Université Larbi ben M'Hidi **15 Jan. 2023**
Faculté des sciences exactes & sciences de la nature et de la vie **Dr. KOUAH Sofia**
Département MI - 3ième Année Licence S.I. -

Examen Final - Génie Logiciel -

Nom & Prénom : **Groupe :**

Exercice 1 (12 Pts) :

On souhaite automatiser la gestion d'une auto-école afin de faciliter les tâches des secrétaires. On considère le fonctionnement suivant :

- Pour sécuriser l'accès au système, tous les acteurs doivent s'authentifier.

- Un élève peut s'inscrire dans une et une seule auto-école. Il faut fournir au secrétaire les renseignements suivants : nom, prénom, date de naissance, adresse, numéros de téléphone.

- Chaque auto-école a ses propres moniteurs. Un moniteur ne peut pas travailler pour deux écoles en même temps.

- Il y a deux types de forfait proposés par les écoles : forfait théorique (code) et forfait pratique. Pour chaque forfait, on connaît le tarif et la durée de validité. Chaque forfait pratique correspond à un nombre fixe d'heures de conduite.

- A la demande d'un élève, le secrétaire peut lui ajouter ou annuler des heures de conduite.

- Pour ajouter des heures de conduite, il faut saisir la date, l'heure de début, la durée, et l'élève peut choisir son moniteur préféré. Si celui-ci est disponible à l'horaire demandé, il est affecté. Sinon, il faut parcourir la liste des moniteurs restants afin de trouver un moniteur libre (qui n'a pas de cours avec un autre élève sur ce même créneau horaire). Si personne n'est disponible, la séance ne sera pas créée. Le tarif correspondant à une séance est calculé selon que l'élève a pris un forfait ou pas. Si l'élève ne prend pas de forfait, ou si le nombre d'heures prévues dans le forfait a été dépassé, il faut payer un prix unitaire plus élevé, qui est fixé par chaque auto-école.

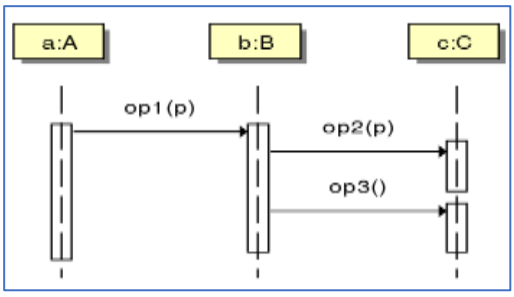
- Pour avoir le permis, un élève doit passer par les étapes suivantes : se former ; passer l'examen de code et s'il réussit cet examen, il peut passer à l'examen pratique, avec une condition d'avoir effectué au moins 20 heures de conduite. Au bout de 3 ans, s'il ne réussit aucun examen pratique, son code sera invalidé.

1) Elaborer le diagramme de classes correspondant (Tous les attributs et les méthodes sont supposés public).
2) Donner le code java de trois classes du diagramme élaboré tout en considérant les associations existantes.
3) Établir le diagramme d'états-transitions pour l'objet de la classe Elève.
4) On suppose que l'auto-école est gérée par un Directeur dont on veut garantir son unicité (instance unique) au sein du système d'autoécole. En utilisant les designs patterns, proposer une solution pour ce problème. Donner le code java correspondant.

Questions de compréhension (8 Pts) :

1. **Répondre à quatre questions parmi les questions suivantes :**
   a) Qu'est-ce qu'un Design Pattern, quel est son objectif ? Citer deux Design Pattern.
   b) Quelle est la différence entre le diagramme d'activités et le diagramme d'états transitions.
   c) Expliquer le rôle du flow final du diagramme d'activités.
   d) Quel est le rôle des partitions dans le diagramme d'activités.
   e) Quelle est la différence entre agrégation et composition ?
   f) Quel est le rôle des fragments combinés dans le diagramme de séquences ?
   g) Quel est le rôle du diagramme de cas d'utilisations ? Citer ses composants de base.
   h) Quelle est la différence entre analyse des besoins et conception ?
   i) Citer trois critères de qualité d'un logiciel. Expliquer un critère parmi les critères cités.

2. **Soit le diagramme suivant :**



Que représente-t-il ? quel est son rôle ? Donner le diagramme de classes qui lui correspond.

Bonne chance !

