IGI 3004 - Système d'exploitation

Responsable Jean Cousty

TD #1

Instructions

- Ne vous contentez pas d'écrire un résultat. Justifiez et commentez vos réponses.
- Pour les réponses qui demandent des programmes ou des morceaux de programmes, vous pouvez utiliser du code C ou n'importe quel style de pseudo-code suffisamment raisonnable (*i.e.*, la syntaxe et la sémantique du programme doivent être claires).
- Des réponses longues ne sont pas nécessaires. La plupart des questions sont conçues pour une réponse brève, de l'ordre d'un paragraphe.
- Lisez l'énoncé d'un exercice jusqu'au bout avant de commencer à répondre à la première question.

Fork et exec

Question 1. Pour chacun des programmes ci-dessous :

- décrivez le déroulement du programme ;
- dessinez l'arbre de création des processus ;
- indiquez le nombre de processus créés à l'exécution du programme ;
- indiquez si l'affichage est le même à chaque exécution et donnez un affichage possible.

```
   /* Programme 1                                                    */
1  int main(){
2  |  int pid, i;
3  |  for (i = 1 ; i <= 4 ; i++) {
4  |  |  pid = fork();
5  |  |  if(pid != 0) printf("%d \n", pid);
   |  }
   }
```

```
   /* Programme 2                                                    */
1  int main(){
2  |  int pid, i;
3  |  for (i = 1 ; i <= 4 ; i++) {
4  |  |  pid = fork();
5  |  |  if(pid == 0) break;
   |  |  else printf("%d \n", pid);
   |  }
   }
```

```
   /* Programme 3                                                    */
1  int main(){
2  |  int pid, i, nb = 4;
3  |  for (i = 1 ; i <= 4 ; i++) {
4  |  |  pid = fork();
5  |  |  if(pid == 0) execlp("ls", "ls", NULL);
6  |  |  if(pid < 0) exit(1);
   |  }
7  |  while(wait(NULL) > 0);
8  |  printf("ls se répète\n");
   }
```

```
   /* Programme 4                                                    */
1  int main(){
2  |  int pid, i;
3  |  for (i = 1 ; i <= 4 ; i++) {
4  |  |  pid = fork();
5  |  |  if(pid < 0) exit(1);
   |  |  execlp("ls", "ls",NULL);
   |  }
   |  while(wait(NULL)>0);
   |  printf("ls se répète\n");
   }
```

```
   /* Programme 5                                                    */
1  int main(){
2  |  int pid1, pid2, i;
3  |  for (i = 1 ; i <= 4 ; i++) {
4  |  |  pid1 = fork();
5  |  |  pid2 = fork();
6  |  |  if(pid1 < 0) exit(1);
7  |  |  if(pid2 < 0) exit(2);
8  |  |  execlp("ls", "ls",NULL);
9  |  |  execlp("pwd", "pwd",NULL);
   |  }
10 |  while(wait(NULL) > 0);
11 |  printf("pwd se répète\n");
   }
```

Question 2. Étant donné deux programmes nommés client et serveur, écrivez un programme qui génère l'exécution simultanée de 4 clients et d'un serveur.

Tubes

Un développeur souhaite utiliser dans un programme une fonction `void mystere(void)`, dont il n'a pas le code source mais qui fonctionne parfaitement bien. La fonction `mystere()` génère un certain volume d'octets sur sa sortie standard. Le développeur souhaite donc écrire un programme qui récupère les octets produits par la fonction `mystere(void)` dans un tampon de nom `buf` pour pouvoir leur appliquer certains traitements ultérieurement. Pour simplifier, on suppose que le volume des octets produits est toujours inférieur à `GRANDE_TAILLE` octets. Voici le code proposé par le développeur pour réaliser cette sorte de redirection.

```
1 int main(){
2 |   char buf[GRANDE_TAILLE];
3 |   int tube[2], n;
4 |   pipe(tube);
  |   /* La ligne suivante redirige la sortie standard sur l'entrée du tube/      */
5 |   dup2(tube[1],1);
  |   close(tube[1]);
6 |   mystere();
  |   /* Lire les données du tube et les mettre dans le tampon buf                */
7 |   n = read(tube[0], buf, GRANDE_TAILLE);
  |   /* Traite le tampon buf...                                                   */
  }
```

Le programme fonctionne lorsque le volume d'octets générés par la fonction `mystere` est petit mais ne se termine pas lorsque le volume devient trop important (tout en restant inférieur à `GRANDE_TAILLE`.

Question 1. Expliquez ce comportement. À quoi correspond la valeur seuil à partir de laquelle le programme ne fonctionne plus ?

Question 2. Proposez une nouvelle version du programme qui fonctionne pour tout volume d'octets compris entre 0 et `GRANDE_TAILLE`.