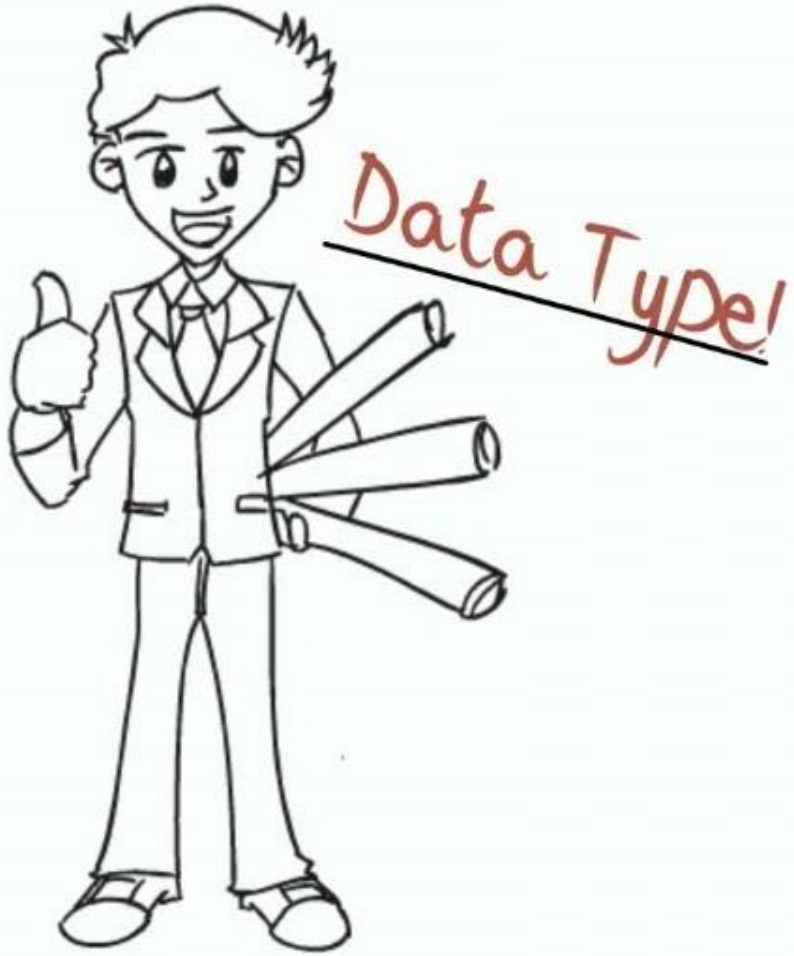
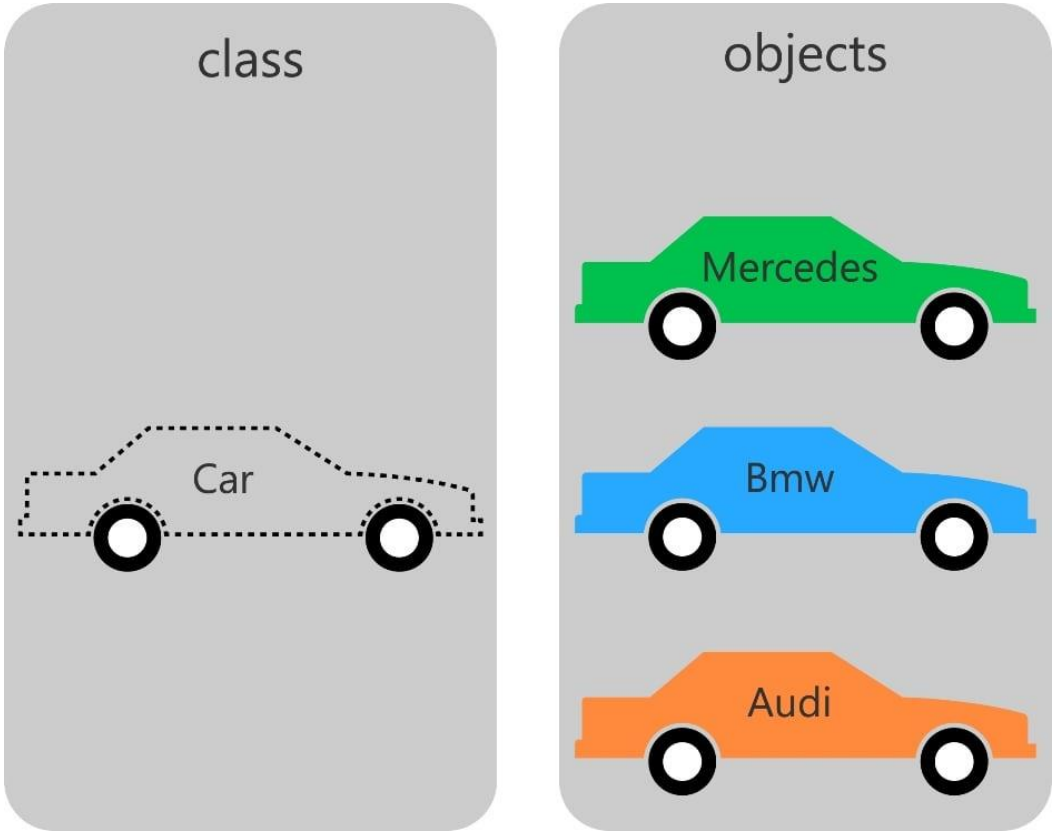
What are classes?

