

# IN405 – Système d’exploitation

## TD 5 – Thread

S. Gougeaud

2017/2018

*L’archive `code-td5.tar` contient un fichier :*

- *`reduc.c` – fichier source de l’exercice 2, contenant les prototypes de fonction et les structures à utiliser.*

### Exercice 1 – Création de thread

En utilisant la fonction `pthread_create()`, écrivez un programme où le processus principal crée un thread pour chacun des comportements suivants :

1. Affichage de 'Hello World!'.
2. Affichage d’un entier aléatoire généré par le processus principal.
3. Affichage d’un entier aléatoire généré par le thread qui sera aussi affiché par le processus principal.
4. Affichage de la moyenne d’un tableau de 5 entiers générés aléatoirement par le processus principal.
5. Affichage de la moyenne d’un tableau de  $n$  entiers générés aléatoirement par le processus principal.

### Exercice 2 – Mécanisme de réduction pour le calcul

L’objectif de l’exercice est d’offrir à l’utilisateur une bibliothèque de fonctions et structures permettant le traitement d’un 'grand' tableau dans un environnement multi-threadé. L’exercice se découpe en plusieurs parties :

1. Programme principal – l’exécution du programme se fait par la ligne de commande suivante :

<pre>\$ ./reduction m n opcode</pre>
--------------------------------------

L'argument `m` définit le nombre de threads générés par le programme principal, `n` la taille du tableau et `opcode` l'opération à réaliser sur le tableau. L'opcode Le programme doit dans un premier temps créer le tableau et générer les entiers le composant (entre 1 et 100), puis créer les threads. Il affiche, une fois l'exécution des threads terminée, le résultat obtenu.

2. Structure message – l'argument à la fonction de thread est une structure comportant quatre champs : le tableau d'entier (dans sa totalité), les indices de début et fin de traitement (partie du tableau que le thread doit traiter) et le résultat (renseignée par le thread à la fin de son exécution).
3. Détermination de l'opération – l'opcode est analysé par le programme principal, et ce dernier sélectionne alors l'opération à réaliser. Le tableau suivant représente les fonctions disponibles et leurs opcodes respectifs.

Code	Nom
+	somme
/	moyenne
M	max
m	min

4. Exécution du thread – chaque thread effectue sur la partie du tableau qu'il doit traiter, la fonction indiquée par l'opcode, puis retourner le résultat local.
5. Pour aller plus loin – vous pouvez gérer les arguments du programme avec la fonction `getopt()` (section 3 du manuel). La fonction `getopt()` permet l'analyse et le décodage des arguments d'un programme en utilisant des options. Il est alors possible de gérer des arguments obligatoires et/ou optionnels, quelque soit l'ordre dans lequel ils sont renseignés. L'exécution du programme se fera par la ligne de commande suivante :

```
$ ./reduction -t m -s n -o opcode
```

On supposera que l'argument `'-t'` pour le nombre de threads est optionnel, et les deux autres obligatoires. Dans le cas où le nombre de threads n'est pas renseigné, ce nombre vaudra 4.