Pemrograman Berbasis Objek
Konsep Dasar PBO

Adam Mukharil Bachtiar

Teknik Informatika UNIKOM

Konsep Dasar PBO

1. Class
2. Class Design
3. Atribut dan behavior
4. Objek
5. Perbedaan class dan objek
6. Translate class diagram into code

Class

1. Kumpulan objek-objek yang memiliki atribut yang sama.
2. Template untuk membuat objek .
3. Prototipe atau blue print yang mendefinisikan variabel-variabel dan method-method secara umum.

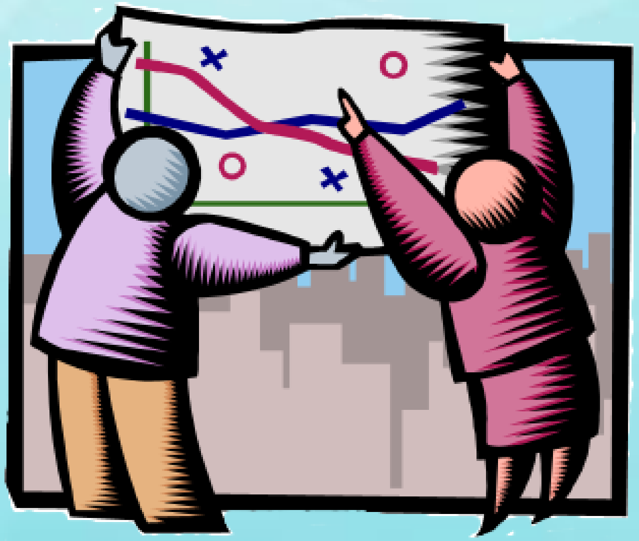
Class (Cont’d)

4. Objek merupakan hasil instansiasi dari class.
5. Proses pembentukan objek dari suatu class disebut **INSTANTIATION**.
6. Objek disebut juga **INSTANCES**.

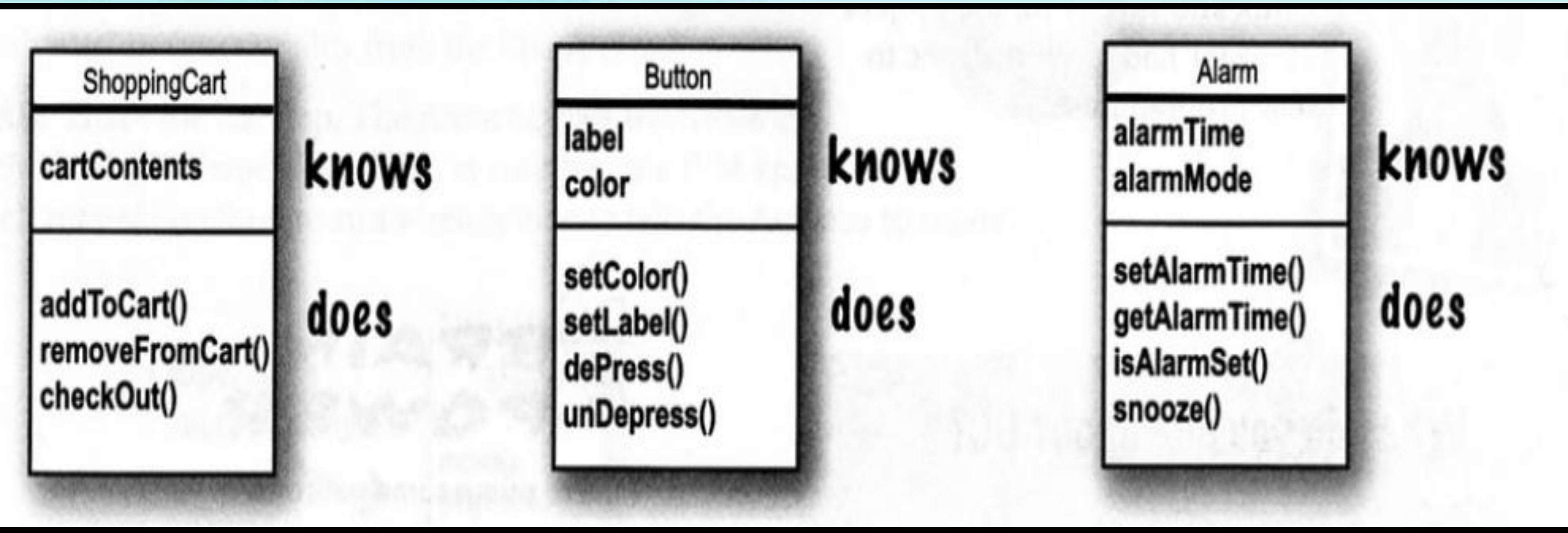
Class Design

In design a class, think about 2 things:

