# Manipulation des signaux

## EXERCICE 1 : Synchronisation père/fils par des signaux

Ecrire un programme qui crée deux processus, père et fils. Le père affiche les nombres entiers impairs compris entre 1 et 100, alors que le fils affiche les entiers pairs compris dans le même intervalle. Synchroniser les processus à l'aide des signaux pour que le résultat d'affichage soit : `1 2 3 ... 100`.

## EXERCICE 2 : Horloge

Ecrire un programme C qui utilise 3 processus `H`, `M`, `S` qui incrémentent les 3 « aiguilles » d'une horloge. `S` reçoit un signal `SIGALRM` chaque seconde (émis par la fonction `alarm(1)`) et émet un signal à `M` quand son compteur passe de `59` à `0`. Quand `M` reçoit un signal, il incrémente son compteur. Quand son compteur passe de `59` à `0`, `M` envoie un signal à `H`. Les paramètres correspondent aux valeurs d'initialisation des compteurs.

## EXERCICE 3 : Roulette russe

Un père fait jouer six fils à la roulette russe.

Le père crée six fils, numérotés de 1 à 6.

- une fois ses fils crées, il envoie un signal au premier pour lui indiquer de commencer à jouer;
- à la réception de ce signal le fils lit un fichier « barillet » alimenté manuellement ou par le père avec un entier de 1 à 6 ;
  - ↘ si l'entier lu par le fils est égal à son numéro, le fils se tue (il s'arrête) ;
  - ↘ sinon, il envoie un signal au père pour l'avertir qu'il est encore vivant.
- si le fils a survécu, le père passe au fils suivant ;
- si le fils est mort, le père annonce à l'aide d'un signal aux autres fils qu'ils sont libres. Ceux-ci s'en réjouissent et s'arrêtent de jouer.

## Annexe

Fonctions relatives au traitement des chaînes de caractères :

- `int atoi(const char *s); // convertit une chaîne de caractères en un entier`
- `char *strcpy(char *dest, const char *src); // copie une chaîne de caractères`
- `size_t strlen(const char *s); // calcule la taille d'une chaîne de caractères`

Fonctions relatives à l'écriture et la lecture d'un entier depuis un fichier (exercice 3) :

```
#include <stdio.h>
#define TAILLE_MAX 20
int lire_valeur(const char *path)
{
	FILE *fichier;
	char chaine[TAILLE_MAX];
	int valeur;
	fichier = fopen(path, "r");
	if (fichier != NULL) {
		/* On lit au maximum TAILLE_MAX caractères du fichier, ce qui est lu est stocké dans `chaine` */
		fgets(chaine, TAILLE_MAX, fichier);
		fclose(fichier);
		valeur = atoi(chaine);
	}
	return valeur;
}
void ecrire_valeur(const char *path, int valeur)
{
	FILE *fichier = fopen(path, "w");
	if (fichier != NULL) {
		fprintf(fichier, "%d", valeur);
		fclose(fichier);
	}
}
```