TD3 – Processus

Exercice 1

Comparez les codes de retour d'un processus et la valeur de retour d'une fonction. Comment y accède-t-on dans chacun des cas, montrez-le sur un exemple de votre choix.

Exercice 2

On considère les programmes sources (langage C), des cinq commandes cmd1, cmd2, cmd3, cmd4 et cmd5.

```
/* source de la commande cmd1*/
f1(){char *p= "aaaaaaaaaaaaaaaaaaaa";
p[2]='s';}
main(){ int i;
for(i=0;i<50000000;i++) f1();
}

/* source de la commande cmd2*/
f2(){char p[]= "aaaaaaaaaaaaaaaaaaaa";
p[2]='s';}
main(){ int i;
for(i=0;i<50000000;i++) f2();
}

/* source de la commande cmd3*/
f3(){static char *p= "aaaaaaaaaaaaaaaaaaaa";
p[2]='s';}
main(){ int i;
for(i=0;i<50000000;i++) f3();
}

/* source de la commande cmd4*/
f4(){static char p[]= "aaaaaaaaaaaaaaaaaaaa";
p[2]='s';}
main(){ int i;
for(i=0;i<50000000;i++) f4();
}

/* source de la commande cmd5*/
#define f5()    {char *p= "aaaaaaaaaaaaaaaaaaaa"; p[2]='s';}
main(){ int i;
for(i=0;i<50000000;i++) f5();
}
```

L'exécution de chacune de ces commandes, par l'intermédiaire de la commande time (qui fournit les temps réels, CPU en mode utilisateur et CPU en mode noyau), dans les mêmes conditions sur une station de travail ou aucune autre activité ne s'exécutait, a fourni les résultats suivants (seuls les temps en mode utilisateur ont été conservés ici, les temps en mode système étant quasi nuls):

	Cmd1	***cmd2***	***cmd3***	***cmd4***	***cmd5***
Temps utilisateur	10.10s	35.30s	10.10s	9.08s	2.03s

1) Expliquez les différences de temps d 'exécution entre ces cinq commandes.
2) Quels temps d'exécution peut-on espérer dans des conditions analogues pour chacune des cinq commandes obtenues avec une initialisation de **p** par une chaîne deux fois plus longue. Indiquez pour chaque commande si le temps espéré est globalement identique, inférieur ou supérieur.

Exercice 3

Ecire un programme C permettant de lancer l'exécution concurrente des binaires dont le nom est passé en argument (le nombre de binaires est quelconque). Affichez les pid et ppid des tâches et analysez-les.

Exemple: $ prog1 cmd1 cmd2 cmd3 cmd4

Exercice 4

On considère le programme C (et le binaire exécutable correspondant) suivant:

```
/* Exo4.c       */
main()
{ int n=100;
        printf("Bonjour --> "); n*=2;
        switch(fork())
{
                case -1: error(); break;
                case 0: sleep(1); printf("dans le fils, adresse de n= %p\n",&n);
                        n+=10; sleep(1);printf("n= %d\n",n);break;
                default: printf("dans le père, adresse de n= %p\n",&n); n+=10;
                        sleep(3); printf("n= %d\n",n);
}
}
```

Qu'est-ce qui est possible et qu'est-ce qui ne l'est pas dans l'exécution suivante?
$ ./Exo4

Bonjour --> dans le père, adresse de n= 142e8
Bonjour --> dans le fils, adresse de n= 23200
n=210
n=220
$

Exercice 5

Donnez le code d'un pogramme C dont la demande d'exécution donne naissance à trois processus P1, P2 et P3, les deux derniers étant descendants de P1 et d'autre part, conduit à cette configuration pour les tables du système.