diset ke masing-masing nilai default-nya (`false`, `0`, `null`, dll)
- Inisialisasi nilai atribut yang diberikan secara eksplisit
- Menjalankan konstruktor
- Assignment antara atribut-atribut dengan obyek



Contoh

```
public class MyDate {
    private int day=1;
    private int month=1;
    private int year=2000;
    public MyDate(int day, int month, int year) {...}
}
```

```
public class TestMyDate {
    public static void main(String args[]) {
        MyDate today=new MyDate(10,10,2005);
    }
}
```



Alokasi memori

```
MyDate today = new MyDate(10, 10, 2005);
```

Sudah dibuat object
today, namun belum
merefence alamat
memorio manapun

today

????



Inisialisasi default value

```
MyDate today = new MyDate(10, 10, 2005);
```

today

???

day

0

month

0

year

0



Menjalankan konstruktor

```
MyDate today = new MyDate(10, 10, 2005);
```

today

???

day

10

month

10

year

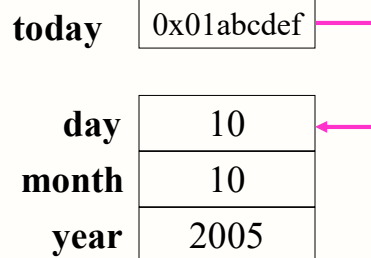
2005



Membuat referensi

```
MyDate today = new MyDate(10, 10, 2005);
```

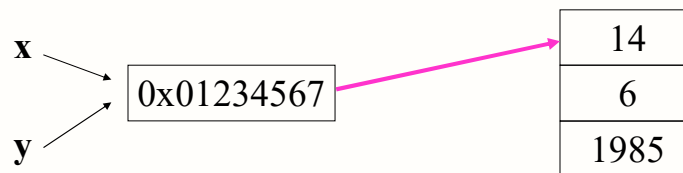
Today merereference
object yang baru
dibuat dalam memori



Men-assign reference variable

```
MyDate x = new MyDate(14, 6, 1985);  
MyDate y = x;
```

Alamat yang direferensi oleh x di assign pada y. Sehingga y dan x
adalah dua object yang merereference alamat memori yang sama



Class Fundamental : main method

- The *main()* Method

```
public static void main(String[] args)
```

- **Public** : method `main()` dapat diakses oleh apa saja, termasuk java technology interpreter.
- **Static** : keyword ini berfungsi untuk memberi tahu kompiler bahwa method `main` bisa langsung digunakan dalam context class yang bersangkutan. Untuk mengeksekusi/menjalankan method yang bertipe `static`, tidak diperlukan instance nya.
- **Void** : menunjukkan bahwa method `main()` tidak mengembalikan nilai
- **Main** : merupakan nama method utama dari program java
- **String [] args** : Menyatakan bahwa method `main()` menerima single parameter yaitu `args` yang bertipe array. Digunakan pada saat memasukkan parameter pada saat menjalankan program.

Contoh: `java TestGreeting args[0] args[1] ...`



Dasar Class Diagram



OOP

tidak mungkin terlepas dari
Class Diagram



CLASS DIAGRAM



Politeknik Elektronika Negeri Surabaya

Apa itu **Class Diagram**?

Diagram yang menggambarkan **Class**

Baris 1:
Nama Class

Baris 2:
Daftar atribut,
variable,
konstanta

Baris 3:
Method
atau fungsi

Class Diagram

Departemen Teknik Informatika & Komputer

51

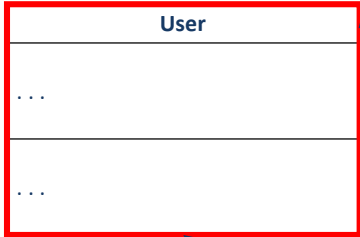
Politeknik Elektronika Negeri Surabaya

CLASS NAME

Departemen Teknik Informatika & Komputer

52

Politeknik Elektronika Negeri Surabaya




UML class diagram for the **User** class. The class name "User" is in the top compartment. Below it are two empty compartments, each containing three dots "...". The entire diagram is enclosed in a red rectangular border.

Jika di implementasikan dalam Bahasa java:

```
public class User{
    ....
}
```

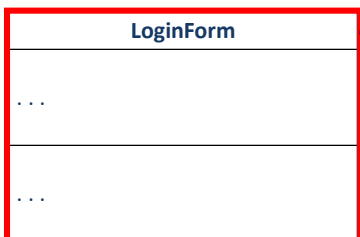
Ini adalah satu buah class dengan nama class : **User**



Departemen Teknik Informatika & Komputer

53

Politeknik Elektronika Negeri Surabaya




UML class diagram for the **LoginForm** class. The class name "LoginForm" is in the top compartment. Below it are two empty compartments, each containing three dots "...". The entire diagram is enclosed in a red rectangular border.

