INF8541: Devoir #7

(À remettre le 13 novembre)

1. Boîte aux lettres

Que calcule le programme Threaded-C présenté plus bas? Expliquez brièvement votre réponse, c'est-à-dire, expliquez dans vos propres mots comment fonctionne ce programme.

```
#include "PRELUDE.h"

THREADED proc( int num_proc, MAILBOX *GLOBAL mb )
{
  DROP_IN( mb, &num_proc, sizeof(int) );
  TERMINATE;
}

THREADED MAIN( int argc, char *argv[] )
{
  int i, n;
  MAILBOX mb;
  int v, v1, v2;
  int nb_recus;

  v = 0;
  INIT_MAILBOX( &mb, VALEUR_RECUE );
  DROP_IN( TO_GLOBAL(&mb), &v, sizeof(int) );

  n = atoi(argv[1]);
  nb_recus = 0;
  INIT_SLOT( FINI, 2*n-1 );
  for( i = 1; i <= n; i++ )
    INVOKE( i % NUM_NODES, proc, i, TO_GLOBAL(&mb) );

  EXCLUSIVE FIBER VALEUR_RECUE <* 1 *> {
    nb_recus += 1;
    if (nb_recus % 2 == 0) {
      RETRIEVE_ITEM( mb, &v1 );
      RETRIEVE_ITEM( mb, &v2 );
      v = v1 + v2;
      DROP_IN( TO_GLOBAL(&mb), &v, sizeof(int) );
    }
    SYNC(FINI);
  }

  FIBER FINI {
    RETRIEVE_ITEM( mb, &v );
    printf( "v = %d\n", v );
    FREE_MAILBOX(&mb);
    TERMINATE;
  }
}
```

2. Processus concurrents et canaux de communications

Soit le processus `Trier`, dont l'interface et la logique générale sont présentées (en partie, sous forme de pseudo-code) plus bas. Ce processus reçoit en entrée une suite de clés (chaînes de caractères) et de nombres point-flottants (chaque clé est associée à un unique nombre) et produit en sortie la version triée de cette suite (triée en fonction de la clé):

```
chan a_trier(String cle, float val);
chan tries  (String cle, float val);

process Trier {
  String cle;
  float val;

  while (TRUE) {
    receive a_trier( cle, val );
    if (cle == EOS) break;
    ajouter_dans_liste( cle, val );  # On ajoute la cle dans la liste des elements a trier.

  }
  # On trie la liste des elements recus.
  trier_liste();
  # On obtient chaque des elements tries (en ordre) et
  # on les transmet sur le canal de sortie.
  while (! liste_vide()) {
    obtenir_prochain_liste( &cle, &val );
    send tries( cle, val )
  }
  send tries( EOS, EOS );
}
```

En utilisant ce processus (vous *n'avez pas* à définir les opérations telles que `ajouter_dans_liste`, `trier_liste`, `liste_vide`, `obtenir_prochain_liste`) et les canaux indiqués, écrivez, dans la notation d'Andrews, un programme concurrent (groupe de processus) utilisant uniquement des canaux de communication et permettant de résoudre le problème suivant:

- Le programme reçoit en entrée (sur le canal `entree`) une série de valeurs de la forme suivante: une chaîne de caractères indique le nom d'un employé, laquelle chaîne est suivie du nombre d'heures travaillées pour chacune des journées de la semaine (peuvent être `0.0`). Les noms ne sont pas nécessairement en ordre. De plus, il est possible que plusieurs items d'informations soient présents dans la suite pour un employé donné, et ce à des positions non-consécutives. La suite est terminée lorsqu'une fin de canal (`EOS`) est rencontrée (pour le nom de l'employé).
- Le programme produit en sortie une liste ordonnée des employés avec le nombre total d'heures travaillées durant la période traitée.

Exemple:

- Entrée:

```
EMP2  8.0  8.0  8.0  8.0  8.0
EMP1  7.0  7.0  8.0  9.0  0.0
EMP3  7.0  7.0  8.0  9.0  0.0
EMP1  0.0  0.0  6.0 10.0  6.0
EMP2  8.0  8.0  8.0  8.0  8.0
```

- Sortie:

```
EMP1  53.0
EMP2  80.0
EMP3  31.0
```