1. Wie unterscheiden sich Objekte und Klassen bzw. wie stehen diese zueinander in Beziehung?

Class:

A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties or methods that are common to all objects of one type. In general, class declarations can include these components, in order:

* Modifiers: A class can be public or has default access .
* Class name: The name should begin with a initial letter (capitalized by convention).
* Superclass (if any): The name of the class’s parent (superclass), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.
* Interfaces (if any): A comma-separated list of interfaces implemented by the class, if any, preceded by the keyword implements. A class can implement more than one interface.
* Body: The class body surrounded by braces, { }.

Object:

It is a basic unit of Object Oriented Programming and represents the real life entities. A typical Java program creates many objects, which as you know, interact by invoking methods. An object consists of:

* State: It is represented by attributes of an object. It also reflects the properties of an object.
* Behaviour: It is represented by methods of an object. It also reflects the response of an object with other objects.
* Identity: It gives a unique name to an object and enables one object to interact with other objects.

When an object of a class is created, the class is said to be instantiated. All the instances share the attributes and the behavior of the class. But the values of those attributes, i.e. the state are unique for each object. A single class may have any number of instances.

1. Was sind Instanzvariablen bzw. Instanzmethoden?

Attributes and methods are different for each object created. So, if you change the attributes of one object the other objects stay the same. Same goes for methods because they mostly use or change the attributes.

1. Was sind Klassenvariablen bzw. Klassenmethoden?

Class variables and methods are declared using the static keyword. This means they can be accessed without creating an Object of this class.

1. Was sind Konstruktoren, wie sind diese gekennzeichnet, und wozu dienen sie?

A constructor gets called whenever an Object is created. With it you can set initial values. The constructor name must match the name of the class.

1. Hat jede Klasse einen Konstruktor?

All classes have constructors by default: if you do not create a class constructor yourself, Java creates one for you. However, then you are not able to set initial values for object attributes.

1. Wozu dient das Keyword this?

With the Keyword this we refer to the reference of the object. Also, we can use it to differ between the attributes of the class or parameters with the same name. e.g.: this.name = name is correct because we refer to the attribute name not the parameter.

1. Was bedeutet null?

It means that there is no value associated with it.

5)

• Welche Vorteile bringt diese Lösung?

The Id’s are fixed via the class.

• Könnte long nextId hier auch als Instanzvariable realisiert werden?

No because then every object would have the same Id.

• Warum sollten nextId und getNextId() hier nicht public sein?

They shouldn’t be public so you can’t edit them without creating an object.

• Könnte man in der Methode private static long getNextId() direkt (ohne neues/konkretes Objekt) auf Instanzvariablen der Klasse Vehicle zugreifen?

No you can’t because the static methods and attributes belong to the class not the objects.