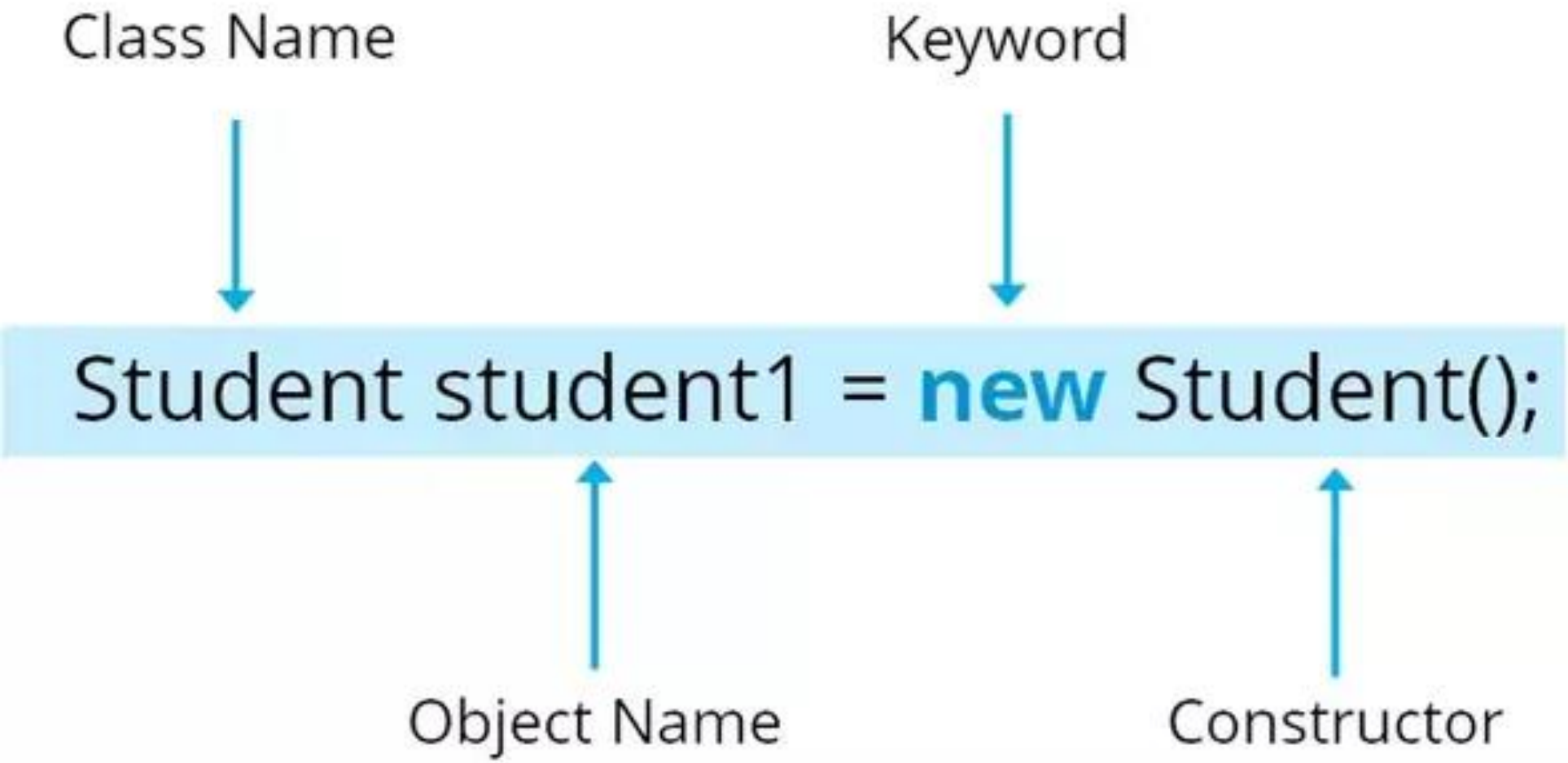


Class adalah ‘blueprint’ dari object. Class digunakan hanya untuk membuat kerangka dasar. Yang akan kita pakai nantinya adalah hasil cetakan dari class, yakni object

