



## Exam 2018

Programmation (École Marocaine des Sciences de l'ingénieur)

**Problème :**

Nous souhaitons réaliser une petite application de gestion des PFE. Dans cette application un encadrant encadre un ensemble de PFE où chaque PFE est attribué à un groupe de 3 étudiants au maximum. Un des encadrant est aussi responsable de l'affectation des PFE et de leur gestion. Cette application est composée de 4 classes (Etudiant, PFE, Encadrant, Responsable) comme suit :

```
public class Etudiant implements Comparable<Etudiant>{ {
    private String cne;
    private String nom;
    private PFE projet;
    //constructeurs, getters & setters sont considérés fournis
    @Override
    public int compareTo(Etudiant etudiant) {}
}
```

```
public class PFE {
    private String sujet;
    private Encadrant encadrant;
    private Set<Etudiant> groupe;
    // getters & setters considérés fournis
    public PFE(String sujet) {}
    public PFE(String sujet, Encadrant encadrant) {}
    public void ajouterEtudiant(Etudiant etudiant) throws MaxEtudiantsDepassé{}
    public void supprimerEtudiant(Etudiant etudiant) {}
    public void remplacerEtudiant(Etudiant etudiant1, Etudiant etudiant2) {}
    public void ajouterGroupe(Set<Etudiant> groupe) throws MaxEtudiantsDepassé{}
    public void viderGroupe()
    public void remplacerGroupe(Set<Etudiant> nouveauGroupe) throws
    MaxEtudiantsDepassé{}
}
```

```
public class Encadrant {
    private String nom;
    private Set<PFE> projets;
    public Encadrant(String nom){}
    public Encadrant(String nom, Set<PFE> projets)
    public Encadrant(Encadrant encadrant){}
    //getters & setters fournis
    public void ajouterProjet(PFE projet){}
    public void proposerProjet(String sujet){}
    public void supprimerProjet(PFE projet){}
    public int NombreEtudiantsParEncadrant(){}}
    public int NombrePFEPParEncadrant(){}}
}
```

```
public class Responsable extends Encadrant{
    private List<Encadrant> listEncadrants;
    //getters et setters fournis
    public Responsable(String nom){}
    public Responsable(String nom, Set<PFE> projets){}
    public Responsable(Encadrant encadrant) {}
    public void echangerEtudiants(Etudiant etudiant1, Etudiant etudiant2) throws
    EtudiantSansPFEException{}
    public void transfererPFE(PFE pfe, Encadrant nouveauEncadrant) throws
    PFESansEncadrantException{}
    public void transfererGroupe(PFE pfeSource, String NouveauSujet, Encadrant
    nouveauEncadrant) throws PFESansGroupeException{}
    public void echangerGroupe(PFE pfeSource, PFE pfeCible){}
    public List<Etudiant> getListEtudiantsTriée(){ }
    public List<PFE> getListProjets(){ }
    public void repartirPFE(Set<PFE> projets, List<Etudiant> etudiants){ }
}
```

**Questions :**

Au niveau de la classe Etudiant, fournir le code de :

1. la méthode **compareTo** qui permet la comparaison de deux objets étudiants.

Au niveau de la classe PFE, fournir le code de:

2. le Constructeur **PFE(String sujet, Encadrant encadrant)**.
3. L'exception **MaxEtudiantsDépassé** .
4. La méthode **ajouterGroupe**.

Au niveau de la classe Encadrant, fournir le code de :

5. le Constructeur **Encadrant(String nom, Set<PFE> projets)**
6. La méthode **supprimerProjet**
7. La méthode **NombreEtudiantsParEncadrant**

Au niveau de la classe Responsable, fournir le code de :

8. Le constructeur **Responsable(String nom, Set<PFE> projets)**.
9. La méthode **echangerEtudiants** qui permet d'échanger deux étudiants appartenant à deux PFE différents.
10. La méthode **transfererGroupe** qui permet de transférer un groupe d'étudiant vers un nouvel encadrant avec un nouveau sujet.
11. La méthode **getListEtudiantsTriée** qui permet de retourner la liste triée par ordre alphabétique de tous les étudiants.
12. La méthode **repartirPFE** qui permet d'effectuer une affectation aléatoire des étudiants, PFE et encadrants.

**Aide :** penser à utiliser la méthode `nextInt(int n)` qui retourne un nombre entier entre 0 et n-1 de la classe `java.util.Random`.