INF8541: Devoir #2

(À remettre le 25 septembre)

Pour cet exercice, vous devez écrire une procédure qui permet de calculer la somme de deux vecteurs (longueur `n`) de nombres entiers (`int`).
Cette procédure doit faire la somme des deux vecteurs en utilisant une stratégie de parallélisme récursif. Plus précisément, une approche diviser-pour-régner bien équilibrée (dichotomique, donc générant deux sous-problèmes) doit être utilisée, chaque appel récursif générant une activation parallèle de la procédure.
Toutefois, contrairement à une solution récursive simple où la décomposition se termine lorsque le problème est vraiment trivial — ici, ce serait lorsque les tableaux à additionner sont de longueur 1 — vous devrez plutôt ajouter un argument supplémentaire à votre procédure (argument appelé le `niveau_coupure`) qui indique à partir de quelle taille de vecteurs la décomposition récursive *doit se terminer*. Ainsi, si la valeur de coupure est de `10`, alors lorsque la taille des vecteurs devient plus petite ou égale à `10`, la décomposition récursive se termine et une approche non-récursive (itérative) est alors utilisée.

Vous devez donnez deux versions de cette procédure:

1. Une version dans la notation d'Andrews;
2. Une version en Threaded-C qui reflète *le plus possible* celle dans la notation d'Andrews.

Voici un exemple d'appel (notation d'Andrews) pour lequel la récursion se terminerait lorsque les sous-tableaux à additionner ont `10` éléments ou moins:

```
#define N ...

int a[N], b[N], c[N];

...
additionner( a, b, c, N, 10 );
/* Le resultat est dans c: c = a + b. */
```

Note importante: Dans la version Threaded-C, vous devrez allouer de la mémoire dynamique (suggestion: utilisez la macro `MALLOC_N` du fichier `TPs/PRELUDE.h` dans votre compte Earthquake). N'oubliez pas que vous devez ensuite *libérer* l'espace ainsi alloué (avec `free`!), ce qui complique un peu le code pour terminer la procédure :(