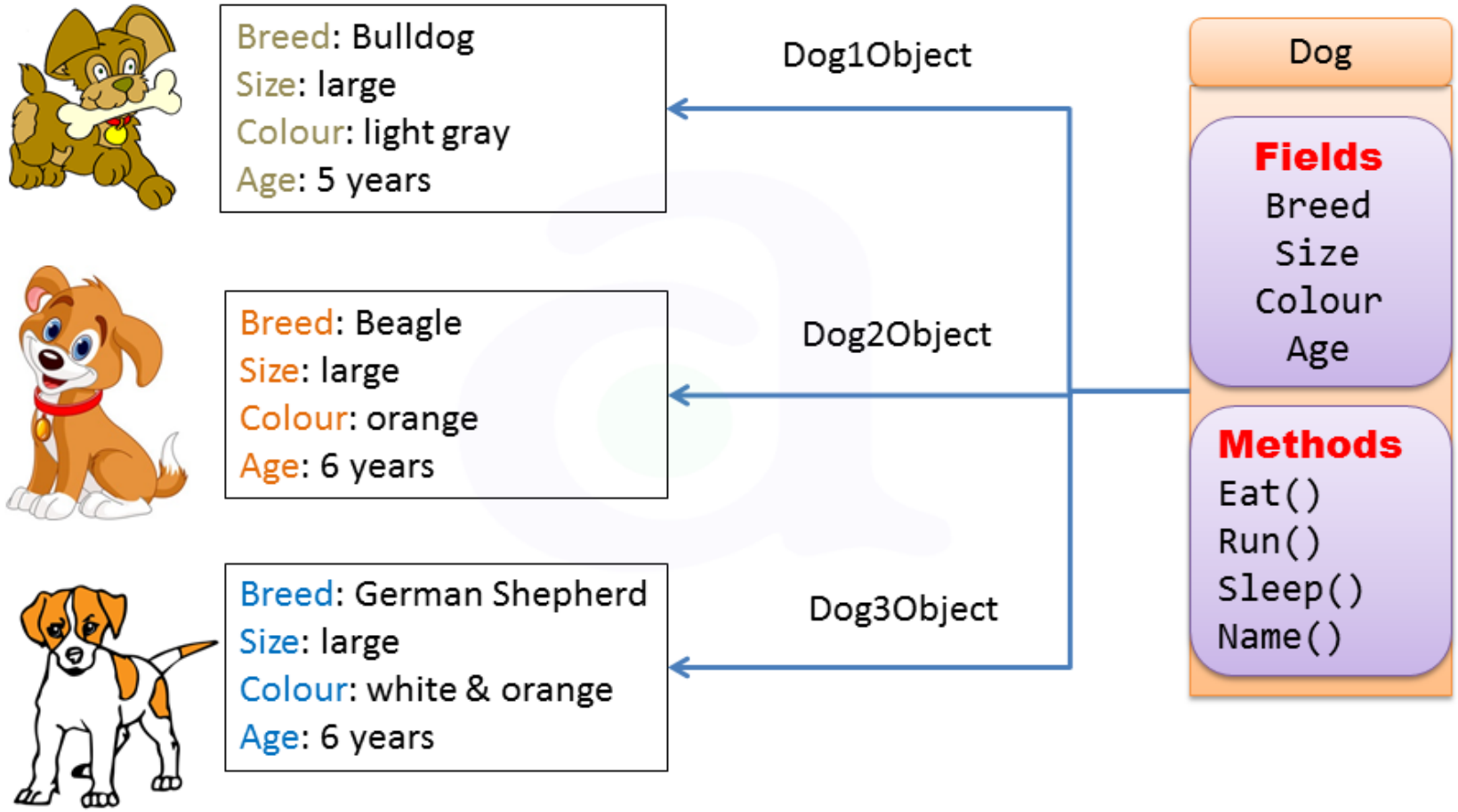


Object adalah *hasil cetak* dari class, atau hasil ‘*konkrit*’ dari class.

Proses '*mencetak*' objek dari class ini disebut dengan ***instantiation***.
Proses *instantiation* dilakukan dengan menggunakan keyword 'new'.