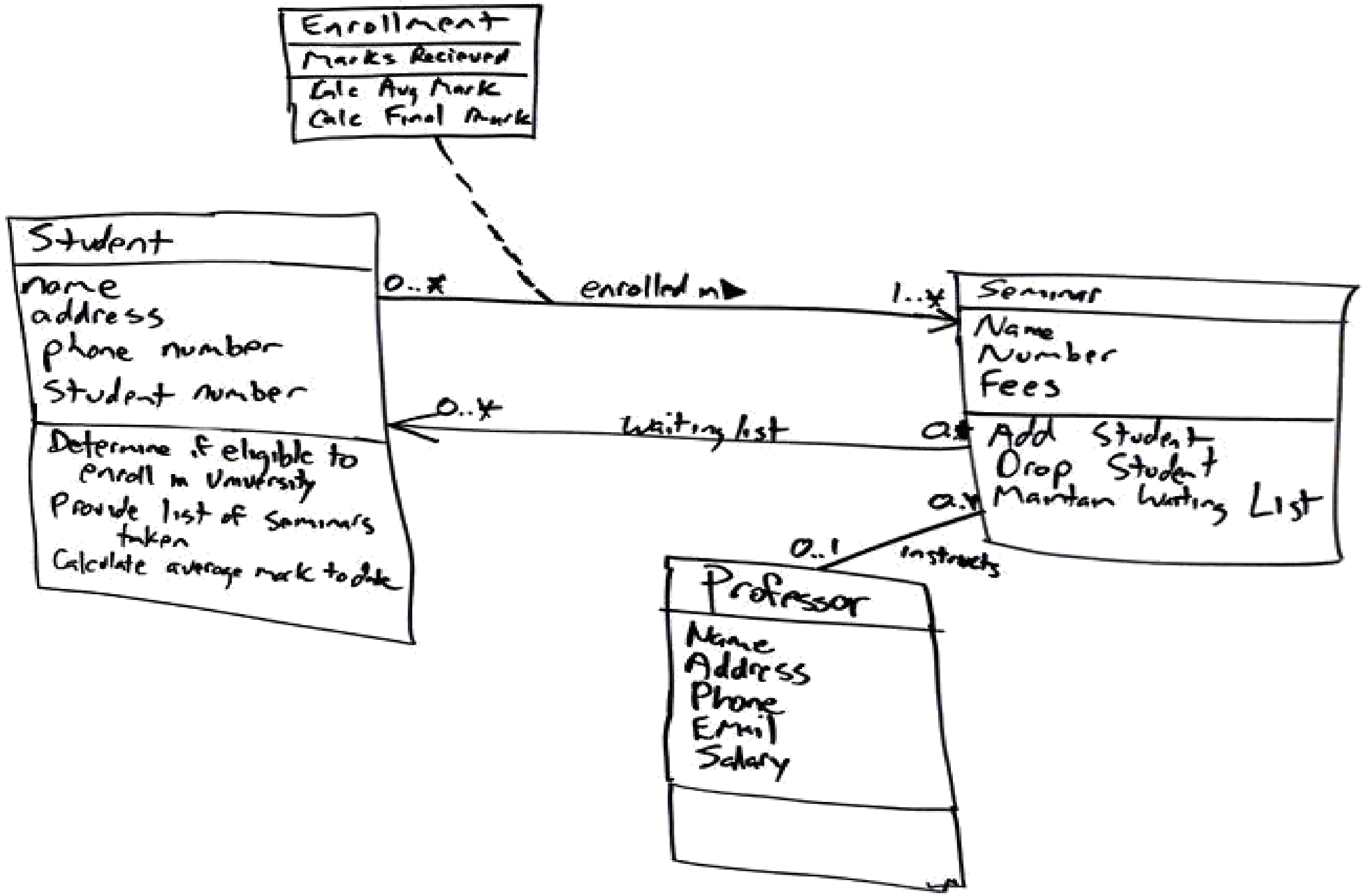
1. Things the object **knows**
2. Things the object **does**.



