Université Mohammed V – Rabat
Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes

Examen

Année Universitaire : 2018 - 2019 **Date** : 27/12/2019
Filière : Ingénieur **Semestre** : S3 **Période** : P2 **Durée** : **1h30**
Elément de Module : **M3.3.2** – Programmation Objet Avancée
Professeur : M. EL HAMLAOUI

Consignes aux élèves ingénieurs :

- *Uniquement un document A4 recto-verso est autorisé*
- *Les questions sont suffisamment expliquées, reportez vos remarques sur votre copie*
- *Il sera tenu compte de la clarté et de la simplicité des réponses*

Cours

Laquelle de ces propositions est fausse ?

- La déclaration d'une méthode comme **synchronized** garantit qu'aucun interblocage ne peut se produire,
- La méthode **sleep** ne consomme pas de temps de processeur pendant qu'un Thread sommeille,
- Les méthodes **suspend** et **resume** de Thread sont obsolètes.

Multithreading

1. Ecrivez un programme dont le thread principal (main) lance et nomme deux nouveaux threads. Chaque thread ainsi créé doit effectuer 10 fois les actions suivantes :
   i. attendre un temps aléatoire compris entre 0 et 200 ms,
   ii. puis afficher son nom,
   iii. le thread principal devra **attendre** la fin de l'exécution des deux threads qu'il a créés avant de terminer son exécution.
2. Que doit-on faire pour que les deux threads affichent leurs noms **en alternance** ?

Socket

On veut écrire une application **client/serveur** qui permet de savoir si une machine est active. Cette application utilise des **sockets TCP**.

i. écrivez le programme **Serveur** qui attend une requête du client sur le port 8182 et lui envoie un message (contenant l'heure locale). Le serveur doit être capable de **gérer plusieurs clients** simultanément,

ii. écrivez le programme **Client** qui lit une adresse au clavier et envoie une requête au serveur de la machine correspondante pour savoir si elle est active. Dans l'affirmative, il affiche le message envoyé par le serveur.

RMI

Nous disposons d'un service qui offre pour le client des opérations de gestion de son compte courant à savoir :

i. void debiter(double montant) { }
ii. void crediter(double montant) { }
iii. double getSolde() { }

a. On souhaite rendre chacune de ces méthodes *accessibles à distance* de manière à ce qu'elles définissent l'interface entre le client et le serveur. Ecrire cette **interface**.
(Respectez les *règles syntaxiques* que doit suivre une *interface Java RMI)*

b. Déduire la classe qui **matérialise** le service qui offre les opérations debiter(), crediter() et getsolde().
(Respectez les *règles syntaxiques* que doit suivre une *implantation* d'un *objet serveur Java RMI*)

c. **Compléter le fichier** Serveur.java suivant afin de permettre **l'enregistrement du service** auprès de RMI Registry.

```
import java.rmi.*; import java.rmi.server.*;
public class Serveur {
    public static void main(String[] args) {
        try {
            System.out.println("Serveur : Construction de l'implémentation");
            Compte cpt= new Compte(2000.0);
            System.out.println("Objet Compte enregistré dans RMIregistry");
            // à compléter
            System.out.println("Attente des invocations des clients ");
        } catch (Exception e) {
            System.out.println("Erreur de liaison de l'objet Compte");
            System.out.println(e.toString());
    } // fin du main
} // fin de la classe Serveur
```

d. Donner la suite de **commandes** ayant pour finalité le lancement du Serveur.