

## L2 – TDTP 5 – Threads Unix

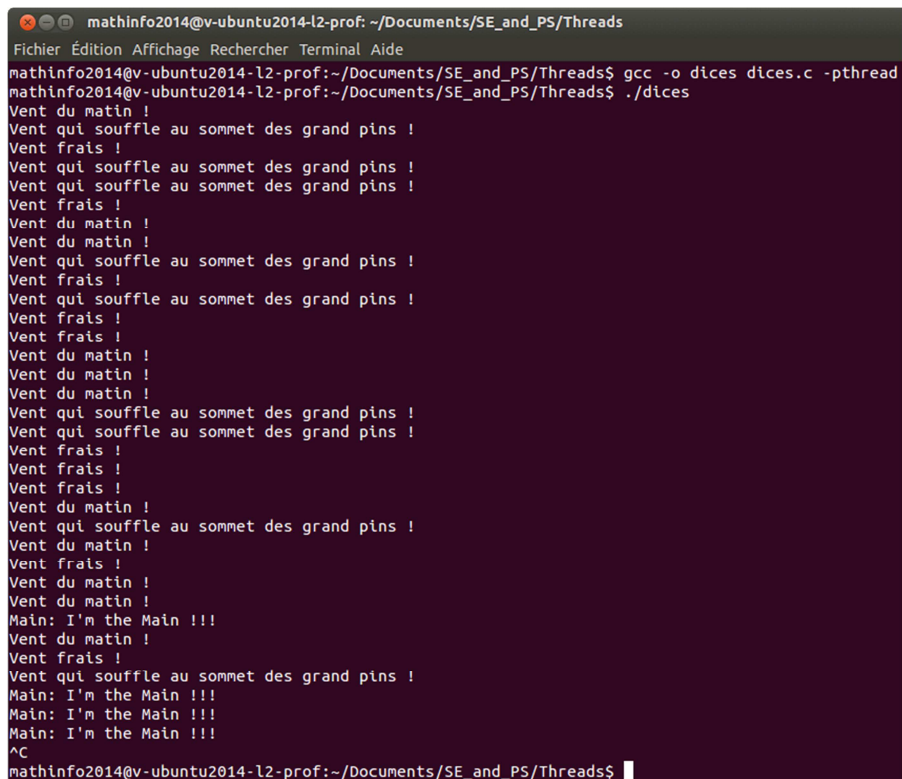
Ce TP se fait sur les machines Unix/Linux. Vous devez rendre un rapport et vos programmes à la fin du cours sur Espadon. Assurez-vous que votre rapport porte un nom du type "1516\_L2\_TDTP\_5\_Threads\_Unix\_votreNom". Mettez dans le rapport des copies d'écran des résultats d'exécutions ("Applications" → "Accessoires" → "Capture d'écran") que vous ferez régulièrement le long du TP pour montrer la réussite ou l'échec de vos opérations. Rendez le tout dans un fichier Zip de même nom que précédemment. Pour tous les TP, ne validez votre dépôt sur Espadon qu'une fois le TP totalement terminé; à la fin de chaque séance, déposez un zip de vos travaux avec votre rapport en l'état, mettez-le à jour à chaque séance, et une fois le TP fini, validez.

### I] Connexion

- 1) Connectez-vous à votre machine virtuelle Unix/Linux (Utilisez le mot de passe que vous avez défini au premier TP).
- 2) Dans le répertoire `"/home/mathinfo2014/Documents/SE_and_PS"`, créez un répertoire `"Threads"` (commande `mkdir`; ce sera le répertoire de travail du TP).

### II] Exercice 1: Le fond de l'air est frais

Dans le fichier `"ventFrais.c"`, écrivez un programme en C qui crée un nombre *len* de threads à partir d'une fonction *print*, et ensuite effectue une boucle infinie en affichant un message toutes les 5 secondes. La fonction *print* tire en boucle un nombre au hasard entre 0 et 3, et affiche une phrase du canon « Vent frais, vent du matin » toutes les secondes.



```

mathinfo2014@v-ubuntu2014-l2-prof: ~/Documents/SE_and_PS/Threads
Fichier Édition Affichage Rechercher Terminal Aide
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ gcc -o dices dices.c -pthread
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ ./dices
Vent du matin !
Vent qui souffle au sommet des grand pins !
Vent frais !
Vent qui souffle au sommet des grand pins !
Vent qui souffle au sommet des grand pins !
Vent frais !
Vent du matin !
Vent du matin !
Vent qui souffle au sommet des grand pins !
Vent frais !
Vent qui souffle au sommet des grand pins !
Vent frais !
Vent frais !
Vent du matin !
Vent du matin !
Vent du matin !
Vent qui souffle au sommet des grand pins !
Vent qui souffle au sommet des grand pins !
Vent frais !
Vent frais !
Vent frais !
Vent du matin !
Vent qui souffle au sommet des grand pins !
Vent du matin !
Vent frais !
Vent du matin !
Vent du matin !
Main: I'm the Main !!!
Vent du matin !
Vent frais !
Vent qui souffle au sommet des grand pins !
Main: I'm the Main !!!
Main: I'm the Main !!!
Main: I'm the Main !!!
^C
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$

