TP6 : Signaux et groupements de processus

Système M1

semaine du 8 novembre

1 Signaux bloqués

Question 1. Écrire une fonction **`void`** `tetue()` qui bloque le signal `SIGINT` le temps de compter jusqu'à trois. Le masque des signaux bloqués sera remis à sa valeur originelle avant le retour de la fonction.

```
$ ./tetue
je compte jusqu'a trois sans me faire interrompre par SIGINT
1
(^C) 2
3
ok c'est fait
```

Question 2. Faites en sorte que votre fonction soit consciente qu'un signal est en attente :

```
$ ./tetue
je compte jusqu'a trois sans me faire interrompre par SIGINT
1
(^C) 2
oui, SIGINT est en attente, mais je continue de compter
3
oui, SIGINT est en attente, mais je continue de compter
ok c'est fait
```

2 Terminaux, sessions, et groupes de processus

Question 3. Que fait la commande `cat > toto` ? Essayez de la lancer en avant puis en arrière-plan. Même question pour la commande `cat toto`. Vous pouvez utiliser les commandes `jobs`, `ps`, `bg` et `fg`.

Question 4. Écrire une fonction **`void`** `disp_proc_info()` qui pour le processus appelant affiche : son *pid*, le *pid* de son père (*ppid*), l'identifiant de son groupe (*pgid*), l'identifiant de sa session (*sid*), et indique s'il s'exécute en avant ou en arrière-plan ou qu'il n'a pas de terminal de contrôle.

```
$ ./infos
pid=12879 ppid=17821 pgid=12879 sid=17821 foreground
```

Question 5. Écrivez un programme qui crée un fils, et où les deux processus appellent la fonction `disp_proc_info`.

Question 6. Faites en sorte de placer successivement le fils :
– dans une session séparée
– en arrière-plan, dans la même session que son père
– seul en avant-plan, dans la même session que son père

Question 7. Modifiez votre Mini-Shell pour lancer en arrière-plan les commandes qui se terminent par `&`. Le shell redonnera alors directement le prompt. Le shell devra détecter quand une commande en arrière-plan se termine et afficher alors un message indiquant la valeur de retour de cette commande (ainsi que le *pid* du processus afin de savoir quelle commande s'est terminée).

```
$ ./mini-shell
> sleep 5 &
[!] Lancement du processus 27280.
> sleep 10
[!] Fin du processus 27280, valeur de retour 0.
Valeur de retour: 0
>
```