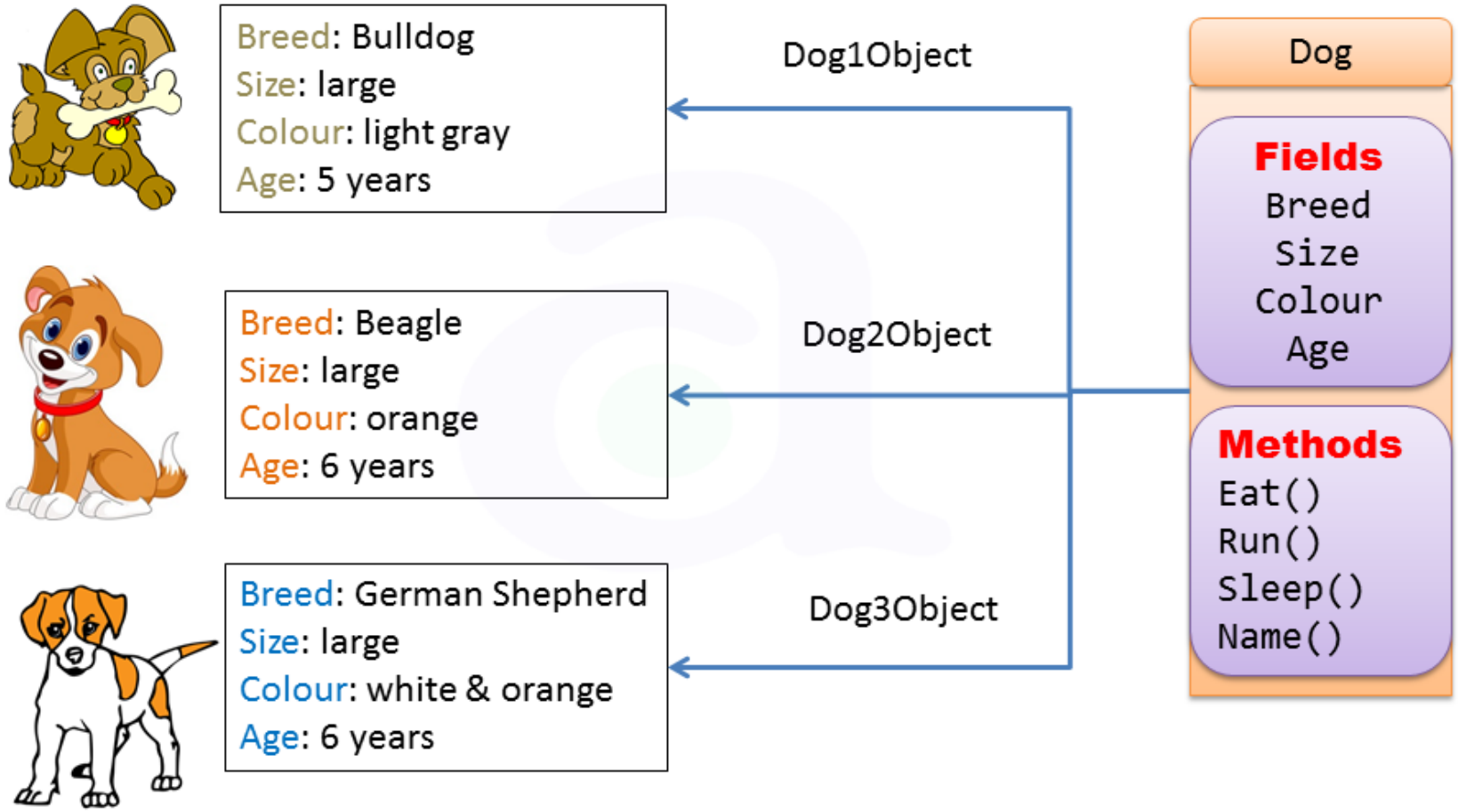


Property (atau disebut juga dengan *atribut*) adalah data yang terdapat dalam sebuah class



Property ini sebenarnya hanyalah *variabel* yang terletak di dalam *class*. Seluruh aturan dan tipe data yang biasa diinput kedalam *variabel*, bisa juga diinput kedalam *property*. Aturan tata cara penamaan *property* sama dengan aturan penamaan *variabel*.

Method adalah tindakan yang bisa dilakukan didalam *class*



Method pada dasarnya adalah ***function*** yang berada di dalam **class**. Seluruh fungsi dan sifat function bisa diterapkan kedalam method, seperti argumen/parameter, mengembalikan nilai (dengan keyword *return*), dll

```
class NamaKelas{
        tipe data1;
        tipe data2;
        …
        tipe dataN;

        tipe method1(daftar-parameter){
                kode untuk method1
        }

        tipe method2(daftar-parameter){
                kode untuk method2
        }

        tipe method3(daftar-parameter){
                kode untuk method3
        }
}
```

```
Class Guguk
{
      String breed;
      String size;
      String colour;
      int age;

      void eat()
      {…}

      void run()
      {…}

      void sleep()
      {…}
}
```

