TP 1 (2019) Systèmes concurrents et distribués

Exercice 1 :

1. Implémentez et testez le programme qui se trouve sur les slides 30-31 du cours.
2. Générez des exécutions différentes en utilisant la fonction *sleep()* et des temps d’attentes aléatoires.
3. Remplacez l’appel à la fonction *System.out.print* par une fonction *print* que vous écrirez. Cette fonction *print* affichera la chaîne de caractères en utilisant *System.out.print* **caractère par caractère**. Observez les nouveaux entrelacements.

Exercice 2 :

On veut accéder des ressources numérotées 1,2,3,… dans un programme concurrent en exclusion mutuelle. Avant chaque accès on appelle la routine *hold(i)* avec *i* est le numéro de la ressource qu’on veut accéder. Cette routine est bloquante, c’est-à-dire que si la ressource n’est pas disponible le thread passe dans l’état *bloqué* et repasse dans l’état *exécutable* lorsqu’elle est libérée. Pour rendre une ressource accessible à un autre thread on utilise la routine *release(i)*.

Montrez que si un thread réserve toujours les ressources dans l’ordre croissant alors il n’y a jamais d’interblocage.

Exercice 3 :

Implémentez un programme multithread qui calcule les solutions entières de l’équation a^2=b^2+c^2 pour a <= 20. Enumérez tous les triples (a,b,c) possibles et testez si l’équation est satisfaite. Pour chaque test un nouveau thread est créé et exécuté.