ED 2
Gestion des processus

Exercice 1 : Création de processus sous UNIX

Pour mieux comprendre le fonctionnement de la fonction de création de processus (fork()), nous l'utilisons dans une boucle itérative.

Question

Ecrire un programme en " C " qui fait appel à la fonction " fork ( ) " dans une boucle " for (i=0; i<3; i++) ". A l'intérieur de chaque itération le programme affichera les informations suivantes :

(i : **valeur_de_i**) je suis le processus : **pid**, mon pere est : **ppid**. Où

- " **valeur_de_i** " est la valeur du compteur de la boucle
- **" pid "** est le PID du processus courant,
- **" ppid "** est le PID du processus père du processus courant,

Dessiner l'arbre des processus correspondant à l'exécution de ce programme.

Exercice 2 : Exemple de processus UNIX

Soit le programme C suivant :

```
#include <unistd.h>

void main() {
pid_t p1, p2, p3, p4;
int i;

if ((p1=fork())==0)
   if ((p2=fork())==0)
printf("je suis le processus p2, mon numero est %d \n", getpid());
   else  execlp("a", "a", NULL);

if ((p3=fork()==0) { execlp("b","b", NULL); }
if ((p4=fork()==0) { execlp("c","c", NULL); }
sleep(5);
/* attente de terminaison des fils*/
while((i=waitpid(-1,NULL,0))>0) printf(" \nFils %d termine\n ",i) ;
}
```

Question 1

Tracer l'arborescence des processus créés par ce programme si les programmes a, b et c se terminent tous par l'instruction : exit(2);

Pour simplifier, nous supposons que :

- les programmes a, b, c durent respectivement 2s, 3s et 1s
- tous les processus partagent le même processeur
- un processus qui obtient le processeur le garde jusqu'à ce qu'il se termine sauf dans le cas où il passe à l'état endormi (exécute la fonction `sleep()`).

Question 2

Dans ces conditions précises, représenter à l'aide d'un diagramme le comportement de chaque processus de l'arborescence.