# Programmation Orientée Objet

## TP d'introduction, première partie – Soyons rationnel (n'ayons l'air de rien)

Ensimag 2ème année

**Résumé**

L'objectif de ce TP est de découvrir les bases de la programmation orientée objet avec Java. Nous nous attacherons à construire et à manipuler une classe très simple pour mettre en pratique les notions de classe, d'objet, d'attribut, de méthode, de rétention d'information (*encapsulation* en anglais), de constructeur, ...

Vous connaissez probablement tous la définition de l'ensemble $\mathbb{Q}$ des rationnels. Un nombre rationnel est un nombre qui peut être défini par un ratio $n/d$, où $n$ est un entier, et $d$ est un entier non nul. Nous allons dans ce TP construire une classe représentant les nombres rationnels.

## 1 On commence doucement

**Question 1** Première version d'une classe `Rational`

**1.a -** Créez une classe `Rational`, possédant deux attributs entiers `num` et `denom`.

**1.b -** Écrivez un programme de test qui crée une fraction $3/2$, affiche son numérateur, et affiche son dénominateur (dans la console).

**1.c -** Compilez tout ça avec le compilateur `javac`, et exécutez avec la machine virtuelle `java`.

**Question 2** Représentation textuelle d'un rationnel

**2.a -** Modifiez votre classe `Rational` afin d'y ajouter une méthode `toString()` sans paramètre, et renvoyant une chaîne de caractères (type `String`) représentant le nombre rationnel (par exemple, cette méthode, appelée sur le rationnel $3/2$, devrait renvoyer la chaîne de caractères `"3 / 2"`).

**2.b -** Modifiez votre programme de test pour utiliser la méthode `toString()` créée précédemment. Compilez et exécutez.

## 2 Encapsuler pour régner

**Question 3** Modifiez votre programme de test pour créer un autre rationnel de dénominateur égal à 0. Est-ce possible ? Si oui, en quoi est-ce un problème ?

**Question 4** Proposez une solution pour empêcher la *création* et la *manipulation* d'objets rationnels ayant pour dénominateur 0. Modifiez votre classe `Rational` en conséquence.

## 3 Des opérations arithmétiques

**Question 5** Pour rappel (au cas où...), la multiplication de deux rationnels est définie de la manière suivante : $\frac{n}{d} \times \frac{n'}{d'} = \frac{nn'}{dd'}$.

**5.a -** Ajoutez à votre classe `Rational` une méthode `mult` prenant en paramètre un autre rationnel, et le multipliant au rationnel sur lequel elle est appliquée. La méthode ne renvoie rien (type de retour `void`).

Cette opération modifie l'état de l'objet sur lequel la méthode est invoquée : `a.mult(b)` revient à

faire $a \leftarrow a * b$. Il n'est pas possible en Java de redéfinir des opérateurs (comme en C++), sinon ce serait équivalent à `a *= b`.

**5.b -** Modifiez votre programme de test en créant un second rationnel (1/3 par exemple), en le multipliant au premier, et en affichant le résultat. Compilez et exécutez.

**Question 6** Mêmes questions pour l'addition. Pour rappel, l'addition de deux rationnels est définie de la manière suivante : $\frac{n}{d} + \frac{n'}{d'} = \frac{nd'+n'd}{dd'}$.

# 4 Des fractions irréductibles

**Question 7** En plus d'interdire la création et la manipulation de rationnels dont le dénominateur est à 0, nous souhaitons que tout rationnel soit toujours sous sa forme irréductible. Il s'agit d'un *invariant de classe* : à tout instant, dès sa création, un objet `Rational` est toujours sous forme irréductible.

Pour rappel, un rationnel $\frac{n}{d}$ est sous forme irréductible si et seulement si $\text{pgcd}(n, d) = 1$, avec :

```
fonction pgcd(a,b)
    si b est égal à 0 renvoyer a
    sinon renvoyer pgcd(b, reste(a,b))
```

Modifiez votre classe `Rational` en conséquence (ou créez un nouvelle classe `ReducedRational`). Testez.

# 5 Pour aller plus loin...

Revenons sur les opérations arithmétiques.

**Question 8** Une autre méthode (surcharge) de multiplication pourrait avoir la signature suivante : `public Rational mult(Rational a, Rational b)`.

- proposez une spécification « logique » de cette méthode (rôle des paramètres, valeur retournée, ...).
- comment s'utilise-t-elle ? Que dire de l'objet sur laquelle la méthode est invoquée ?
- renseignez-vous sur les notions de *méthodes de classes* (par rapport aux *méthodes d'instances*), et le mot-clé Java `static` - par exemple au moyen des documents suivants :
  - Fiche de synthèse #1, Section 8
    https://programmation-orientee-objet.pages.ensimag.fr/poo/resources/fiches/01-ClassesEtObjets/#attributs-et-methodes-de-classe
  - Tutoriel Java officiel
    https://docs.oracle.com/javase/tutorial/java/javaOO/classvars.html.