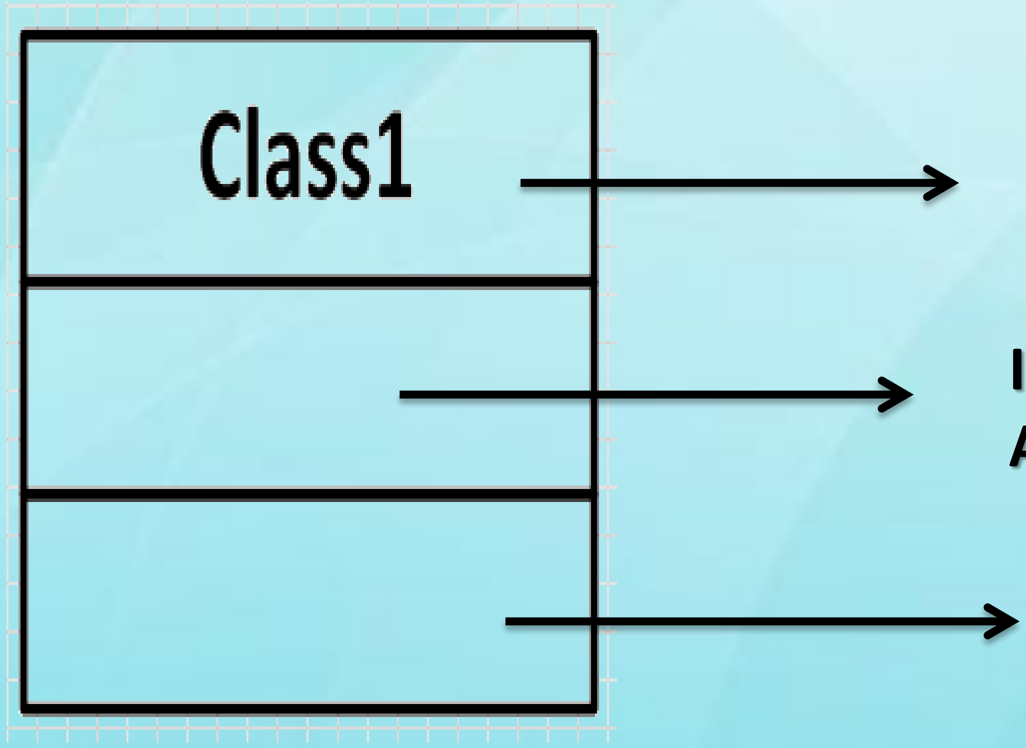
Class Design (Cont'd)



Class Design (Cont'd)



Class Design (Cont’d)



