Programmation Multi-cœurs — TP 1
Introduction à la concurrence

Le TP peut être fait en Java ou C++, mais Java est conseillé. Téléchargez l'archive contenant le code à exécuter sur Madoc. Pour les inconditionnels du C++, vous pouvez compiler un fichier avec la commande "`g++ Programme.cpp -lpthread -O1 -o Programme.exe`".

Pour chaque programme du dossier TP1, vous devez en comprendre le code, prédire la sortie attendue, et enfin confronter vos prédictions à l'exécution du programme.

Exercice 1 (Asynchronie)**.** Étudiez le code du programme `HelloWorld`.

a. Prédisez-en la sortie.

b. Exécutez le programme plusieurs fois. Comment expliquez-vous les résultats ?

c. Utilisez le débogueur pour obtenir les sorties décrites en commentaires.

Exercice 2 (Section critique)**.** Étudiez le code du programme `SharedCounter`.

a. Prédisez-en la sortie

b. Exécutez le programme en faisant varier la valeur du paramètre : 10, 100, 1000, 10000, 100000. Que constatez-vous ?

c. Lisez la page `man javap`, puis exécutez `javap -c SharedCounter.class`. Expliquez les résultats obtenus lors de l'exécution du programme.

d. Décommentez les sections de code prévues à cet effet dans les fonctions `increment` et `decrement` puis relancez l'exécution.

e. Proposez une définition du mot-clé Java **synchronized** ou de l'objet C++ `mutex`.

Exercice 3 (Vivacité)**.** Étudiez le code du programme `Friend`.

a. Exécutez le programme. Que se passe-t-il ?

b. Utilisez le débogueur pour obtenir les sorties décrites en commentaires.

Exercice 4 (Modèles de mémoire)**.** Étudiez le code du programme `Volatile`.

a. Prédisez-en la sortie.

b. Exécutez le programme.

c. Ajoutez le mot-clé `volatile` devant la déclaration de la variable `shared` et ré-exécutez.

d. Que fait le mot-clé `volatile` ?

Exercice 5 (Moniteurs)**.** Étudiez le code du programme `Semaphore`.

a. Exécutez le programme en faisant varier la valeur du paramètre : 6, puis 4, puis 2. Que constatez-vous ?

b. Proposez une définition de sémaphore.

Exercice 6 (Réentrance)**.** Étudiez le code du programme `Reentrance`.

a. Exécutez le programme. Que remarquez-vous ?

b. Lors de la déclaration de `lock`, remplacez ReentrantLock par NonReentrantLock. Que constatez-vous ?

c. Proposez une définition de la réentrance.