Gremaud Lucien
Paunoiu Bruno

IN.2028
March 5, 2019

Serie 2

1 Theory

a) Integer is an object types and int is a primitive type. The difference is that object types are reference to objects which means that:

- Two variables can represent the same object.
- The affectation only create a copy of the reference.
- The comparison operator only compares the references.

b) A disadvantage to an object having multiple reference to itself would be that if a change is made through one of its reference it might cause trouble for the other reference since the object will be modified for each of them.
An advantage would be that it can be more efficient to make a reference to an already existing object than destroying and recreating a new object.

c) The comparison operator "==" compares two object references and not their field's value. That is why if we compare two String together it will simply return false unless the two string have the same reference.

d) "null" represents the lack of object it will therefore return false if compared to any type of object. It might be useful to check if a reference points to no object at all.
"NaN" represents a value that is not a number. If we directly compare this to any object, it will always return false, even if compared to itself. That is why we should always use the method isNaN() to check whether our variable is a number or not.

e) The method hashCode() should also be overwritten.

f) The equals() method might return an undesired value. For example if we wanted to compare if two cars were the same, we need to know whether they are of the same manufacturer and same version. We don't need to know if their fuel tanks are both empty or full, or if their owner is the same. If we used equals(), the method would return true only if all of those object fields are the same.

g) It is used to invoke another constructor of the same calls inside a constructor.

h) A static attribute is shared among every object of the class, an non-static one is not.

i) Since a static method is shared among every object of the class, the non-static attribute of an object could be affected in undesirable ways.

j) Factory Methods:

1. A factory method is a method inside a class that is called in order to create an object of this class.

2. The advantage of the factory methods is that an object can be created in many different ways if we have multiple factory methods declared. Also we can create an object without giving access to the constructor.
3. To be more lenient in the way of creating an object of the class.
4. As many as we want.
5. The class constructor should be set to private.

3 Debugging

a)
```
Static member 'week2.src.ex3.Car.NUMBER_OF_WHEELS' accessed via
    instance reference
```

The field NUMBER_OF_WHEELS is static and we are accessing it through the object. We can fix this by accessing it through the class like this Car.NUMBER_OF_WHEELS

b)
```
Error:(21, 28) java: non-static variable this cannot be
    referenced from a static context
```

We are accessing a private object field with a public static method, which is not possible.

c) We could make the method printModel non-static.

d) In the equals method we are comparing the String model with == so only the references are compared.

e) We should use model.equals(car.model) to compare them.

f) 74783336

4 Bonus

a) We can have as many as we want, as long as we dont have 2 constructors that takes the same types as arguments and in the same order.

b) If we have a class that requires the attributes to have default values. We can replace the default constructor with a constructor that takes no arguments but that uses another constructor with default values.

c) The default privacy level is private package, which means that any other class from the same package can access its attributes and methods. It is similar to the protected level but the protected level also allows any other subclass to access the attributes and methods of the class.

d) Encapsulation is a technique used to keep data unchangeable by other users of the class. An example of this would be if we make a class package for someone else and we make every field private in order to not let him change them (by mistake) if it isn't necessary. We can then make public methods that can change some of the private fields if it really is necessary for the end user.

e) We can never be sure if an object was destroyed. We can only know when a variable is eligible to be collected by the garbage collector. This is because it is an automated function that we do not control. It is not deterministic at all.

f) If there is no way to access any of the 2 objects, both of them are eligible to be collected by the garbage collector.