UVHC/ISTV	**Année universitaire : 2018/2019**
L2 Informatique	
Module :	**Programmation Orientée Objet / S4 – Session 1**
Epreuve :	**2 heures, aucun document autorisé**
Enseignants :	**M. Desertot**

Remarques : Lisez bien, pour chaque question, le travail qui vous est demandé.

AUCUN code n'est demandé, mais vous pouvez utiliser des schémas pour illustrer vos propos au besoin.

Exercice 1 : Questions de cours (10 pts)

1) Expliquez ce que définit le code suivant : public class Box<T> {….}
2) Comment est réalisée la configuration de l'accès à la BDD en JPA ?
3) Quel est le rôle de l'interface Serializable ?
4) Pourquoi ne faut-il jamais oublier de fermer un flux ?
5) Pourquoi est-il indispensable qu'un objet soit défini dans un package ?
6) Quelle est l'utilité d'utiliser un débuggeur pour comprendre les problèmes d'un programme ?
7) Qu'est-ce que l'introspection ?
8) Un objet peut-il implémenter plusieurs interfaces ? Pourquoi ?
9) Expliquez comment est gérée la taille d'un ArrayList.
10) Quel est l'intérêt d'outils comme Maven ou Gradle pour le développement logiciel ?

Exercice 2 : Cas d'étude (10 pts)

On souhaite réaliser une application qui permettra de gérer des personnes et des parties d'un jeu vidéo en réseau. Les différentes personnes ont trois rôles possibles : joueur, créateur ou spectateur. Toutes ces personnes ont un nom un prénom un âge et un identifiant. Le créateur a créé un certain nombre de parties que des joueurs peuvent rejoindre. Un joueur peut être dans une ou plusieurs parties (on suppose que les parties ne peuvent pas se dérouler en même temps) et a une somme d'argent qu'il peut utiliser pendant les parties. Un spectateur peut intégrer n'importe quelle partie. Un compteur cumul le nombre de parties visualisées. Toutes les parties sont associées à un serveur. Plusieurs serveurs peuvent être disponibles en même temps.

1) Proposez une architecture (utilisez le formalisme vu en cours) répondant aux besoins de l'énoncé et utilisant l'héritage à bon escient. Les objets doivent pouvoir représenter tous les concepts évoqués dans l'énoncé.
2) Quels objets devraient se voir définir une interface ? Justifiez.
3) En fonction de votre architecture, comment feriez-vous pour mettre en place une limitation du nombre de joueurs à 8 par partie ?
4) Que faut-il ajouter à vos objets pour permettre de modéliser le fait qu'un joueur peut avoir des amis joueurs (et uniquement joueurs).
5) Que faut-il modifier pour que des parties puissent opposer deux équipes de joueurs en plus des parties pour joueurs solo déjà existantes ?
6) Comment et où placeriez-vous une méthode recherchant le joueur ayant le plus grand montant d'argent ?
7) Comment feriez-vous pour qu'à chaque démarrage de l'application, les données existantes soient conservées ?
8) Que faut-il ajouter au code écrit pour mettre en place des tests unitaires ? Qu'est-ce qu'une assertion dans les tests ?
9) Quelle(s) modification(s) devriez-vous apporter à votre programme pour pouvoir afficher les information d'un joueur au format JSON.