Nama Class

Instance Variable / Atribut

Behaviour / Method

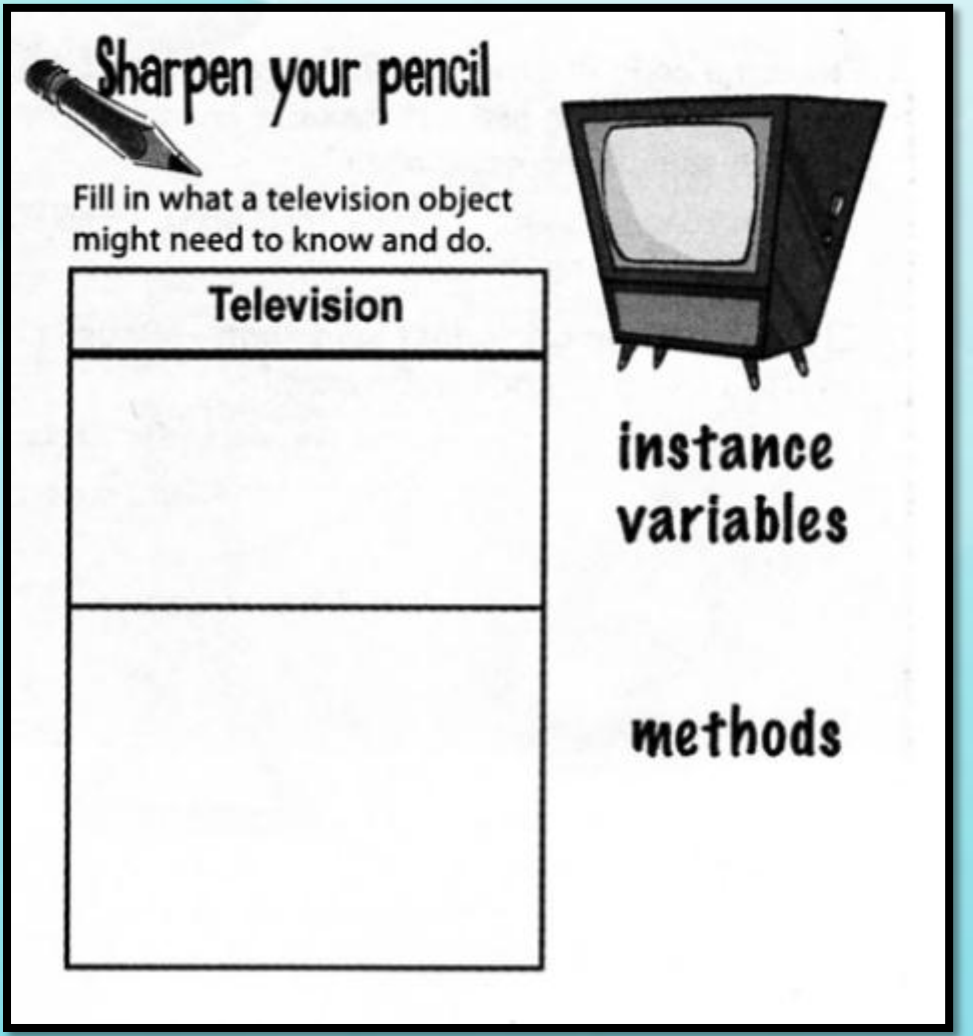
Atribut

1. Data yang membedakan antara object yang satu dengan yang lain.
2. Contoh: Manusia → Salah satu mahluk hidup. Atributnya: status, berat badan, dan tinggi badan.
3. Di dalam class atribut disebut sebagai **VARIABEL**.

Method

1. Serangkaian statements dalam suatu class yang menghandle suatu task.
2. Cara objek berkomunikasi dengan objek lain adalah dengan menggunakan methods.

Brain Storming!!!!



Sharpen your pencil

Fill in what a television object might need to know and do.

Television

instance variables

methods

Objek

Abstraksi dari sesuatu yang mewakili sesuatu yang ada di dunia nyata dan harus dapat dibedakan dengan objek lain.

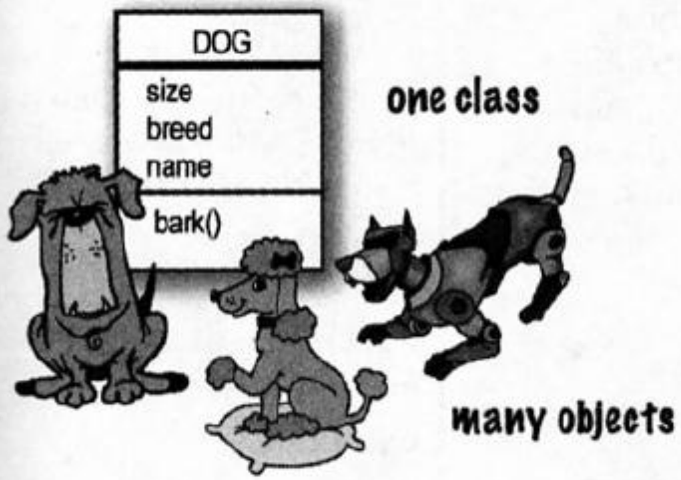
Objek

1. Semua benda di dunia nyata bisa dianggap sebagai objek.
2. Contoh: kursi, meja, buku, sepeda, komputer.
3. Penggambaran pemrograman berorientasi objek = penggambaran di dunia nyata.

Jadi apa hubungan class dengan objek???

What's the difference between a class and an object?

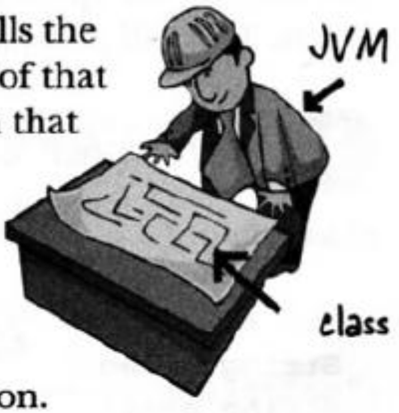
1 CLASS = N BUAH OBJEK



A class is not an object.

(but it's used to construct them)

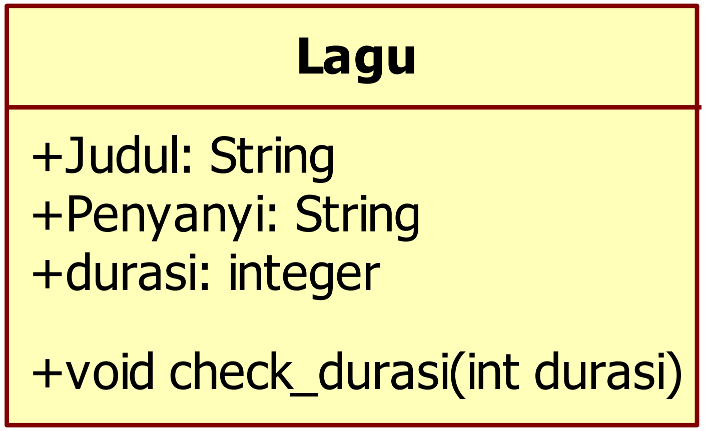
A class is a *blueprint* for an object. It tells the virtual machine *how* to make an object of that particular type. Each object made from that class can have its own values for the instance variables of that class. For example, you might use the Button class to make dozens of different buttons, and each button might have its own color, size, shape, label, and so on.



