TP1 – Programmation parallèle : CPU

Exercice 1 : Hello World !

Écrire un programme qui lance deux threads où chaque thread affiche le message "Hello World !".

Exercice 2 : Produit de deux matrices

Écrire un programme qui permet de calculer le produit de deux matrices (P=A*B).

Pour le calcul du produit matriciel, on remarque que les valeurs des (Pij) sont indépendantes les unes des autres. On peut donc facilement effectuer ces calculs en parallèles.

Écrivez une classe Matrice qui permet d'effectuer ce produit matriciel (n'implantez que les méthodes utiles pour effectuer le produit de 2 matrices en parallèle).

1. Implémenter une version séquentielle (ie. sans thread)
2. Implémenter une version par valeur (ie. chaque valeur (Pij) de la matrice P résultant du produit matriciel sera calculée par un thread)
3. Implémenter une version par bloc (ie. la matrice P est divisée en m blocs et les valeurs (Pij) composant chaque bloc seront calculées par un thread)

Exercice 3 : Somme d'un tableau

Écrire un programme qui calcule la somme des éléments d'un tableau T de n éléments :

1. Implémenter une version séquentielle
2. Implémenter une version en utilisant des variables locales (ie. le tableau est divisé en m sous-tableaux et chaque thread calcule la somme de ce sous-tableau dans une variable locale -- variable utilisée uniquement par ce thread)
3. Implémenter une version en utilisant une variable partagée (ie. le tableau est divisé en m sous-tableaux en utilisant une variable partagée)

Exercice 4 : Mutex

Écrire un programme qui lance deux threads :
- Le premier thread (T1) affiche les nombres pairs (de 0 à 1000).
- Le second thread (T2) affiche les nombres impairs (de 0 à 1000).

1. Observer le résultat de ce programme lors de son exécution.
2. Donner une nouvelle version du programme de sorte que l'exécution s'effectue en alternance entre T1 et T2 afin d'avoir un affichage de 0 à 1000 en ordre.

Référence : https://fr.cppreference.com/w/cpp/thread/mutex