Laboratoire no. 6

Objectifs

- Pratiquer les associations bidirectionnelles
- Prendre conscience des problèmes de synchronisation que posent ces associations

Donnée

On considère le diagramme de classes UML suivant :

1. Compléter la classe Etudiant du diagramme de classes ci-dessus de sorte que celle-ci permette de :
   a. construire un objet de type Etudiant
   b. obtenir le nom de l'étudiant
   c. obtenir le groupe auquel appartient l'étudiant
   d. fixer le groupe auquel appartient l'étudiant

2. Compléter la classe Groupe du diagramme de classes ci-dessus de sorte que celle-ci permette de :
   a. construire un objet de type Groupe
   b. obtenir le no du groupe
   c. obtenir la liste des étudiants du groupe
   d. ajouter un ou plusieurs étudiants au groupe
   e. supprimer un ou plusieurs étudiants du groupe
   f. supprimer tous les étudiants du groupe
   g. transférer un ou plusieurs étudiants du groupe vers un autre groupe
   h. transférer tous les étudiants du groupe vers un autre groupe

3. Implémenter en Java les classes Etudiant et Groupe

4. Implémenter un programme de test (pertinent !) permettant de vérifier :
   a. le bon fonctionnement des méthodes des classes Etudiant et Groupe
   b. la préservation de la cohérence des données

Prescriptions / indications

- **IMPORTANT !**
  Coder de sorte à préserver en tout temps la cohérence des données
  Exemple : Si l'étudiant e se voit affecté au groupe g, il faut "symétriquement" ne pas oublier d'ajouter l'étudiant e dans la liste des étudiants du groupe g.
- Faire en sorte que toString() appliqué sur l'étudiant appelé "Toto", retourne :
  - la chaîne "`Toto`", si Toto n'est associé à aucun groupe
  - la chaîne "`Toto(g`*n*`)`", si Toto appartient au groupe numéro *n*
- Faire en sorte que toString() appliqué sur le groupe numéro *n*, retourne :
  - la chaîne "`g`*n* `: []`", si le groupe ne contient aucun étudiant
  - la chaîne "`g`*n* `: [E1(g`*n*`), E2(g`*n*`)]`", si le groupe comprend, par exemple, les étudiants appelés E1 et E2
- Il n'est pas demandé ici :
  - de commenter systématiquement "à la Javadoc" le code Java (quelques commentaires "classiques" sont toutefois, bien sûr, les bienvenus !)
  - de vérifier la validité des paramètres passés aux constructeurs.
- En ce qui concerne LinkedList : voir documentation de l'API Java (paquetage java.util) et chap 16 du cours.
- Convention :
  Aucun constructeur et aucune méthode (de nos 2 classes Groupe et Etudiant) ne doit lever d'exception

A réaliser

☐ Seul
☒ Par groupe de 2

Travail à rendre **le 07.12.2017, au début de la séance de laboratoire**

☒ Fiche de laboratoire (sur papier)
☒ Copie papier du diagramme de classes UML réalisé avec StarUML
☒ Listings des fichiers source java (imprimés avec NotePad++ ou équivalent)
☒ Fichiers sources uniquement, dans :
\\eistore1\cours\tic\RRH\POO1\Rendus\<votre répertoire>\Labo_6
où <votre répertoire> = répertoire du membre du groupe venant en premier dans l'ordre alphab.