Ecole Supérieur en Informatique de Sidi Bel Abbès

Module: **Système d'exploitation 1** **1ère année Cycle Secondaire**
Semestre **: S1** Année universitaire: **2020-2021**

Fiche de TD/TP
Processus

Exercice 1

1) Pour chacune des transitions suivantes entre les états des processus, indiquer si la transition est possible. Si c'est le cas, donner un exemple d'un élément qui pourrait en être à l'origine.
   a. en exécution – prêt
   b. en exécution – bloqué
   c. bloqué – en exécution
   d. en exécution – terminé

2) Sous UNIX, un processus Zombie est un processus :
   a. Qui a perdu son père ?
   b. Qui a terminé son exécution en erreur ?
   c. Qui a terminé son exécution et attend la prise en compte de cette fin par son père ?

3) Dans le système UNIX, est-ce que tout processus a un père ? Que se passe-t-il lorsqu'un processus devient orphelin (mort de son père) ?

4) Dans le cas d'UNIX, la création de processus est réalisée par duplication. Citez un avantage et un inconvénient.

5) Quelle est la différence entre un thread et un processus ?

6) Soit un système avec *n* processus, combien existe-t-il de manières d'ordonnancer ces processus ?

Exercice 2 (TP)

Le programme suivant permet d'illustrer le comportement du fork.

```
int main ()
{
      int code_retour ;
      printf ("Debut du test fork ()\n");
      code_retour = fork ();
      switch (code_retour ) {
          case -1 :
               printf ("Pbm lors de la creation du processus\n");
               break;
          case 0 :
               printf ("Je suis le processus fils \n");
               break;
          default :
               printf ("Je suis le processus père\n");
               printf ("Je viens de créer le processus fils dont le pid est %d
               \n",code_retour);
               break;
      }
       printf ("code_retour %d\n", code_retour);
       printf ("Fin du test fork ()\n");
      return 0 ;
}
```

Exécuter ce programme puis expliquer le résultat obtenu.

Exercice 3

Donner les résultats de l'exécution du programme ci-dessous en supposant que l'identificateur du processus (PID) correspondant à ce programme est 2500 et que le système affecte des identificateurs séquentiels aux nouveaux processus.

```
/* Programme p3.c */
int main ( )
{
      int i=4, j=10;
      int p; /* ou pid_t p */
      p = fork(); /* 1 */
      j += 2;
      if (p == 0){
             p = fork(); /* 2 */
             i += 3; j += 3;
      }
      else{
             p = fork(); /* 3 */
             i *= 2; j *= 2;
      }
      printf("\nprocessus=%d, i=%d, j=%d; Processus Père=%d" , getpid(), i , j ,
getppid() );
      exit(0);
}
```

Exercice 4

1. Combien de processus engendre l'évaluation de la commande C :
   fork() && (fork() || fork()) ;

2. Dessiner l'arbre généalogique des processus engendrés par cette ligne.