```

### III] Exercice 2: Dans la jungle

Dans le fichier "dansLaJungle.c", écrivez un programme en C qui crée 6 threads et leur passe à chacun une chaîne de caractère issue de la liste ci-après : «Le», «lion», «est», «mort», «ce», «soir». Chaque thread est créé à partir d'une fonction *print* qui affiche la chaîne de caractère reçue dans une boucle infinie.

```
mathinfo2014@v-ubuntu2014-l2-prof: ~/Documents/SE_and_PS/Threads
Fichier Édition Affichage Rechercher Terminal Aide
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ gcc -o dansLaJungle dansLaJungle.c -pthread
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ ./dansLaJungle
Creating thread for Le.
Creating thread for lion.
Creating thread for est.
Creating thread for mort.
Creating thread for ce.
Creating thread for soir.
Main: I'm still alive !!!!
Thread: soir
Thread: ce
Thread: mort
Thread: est
Thread: lion
Thread: Le
Thread: soir
Thread: ce
Thread: mort
Thread: est
Thread: lion
Thread: Le
Thread: ce
Thread: soir
Thread: mort
Thread: est
Thread: lion
Thread: Le
Main: I'm still alive !!!!
Thread: soir
Thread: mort
Thread: ce
Thread: est
Thread: lion
Thread: Le
^C
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$
```

## IV] Exercice 3: Integrator

Dans le fichier "integrator.c", écrivez un programme en C qui crée un tableau dynamique de taille  $n$  des  $n$  premiers entiers naturels dans l'ordre croissant (par exemple  $n = 5000000$ ). Le programme crée ensuite  $nb$  thread (par exemple  $nb=5$ ) en leur passant comme argument une structure contenant le pointeur sur le tableau d'entiers, et deux nombres entiers *start* et *end*. Chaque thread est basé sur une fonction qui calcule la somme des nombres dans le tableau entre les *start* et *end*, et renvoie la valeur calculée au programme principal. Faites ensuite la somme des réponses des différents threads, et comparez la avec la somme obtenue en la calculant directement sur tout le tableau dans le 'main'.

```

mathinfo2014@v-ubuntu2014-l2-prof: ~/Documents/SE_and_PS/Threads
Fichier Édition Affichage Rechercher Terminal Aide

mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ gcc -o integrator integrator.c -pthread
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ ./integrator
Creating thread for segment 1 (from 0 to 999999).
Creating thread for segment 2 (from 1000000 to 1999999).
Creating thread for segment 3 (from 2000000 to 2999999).
Creating thread for segment 4 (from 3000000 to 3999999).
Creating thread for segment 5 (from 4000000 to 4999999).

This thread will perform summation from 4000000 to 4999999 (inclusive)
This thread will perform summation from 3000000 to 3999999 (inclusive)
This thread will perform summation from 2000000 to 2999999 (inclusive)
Done...
This thread will perform summation from 1000000 to 1999999 (inclusive)
This thread will perform summation from 0 to 999999 (inclusive)
Done...
Done...
Done...
Done...
La somme calculée par les thread est de : 1642668640.
La somme calculée par le main est de : 1642668640.

mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$

```

## V] Exercice 4: Syracuse Inverse

Dans le fichier "syracuseInverse.c", écrivez un programme en C qui calcule l'arborescence de la Conjecture Tchèque à partir de sa racine (1). Le programme principal crée un thread de calcul et lui passe le premier élément d'une liste chaînée contenant l'élément racine. Ce Thread crée l'élément suivant de la liste chaînée en multipliant le nombre *value* reçu par deux, et si ce nombre moins un est un multiple de trois, il crée un nouveau thread en lui passant le nouvel élément avec  $(value-1)/3$  pour calculer la nouvelle branche. Chaque thread qui crée lui-même des threads doit attendre que chacun de ses fils se termine avant de finir son exécution. L'exploration de l'arborescence de la conjecture se fait jusqu'à un degré *level* spécifié par l'utilisateur, et qui doit être stocké dans chaque élément de la liste chaînée. Ces éléments sont donc composés d'une valeur entière, d'un niveau d'exploration *level*, et de deux éléments suivants : un pair et parfois un impair (quand il existe). Commencez par écrire l'arborescence sur un papier en partant de 1.

```

mathinfo2014@v-ubuntu2014-l2-prof: ~/Documents/SE_and_PS/Threads
Fichier Édition Affichage Rechercher Terminal Aide
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ gcc -o syracuseInverse syracuseInverse.c -pthread
mathinfo2014@v-ubuntu2014-l2-prof:~/Documents/SE_and_PS/Threads$ ./syracuseInverse
Receive element value: 1, with remaining levels 7.
Creating next even (2).
Nexting.
Creating next even (4).
Nexting.
Creating next even (8).
Nexting.
Creating next even (16).
Nexting.
Creating next even (32).
Creating next odd (5).
Launching thread for odd branch.
Nexting.
Creating next even (64).
Nexting.
Creating next even (128).
Creating next odd (21).
Launching thread for odd branch.
Nexting.
Branch explored.
Waiting for threads...
Received.
Receive element value: 21, with remaining levels 0.
Branch explored.
Thread finished for branch value: 21.
Receive element value: 5, with remaining levels 2.
Creating next even (10).
Nexting.
Creating next even (20).
Creating next odd (3).
Launching thread for odd branch.
Nexting.
Branch explored.
Waiting for threads...
Received.
Receive element value: 3, with remaining levels 0.
Branch explored.
Thread finished for branch value: 3.
Received.
Thread finished for branch value: 5.
Received.
Thread finished for branch value: 1.

```