

# Exposés

Toutes les personnes d'un groupe de TD doivent être passées à la fin des 8 séances de présentations.

Les groupes de présentation doivent être homogènes (3 à 4 personnes). À Chaque groupe de TD sa page qui doit être tenue à jour.

## Problèmes

### 1- pour le TP2

Testez et expliquez la classe Dice du TP1 (vous pourrez vous aider du corrigé). En particulier :

- ses constructeurs (utilisation d'un constructeur et utilité d'en avoir plusieurs),
- le fonctionnement de la méthode *void roll()*
- utilisation et utilité des méthodes générées (*toString*, *equals* et *hashCode*)
- que sont des *setter* et des *getter* ?

### 2- pour le TD1

En Java, il y a en gros deux façons de stocker un groupe d'objets :

- les tableaux
- les conteneurs, en particulier la classe *ArrayList*.

On vous demande de comparer les deux façons de faire. Cette fois ci, c'est à vous de trouver l'information sur le net. Donnez les urls que vous avez utilisé pour votre présentation.

On vous demande en particulier :

- d'expliquer ce qu'est un tableau d'objets (création et utilisation),
- d'expliquer ce qu'est une *ArrayList* (création et utilisation),
- de montrer les différences entre les deux façons de faire. Quand préférer l'une par rapport à l'autre ?
- Une *ArrayList* est souvent associée au terme *generics* en Java. Explicitez en 2 mots ce terme.

### 3- pour le TP3

Gestion des constantes en Java.

On vous demande :

- D'expliquer une des règles fondamentales en programmation concernant les constantes : *no magic number*.
- De montrer les 2 grandes façons de gérer des constantes en Java :
  - utilisation du mot clé *final*

- utilisation des enum

## 4- pour le TD2

L'héritage en Java.

Tout ce dont vous avez besoin est disponible en suivant les 2 ressources suivantes :

- <http://www.loria.fr/~abelaid/Enseignement/FC/Cours5-heritage.pdf>
- <https://docs.oracle.com/javase/tutorial/java/landl/subclasses.html>

Il faudra en particulier expliciter :

- à quoi sert l'héritage
- comment on l'utilise en java (mots clés extends et protected)
- utilisation du mot clé super (constructeur et autres méthodes)

## 5- pour le TP4

Méthodes et attributs de classes.

Il faudra en particulier expliciter :

- les différences avec les attributs et méthodes classiques (utilisés pour les objets)
- le mot clé `static`
- l'utilisation de méthodes/attributs de classes pour :
  - la gestion des constantes
  - le design pattern factory (qu'est-ce ? Comment on l'utilise ?)
  - On pourra prendre comme exemple la classe `Color` en javafx (voir TP3)
- pourquoi est-ce que la méthode `main` (le programme principal) est `static` ?

## 6- pour le TD3

Les interfaces en Java.

Une interface, à ne pas confondre avec une UI, est un moyen de gérer les dépendances entre classes. L'héritage gère les dépendances de code, les interfaces les dépendances de fonctionnalités. Voir le cours d'oracle [sur les interfaces](#).

On vous demande de :

- chercher et lire de la documentation sur les interfaces,
- comprendre la syntaxe d'une interface,
- de montrer à quoi cela peut servir,
- de les différencier de l'héritage.

## 7- pour le TP5

Notions de *design pattern*.

On vous demande d'expliquer en quelques phrases ce qu'est un design pattern en conception objet. Vous vous attarderez ensuite sur un design pattern spécifique : le design pattern strategy. En particulier :

- vous motiverez son utilisation,
- vous donnerez et explicitez son diagramme UML et des éléments de son implementation en Java,
- vous donnerez quelques exemples pratiques d'utilisation.

## 8- pour le TP6

Programmation événementielle

On vous demande d'expliquer ce qu'est la programmation événementielle en informatique. Vous pourrez vous aider de documents comme :

- la [documentation de javafx 8](#)
- un [cours assez general](#) mais qui met en avant le design pattern observer très utile
- un [cours orienté javafx](#)

From:

<https://wiki.centrale-med.fr/informatique/> - **WiKi informatique**

Permanent link:

<https://wiki.centrale-med.fr/informatique/public:mco-2:expose>

Last update: **2016/04/29 08:31**

