

Question 1 : Pourquoi est-ce qu'il n'y a pas de méthodes `setId` et `setYearOfBirth` ?

Les deux attributs `id` et `yearOfBirth` étant `final`, il n'est pas possible de les modifier après la création de l'objet. Il est donc impossible de définir des méthodes modifiant les valeurs de ces deux attributs.

Question 2 : Quel est le rôle du mot-clé `static` dans la définition de `studentCount` ?

Le mot-clé `static` sert à indiquer que `studentCount` est un attribut de classe et non d'instance. Le nombre d'étudiants créés est une notion qui ne dépend pas d'une instance mais de la classe.

Question 3 : Pourquoi l'identifiant `FIRST_ID` est écrit en majuscules (`SNAKE_CASE`) ?

C'est la norme en java d'écrire les constantes de classe (déclarées avec `static final`) en `SNAKE_CASE`, c'est-à-dire avec que des majuscules et des `_` pour séparer les mots.

Question 4 : Pourrait-on rajouter l'instruction `this.id = String.valueOf(FIRST_ID + studentNumber)` dans la méthode `toId` sans faire aucune autre modification ?

Ce n'est pas possible car la méthode `toId` étant `static` (donc liée à la classe et pas à une instance), elle ne peut pas accéder à un attribut d'une instance courante avec `this`.

Question 5 : Quel est l'affichage produit par l'exécution de `java StudentApp` ?

L'affichage est le suivant :

```
David Goodenough (1976) : 21000000  
Jean-Michel Bruitage (1989) : 21000001
```

Question 6 : Donner le code de `boolean equals(Object o)` qui renvoie `true` si `o` est une instance de `Student` (`o instanceof Student` vaut `true`) et qu'il a le même `id` que `this`.

```

public boolean equals(Object o) {
    if (this == o) return true;
    if (!(o instanceof Student)) return false;
    Student student = (Student) o;
    return id.equals(student.id);
}

```

Question 7 : Donner le code de la méthode `int compareTo(Student s)`.

```

public int compareTo(Student s){
    int compareLastName = this.lastName.compareTo(s.lastName);
    if(compareLastName != 0)
        return compareLastName;
    int compareFirstName = this.firstName.compareTo(s.firstName);
    if(compareFirstName != 0)
        return compareFirstName;
    int compareYearOfBirth = s.yearOfBirth - this.yearOfBirth;
    if(compareYearOfBirth != 0)
        return compareYearOfBirth;
    return this.id.compareTo(s.id);
}

```