Jika di implementasikan dalam Bahasa java:

```
public class LoginForm{
    ....
}
```

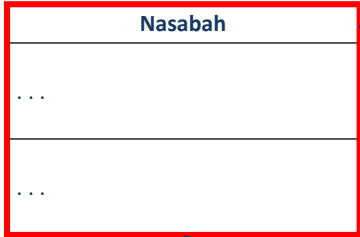
Ini adalah satu buah class dengan nama class : **LoginForm**



Departemen Teknik Informatika & Komputer

54

Politeknik Elektronika Negeri Surabaya




The diagram shows a class box for 'Nasabah' with a header section containing the name and two body sections, each containing three dots to indicate fields and methods.

Jika di implementasikan dalam Bahasa java:

```
public class Nasabah{
    ....
}
```

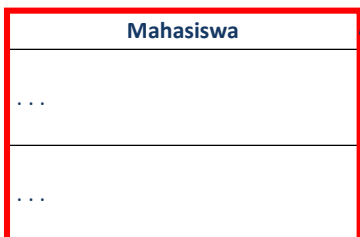
Ini adalah satu buah class dengan nama class : **Nasabah**



Departemen Teknik Informatika & Komputer

55

Politeknik Elektronika Negeri Surabaya




The diagram shows a class box for 'Mahasiswa' with a header section containing the name and two body sections, each containing three dots to indicate fields and methods.

Jika di implementasikan dalam Bahasa java:

...?

Ini adalah satu buah class dengan nama class : **Mahasiswa**



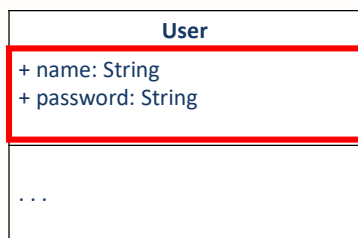
Departemen Teknik Informatika & Komputer

56

ATTRIBUTE



57



Jika di implementasikan dalam Bahasa java:

```
public class User{
    public String name;
    public String password;
}
```

Dalam class User terdapat dua variable : name dan password



58


Politeknik Elektronika Negeri Surabaya

Nasabah
+ saldo: Long
+ id: String
+ rekening: String
...

Jika di implementasikan dalam Bahasa java:

```
public class Nasabah{
    public Long saldo;
    public String id;
    public String rekening;
}
```

Dalam class Nasabah terdapat 3 buah variable: saldo, id, rekening



Departemen Teknik Informatika & Komputer

59


Politeknik Elektronika Negeri Surabaya

Mahasiswa
+ nrp: String
+ nama: String
+ jurusan: String
...

Jika di implementasikan dalam Bahasa java:

...?

Dalam class Mahasiswa ada 3 variable: nrp, nama, jurusan



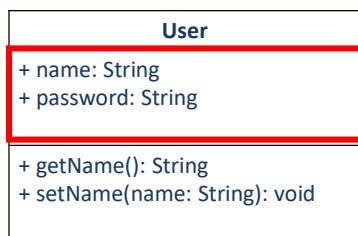
Departemen Teknik Informatika & Komputer

60

METHOD



61



Dalam class User terdapat dua method : getName() dan setName()

Jika di implementasikan dalam Bahasa java:

```

public class User{
    public String name;
    public String password;

    public String getName(){...}
    public void setName(String
name){...}
}
  
```



62

Politeknik Elektronika Negeri Surabaya


Nasabah
+ saldo: Long + id: String + rekening: String
+ saving(value:Long): void + withdraw(value:Long): boolean

Jika di implementasikan dalam Bahasa java:

```
public class Nasabah{
    public Long saldo;
    public String id;
    public String rekening;

    public void saving(Long value){...}
    public Boolean withdraw(Long value){...}
}
```

Dalam class Nasabah terdapat 2 buah method: saving() dan withdraw()



Departemen Teknik Informatika & Komputer

63


Politeknik Elektronika Negeri Surabaya

Mahasiswa
+ nrp: String + nama: String + jurusan: String
+ getNrp(): String + setName(nama:String): void

Jika di implementasikan dalam Bahasa java:

...?

Dalam class Mahasiswa ada 2 method: getNrp() dan setName()



Departemen Teknik Informatika & Komputer

64

MODIFIER



+	adalah public
-	adalah private
#	adalah protected
~	adalah default



Mahasiswa	
+	nrp: String
-	nama: String
#	jurusan: String
~	wali: String
...	



