amU Faculté des sciences Aix Marseille Université

Année 2025-26

Domaine Sciences et Technologies
Licence Informatique : math-info
Initiation au Génie logiciel : TD 6
Codes UE : SIN5U34
TD 6 : Étudiants

Classe `Student`

Dans cette exercice, on va travailler sur la classe `Student` ci-dessous qui permet de créer des étudiants.

```
public class Student {
  private static int studentCount = 0;
  private static final int FIRST_ID = 21000000;

  private static String toId(int studentNumber){
    return String.valueOf(FIRST_ID + studentNumber);
  }

  private final int yearOfBirth;
  private final String id;
  private String firstName;
  private String lastName;

  public Student(int yearOfBirth, String firstName, String lastName) {
    this.id = toId(studentCount++);
    this.yearOfBirth = yearOfBirth;
    this.firstName = firstName;
    this.lastName = lastName;
  }
  public int getYearOfBirth() {
    return yearOfBirth;
  }
  public String getId() {
    return id;
  }
  public String getFirstName() {
    return firstName;
  }
  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }
  public String getLastName() {
    return lastName;
  }
  public void setLastName(String lastName) {
    this.lastName = lastName;
  }
  public String toString() {
    return firstName + " " + lastName + " (" + yearOfBirth + ") : " + id;
```

```
  }
}
```

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

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

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

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

```
public class StudentApp {
  public static void main(String[] args){
    Student s1 = new Student(1976, "David", "Goodenough");
    Student s2 = new Student(1989, "Jean-Michel", "Bruitage");
    System.out.println(s1);
    System.out.println(s2);
  }
}
```

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

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`.

On souhaite coder une méthode `int compareTo(Student s)` qui renvoie :

- un entier négatif si `this` est avant `s` pour l'ordre
- `0` si `this` et `s` sont considérés égaux pour l'ordre
- un entier positif si `this` est après `s` pour l'ordre

La classe `String` implémente une méthode `int compareTo(String s)` avec l'ordre lexicographique (`"aa".compareTo("ac")` renvoie `-2`). On dit qu'une instances de `Student s1` est avant `s2` si :

— le nom de famille de `s1` est avant `s2`
— les noms de familles de `s1` et `s2` sont les mêmes et le prénom de `s1` est avant `s2`
— `s1` et `s2` ont les mêmes prénoms et noms et `s1` est plus jeune que `s2`
— Dans le cas où les attributs `firstName`, `lastName` et `yearOfBirth` sont les mêmes pour `s1` et `s2`, l'ordre est défini par les `id` (plus petit `id` avant dans l'ordre).

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