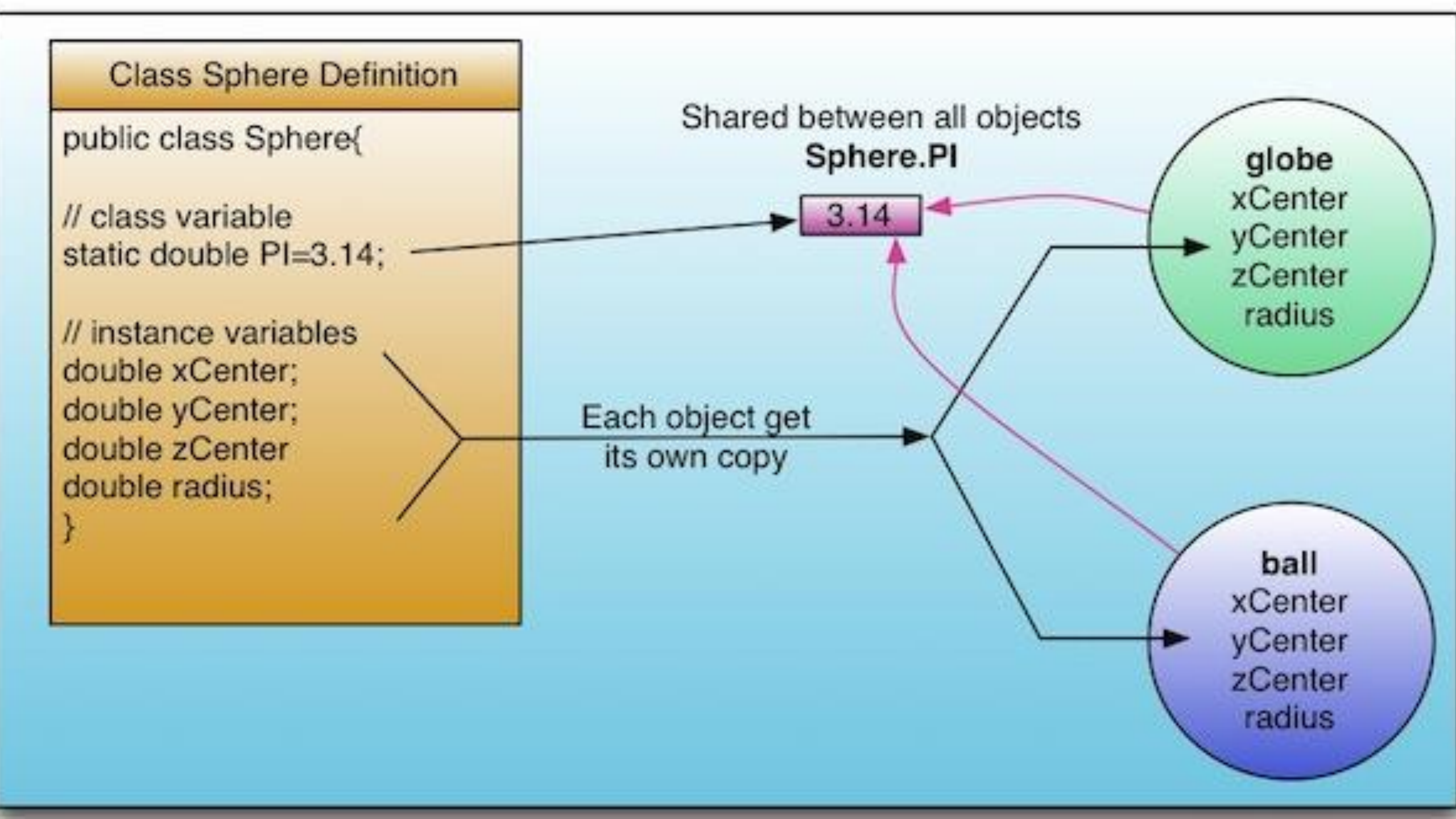
Jenis Variabel

Class Variable

- Variabel class adalah variabel yang dapat diakses oleh semua method di dalam class, termasuk *static method* seperti *main*.
- Menempatkan deklarasi variabel dalam blok class, tetapi tidak dalam method yang berada dalam class.
- Kita harus memasukkan kata static dalam pendeklarasian variabel. Kata static ditempatkan sebelum tipe variabel.

Instance Variable

- Variabel instance sama halnya dengan variabel class, tetapi tidak menggunakan kata static dalam pendeklarasiannya.
- Variabel instance terhubung dengan instance dari class. Jadi hanya dapat digunakan ketika membuat instance dari class.
- Karena static method tidak terhubung dengan instance dari class, maka tidak dapat menggunakan variabel instance dalam static method dan memasukkannya dalam main method.


Class Sphere Definition
public class Sphere{
// class variable
static double PI=3.14;
// instance variables
double xCenter;
double yCenter;
double zCenter
double radius;
}
Shared between all objects
Sphere.PI
3.14
Each object get
its own copy
globe
xCenter
yCenter
zCenter
radius
ball
xCenter
yCenter
zCenter
radius

Kelas

```
public class Kotak {
    double panjang;
    double lebar;
    double tinggi;
}
```

```java
public class DemoKotak1 {
    public static void main (String [] args){
        double volume;
        Kotak k = new Kotak();
        k.panjang = 4;
        k.lebar = 3;
        k.tinggi = 2;
        volume = k.panjang * k.tinggi * k.lebar;
        System.out.println("Volume kotak = " + volume);
    }
}
```

Membuat objek kotak dengan nama k

DemoKotak1 > main >

put - Pertemuan5 (run)

```
run:
Volume kotak = 24.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

```java
public class DemoKotak2 {
    public static void main (String [] args){
        double volume1, volume2;
        Kotak k1 = new Kotak();
        Kotak k2 = new Kotak();
        k1.panjang = 4;
        k1.lebar = 3;
        k1.tinggi = 2;
        k2.panjang = 6;
        k2.lebar = 5;
        k2.tinggi = 4;
        volume1 = k1.panjang*k1.lebar*k1.tinggi;
        volume2 = k2.panjang*k2.lebar*k2.tinggi;
        System.out.println("Volume k1 = "+ volume1);
        System.out.println("Volume k2 = "+ volume2);
    }
}
```

Setiap objek atau instance dari suatu kelas akan memiliki salinan data sendiri-sendiri.

put - Pertemuan5 (run)

```
run:
Volume k1 = 24.0
Volume k2 = 120.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

```java
public class DemoReferensi1 {
    public static void main (String [] args){
        double volume1, volume2;
        Kotak k1, k2;
        k1 = new Kotak();
        k2 = k1;
        k1.panjang = 4;
        k1.lebar = 3;
        k1.tinggi = 2;
        volume1 = k1.panjang*k1.lebar*k1.tinggi;
        volume2 = k2.panjang*k2.lebar*k2.tinggi;
        System.out.println("Volume k1 = "+ volume1);
        System.out.println("Volume k2 = "+ volume2);
    }
}
```

Variabel k2 berperan sebagai referensi ke objek yang sedang ditunjuk k1

DemoReferensi1 > main >

put - Pertemuan5 (run)

```
run:
Volume k1 = 24.0
Volume k2 = 24.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

