Théorie et pratique de la concurrence – Master 1 Informatique
TP 6 : Sémaphores en Java

Exercice 1: *La piscine*

Une piscine dispose d'un certain nombre N_c de cabines et d'un certain nombre N_p de paniers. Un client qui se présente à l'entrée de la piscine effectue les étapes suivantes :

a. il doit attendre une clef de cabine,

b. il doit demander et obtenir un panier vide,

c. il occupe la cabine pour se changer,

d. il libère la cabine et la clef (et dépose son panier plein au vestiaire),

e. il va nager,

f. il cherche son panier plein au vestiaire et attend qu'une cabine se libère,

g. il prend une clef de cabine et occupe la cabine pour se changer, et

h. libère la cabine et rend la clef et le panier vide (et quitte la piscine).

Le but de cet exercice est de développer un programme concurrent simulant le comportement des clients de cette piscine.

1. Créez une classe `Client` étendant la classe `Thread` et dont la méthode `run` caractérise le comportement d'un client de la piscine. Les paniers et les cabines seront représentés par deux sémaphores `semCabine` et `semPanier` qui seront des champs de la classe `Client` (ces sémaphores seront partagés par tous les threads représentant les clients de la piscine).
2. Testez votre programme avec différents nombres de clients et différents nombres de cabines et de paniers (le nombre de cabines et de paniers étant déterminé par la valeur avec laquelle le sémaphore correspondant est initialisé). En particulier, essayez d'obtenir (en faisant dormir les threads à un certain endroit) un deadlock pour le cas où le nombre de cabines et plus petit que le nombre de paniers, lui même étant plus petit que le nombre de clients.
3. Dans le cas précis où il y a un deadlock, proposez une solution en modifiant le code des clients sans sans changer les nombres de clients, paniers ou cabines. **Indice :** Vous pouvez vous servir des méthodes proposées par la classe `Semaphore`.