INF8541: Devoir #3

(À remettre le 11 octobre)

1. Exécution parallèle et instructions atomiques

Indiquez ce qui sera imprimé par chacun des programmes suivants. Si plusieurs réponses sont possibles, indiquez-les toutes.

1.
```
int x  = 0;
co  < await (x != 0) x = x - 2; >
//  < await (x != 0) x = x - 3; >
//  < await (x == 0) x = x + 5; >
oc
printf( "x = %d\n", x );
```

2.
```
int x  = 0;
co  if (x != 0) x = x - 2;
//  if (x != 0) x = x - 3;
//  if (x == 0) x = x + 5;
oc
printf( "x = %d\n", x );
```

2.

Les procédures présentées à la Figure 2 (p. 2) permettent de déterminer la valeur maximum d'un tableau d'entiers `elems` de taille `n`. On suppose que les éléments sont non-négatifs et, donc, que `0` est plus petit ou égal à n'importe quel élément du tableau.
Complétez la version Threaded-C de la procédure `trouver_max` présentée plus bas (Fig. 1). Vous pouvez évidemment utiliser n'importe quelle primitive ou opération de librairie vue en classe. Point important, votre mise en oeuvre Threaded-C doit réfléter *le plus fidèlement possible* la version dans la notation d'Andrews.

Note: Il ne faut pas oublier de libérer l'espace local alloué pour le tableau (avec `free`).

```
THREADED trouver_max( int *GLOBAL elems, int n, int *GLOBAL max_global,
                      LOCK verrou, SPTR fini )
{
  int i;
  int max_local = 0;
  int *elems_local;
  int max_global_local;

  MALLOC( elems_local, int, n );


  .
  .
  .

}
```

Figure 1: Procédure `trouver_max` à compléter

```
void trouver_max( int elems[], int n, int *max_global )
/* elems      = adresse de depart de la portion de tableau a fouiller
   n          = nombre d'elements de la partie de tableau a fouiller
   max_global = addresse de la variable qui doit etre modifiee
                s'il faut mettre a jour le maximum global
 */
{
  int i;
  int mon_max = 0;

  for [i = 0 to n-1] {
    if( elems[i] > mon_max )
      mon_max = elems[i];

  < if( mon_max > *max_global )
      *max_global = mon_max;    >
}


int borne_inf( int i, int n ){ return( i * n / NUM_NODES ); }

int max( int elems[], int n )
{
   int i;
   int max_global = 0;

   assert( n % NUM_NODES == 0 );

   co[i = 0 to NUM_NODES-1] {
     trouver_max( &elems[borne_inf(i, n)], n/NUM_NODES, &max_global );
   }
   return( max_global );
}
```

Figure 2: Procédures, dans la notation d'Andrews, pour trouver le maximum dans un tableau (nombre statique de processus)