MODULE SYSTÈME D'EXPLOITATION
TP – INTER PROCESS COMMUNICATION

Exercice n°1 – signal()

Soit le code suivant :

```
#include <signal.h>
#include <stdio.h>

int var;

void hand(int n)
{
	printf("Signal reçu : %d\n",n);
	if (n==SIGINT) signal(SIGINT,SIG_DFL);
	else signal (SIGQUIT, SIG_IGN);
}

int main ()
{
	signal(SIGINT,hand);
	signal(SIGQUIT,hand);
	while(1);
}
```

Expliquez le comportement du programme lorsque vous appuyez sur **Ctrl+C.**

Exercice n°2 – pipe()

Soit le code suivant :

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

int main(int argc, char *argv[])
{
  int filedes[2];
  int PID;
  pipe(filedes);
  if ( (PID=fork()) > 0)
  {
	write(filedes[1],"Message du pere a son fils...q",strlen("Message du pere a son fils...q"));
	printf("Le père a écrit\n");
	while(getchar());
  }

  else if (PID==0)
  {
	char car=0;
	char tab[200];
	int i=0;
	printf("Le fils essaie de lire le message\n");
	do
	{
		if ( read(filedes[0],&car,1)==1 )
		{
			tab[i]=car;i++;
			printf("Caractere lu : %c\n",car);
		}
	}while(car!='q');
	tab[i]=0;printf("Message : %s\n",tab);
  }

  else
  {
	printf("Erreur lors du fork()\n");
  }

  return EXIT_SUCCESS;
}
```

1. Que réalise l'appel système **pipe()** sur la variable **filedes** ?

2. A quoi sert le caractère 'q' dans le message envoyé par le père ?
3. Modifiez votre programme pour que le père puisse répondre au fils.

Exercice n°3 – Mémoire partagée

Tapez le code suivant pour le processus A :

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <math.h>
#include <stdio.h>

#define CLEF 12345 // je définis une clef au hasard

// Je définis une structure quelconque qui comporte un entier et un double.
typedef struct
{
int a;
double b;
}structure_partagee;

int main()
{
// variable quelconque qui me sers pour un compteur plus bas...
int i = 0;

int mem_ID; // identificateur du segment de mémoire partagée associé à CLEF
void* ptr_mem_partagee; // pointeur sur l'adresse d'attachement du segment de mémoire partagée

structure_partagee Data;

if ((mem_ID = shmget(CLEF, sizeof(Data), 0666 | IPC_CREAT)) < 0)
{
perror("shmget");
exit(1);
}

if ((ptr_mem_partagee = shmat(mem_ID, NULL, 0)) == (void*) -1)
{
perror("shmat");
exit(1);
}

// J'alloue des valeurs aux variables de ma structure
Data.a = 2;
Data.b = 2.6544;

*((structure_partagee*)ptr_mem_partagee) = Data;

 // je vais modifier en permanence le champ a de ma structure et le mettre à jour, le processus B lira la
structure Data.
while(1)
{
Data.a = i;
*((structure_partagee*)ptr_mem_partagee) = Data;
i++;
if(i == 100000000) // je remets à 0 de temps en temps...
i = 0;
}

shmdt(ptr_mem_partagee);

// je quitte le programme
return 0;
}
```

Tapez le code suivant pour le processus B :

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>

#define CLEF 12345 // !!je définis la même clef que celle du processus A!!


typedef struct
{
int c;
double d;
}structure_partagee_B;

int main()
{
// je déclare des variables aptes à recevoir les variables de la structure "structure_partagee" définie dans le
processus A
int var1;
double var2;

int mem_ID_B; // identificateur du segment de mémoire partagée associé à CLEF
void* ptr_mem_partagee_B; // adresse d'attachement du segment de mémoire partagée (idem)

structure_partagee_B Data_B;

if ((mem_ID_B = shmget(CLEF, sizeof(Data_B), 0444)) < 0)
{
perror("shmget");
exit(1);
}
if ((ptr_mem_partagee_B = shmat(mem_ID_B, NULL, 0)) == (void*) -1)
{
perror("shmat");
exit(1);
}

// j'affiche le contenu des variables inscrites par A dans la mémoire partagée
while(1) {

var1 = ((structure_partagee_B*)ptr_mem_partagee_B)->c;
var2 = ((structure_partagee_B*)ptr_mem_partagee_B)->d;

printf("data.a = %d\n",var1) ;
printf(" data.b = %d\n", var2);
}

// Je détruis le segment (le segment n'est pas détruit tant qu'au moins un processus est lié au segment)
shmdt(ptr_mem_partagee_B);

// je quitte le programme
return 0;
}
```

1. Compilez et exécutez les deux processus. Vérifiez que les variables du processus B sont bien mises à jour.
2. Commentez les appels systèmes shmget(), shmat() et shmdt() dans chaque processus (leur rôle).

Annexe

% ipcs –m

––– Segments de mémoire partagée –––

clé shmid propriétaire perms octets nattch état

- 0x00000000 1627649 user 640 25600 0

% ipcrm shm 1627649