



Génie Logiciel (DSI310)

Examen Final

Date : 19/02/2025

Durée : 1h30

Filière : DSI2

*** Tous les documents sont interdits ***

Exercice 1 : Questions théoriques (5 points)

1. Expliquez la différence entre la *composition* et l'*agrégation* dans le contexte d'une base de données.
2. Donnez un exemple illustrant la *composition* et un autre illustrant l'*agrégation*.
3. Quelle différence entre un acteur principal et un acteur secondaire ?
4. Quelles différences entre les assertions *@BeforeAll* et *@BeforeEach* de Junit?
5. Définir brièvement les trois principes de développement logiciel : *Yagni*, *Kiss* et *Dry*.

exéc

Exercice 3 : Conception (15 points)

Le ministère de l'enseignement supérieur souhaite moderniser l'organisation des examens dans ses différents établissements. Pour ce faire, il met en place une base de données relationnelle permettant de gérer l'ensemble des informations relatives aux examens, depuis l'inscription des étudiants jusqu'à la correction des épreuves. L'objectif est de garantir une organisation fluide et cohérente des sessions d'examens, en s'appuyant sur les données des établissements, des examens, des épreuves et des enseignants.

- Chaque établissement supérieur est caractérisé par un code, un nom et la ville où se situe l'établissement. Dans chaque établissement on a plusieurs examens et plusieurs comités chargés de proposer ou corriger les épreuves des examens. Un établissement peut être une *université*, un *institut* ou une *école*. Pour les universités, on doit connaître également le nombre de facultés qui la compose, son budget annuel, son effectif total des étudiants inscrits et le nombre projets de recherche encours. Pour un institut, on gardera aussi son domaine d'expertise et le nombre de formations. Quant aux écoles, on gardera le type de l'école et le taux d'insertion.
- Chaque examen est constitué d'une série d'épreuves qui lui est propre, chacune dotée d'un id et d'un coefficient. Un examen est caractérisé par son nom, la date de début et la date de fin. Pour faciliter l'organisation, chaque examen n'aura lieu que dans un seul établissement. Bien sûr, par soucis de cohérence de la base, si un examen est annulé pour une raison ou une autre, alors toutes ses épreuves le sont également.
- Chaque épreuve est proposée et est corrigée par un seul comité d'une dizaine d'enseignants. Bien sûr un comité peut proposer et/ou corriger plusieurs épreuves. Les corrections ont lieu le lendemain de l'épreuve. Les épreuves se déroulent donc à la même date dans tous les établissements.
- Un enseignant est connu par son matricule, son nom, son téléphone et son adresse. Un enseignant appartient au plus à un seul comité mais appartient à un seul établissement.
- Chaque comité est identifié par un code unique et n'appartient qu'à un seul établissement. Chaque comité est composé d'une dizaine d'enseignants.

vail demandé :

1. Donner un diagramme de classes correspondant à ce scénario en utilisant les concepts de composition, agrégation et d'héritage dès que possible.

Don fonction
Sud



2. Si un enseignant change de comité, quelles implications cela a-t-il sur la relation entre l'enseignant et le comité dans le modèle ?
3. Que se passe-t-il dans le modèle de données si un établissement est supprimé ? Quels éléments (comités, examens, étudiants, université, institut ou école) doivent être supprimés ?
4. Proposer une structure de la classe *Comité* en java en mettant en évidence les associations.
5. On suppose que la classe *Comité* contient des méthodes
 - a. *getEnseignants()* qui retourne la liste des enseignants d'un comité
 - b. *addEnseignant()* qui ajoute un enseignant dans un comité
 - c. *removeEnseignant()* qui supprime un enseignant d'un comité

On veut écrire une méthode de test *testAddAndRemoveEnseignant* pour tester les méthodes *addEnseignant* et *removeEnseignant*. Complétez la méthode de test en remplaçant ? par le contenu qu'il faut.

```
public class ComiteTest {  
  
    private Comite comite;  
    private Enseignant ens1;  
    private Enseignant ens2;  
    ...  
  
    @BeforeEach  
    public void setUp() {  
        // Création du comité associé  
        comite = new Comite("CMT001", "Comité d'examen", etab);  
        // Création de quelques enseignants  
        ense1 = new Enseignant("ENS001", "Moussa", "45879652", "1 Rue ELHadj Oumar Tall");  
        ense2 = new Enseignant("ENS002", "Cheikh", "35698587", "2 Rue Gamal Abdel Nacer");  
    }  
  
    @Test  
    public void testAddAndRemoveEnseignant() {  
        // Vérifier l'ajout d'un enseignant  
        comite.addEnseignant(ens1);  
        assertEquals(?, comite.getEnseignants().size());  
        assertTrue(comite.getEnseignants().contains(...));  
  
        // on essaie d'ajouter à nouveau le même enseignant, ça ne doit pas être ajouté  
        // deux fois  
        comite.addEnseignant(ens1);  
        assertTrue(=? comite.getEnseignants().size());  
  
        // Suppression d'un enseignant  
        comite.removeEnseignant(ens1);  
        assertEquals(?, comite.getEnseignants().size());  
    }  
}
```

Bonne chance !