TP n°8 : Communication et synchronisation (les signaux)

SOLUTIONS (ne pas distribuer aux étudiants)

1)

Exemple d'execution :

```
processus : 27422
reception d'un signal SIGINT
reception d'un signal SIGINT
reception d'un signal SIGINT
reception d'un signal SIGINT
reception d'un signal SIGINT
reception d'un signal SIGINT
reception d'un signal SIGINT
reception d'un signal SIGINT
```

pour finir, lancer un autre shell :

```
   $ kill -KILL 27422
   (ou kill -9 27422)
```

2)

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

/* code du handler pour le signal SIGUSR1 */
void handUSR1 ( int signo) {
  printf ("BONJOUR\n") ;
  signal(SIGUSR1, SIG_IGN); /* signal pris en compte une fois seulement */
}

/* code du handler pour le signal SIGUSR2 */
void handUSR2 ( int signo) {
  printf ("BONSOIR\n") ;
  /* selon le système, il faut parfois réarmer */
  /*     signal(SIGUSR2, handUSR2); */
}

int main ( ) {
  /* on arme les deux signaux */
  signal (SIGUSR1, handUSR1) ;
  signal (SIGUSR2, handUSR2) ;

  printf("processus : %d\n", getpid ( ));

  /* boucle infinie */
  while (1) ;

  return 0;
}
```

Exemple d'execution :

```
processus : 27414
BONJOUR
BONSOIR
BONSOIR
BONSOIR
BONSOIR
BONSOIR
```

Sortie obtenue en tapant dans un autre shell :

`$ kill -USR1 27414` *(3 fois de suite)*

`$ kill -USR2 27414` *(5 fois de suite)*

3)

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

/* code du handler pour le signal SIGUSR1 */
void handUSR1 ( int signo) {
  printf ("BONJOUR\n") ;
}

int main ( ) {
  int pid;
  setbuf(stdout, NULL);
  /* il faut prendre garde que le fils n'envoie pas le signal avant
   *  que le père ait armé le signal. Pour simplifier, on le fait
   *  avant l'appel à fork(). Le signal est alors armé également pour
   *  le fils */
  signal(SIGUSR1, handUSR1);
  if (pid =  fork()) { /* processus père */
    printf("père : Je suis le processus %d\n", getpid ( ));
    printf("père : j'attends un signal\n");
    pause();
    waitpid(pid, NULL, 0);
    printf("père : fin du processus\n");
  }
  else { /* processus fils */
    printf("fils : Je suis le processus %d\n", getpid ( ));
    /* le fils dort pour être sûr de ne pas envoyer de signal avant
       que le père n'ait appelé pause() */
    sleep(1);
    /* envoie du signal SIGUSR1 */
    kill(getppid(), SIGUSR1);
    sleep(2);
    printf("fils : fin du processus\n");
  }
  return 0;
}
```

Exemple d'execution :

```
père : Je suis le processus 27406
fils : Je suis le processus 27407
père : j'attends un signal
BONJOUR
fils : fin du processus
père : fin du processus
```