Faculté des Sciences & Techniques — Université de Limoges

Licence 3ème année

Programmation Concurrente

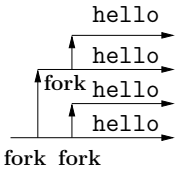
TD n°1

Manipulation des appels système fork, pipe

Manipulation de fork

1 – Soit l'exemple de schéma d'exécution suivant :

```
1 int main() {
2    fork();
3    fork();
4    printf("Hello\n");
5    exit(0);
6 }
```



Donnez le schéma d'exécution de :

```
1 int main() {
2    fork();
3    fork();
4    fork();
5    printf("Salut\n");
6    exit(0);
7 }
```

2 – Combien de lignes « salut » affiche chacun de ces programmes :

Programme 1 :

```
1 int main() {
2    int i;
3    for (i=0; i<2; i++)
4       fork();
5    printf("salut\n");
6    exit(0);
7    }
```

Programme 2 :

```
1 void go() {
2    fork();
3    fork();
4    printf("salut\n");
5 }
6 int main() {
7    go();
8    printf("salut\n");
9    exit(0);
10 }
```

Programme 3 :

```
1 int main() {
2    if (fork())
3       fork();
4    printf("salut\n");
5    exit(0);
6    }
```

Programme 4 :

```
1 int main() {
2    if (fork() == 0)
3    { if (fork())
4       {
5          printf("salut\n");
6       }
7    }
8 }
```

3 – Écrire un programme qui crée 10 processus fils :

▷ chacun d'entre eux devra afficher dix fois son numéro d'ordre entre 0 et 9.

Vérifiez que votre programme affiche 100 caractères.

▬▬▬ Manipulation de fork & pipe

4 – Crible d'Erathostène :
Tony Hoare a proposé, il y a longtemps, de calculer les nombres premiers inférieurs à n^2 en utilisant $n + 2$ processus, qui :

a. reçoivent de leur prédécesseur une suite croissante de nombres naturels ;
b. considérent le premier, p , reçu comme un nombre premier qui est imprimé ;
c. transmettent à leur successeur les nombres de cette suite, dans le même ordre, à l'exclusion de p et de ses multiples.

En utilisant des tubes et des fork, **écrire un programme C** qui affiche la suite des nombres premiers par la méthode du crible d'Erathostène :

i. un processus, P_1 est chargé de générer les entiers naturels, dont on élimine d'abord les multiples de 2, puis 3, 5 etc. au moyen de processus filtrants successifs ;
ii. chaque processus P_i crée son successeur, P_{i+1}, auquel il est relié par un tube et lui envoie la suite de nombre entiers filtrés.

▬▬▬ Manipulation des signaux

Remarques : Pour utiliser les signaux, il est recommandé d'utiliser les instructions suivantes :

```
1 struct sigaction nvt, old;
2 memset(&nvt,0,sizeof(nvt));
3 nvt.sa_handler = ma_fonction;
4 sigaction(SIGUSR1, &nvt, &old);
```

5 – Écrire un programme qui se termine uniquement au 5ème « `Ctrl-C` ».

6 – Écrire un programme P créant deux fils :

a. P envoie le signal SIGUSR1 à son second fils ;
b. à la réception de ce signal, le second fils envoie le signal SIGUSR2 au premier fils (qui provoque sa terminaison) avant de s'arrêter.