# HMIN101M - TP noté

Durée : 1h30

24 novembre 2015

**REMARQUE : Lisez attentivement l'énoncé en entier avant de commencer le travail.**

L'objectif de cet exercice est de mettre en place un mécanisme appelé $N^{eme}$-*mutex*. Un $N^{eme}$-*mutex* est un système de synchronisation qui permet à $k$ ($1 \leq k \leq N$) threads d'être présents simultanément en section critique, mais pas plus de $N$ threads à la fois (pensez à l'exemple du pont à capacité maximum en nombre de voitures).

Pour utiliser des $N^{eme}$-*mutex*, nous souhaitons disposer des 4 fonctions suivantes :

- **int n_mutex_init(n_mutex * v, int n)** pour initialiser une structure n_mutex. $n$ est le nombre maximum de threads pouvant être simultanément en section critique.
- **int n_mutex_lock(n_mutex * v)** pour demander l'entrée dans la section critique.
- **int n_mutex_unlock(n_mutex * v)** pour avertir de la sortie de la section critique.
- **int n_mutex_destroy(n_mutex * v)** pour détruire les éléments de la structure n_mutex.

L'ensemble de ces fonctions renvoi 0 en cas de succès, $-1$ en cas d'échec.

1. Définir la structure n_mutex.
2. Implémnter les fonctions précédentes. Ne pas oublier le traitement des cas d'erreurs.
3. Utiliser ces fonctions dans un exemple de sorte à montrer qu'elles répondent bien au problème posé. Attention, le nombre de threads et la valeur $n$ doivent être des paramètres de votre application. Aussi, vous devez mettre l'accent sur la trace d'exécution qui doit permettre de bien illustrer le comportement de votre application.

**Dépôt de votre travail** Créer une archive votre_nom_TP2.tgz, contenant le code soucre et un Makefile. Déposer l'archive dans l'espace pédagogique (section "TP-noté 2").