Licence Informatique

Université Paul Sabatier
118 route de Narbonne
31062 Toulouse Cedex 09

SR2 – Partie « Systèmes » - Travaux pratiques n°1
Processus Unix – Signaux

Exercice 1 – Se protéger ainsi que ses fils

a) Écrire une application dans laquelle un processus boucle indéfiniment en affichant, toutes les secondes, un message contenant son identifiant (utiliser la fonction *sleep()* pour temporiser). Lorsque le signal SIGINT lui parviendra, il affichera le message « Ctrl-C/SIGINT reçu par le processus de n° <pid> » avant de continuer sa boucle d'affichage normale.

Exécuter cette application pour vérifier ce qu'il se passe en envoyant de temps en temps le signal SIGINT par appui des touches Ctrl & C du clavier.

Exemple d'exécution : ./exo1a

```
Je suis le processus 104
Je suis le processus 104
Je suis le processus 104
Je suis le processus 104
Je suis le processus 104
^C>> Ctrl-C/SIGINT recu par 104
Je suis le processus 104
Je suis le processus 104
```

b) Modifier l'application pour que le père crée un processus fils qui aura le même comportement que le processus de la version a) c'est-à-dire boucler indéfiniment en affichant, toutes les secondes, un message l'identifiant et en affichant un message spécifique lorsque Ctrl-C lui parvient. Le père attendra la fin de son fils pour se terminer et ne pourra pas se terminer avant ce dernier.

Exécuter à nouveau l'application pour vérifier son bon fonctionnement.

Exercice 2 – Signaux et primitives de recouvrement

a) Écrire une application (nommée *boucler*) qui affiche un message, toutes les NBS secondes (utiliser la fonction *sleep()* pour temporiser), un certain nombre NBF de fois. NBS et NBF sont les **paramètres** de cette application.

Exécuter cette application pour vérifier son comportement.

Exemple d'exécution : ./boucler 10 2

/ Afficher 10 fois toutes les 2 secondes */*

```
Je suis le processus 414, il est Wed Mar  8 16:22:51 2023
Je suis le processus 414, il est Wed Mar  8 16:22:53 2023
Je suis le processus 414, il est Wed Mar  8 16:22:55 2023
Je suis le processus 414, il est Wed Mar  8 16:22:57 2023
Je suis le processus 414, il est Wed Mar  8 16:22:59 2023
Je suis le processus 414, il est Wed Mar  8 16:23:01 2023
Je suis le processus 414, il est Wed Mar  8 16:23:03 2023
Je suis le processus 414, il est Wed Mar  8 16:23:05 2023
Je suis le processus 414, il est Wed Mar  8 16:23:07 2023
Je suis le processus 414, il est Wed Mar  8 16:23:09 2023
```

b) Modifier le code écrit pour l'exercice 1a en faisant en sorte que le processus principal, au lieu de boucler indéfiniment, remplace son code (par recouvrement ou commutation d'image – voir rappels de cours) par celui de l'exécutable *boucler* créé précédemment. Ce programme sera **paramétré** par le nombre de messages à afficher et leur périodicité.

Exécuter cette application pour vérifier ce qu'il se passe lorsqu'on envoie le signal SIGINT.

Exemple d'exécution : ./exo2b 10 2

/ Afficher 10 fois toutes les 2 secondes */*

```
Processus de pid 420 : protege contre SIGINT
Processus de pid 420 : Je vais devenir l'executable boucler pour afficher 10 fois toutes les 2 secondes
Je suis le processus 420, il est Wed Mar  8 16:24:26 2023
Je suis le processus 420, il est Wed Mar  8 16:24:28 2023
Je suis le processus 420, il est Wed Mar  8 16:24:30 2023
Je suis le processus 420, il est Wed Mar  8 16:24:32 2023
Je suis le processus 420, il est Wed Mar  8 16:24:34 2023
```

. . . . à vous de tester . . .

Exercice 3 – Envois de signaux entre processus

Écrire une application dans laquelle un processus père et deux processus fils s'exécutent en **parallèle**.

Chaque processus fils joue le rôle d'un capteur comptant le nombre de véhicules qui passent sur la route sur laquelle il est placé. À chaque fois que son compte atteint un multiple de NB, il envoie un signal à son père pour lui indiquer que NB véhicules de plus sont passés.

Le processus père affiche le nombre de véhicules qui sont passés sur chaque route **au fur et à mesure** de la réception des informations.

Pour que l'application se termine **proprement**, on va supposer qu'un capteur termine sa tâche lorsque NBL véhicules sont passés sur sa route et que le processus père se termine lorsque ses fils se sont terminés.

NBL et NB sont des **paramètres** de l'application.

Attention : Exécutez votre programme **plusieurs fois**, que constatez-vous parfois ? Est-ce **normal** ? Et si oui, **pourquoi** ?