CLASS VS OBJECT = DATA TYPE VS VARIABEL

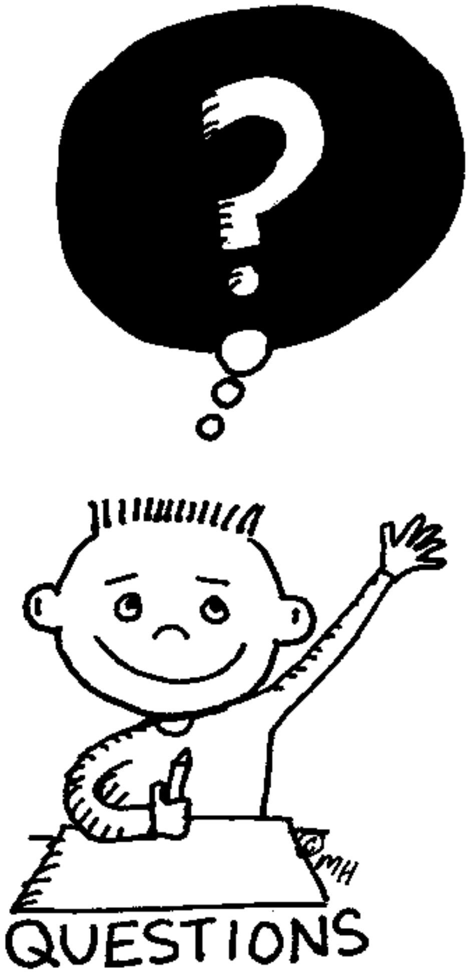
Class Diagram

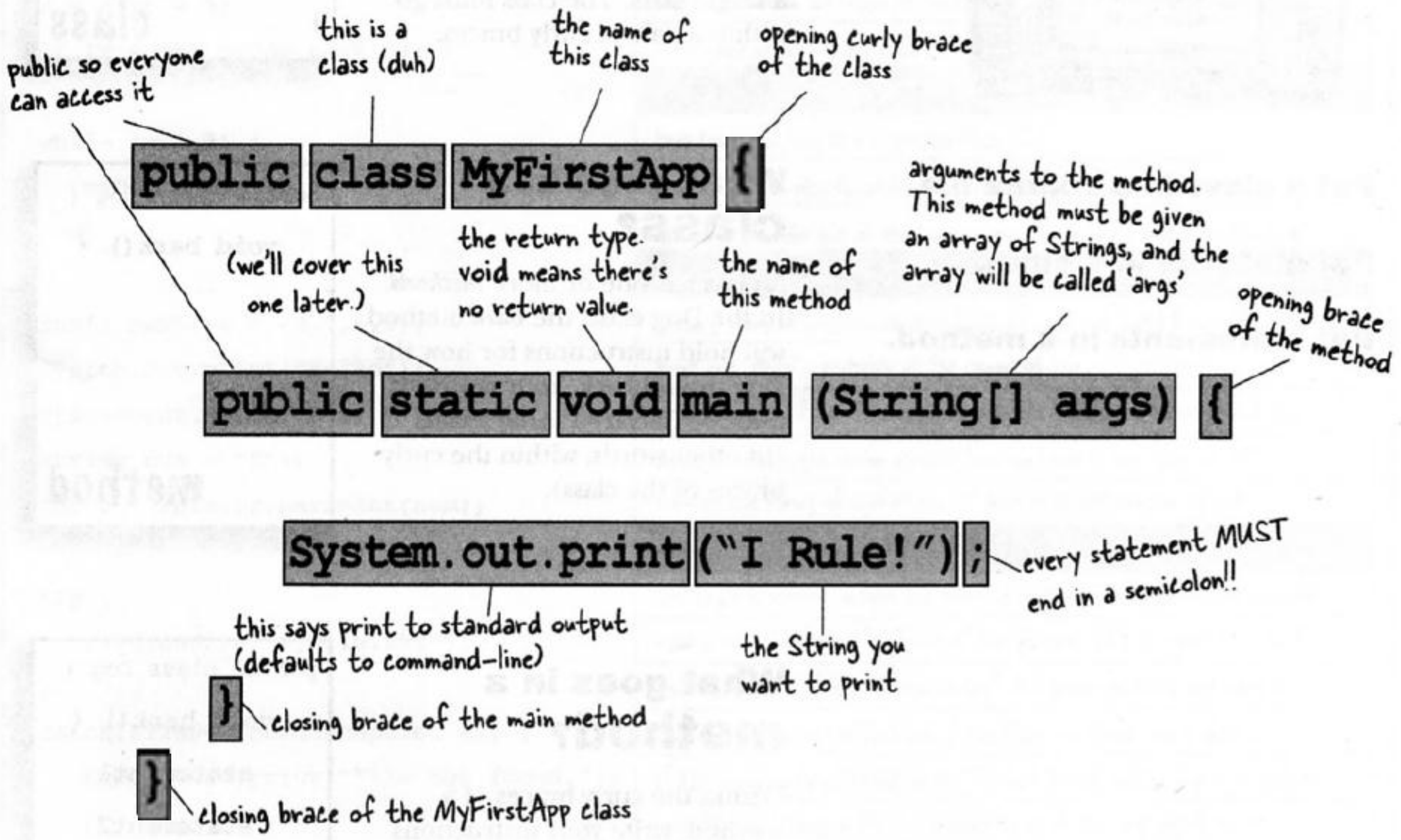
Kode Program



```
class lagu{
   char judul[35];
   char penyanyi[25];
   int durasi;

   void check_durasi(int durasi){
      if(durasi>0)
         cout<<"Valid";
   }
}
```