Java Naming Convention



Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika & Komputer

67

Politeknik Elektronika Negeri Surabaya

Java Naming Convention

- Code Convention
 - Cara penulisan kode program yang telah disepakati sekelompok programmer
- Java Code Convention
 - <https://www.oracle.com/technetwork/java/codeconventions-150003.pdf>
 - 12 September 1997
 - Revisi: 20 April 1999
<https://www.oracle.com/java/technologies/javase/codeconventions-contents.html>
- Java Naming Convention
 - <https://www.oracle.com/java/technologies/javase/codeconventions-namingconventions.html>



Departemen Teknik Informatika & Komputer

68

Naming Convention: Classes

```
class Raster;  
class ImageSprite;
```

- Class names should be **nouns**, in **mixed case** with the **first letter** of **each internal word** **capitalized**
- Try to keep your class names **simple** and **descriptive**
- Use **whole words**—avoid acronyms and abbreviations (unless the abbreviation is much more widely used than the long form, such as URL or HTML)



Naming Convention: Interfaces

```
interface RasterDelegate;  
interface Storing;
```

- Interface names should be **capitalized like class** names



Naming Convention: Methods

```
run();
runFast();
getBackground();
```

- Methods should be **verbs**
- It is written in **mixed case** with the **first letter lowercase**, with the **first letter of each internal word capitalized**



Naming Convention: Variables

```
int      i;
char     *cp;
float    myWidth;
```

- Variable names should be **short yet meaningful**
- The choice of a variable name should be **mnemonic**— that is, designed to indicate to the casual observer the intent of its use
- **One-character** variable names should be **avoided** except for temporary “throwaway” variables



Naming Convention: Constants

```
int MIN_WIDTH = 4;  
int MAX_WIDTH = 999;  
int GET_THE_CPU = 1;
```

- The names of variables declared class constants should be **all uppercase**



Tugas

1. Apakah yang dimaksud dengan kelas, method, atribut dan obyek?
2. Buatlah contoh suatu kelas dan definisikan atribut dan methodnya!
3. Buatlah kode program soal no. 2 diatas!
4. Buatlah kelas yang berisi main method yang membuat obyek dari kelas yang telah dibuat di soal no. 3. Selanjutnya obyek tersebut mengakses atribut dan methodnya.



1. Oracle Java Documentation, The Java™ Tutorials, <https://docs.oracle.com/javase/tutorial/>, Copyright © 1995, Oracle 2015.
2. Tita Karlita, Yuliana Setrowati, Rizky Yuniar Hakkun, Pemrograman Berorientasi Obyek, PENS-2012
3. Sun Java Programming, Sun Educational Services, Student Guide, Sun Microsystems, 2001.
4. John R. Hubbard, Programming With Java, McGraw-Hill, ISBN: 0-07-142040-1, 2004.
5. Patrick Niemeyer, Jonathan Knudsen, Learning Java, O'reilly, CA, ISBN: 1565927184, 2000.
6. Philip Heller, Simon Roberts, Complete Java 2 Certification Study Guide, Third Edition, Sybex, San Francisco, London, ISBN: 0-7821-4419-5, 2002.
7. Herbert Schildt, The Complete Reference, Java™ Seventh Edition, Mc Graw Hill, Osborne, ISBN: 978-0-07-163177-8, 2007

75



76

Quiz

- Bagaimana aturan penamaan class?
- Berikan contoh nama class!
- Bagaimana aturan penamaan attribute?
- Berikan contoh nama attribute!
- Bagaimana aturan penamaan method?
- Berikan contoh nama method!
- Apa yang dimaksud constructor dan bagaimana aturannya?



Point
+ coordinateX: int
+ coordinateY: int
+ coordinateZ: int
+ Point(coordinateX: int, coordinateY: int, coordinateZ: int)



Line
+ firstPoint: Point
+ secondPoint: Point
+ length: double
+ Line(firstPoint: Point, secondPoint: Point)
+ getLength(): double

```
public class Line{
    public Point firstPoint;
    public Point secondPoint;
    public double length;

    public Line(Point firstPoint, Point secondPoint){
        this.firstPoint = firstPoint;
        this.secondPoint = secondPoint;
    }

    public double getLength(){
        ...
    }
}
```



Square
+ lines: Line[]
+ area: double
+ Square(lines: Line[])
+ getArea(): double



Cube
+ squares: Square[]
+ volume: double
+ Cube(squares: Square[])
+ getVolume(): double

```
public class Test{
    public void percobaan() {
        Cube kubusSaya = new Cube(...);
        kubusSaya.squares[0].lines[0].firstPoint.coordinateX
    }
}
```

```
public class Cube{
    public Square[] squares;
    public double volume;

    public Cube(Square[] squares){
        this.squares = squares;
    }

    public double getVolume(){
        return volume;
    }
}
```

