# Examen intra (20%)

(Durée : 1h30, 2 heures seront accordées)

**Critères d'évaluation /20**

|  |  |  |  |
| --- | --- | --- | --- |
| **Critères** | **Exercice 1**  Diagramme de classe | **Exercice 2**  Diagramme de cas d’utilisation | **Exercice 3**  Code Python |
| Respect des normes (noms significatifs, docstring, commentaires, clarté)  (diagramme de classe  Diagramme de cas d’utilisation) | 3 | 2 | 2 |
| Fonctionnement et/ou logique | 5 | 2 | 6 |

Directives pour la remise

* Créer un dossier nommé Examen\_intra\_<#etud>. Exemple : Examen\_intra\_9962727
* Dans ce dossier ajouter :
  + Le diagramme de classe de l’exercice 1.
  + Le diagramme de cas d’utilisation de l’exercice 2.
  + Le dossier contenant le code des classes et le programme principal de l’exercice 3.
* Compresser le dossier Examen\_intra\_<#etud> et le remettre sur LÉA. **Aucune remise par mio ne sera acceptée.**

Exercice 1: Gestion de Pizzeria – Diagramme de classe UML

Pour créer un système de gestion de pizzeria, on commence par créer un diagramme de classe UML qui modélise les classes du système et leurs relations selon l’énoncé suivant :

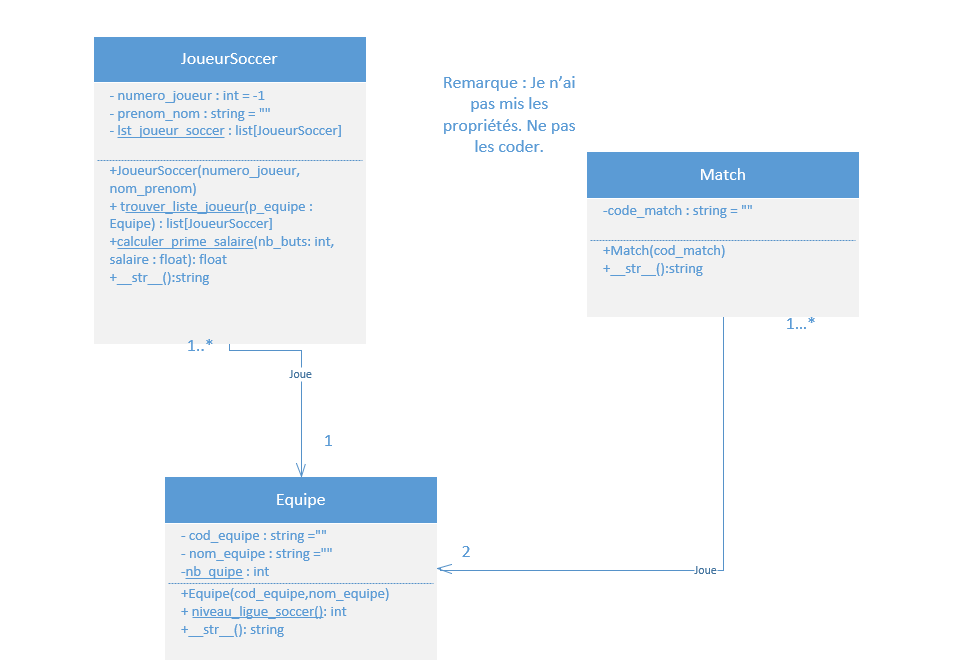
chaque client peut passer des commandes. Il a un nom et un numéro de téléphone. Les commandes peuvent être emportées ou livrées. Une commande à emporter se caractérise par une heure de retrait. Une commande à livrer comporte une adresse et une heure de livraison. Les commandes sont constituées d’un ou plusieurs d’articles. Tous les articles ont un prix. Le système doit être capable d’afficher le prix total d’une commande ainsi que le prix d’un article avec taxes. Il doit aussi être capable d’afficher les informations sur la commande d’un client. On veut avoir la liste de tous les clients de la pizzeria et de tous les articles vendus par la pizzeria. Le système doit aussi être capable de trouver nombre et la liste de commandes livrées à une certaine heure.

Exercice 2 : Gestion d’un parc d’attraction – Diagramme de cas d’utilisation

Dans un système de gestion d'un parc d'attractions, les clients peuvent être des visiteurs réguliers ou des membres VIP. Les visiteurs réguliers peuvent acheter des billets pour des attractions individuelles ou des forfaits pour plusieurs attractions. Les membres VIP ont accès à des avantages exclusifs tels que des files d'attente prioritaires et des réductions sur les repas. Les membres VIP ont droit à la réduction sur les repas s’ils sont membres depuis plus de 5 ans. Au moment de l’achat d’un billet, le client peut recommander un(e) ami(e) à faire une visite gratuite au parc. Tous les clients peuvent consulter les informations sur les attractions.

Exercice 3 : Coder à partir d’un diagramme de classe UML

Soit le diagramme de classe fourni également dans le fichier « exercice3.vsdx ».



Compléter ce diagramme de classe avec les attributs issus des associations avant de répondre aux questions suivantes :

1. Coder les classes **JoueurSoccer**, **Equipe** et **Match** selon le diagramme fourni complété. Voici la description des méthodes :
   1. trouver\_liste\_joueur() : permet de retourner la liste des joueurs qui jouent dans une équipe.
   2. niveau\_ligue\_soccer(): permet de retourner le niveau (1, 2 ou 3) de la ligue en fonction du nombre d’équipes qu’elle contient :
      1. Niveau 1 : entre 1 et 10 équipes.
      2. Niveau 2 : entre 11 et 20 équipes.
      3. Niveau 3 : plus de 20 équipes.
   3. calculer\_prime\_salaire() : permet de calculer la prime du joueur en fonction du nombre de buts. Chaque but marqué offre une prime de 10% sur le salaire du joueur.
2. Dans le programme principal,
   1. instancier 3 joueurs et une équipe de soccer. Les trois joueurs instanciés jouent dans cette équipe.
   2. Afficher les informations des joueurs qui jouent dans cette équipe-là. Pour cela, écrire une méthode et l’utiliser.

Bonne réussite 😊