**YouTube tutorial 57 – Overriding rules**

Whenever you override a method, you must have the same argument. For example, if the method “eat()” don’t take any variables, then the baby classes also can’t take any variables.

It has to return the same thing too. If “food” class had to return an integer, same goes for “potpie” and “tuna” classes.

Bucky explains how “food” class is too generic, and there’s no need to make objects on it. That’s why Bucky makes the “food” class abstract, which makes it unable to make objects from.

**YouTube tutorial 58 – Abstract and concrete classes**

The abstract is only useful for polymorphism or inheritance. The classes without *abstract* are called concrete classes.

Methods can be abstract too. Those methods are methods that **needs to be overridden**. For example:

**4th class – food.java:**

Abstract public class food{

public abstract void eat();

// It doesn’t need a body.

}

**3rd class – tuna.java:**

Public class tuna extends food{

void(){

System.out.println(“This tuna is good”);

}

}

**Important notes:**

* Note that in order to make a method abstract, the class has to be abstract too.
* The “tuna” class and “potpie” class **MUST** use or override the eat method.