Contrôle intermédiaire 2 - 2017

Durée 2 h.

Programme : Tout le début du cours jusqu'à la généricité (compris).

Documents autorisés : Supports de cours distribués pendant le cours (mais **les TPs et corrections de TPs ne sont pas autorisés**).

Exercice 1 :

Vous allez écrire des classes `Personne`, `Etudiant` et `Enseignant` dans le paquetage `fac.personne`.

Un étudiant a un nom et des notes. Une note est représentée par un `double`. Vous utiliserez une collection (pas un tableau) pour conserver les notes des étudiants.

Un enseignant a un nom et une adresse (uniquement la ville sous la forme d'une `String`).

`Etudiant` et `Enseignant` héritent de `Personne`.

La méthode `main` suivante doit pouvoir s'exécuter avec les classes que vous allez écrire. N'oubliez pas d'écrire dans vos classes tout ce qui est nécessaire, **en particulier pour la boucle « for-each »**. Dans vos classes, n'ajoutez rien d'autre que ce qui est indispensable à l'exécution de cette méthode `main`.

```
public static void main(String[] args) {
  Etudiant etudiant = new Etudiant("Bob");
  etudiant.ajouterNote(10.5);
  etudiant.ajouterNote(12);
  Enseignant enseignant = new Enseignant("John", "Paris");
  System.out.println("Les notes de " + etudiant.getNom() + " : ");
  for (double note : etudiant) {
    System.out.println(note);
  }
  System.out.println("Adresse de " + enseignant.getNom() + " : "
    + enseignant.getAdresse());
}
```

Est-ce qu'on aurait pu écrire

« `Personne etudiant = new Etudiant("Bob");` » ?

Pourquoi ?

Suite au verso de la feuille

Exercice 2

Ecrivez une classe `Universite` dans le paquetage `fac`. Une université a un nom de type `String`. La classe a un constructeur et les méthodes suivantes :

- `ajouterPersonne` pour ajouter un étudiant ou un enseignant dans l'université. Vous utiliserez une Map pour enregistrer l'étudiant ou l'enseignant dans l'université. Cette Map utilisera le nom comme clé.
  Une exception `IllegalArgumentException` est lancée s'il existe déjà une personne avec le même nom dans l'université. Rappel : `IllegalArgumentException` est une exception qui n'est pas contrôlée par le compilateur. Le message associé à cette exception sera le suivant :
  <<nom université>> a déjà une personne nommée <<le bon nom>>
- `getPersonnes` qui retourne une collection de tous les enseignants et étudiants de l'université.
- `getEtudiants` qui retourne une liste des **étudiants** de l'université (type retour `List<Etudiant>`. Aide : vous aurez besoin de `instanceof` (pour savoir si une personne est un étudiant) et d'un cast.

Où avez-vous utilisé la généricité dans votre code ? Expliquez pourquoi vous avez choisi ces paramètre de type.

Exercice 3

Ecrivez une classe `Main` avec une méthode `main` qui crée une université et ajoute un enseignant et un étudiant dans l'université.

La méthode `main` ajoute un 2ème étudiant qui a le même nom que l'enseignant et elle affiche le message d'erreur associé à l'exception lancée par `ajouterPersonne`.

Elle affiche ensuite le nom de tous les enseignants et étudiants ajoutés dans l'entreprise (un nom par ligne).