```java
public class DemoReferensi2 {
    public static void main (String [] args){
    double volume1, volume2;
        Kotak k1, k2;
        k1 = new Kotak();
        k2 = k1;
        k1.panjang = 4;
        k1.lebar = 3;
        k1.tinggi = 2;
        volume1 = k1.panjang*k1.lebar*k1.tinggi;
        volume2 = k2.panjang*k2.lebar*k2.tinggi;
        System.out.println("Volume k1 = "+ volume1);
        System.out.println("Volume k2 = "+ volume2);
        k1 = new Kotak();
        k1.panjang = 6;
        k1.lebar = 5;
        k1.tinggi = 4;
        volume1 = k1.panjang*k1.lebar*k1.tinggi;
        volume2 = k2.panjang*k2.lebar*k2.tinggi;
        System.out.println("Volume k1 = "+ volume1);
        System.out.println("Volume k2 = "+ volume2);
}
}
```

Pada saat memasukkan suatu objek dari salah satu variabel referensi lain, maka sebenarnya kita tidak membuat salinan objek, melainkan hanya membuat salinan referensinya saja.

out - Pertemuan5 (run)

```
run:
Volume k1 = 24.0
Volume k2 = 24.0
Volume k1 = 120.0
Volume k2 = 24.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

JENIS METHOD

VOID

- Method yang tidak mengembalikan nilai

NON VOID

- Method yang mengembalikan nilai

Method VOID (tidak mengembalikan nilai)

```
public class Kotak2 {
    double panjang;
    double lebar;
    double tinggi;
    void cetakvolume(){
    System.out.println ("Volume kotak = " + (panjang*lebar*tinggi));
    }
}
```

WAJIB digunakan untuk method VOID

```java
public class DemoMethod1 {
    public static void main (String [] args){
        Kotak2 k1,k2,k3;
        k1 = new Kotak2();
        k2 = new Kotak2();
        k3 = new Kotak2();
        k1.panjang = 4;
        k1.lebar = 3;
        k1.tinggi = 2;

        k2.panjang = 6;
        k2.lebar = 5;
        k2.tinggi = 4;

        k3.panjang = 8;
        k3.lebar = 7;
        k3.tinggi = 6;

        k1.cetakvolume();
        k2.cetakvolume();
        k3.cetakvolume();
    }
}
```

Semua method dalam suatu kelas dapat mengakses data-datanya secara langsung tanpa melalui referensi

DemoMethod1 > main >

put - Pertemuan5 (run)

```
run:
Volume kotak = 24.0
Volume kotak = 120.0
Volume kotak = 336.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Method NON-VOID (mengembalikan nilai)

```
public class Kotak3 {
    double panjang;
    double lebar;
    double tinggi;
```

Mengembalikan nilai vol dengan tipe data double

```
    double hitungvolume(){
    double vol = panjang*lebar*tinggi;
    return vol;
}
}
```

Bila ingin menyimpan nilai kembalian method ke dalam sebuah variabel, maka tipe data dari variabel tersebut harus kompatibel atau cocok dengan tipe kembalian yang didefinisikan dalam method

```java
public class DemoMethod2 {
    public static void main (String [] args){
        Kotak3 k1,k2,k3;
        k1 = new Kotak3();
        k2 = new Kotak3();
        k3 = new Kotak3();

        k1.panjang = 4;
        k1.lebar = 3;
        k1.tinggi = 2;

        k2.panjang = 6;
        k2.lebar = 5;
        k2.tinggi = 4;

        k3.panjang = 8;
        k3.lebar = 7;
        k3.tinggi = 6;

        System.out.println("Volume k1 = " + k1.hitungvolume());
        System.out.println("Volume k1 = " + k2.hitungvolume());
        System.out.println("Volume k1 = " + k3.hitungvolume());
    }
}
```

DemoMethod2 > main >

put - Pertemuan5 (run)

```
run:
Volume k1 = 24.0
Volume k1 = 120.0
Volume k1 = 336.0
BUILD SUCCESSFUL (total time: 1 second)
```

Tipe data yang dikembalikan oleh method harus kompatibel atau cocok dengan tipe data kembalian yang didefinisikan pada saat pembuatan method.

Method Berparameter

```
public class Kotak4 {
    double panjang;
    double lebar;
    double tinggi;

    void isiData(double p,double l,double t){
        panjang = p;
        lebar = l;
        tinggi = t;
    }
    double hitungVolume(){
        return (panjang*lebar*tinggi);
    }
}
```

Dengan adanya parameter, sebuah method dapat bersifat dinamis dan general

Parameter adalah variabel yang didefinisikan pada saat method dibuat

