TP5- Synchronisation de processus- SEMAPHORE-

Lorsque 2 processus veulent se partager une ressource unique, on met en place un système de sémaphore afin d'arbitrer l'accès à cette ressource. Un sémaphore est une variable gérée par le système d'exploitation dont le test et l'affectation se font de manière atomique ; c'est à dire que le système ne peut pas être interrompu au milieu du traitement de ces 2 opérations. De plus, plusieurs processus peuvent avoir accès à cette variable. En général la prise (décrémente de 1) d'un sémaphore par un processus bloquera un autre processus qui voudrait prendre par la suite ce sémaphore ; ce dernier processus sera débloqué quand le premier rendra (incrémente de 1) le sémaphore.

```
#include <stdio.h>
#include <pthread.h>
#include <sys/ipc.h>
#include <sys/sem.h>
int i;
int semid;
struct sembuf operation;

void addition()
{
/* operation P */
operation.sem_num = 0;
operation.sem_op = -1;
operation.sem_flg = 0;
semop (semid, &operation, 1);

i = i + 10;
printf ("hello, thread fils %d\n", i);
i = i + 20;
printf ("hello, thread fils %d\n", i);

/* operation V */
operation.sem_num = 0;
operation.sem_op = 1;
operation.sem_flg = 0;
semop (semid, &operation, 1);
}
```

```
/*****************PROGRAMME PRICIPALE***************/

main()
{
pthread_t num_thread;
i = 0;
/* creation d'un semaphore initialisé à la valeur 1 */
if ((semid = semget (12, 1, IPC_CREAT|IPC_EXCL|0600)) ==
-1)
perror ("pb semget");
if ((semctl (semid, 0, SETVAL, 1)) == -1)
perror ("pb semctl");
if (pthread_create(&num_thread, NULL, (void
*(*)())addition, NULL) == -1)
perror ("pb fork\n");

/* operation P */
operation.sem_num = 0;
operation.sem_op = -1;
operation.sem_flg = 0;
semop (semid, &operation, 1);

i = i + 1000;
printf ("hello, thread principal %d\n", i);
i = i + 2000;
printf ("hello, thread principal %d\n", i);

/* operation V */
operation.sem_num = 0;
operation.sem_op = 1;
operation.sem_flg = 0;
semop (semid, &operation, 1);

pthread_join(num_thread, NULL);
}
```

Quelles traces résultent de cette exécution ?

Quel schéma de synchronisation reconnaissez-vous ?