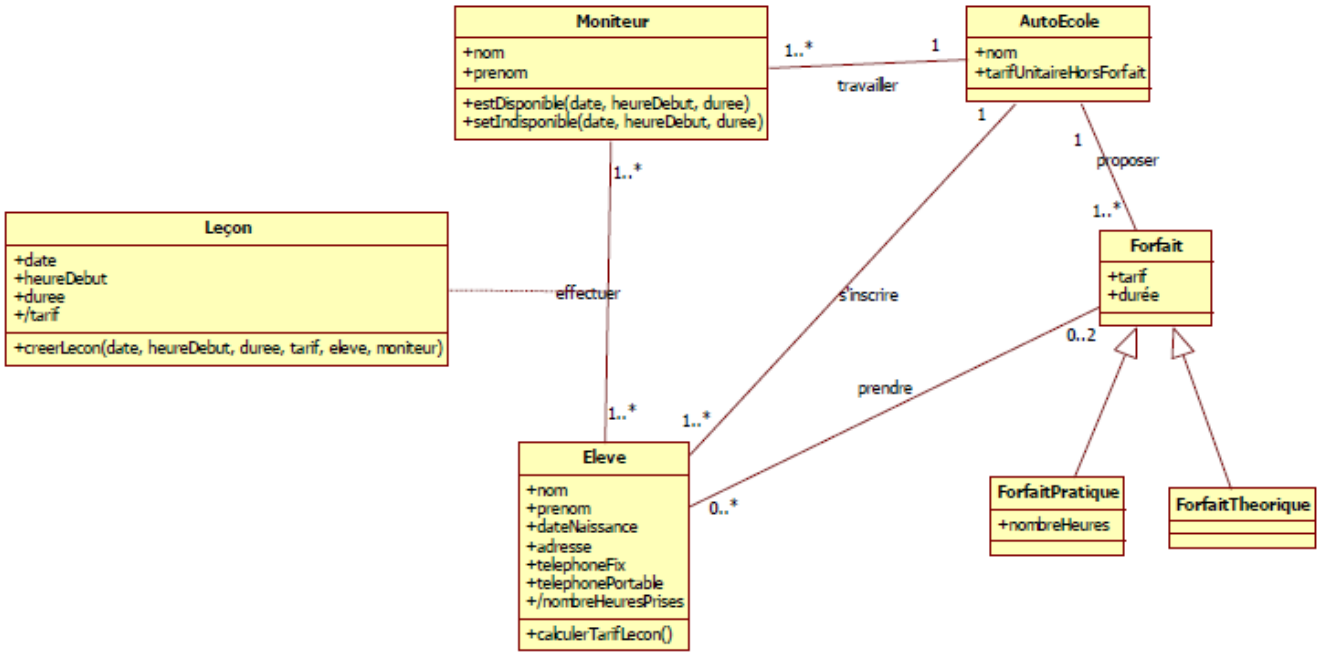
Université Larbi ben M'Hidi
Faculté des sciences exactes & sciences de la nature et de la vie
Département MI 3ième Année Licence S.I.

15 Jan. 2023
Dr. KOUAH Sofia

Corrigé type - Génie Logiciel-

Exercice :

1) Diagramme de classes : (4,5 Pts)



2) Code Java (2 Pts)

```
public class Moniteur {
public String nom, prenom ;
public Eleve[ ] lesEleves;
public AutoEcole au;
public void estDisponible (Date d, int
HeureDebut, int dr) {... }
public void setIndisponible (Date d, int
h, int dr) {... }
...  }
```

```
public class Eleve {
public String nom, prenom, adresse;
public int tarif,telephoneFix,
telephonePortable, nombreHeurePrises;
public Moniteur[] lesMoniteurs;
Forfait[] lesForfaits = new Forfait[2];
AutoEcole au;
public int calculerTarifLecon(){ ...
return tarif;     }   ... }
```

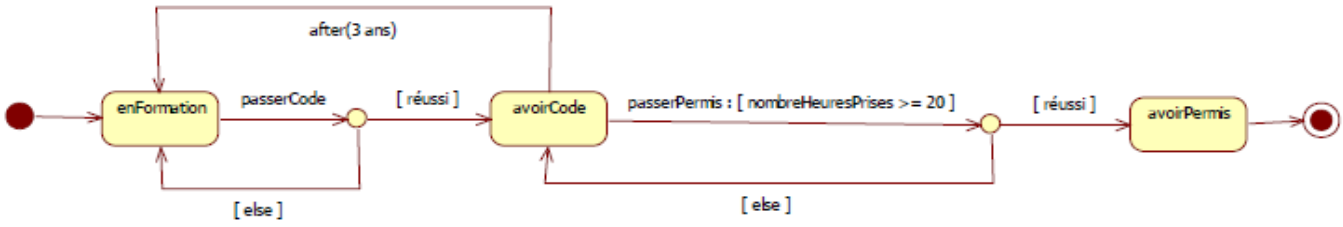
```
public class Lecon {
public Date date ;
public int heureDebut, duree, tarif;
public Moniteur m;
public Eleve e;
public void
creerLecon(date,heureDebut,duree,tarif,e,m)
{... }    ...  }
```

```
public class AutoEcole {
public String nom ;
public int tarifHorsForfait ;
public Eleve[] lesEleves;
public Moniteur[] lesMoniteurs;
public Forfait[] lesForfaits;
...  }
```

```
public class Forfait {
public int tarf, duree ;
public Eleve[] lesEleves;
public AutoEcole auto;
...      }
```

```
public class ForfaitPratique extends Forfait{
        int nombreHeures;
        public ForfaitPratique() {
                super();  }  ... }
public class ForfaitTeorique extends Forfait
{
        int nombreHeures;
        public ForfaitTheorique() {
                super();  }  ... }
```

