# Vragen op sollicitaties

## wat je al na JPF zou moeten kunnen beantwoorden:

* Wat is overerving?
* generics (>< raw types)
* boxing?
* Waarom best primitieve types ipv Object types?
* Collections? En noem er een paar... wat zijn de verschillen?
* synchronised?
* try catch finally: finally wat doet dat en moet dat er altijd staan?
* Wat is normalisatie? En denormalisatie?
* List<Object> iets zoeken op titel (Java 8: lambda expressions)
* wat betekent: iets.equals(“” + anderIets) waarbij anderIets een Integer is? En zou jij dit doen in je code?
* Wat is het verschil tussen Java SE en EE?

EE = SE + web, beans, ...

* hoe maak je een class immutable?
* wat is er nieuw in Java 8? ( lambda’s & streams, default methods in interfaces, nieuwe LocalTime en LocalDate API, …)
* Abstract class en interface (Java 8 is een game changer op dit gebied, zie documentje...)  
  (pre java 1.8 : )
  + wat is het verschil en wanneer gebruik je wat?  
    class inheritance defines implementation  
    ><  
    interface inheritance (subtyping): common type, describes usage
  + Voor welke reden(en) kies je voor een abstract class of voor een interface?  
    abstract class: implementation (class inheritance = code and representation sharing)  
    interface: when can an object be used instead of another (object’s type: the set of requests to which it can respond)

## ‘latere’ vragen:

* wat zou je doen met een constructor met veel parameters? (Builder-pattern)
* SQL: verschil tussen left join en inner join?
* vertel iets over: load testing, integration testing, unit testing
* Waarvoor schrijf je allemaal unit-tests?
* Scrum?
* responsive webdesign
* media queries
* design patterns: noem er een paar op (en dan mag je er een van degene die je opnoemde uitleggen) MVC?, DTO?
* orm: ken je een andere dan hibernate? (bv. EclipseLink)
* Wat is spring? Wat doet dat, waarvoor wordt dat gebruikt?
* GIT?
* Expression language?
* AJAX?
* REST services? HTTP methodes?
* verschil tussen request en session scope

# OCA certificaat (dit zou je na JPF al moeten weten)

* method / constructor overloading
* switch alleen int (geen float,... byte, ... long) (en natuurlijk ook String etc. maar hier even alleen over getallen)
* static variabele
* exception hiërarchy (checked/ unchecked)
* strings & stringbuilder
* equals vs ==
* overerving
* polymorfism
* access modifiers
* pre and post increment & ...
* ...

Wat voorbeelden van vraagjes / testjes.

+ What is the output?

* String s1 = “ Hello “;  
  s1.trim().concat(“ everyone);  
  System.out.println(s1);
* int b = 0;  
  if (1 == 2 && 2 = ++b){}  
  System.out.println(b);
* int b = 0;  
  if (1 == 2 & 2 = ++b){}  
  System.out.println(b);

+ How many objects are created?

* MyClass o1 = new MyClass();  
  Myclass o2 = null;  
  MyClass o3 = o1;

+ Wat is voor jou het grootste verschil tussen een Set en een List?

+ Shallow copy vs deep copy?

+ public void mijnMethode(String naam) {  
 if (“”.equals(naam) ...  
 of  
 if (naam.equals(“”)) ...  
 }

+ Verschil tussen method overriden en method overloaden.

+ Welke zijn correcte manieren om een method te overloaden?

* public void doeIets(String naam) { ... }
* public int doeIets(String naam) { ... }
* private void doeIets(String naam) { ... }
* public void doeIets(String naam, Integer getal) { ... }

Denk aan controle op null als je een parameter binnen krijgt in een methode of constructor.

Denk aan equals en hashCode en aan toString (best practices).

Fout meldingen in je IDE vertellen (meestal) waar het probleem zit. En leer werken met een IDE (auto aanvullen, short cut keys, project properties, ...).

**Je vindt online allerlei testjes voor Java. let wel op dat dit niet altijd goede code is maar wil testen of je iets kent van een onderwerp / een program flow kan inschatten.**

**JPF: hfdstk 14 en 15: multithreading en swing:** lezen en oefeningen doen & stoppen, later zeker naar multithreading terug kijken voor solliciteren (swing indien gevraagd, swing zal zelden een hefboom naar werk zijn vandaar niet teveel tijd insteken).