UQÀM
Université du Québec à Montréal

Travail pratique # 1	**Cours : INF3172 groupe 20 et 30**
Pondération : 20 points	**Date de remise : 15 octobre 2019 via Moodle**

Objectifs pédagogiques

Le but de ce travail est de mettre en pratique l'utilisation des processus et des *threads*, cela se traduira par une bonne compréhension de la matière du cours. Et aussi vous préparera pour le deuxième travail.

Objectif de TP: implémentation en C un validateur d'un *Sudoku*.

Description de travail :

Un *Sudoku* utilise une grille de 9 x 9 dans laquelle chaque colonne et chaque ligne, ainsi que chacune des neuf sous-grilles 3 x 3, doivent contenir tous les chiffres de 1 à 9. La figure 1.0 présente un exemple de *Sudoku* valide. Ce projet consiste à concevoir une application *multithread* permettant de déterminer si la solution à un *Sudoku* est valide. Il existe plusieurs manières différentes d'utiliser le *multithreading* pour cette application. Une stratégie suggérée consiste à créer des *threads* qui vérifient les critères suivants :

- Un *thread* pour vérifier que chaque colonne contient les chiffres 1 à 9
- Un *thread* pour vérifier que chaque ligne contient les chiffres 1 à 9
- Neuf *threads* pour vérifier que chacun des sous-grilles 3x3 contient le chiffre 1

Cela donnerait un total de onze *threads* distincts pour valider un *Sudoku*. Cependant, vous pouvez créer encore plus de *threads* pour le projet. Par exemple, au lieu de créer un *thread* qui vérifie les neuf colonnes, vous pouvez créer neuf *threads* distincts et demander à chacun de vérifier une colonne (les deux façons sont correctes).

6	2	4	5	3	9	1	8	7
5	1	9	7	2	8	6	3	4
8	3	7	6	1	4	2	9	5
1	4	3	8	6	5	7	2	9
9	5	8	2	4	7	3	6	1
7	6	2	3	9	1	4	5	8
3	7	1	9	5	6	8	4	2
4	9	6	1	8	2	5	7	3
2	8	5	4	7	3	9	1	6

Figure 1.0 Solution a un *Sudoku* 9 x 9

Passer des paramètres à chaque thread : Le *thread* parent créera les *threads*, en transmettant à chaque utilisateur l'emplacement qu'il doit vérifier dans la grille de *Sudoku*. Cette étape nécessitera de transmettre plusieurs paramètres à chaque *thread*. L'approche la plus simple consiste à créer une structure de données à l'aide d'une *struct*. Par exemple, une structure pour passer la ligne et la colonne où le *thread* doit commencer à valider apparaîtra comme suit :

```
/* structure pour transmettre des données aux threads */

typedef struct

{

    int ligne;

    int colonne;

}   parameters;
```

Pthreads créera des *threads* de production en utilisant une stratégie similaire à celle présentée ci-dessous :

```
Parameters *data = (parameters *) malloc (sizeof (parameters))
data –> ligne = 1;
data –> colonne = 1;

/* Maintenant créer le thread en lui passant les données en tant que paramètre*/
```

Le pointeur de données sera transmis à la fonction pthread_create (); à son tour, il le transmettra en tant que paramètre à la fonction devant s'exécuter en tant que *thread* séparé.

Renvoyer les résultats au *thread* parent : Chaque *thread* est assigné à une tâche pour déterminer la validité d'une région particulière de *Sudoku*. Une fois qu'un *thread* a effectué cette vérification, il doit transmettre ses résultats au parent. Un bon moyen de gérer cela consiste à créer un tableau de valeurs entières visibles par chaque *thread*. L'indice i dans ce tableau correspond à la tâche de *thread* i. Si un *thread* définit sa valeur correspondante à 1, cela indique que sa région du *Sudoku* est valide. Une valeur de 0 indiquerait le contraire. Lorsque tous les tâches de *threads* sont terminées, le *thread* parent vérifie chaque entrée du tableau de résultats pour déterminer si le *Sudoku* est valide et il l'affiche.

Contraintes : Il faut le langage C et le thread pour réaliser ce travail.

Réalisation :

- La réalisation de ce travail se fera par groupes d'au plus deux personnes;
- Chaque membre peut être questionner pour savoir s'il maîtrise le travail;
- Le programme doit être en C sous linux (Unix);

Fichiers à rendre via Moodle (Rendre TP1) : Date de remise est : le mardi 15 octobre 2019

- 1 fichier TP1.c;
- 1 fichier de manuel d'utilisation (assez simple, comment on compile votre programme? Comment on l'exécute? Et toutes autres informations que vous jugiez pertinentes pour que je puisse tester adéquatement votre TP.

Évaluation :

- Le respect des instructions demandées;
- La qualité de la solution technique proposée;
- La qualité du code fourni (commentaires clairs, utilisez juste les appels systèmes ou les fonctions dont vous aurez besoin, etc.);
- Le traitement automatique des cas d'erreurs.