3) Diagramme d'états transitions (3,5 Pts)



4) Design Pattern Singleton (2Pts)

Pour assurer l'unicité du Directeur on utilise le design pattern **Singleton** définit par : un **constructeur privé** et une **méthode de création** permettant de créer une instance unique. Le code correspondant est décrit comme suit :

Classe Directeur :

```
public class Directeur {
private static Directeur instance = null;
private Directeur ( ) { }
public static Directeur getInstance ( ) {
if (instance == null)
instance = new Directeur( );
return instance;   }   }
```

Utilisation de la classe Directeur :

```
public class Program {
public void  Method ( ) {
Directeur D= Directeur.getInstance (); }   }
```

Questions de compréhension (8 Pts) :

1. **1,5 Pts par réponse** (4 questions uniquement)

a) Un **design pattern** : est une solution à un problème récurrent dans la conception d'applications orientées objets.
Rôle : permet de décrire une solution éprouvée pour résoudre des problèmes récurrents lies à l'architecture de logiciel.
Examples design pattern: singleton, composite, factory, Abstract Factory, …etc.

b) La **différence** entre le diagramme d'activités et le diagramme d'états transitions :
Les diagrammes **d'états-transitions** sont définis pour **chaque classeur** et **n'en font pas intervenir plusieurs**. A l'inverse, les **diagrammes d'activité** permettent une description s'affranchissant (partiellement) de la structuration de l'application en classeurs.

c) Rôle **du flow final** du diagramme d'activités :
Un nœud **flow final** permet de mettre fin (détruire) le flot de contrôle correspondant sans affecter les autres flots de contrôle du diagramme.

d) **Rôle des partitions** dans le diagramme d'activités.
Les partitions permettent d'attribuer les activités à des éléments particuliers du modèle. Donc, on peut spécifier le classeur responsable de chaque activité.

e) La différence entre **agrégation et composition** :
Une agrégation (représentée par un losange vide) est une **relation d'inclusion** d'un élément dans un ensemble. Par contre la composions (représentée par un losange plein) décrit une **contenance structurelle** entre instances. Dans la composition, la **destruction et la copie de l'objet composite** (l'ensemble) implique respectivement **la destruction ou la copie de ses composants** (les parties).

f) Rôle des **fragments combinés** : Un fragment combiné permet de décomposer une interaction complexe en fragments suffisamment simples pour être compris. Il permet de faire face à la complexité et favorise la réutilisation.

g) **Rôle du diagramme de cas d'utilisations** : permet de donner une vision globale du comportement fonctionnel d'un système. Il décrit le « quoi » du système (les fonctionnalités).
Ses composants : acteurs, cas d'utilisation, association et relation entre cas d'utilisation (include, extend, generalisation), limite du système.

h) Quelle est la différence entre **analyse des besoins et conception** ?
- **Analyse des besoins** : Comprendre les besoins du client (Déterminer les fonctionnalités du système et ses contraintes d'utilisation)
- **Conception** : Élaborer une solution concrète réalisant la spécification (Description très proche d'un programme), généralement décomposée en deux phases successives : Conception architecturale et conception détaillée.

i) Trois critères de qualité d'un logiciel.
Par exemple : validité, fiabilité, efficacité, portabilité, et facilite d'emploi.
Fiabilité : fonctionner dans des conditions anormales.
Validité : remplir exactement ses fonctions, définies par le cahier des charges et les spécifications.

2. **(2 Pts)**

Diagramme de séquences.
Rôle : diagramme comportemental permettant des représenter des échanges de messages entre les différents objets et acteurs du système en fonction du temps.
Le diagramme de classes correspondant :