```java
public class DemoMethod3 {
    public static void main (String [] args){
        Kotak4 k;
        k = new Kotak4();
        k.isiData(4, 3, 2);
        System.out.println("Volume kotak = " + k.hitungVolume());
    }
}
```

Dapat mengembalikan nilai yang beragam sesuai dengan nilai parameter yang dilewatkannya

Argumen adalah nilai yang digunakan pada saat pemanggilan method

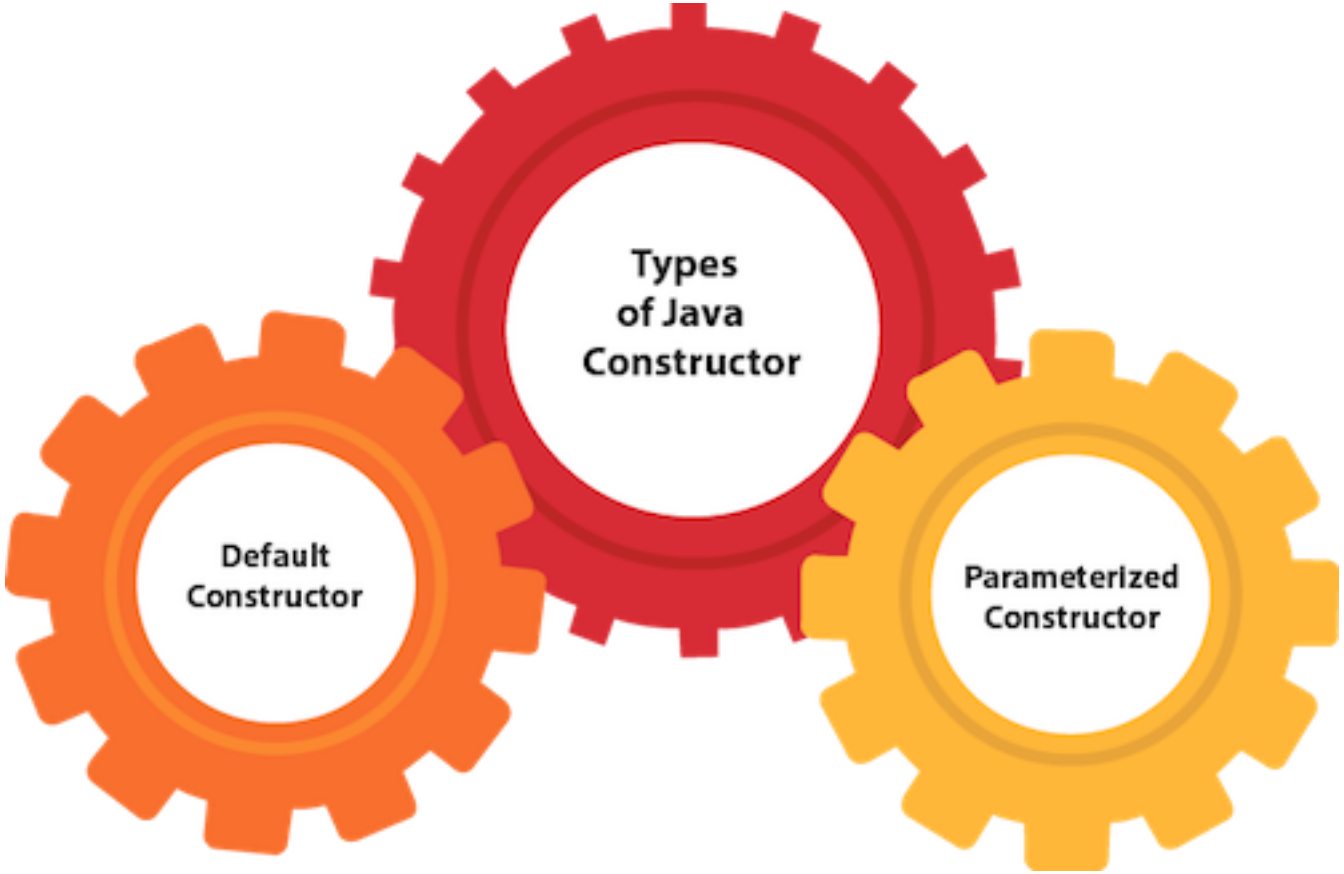
DemoMethod3 > main >

put - Pertemuan5 (run)

```
run:
Volume kotak = 24.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
class NamaKelas{
        tipe data1;
        tipe data2;
        …
        tipe dataN;

        NamaKelas (daftar-parameter){
                kode untuk konstruktor
        }

        tipe method1(daftar-parameter){
                kode untuk method1
        }

        tipe method2(daftar-parameter){
                kode untuk method2
        }

        tipe method3(daftar-parameter){
                kode untuk method3
        }
}
```

Constructor adalah method khusus yang akan dijalankan secara otomatis pada saat sebuah objek dibuat (instansiasi), yakni ketika perintah “new” dijalankan.



Constructor biasa digunakan untuk membuat proses awal dalam mempersiapkan objek, seperti memberi nilai awal kepada property, memanggil method internal dan beberapa proses lain yang digunakan untuk ‘mempersiapkan’ objek.

Constructor

```
public class Kotak5 {
    double panjang;
    double lebar;
    double tinggi;
```

Constructor adalah method khusus yang didefinisikan di dalam kelas dan akan dipanggil secara otomatis setiap kali terjadi instansiasi objek

```
    Kotak5 (){
        panjang = 4;
        lebar = 3;
        tinggi = 2;
    }
    double hitungVolume(){
        return (panjang*lebar*tinggi);
    }
}
```

Nama Constructor harus sama dengan nama kelas

Constructor tidak memiliki nilai kembalian, tidak juga void.