Remarque : compter les véhicules pourra se faire en comptant les caractères frappés au clavier.

Exemple d'exécution : ./exo3 6 2

/ Les capteurs se terminent quand ils ont compté 6 véhicules (chacun) et envoient la mise à jour au père tous les 2 véhicules comptés – On saisit les caractères abcdefghi (<entrée> est un caractère)*/*

```
abcde
Pere (377) - Capteur 0 : nombre de vehicules = 2
        Capteur 1 (379) : 2 vehicules de plus => 2
        Capteur 0 (378) : 2 vehicules de plus => 2
Pere (377) - Capteur 1 : nombre de vehicules = 2
        Capteur 1 (379) : 2 vehicules de plus => 4
Pere (377) - Capteur 1 : nombre de vehicules = 4
fgh
        Capteur 1 (379) : 2 vehicules de plus => 6
        Capteur 0 (378) : 2 vehicules de plus => 4
Pere (377) - Capteur 0 : nombre de vehicules = 4
        Capteur 1 (379) : Termine
Pere (377) - Capteur 1 : nombre de vehicules = 6
i
        Capteur 0 (378) : 2 vehicules de plus => 6
Pere (377) - Capteur 0 : nombre de vehicules = 6
        Capteur 0 (378) : Termine
        Pere (377) - Je me termine en dernier
```

Exercice 4 – Affichages périodiques

On veut modifier l'application précédente pour que le père affiche la valeur des compteurs à intervalles de temps **réguliers** et non pas à chaque réception de l'information.

Le comportement des fils reste identique.

Le père se termine quand les compteurs des fils ont atteint la valeur limite NBL.

L'application doit **pouvoir** être arrêtée par un signal SIGINT.

Dans cet exercice, il est **interdit** d'utiliser la fonction *sleep()*.

Exemple d'exécution : ./exo4 6 2 2

/ Les capteurs se terminent quand ils ont compté 6 véhicules (chacun) et envoient la mise à jour au père tous les 2 véhicules comptés. Le père affiche les compteurs toutes les 2 secondes – On saisit les caractères abcdefghij */*

```
abcPere (390) - Wed Mar  8 16:14:42 2023 : Capteur 0 : nombre de vehicules = 0
Pere (390) - Wed Mar  8 16:14:42 2023 : Capteur 1 : nombre de vehicules = 0
defPere (390) - Wed Mar  8 16:14:44 2023 : Capteur 0 : nombre de vehicules = 0
Pere (390) - Wed Mar  8 16:14:44 2023 : Capteur 1 : nombre de vehicules = 0

        Capteur 1 (392) : 2 vehicules de plus => 2
        Capteur 0 (391) : 2 vehicules de plus => 2
        Capteur 1 (392) : 2 vehicules de plus => 4
ghPere (390) - Wed Mar  8 16:14:46 2023 : Capteur 0 : nombre de vehicules = 2
Pere (390) - Wed Mar  8 16:14:46 2023 : Capteur 1 : nombre de vehicules = 4
iPere (390) - Wed Mar  8 16:14:48 2023 : Capteur 0 : nombre de vehicules = 2
Pere (390) - Wed Mar  8 16:14:48 2023 : Capteur 1 : nombre de vehicules = 4

        Capteur 1 (392) : 2 vehicules de plus => 6
        Capteur 0 (391) : 2 vehicules de plus => 4
        Capteur 1 (392) : Termine
Pere (390) - Wed Mar  8 16:14:50 2023 : Capteur 0 : nombre de vehicules = 4
Pere (390) - Wed Mar  8 16:14:50 2023 : Capteur 1 : nombre de vehicules = 6
j
        Capteur 0 (391) : 2 vehicules de plus => 6
        Capteur 0 (391) : Termine
Pere (390) - Wed Mar  8 16:14:52 2023 : Capteur 0 : nombre de vehicules = 6
Pere (390) - Wed Mar  8 16:14:52 2023 : Capteur 1 : nombre de vehicules = 6
        Pere (390) - Je me termine (en dernier)
```

Exemple d'exécution : ./exo4 6 2 1

/* Ici, le père affiche toutes les secondes. On a arrêté l'exécution par un Ctrl-C */

```
Pere (393) - Wed Mar  8 16:17:34 2023 : Capteur 0 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:34 2023 : Capteur 1 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:35 2023 : Capteur 0 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:35 2023 : Capteur 1 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:36 2023 : Capteur 0 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:36 2023 : Capteur 1 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:37 2023 : Capteur 0 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:37 2023 : Capteur 1 : nombre de vehicules = 0
^CPere (393) - Wed Mar  8 16:17:38 2023 : Capteur 0 : nombre de vehicules = 0
Pere (393) - Wed Mar  8 16:17:38 2023 : Capteur 1 : nombre de vehicules = 0
```