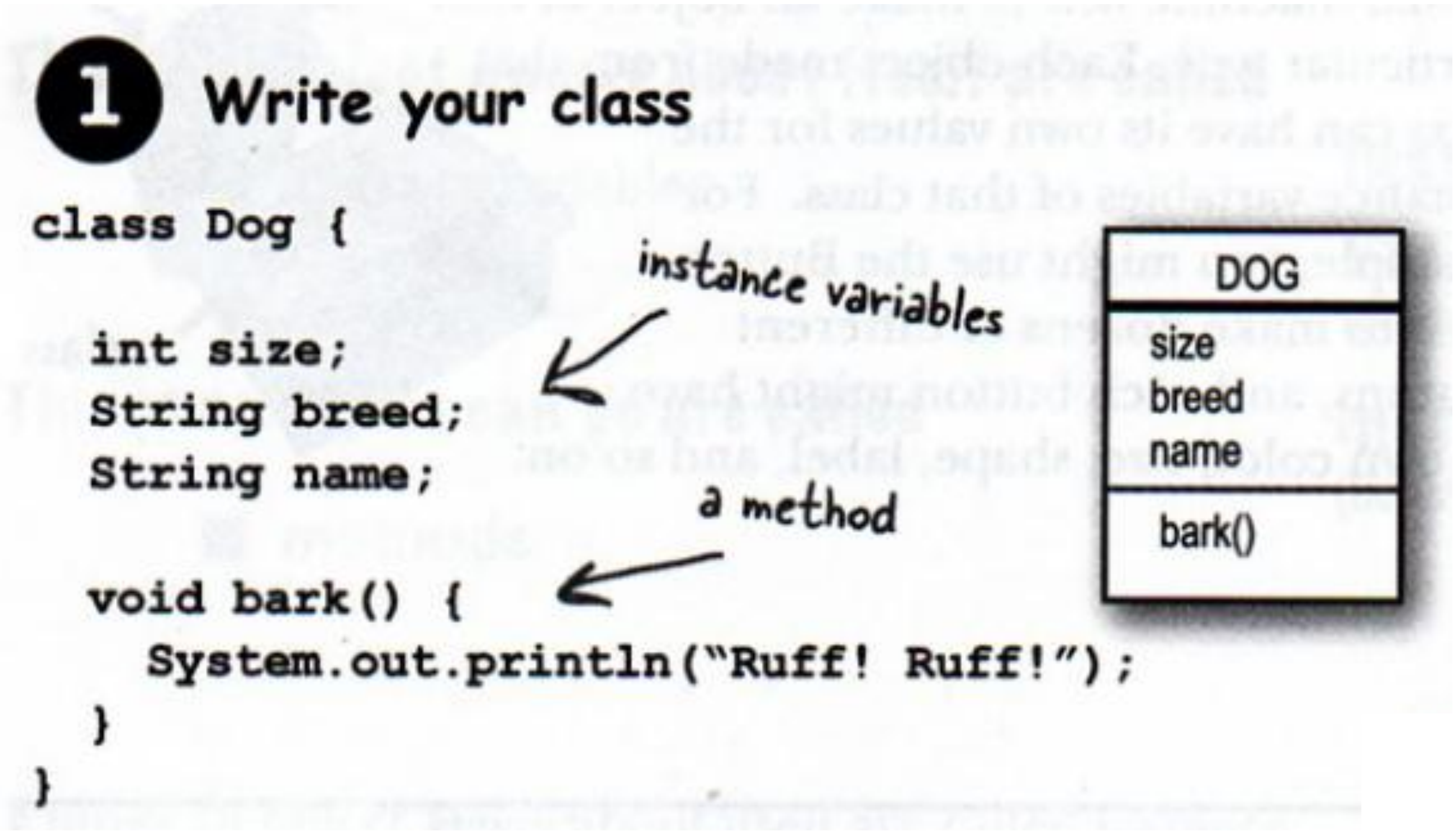
Next Question:
What about coding in JAVA?





```
public class MyFirstApp {
public static void main (String[] args) {
System.out.print("I Rule!");
}
}
```

