Université de Reims
Champagne-Ardenne
U.F.R. de Sciences
Exactes et Naturelles

Licence
Informatique
Info 0403
2020/2021

Travaux Dirigés 1 & 2

Processus

Exercice 1

Soit le code suivant :

```
int main(int argc, char* argv[]){
        int k = 0;
        for ( i = 0 ; i < 5 ; i++ ){
                k = fork();
                printf(" i =  d, k =  d, pid =  d \n", i, k, getpid());
        }

        exit(0);
}
```

1) Qu'affiche ce code ?

Il affiche les processus père et fils.

2) Expliquez le nombre de processus crées.

On a 62 affichages car à chaque tour de boucle les processus se dédoublent.

3) Corrigez ce code pour qu'il crée 5 processus.

```
int main(int argc, char* argv[]){
   int k = 0;
   for ( i = 0 ; i < 5 ; i++ ){
      if(( k = fork()) == 0)
         printf(" i =  d, k =  d, pid =  d \n",i,k,getpid());
   }

   exit(0);
}
```

Exercice 2

1) Écrire un programme qui exécute la commande "ls -al".

```
int main(int argc, char* argv[]){
        execl("/bin/ls", "ls", "-al", NULL);
        exit (0);
}
```

2) Écrire un programme qui exécute au choix : "ls", "pwd", "ps -e".

```
int main(int argc, char* argv[]){
   int rep;
   scanf("%d", &rep);

   if(rep == 1) execl("/bin/ls", "ls", NULL);
   if(rep == 2) execl("/bin/pwd", "pwd", NULL);
   if(rep == 3) execl("/bin/ps", "ps", "-e", NULL);

   exit(0);
}
```

3) Écrire un programme qui exécute tant que l'utilisateur le désire une commande parmi : "ls", "pwd", "ps".

```
int main(int argc, char* argv[]){
   int rep;
   printf("0- Quitter\n1- ls\n2- pwd\n3- ps\nChoix: \n")
   scanf("%d", &rep);

   while(rep != 0){
      if(fork() ==0 ){
         if(rep == 1) execl("/bin/ls", "ls", NULL);
         if(rep == 2) execl("/bin/pwd", "pwd", NULL);
         if(rep == 3) execl("/bin/ps", "ps", NULL);

         exit (0);
      }
      printf("0- Quitter\n1- ls\n2- pwd\n3- ps -e\nChoix: \n")
      scanf("%d", &rep);
   }

  exit(0);
}
```

Exercice 3

1) Écrire un programme qui crée 10 processus fils. Ces processus fils afficheront leur Pid.

```
int main(int argc, char* argv[]){
   int i;
   for(i = 0; i < 10; i++){
      if(fork() == 0){
         printf("pid: %d", getpid());
         exit(0);
      }
   }
   printf("pid: %d", getpid());

   exit(0);
}
```

2) Écrire un programme qui crée séquentiellement 10 processus fils. Ces processus fils afficheront leur Pid.

```
int main(int argc, char* argv[]){
   int i;
   for(i = 0; i < 10; i++){
      if(fork() == 0){
         printf("pid: %d", getpid());
         exit(0);
      }
      wait(NULL);
   }
   printf("pid: %d", getpid());

   exit (0);
}
```

3) Écrire un programme qui crée s´séquentiellement 10 processus fils. Ces processus fils créeront 10 processus fils. Chaque processus affichera son Pid et chaque processus père attendra la terminaison de tous ses processus fils avant de terminer.

```
int main(int argc, char* argv[]){
   int i, j;

   for(i = 0; i < 10; i++){
      if(fork() == 0){
         for(j = 0; j < 10; j++){
            if(fork() == 0){
               printf("pid: %d", getpid());
               exit(0);
            }
         }

         for(j = 0; j < 10; j++){
            wait(NULL);
         }

         printf("pid: %d", getpid());
         exit(0);
      }
      wait(NULL);
   }
   printf("pid: %d", getpid());

   exit (0);
}
```

Exercice 4

Écrire un programme qui lance dix fils qui effectuent une course. À la fin du programme, l'ordre d'arrivée des fils est affiche (chaque fils effectuera, par exemple une boucle vide de n tours, avec n choisit au hasard entre 5000 et 10000).

```c
int main(int argc, char* argv[]){
    int i;

    for (i = 0; i < 10; i++) {
        if (fork() == 0) {
            srand(time(NULL));
            int k = (rand() % 5001) + 5000;
            while(k != 0) {k -= i;}
            printf("Processus: %d, de pid : %d", i, getpid());
            exit(0);
        }
    }
    exit (0);
}
```