TP n°9: Communication et synchronisation (les sémaphores)

SOLUTIONS (ne pas distribuer aux étudiants)

1) et 2)

- Initialisation d'un sémphore :

```
int sem_init(sem_t *sem, int pshared, unsigned int value);
```

- Destruction d'un sémaphore :

```
int sem_destroy(sem_t *sem) ;
```

- Attente sur un sémaphore :

```
int sem_wait(sem_t *sem) ;
```

- Libération d'un sémaphore :

```
int sem_post(sem_t *sem) ;
```

3)

```
#include <stdio.h>
#include <stdlib.h>
#include <semaphore.h>
#include <pthread.h>

// Déclaration des sémaphores de synchronisation
sem_t synchro1vers2;
sem_t synchro2vers1;

// Code Thread en charge de l'affichage de majuscules
void* codeThreadMaj(void *arg)
{
	int i;

	for (i=65; i<91; i++)
	{
		sem_wait(&synchro2vers1); /* Synchro depuis ThreadMin */
			printf("%c\n",i);
		sem_post(&synchro1vers2); /* Synchro vers ThreadMin */
	}

	return NULL;
}

// Code Thread en charge de l'affichage des minuscules
void* codeThreadMin(void *arg)
{
	int i;

	for (i=97; i<123; i++)
	{
		sem_wait(&synchro1vers2); /* Synchro depuis ThreadMaj */
			printf("%c\n",i);
```

```
            sem_post(&synchro2vers1); /* Synchro vers ThreadMaj */

      }

      return NULL;
}


// Code du main()
int main(int argc, char *argv[])
{
      // Déclaration du thread
      pthread_t ThreadMaj, ThreadMin;

      /* Initialisation des sémaphores */
      sem_init(&synchro1vers2, 0, 0);
      sem_init(&synchro2vers1, 0, 1);

      // Création des 2 threads
      if (pthread_create(&ThreadMaj, NULL, codeThreadMaj, (void *)NULL))
            {
                  perror("pthread_create Maj");
                  exit(EXIT_FAILURE);
      }
      if (pthread_create(&ThreadMin, NULL, codeThreadMin, (void *)NULL))
            {
                  perror("pthread_create Min");
                  exit(EXIT_FAILURE);
      }

      // Attente de la terminaison des threads
      pthread_join(ThreadMaj, NULL);
      pthread_join(ThreadMin, NULL);

      /* Destruction des sémaphores */
      sem_destroy(&synchro1vers2);
      sem_destroy(&synchro2vers1);

      return 0;
}
```