public so everyone can access it
this is a class (duh)
the name of this class
opening curly brace of the class
(we'll cover this one later.)
the return type. void means there's no return value.
the name of this method
arguments to the method. This method must be given an array of Strings, and the array will be called 'args'
opening brace of the method
every statement MUST end in a semicolon!!
this says print to standard output (defaults to command-line)
the String you want to print
closing brace of the main method
closing brace of the MyFirstApp class

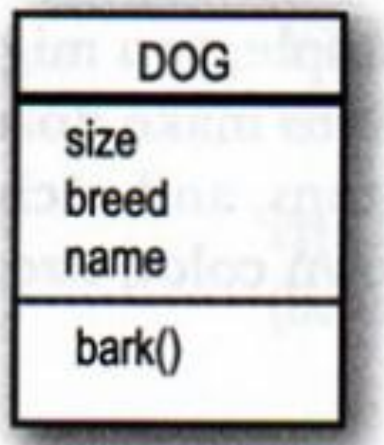


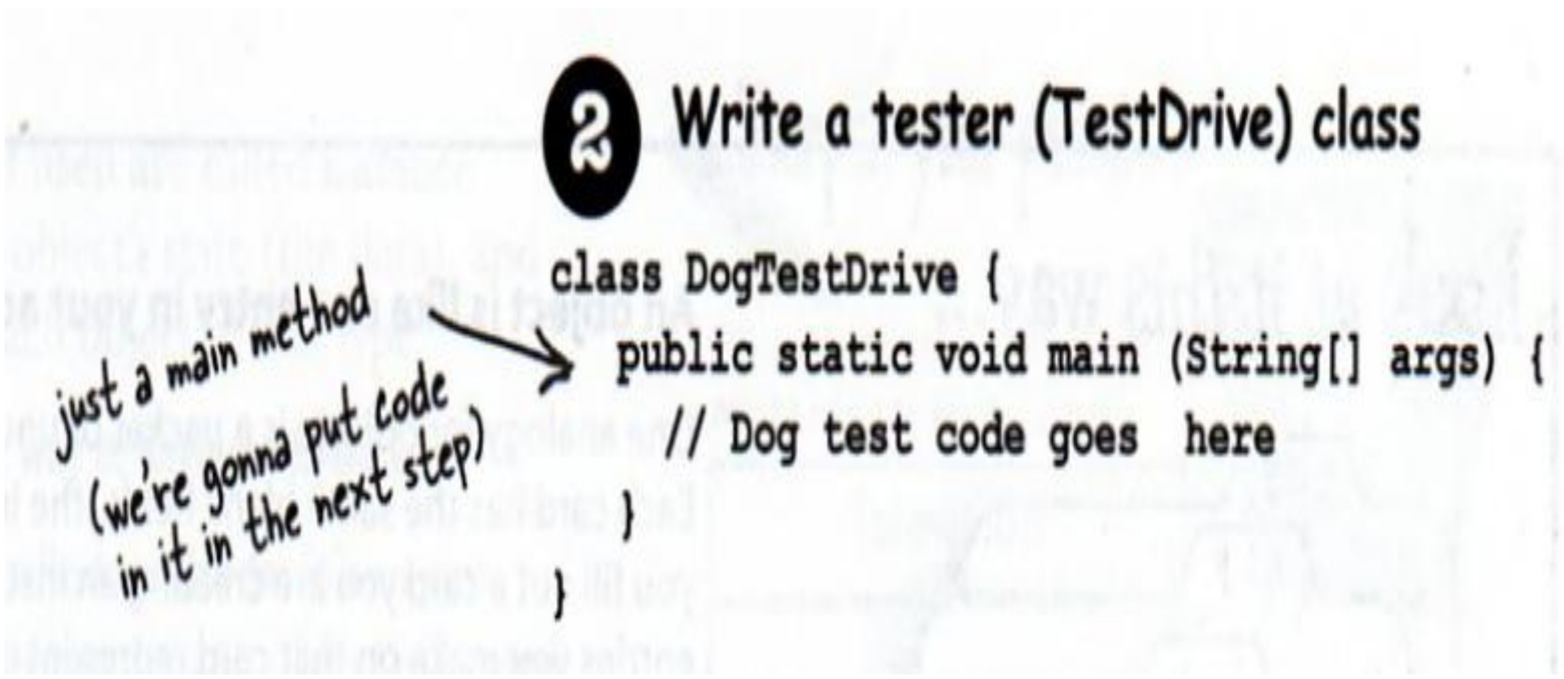
1 Write your class

```
class Dog {

  int size;
  String breed;
  String name;

  void bark() {
    System.out.println("Ruff! Ruff!");
  }
}
```