```java
public class DemoConstructor1 {
    public static void main (String[]args){
        Kotak5 k1,k2;

        k1 = new Kotak5();
        k2 = new Kotak5();

        System.out.println ("Volume k1 = " + k1.hitungVolume());
        System.out.println ("Volume k2 = " + k2.hitungVolume());
    }
}
```

DemoConstructor1 > main >

put - Pertemuan5 (run)

```
run:
Volume k1 = 24.0
Volume k2 = 24.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

Constructor Berparameter

```
public class Kotak6 {
    double panjang;
    double lebar;
    double tinggi;

    Kotak6 (double p, double l, double t){
        panjang = p;
        lebar = l;
        tinggi = t;
    }
    double hitungVolume(){
        return (panjang*lebar*tinggi);
    }
}
```

Nilai awal dapat bersifat dinamis tergantung dari nilai-nilai yang dilewatkan pada saat pembentukan objek

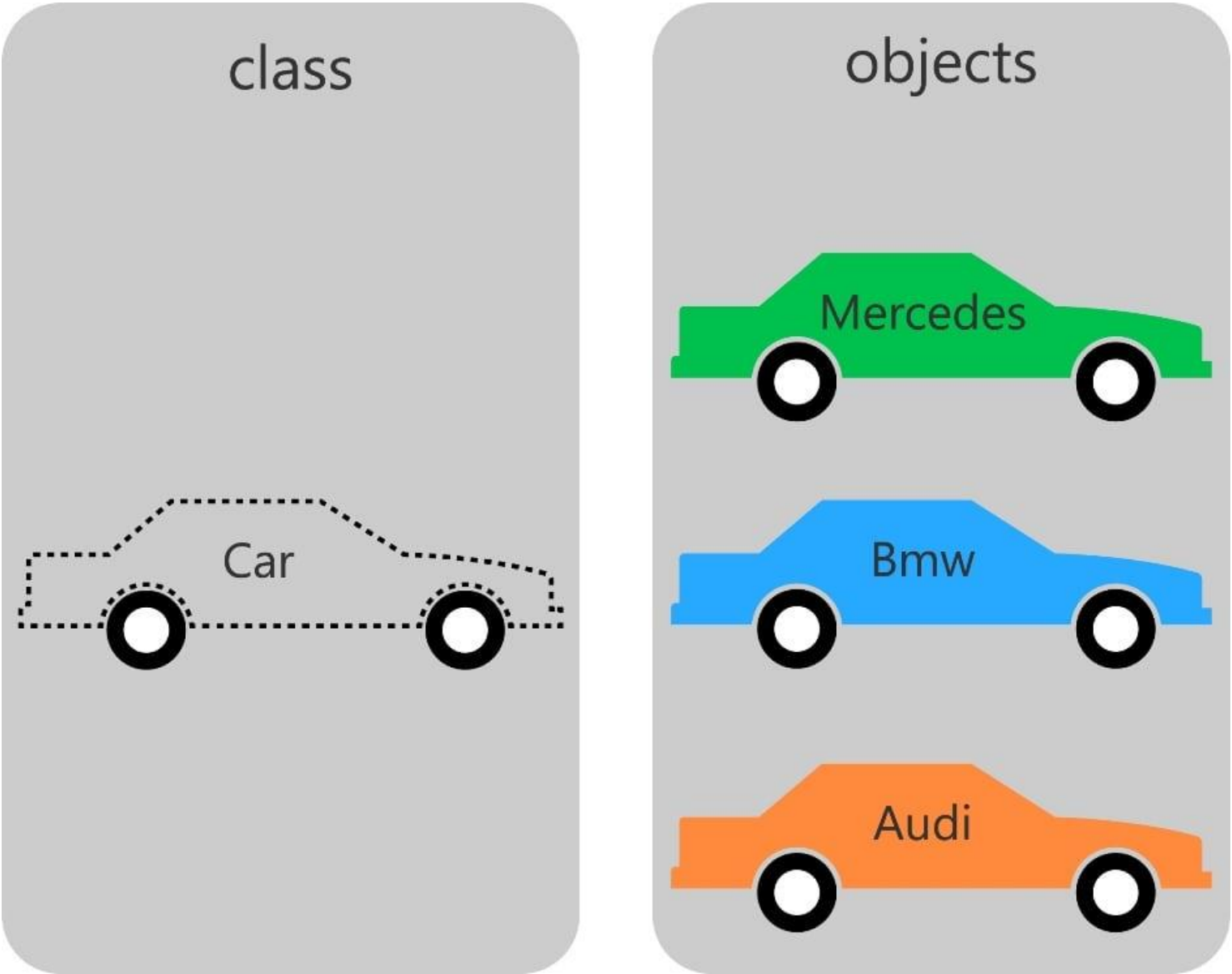
```java
public class DemoConstructor2 {
    public static void main (String [] args){
    Kotak6 k1,k2;
    k1 = new Kotak6 (4,3,2);
    k2 = new Kotak6 (6,5,4);

    System.out.println ("Volume k1 = " + k1.hitungVolume());
    System.out.println ("Volume k2 = " + k2.hitungVolume());
}
}
```

DemoConstructor2 > main >

put - Pertemuan5 (run)

```
run:
Volume k1 = 24.0
Volume k2 = 120.0
BUILD SUCCESSFUL (total time: 1 second)
```

1. Buatlah objects dari class Car



2. Apa isi dari spacer.health ?

```
class Player {
    int health;
    int shield;

    // constructor to initialize health
    Player(int initHealth, int initShield) {
        health = initHealth;
        shield = initShield;
    }
}
```

```
Player spacer = new Player(75, 100);
```

3. **Buatlah sebuah class**
4. **Tentukan atribut dan method dari class tersebut**
5. **Buatlah object dari class yang anda miliki**
6. **Buatlah ilustrasi untuk memanggil atribut/property dan method dari object yang anda miliki**