# Cours de programmation orientée-objet
# Examen du 8 juin 2011

*Livres fermés. Durée : 3 heures 1/2.*

*Veuillez répondre à chaque question sur une feuille séparée sur laquelle figurent nom, prénom et section. Soyez bref et concis, mais précis.*

1. Dans le cadre de la programmation d'un agenda, une équipe désire disposer d'une classe `Evenement`. Un événement est caractérisé par une brève description textuelle ainsi que par une date. Une date est composée de trois nombres naturels représentant un jour, un mois et une année. La description est une chaîne de caractères.

   La consultation étant une fonctionnalité très importante de l'agenda, on désire mettre au point un système de filtrage. Un filtre est un objet possédant une méthode `filtre` qui reçoit en argument une instance de la classe `Evenement` et retourne `true` ou `false` selon que cet événement satisfait ou non son critère de filtrage.

   Les filtres pouvant être de différentes natures, la classe `Filtre` est une classe abstraite qu'il convient de spécialiser pour définir un filtre particulier.

   On demande de réaliser les actions de développement suivantes en Java :

   (a) Donner une implémentation des classes `Evenement`, `Filtre` et `FiltreDate`. Cette dernière est une spécialisation de la classe `Filtre` permettant de sélectionner les événements se déroulant à une certaine date. Cette date est spécifiée lors de l'instanciation du filtre.

   (b) Une méthode `filtreEnsemble` ayant l'interface suivante :
   `Vector filtreEnsemble(Vector ensemble, Filtre f)`.
   Cette méthode a pour but de renvoyer une nouvelle instance de `Vector` contenant toutes les références des événements présents dans le vecteur passé en premier argument et qui satisfont le filtre `f` passé en second argument. ***Note*** *: Si le vecteur référencie des instances de classes autres que* `Evenement`*, il conviendra de le signaler au moyen d'une exception.*

   La classe `Vector` est une classe présente dans le package `java.util`, dont voici les principaux éléments d'interface :
   - `Vector()` : est un constructeur permettant d'instancier un vecteur vide.
   - `void add(Object x)` : ajoute l'élément `x` au vecteur.
   - `int size()` : renvoie le nombre d'éléments présents dans le vecteur.
   - `Object elementAt(int i)` : permet de récupérer l'élément d'indice `i` dans le vecteur (le premier élément a l'indice 0).

2. (a) Si `i1` et `i2` sont deux variables de références, quelle est la différence entre les instructions `i1 == i2` et `i1.equals(i2)` ?

   (b) En Java, à quoi peut servir une interface vide telle que `Cloneable` ?

   (c) Citez deux situations où l'utilisation du mot clé `this` est requise en Java.

3. Voici le code d'une classe Java :

```
public class Rationnel implements Cloneable
 {
   private int n, d;
   public Rationnel(int n, int d)
     {
       this.n = n;
       this.d = d;
     }

   public int getNumerateur() { return n; }

   public int getDenominateur() { return d; }

   public Object clone()
     {
       Rationnel copie = new Rationnel(n, d);
       return copie;
     }
 }
```

   (a) L'implémentation de la méthode `clone()` est incorrecte. Expliquez pourquoi.

   (b) Donnez un scénario (sous la forme d'un petit programme Java) qui montre une situation problématique.

   (c) Corrigez la méthode `clone()`.