instance variables

a method

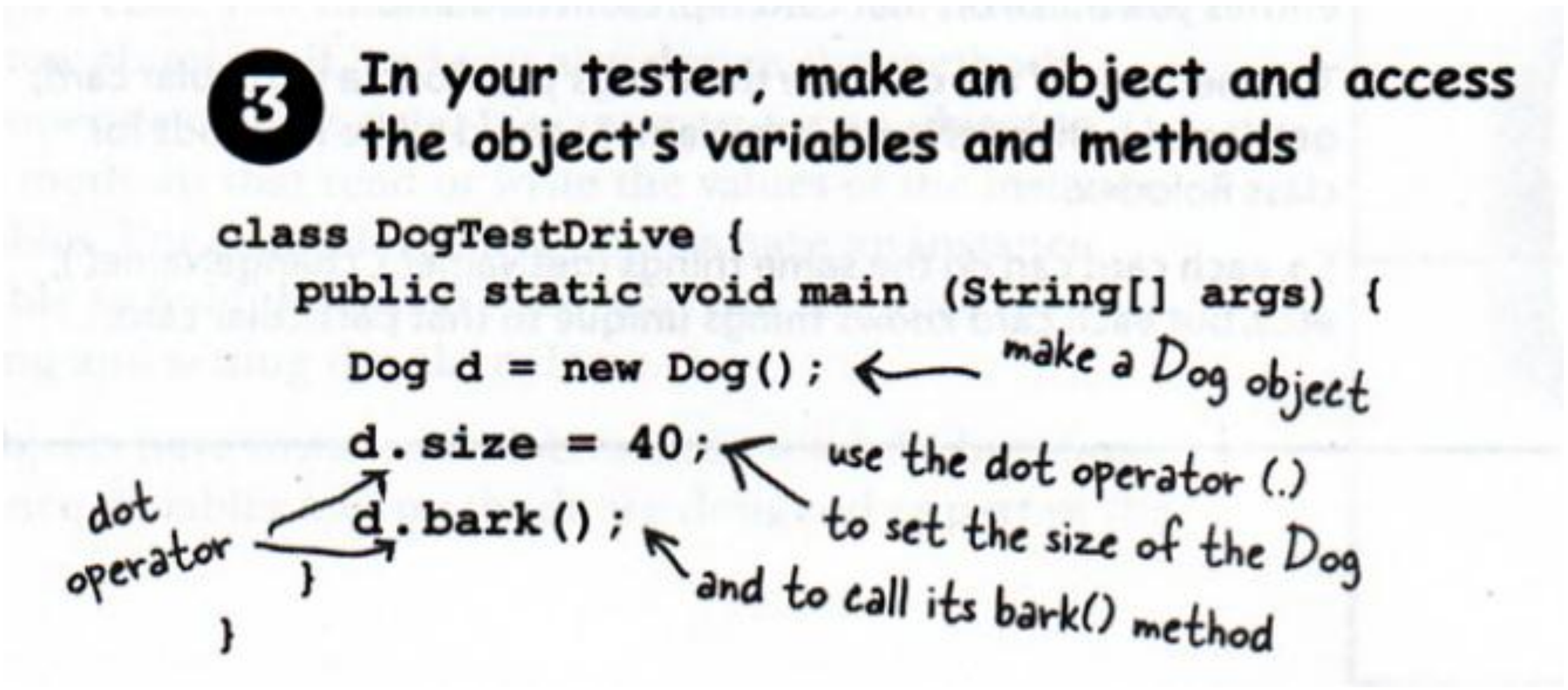




2 Write a tester (TestDrive) class

just a main method
(we're gonna put code
in it in the next step)

```
class DogTestDrive {
  public static void main (String[] args) {
    // Dog test code goes  here
  }
}
```



3 **In your tester, make an object and access the object's variables and methods**

```
class DogTestDrive {
   public static void main (String[] args) {
     Dog d = new Dog();
     d.size = 40;
     d.bark();
   }
}
```

make a Dog object

use the dot operator (.) to set the size of the Dog

and to call its bark() method

dot operator

Still Confused???

We'll Continue Next Week: JAVA and Class!!!

Is there any interview???